Theory and Numerical Implementation of Greedy Algorithms in Highly Nonlinear Approximation

by

Philip C. Mendelsohn

A Thesis submitted to

the Faculty of Graduate Studies

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Mathematics

University of Manitoba

Winnipeg, Manitoba

defended June 2, 2006

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES

*****

COPYRIGHT PERMISSION

Theory and Numerical Implementation of Greedy Algorithms in Highly Nonlinear Approximation

BY

Philip C. Mendelsohn

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of Manitoba in partial fulfillment of the requirement of the degree

OF

MASTER OF SCIENCE

Abstract

Greedy algorithms in nonlinear approximation have been studied in mathematics, signal processing, and statistics. In this thesis, they are investigated as part of approximation theory, with applications to one-dimensional signal processing.

Necessary background is given for understanding the germane properties of bases. Atoms and redundant dictionaries are introduced. Approximation results of greedy algorithms are stated. The computational complexity of greedy algorithms is discussed.

Some ways that greedy algorithms can be rendered tractable are considered. Past numerical implementation of greedy algorithms are examined, as are possible adaptations. Theoretical aspects and practical issues of implementation are discussed.

Acknowledgments

I would like to thank, above all, Kirill Kopotun for his patience, tolerance, and encouragement in my attempts to approximate a mathematician.

I gratefully acknowledge Joe Williams and Alex Leblanc for taking the time out of their busy schedules to serve as examiners and wade through this stack of papers.

As well, I would like to thank Peter Aitchison specifically, for being instrumental in bringing me to the University of Manitoba, and I would like to thank Tom Berry, generally.

Thanks to my colleague, Xuan Li, for being quiet, helpful, and fun, each at exactly the right time. Thanks to Dave Gabrielson for being clueful, a catalyst, and always cooperative.

General thanks go to each of the faculty members and students in the Math Department for maintaining an open, friendly, and collegial atmosphere that is second to none.

Finally, thanks to the Department of Mathematics and Faculty of Science at the University of Manitoba for providing the opportunity and funding to pursue these studies.

To my family:

Becky,

Mom, August, Emily, Steve, Dad,

Garth and Arlene.

Some may enumerate, but these are the people who count.

Table of Contents

List of Figures

List of Tables

Chapter 1

Preliminaries and Introduction

"To separate mathematics from the sciences is to invite the sterility of a cow locked away from the bulls."

—P. Chebyshev

This section will provide a description of some of the mathematics and terminology required to look at the subject under consideration. It is intended to be brief, assuming the reader has some familiarity with the underlying mathematics, but references are included for those who would like to read more.

It is tacitly and wrongly assumed that there exist no interesting applications of these mathematics outside the field of signal processing. The author wishes to thank the reader for restraint in sharing counterexamples.

1.1 Terms, Usage, and Notation

Since the aim of this thesis is to explain rather than confuse, a few words about the conventions adopted are in order. Some notation is also included; Appendix A is an attempt to accompany the text with a convenient guide to notation.

The word "signal" will be used to mean a real-valued function of a single real variable. The term "time" will mean a real variable, $t \in \mathbb{R}$, and the expression "a function in the time-domain" is then a signal. A periodic function is known as "stationary," while we shall refer to non-periodic signals as "non-stationary". This is looking at the signal through a time-window, which will be discussed in section 1.5.

We use the word "spectrum" to mean the description of a function in the frequency domain; this is the Fourier transform of a time-domain signal. This is a signal processing convention, and is related, but not directly, to the spectrum of an operator in functional analysis.

We may refer to (digital) signal processing as "DSP". We use the word "filter" in a DSP context, which derives from the natural language notion of a filter. A water filter stops particulate matter while allowing water to pass through. Mathematically, we use a filter to process the spectrum of a signal. The idea that a filter is like a tone control on a stereo is fairly accurate, if not precise. As practitioners of signal processing like to say, "everything is a filter." We shall discuss this at more length in section 1.4.

A dyadic interval is a natural concept, and in a sense is the simplest fractal known. If you divide an interval into half, then each sub-interval into half, and so forth, you have made a set of dyadic intervals. Allowing for translation, we have

Definition 1.1.1 (Dyadic Interval). For $j, k \in \mathbb{Z}$,

$$I_{j,k} = [k2^{-j}, (k+1)2^{-j}] \tag{1.1.1}$$

We use the term "regularity," as in the regularity of a wavelet, to mean the number of continuous derivatives. This is used as a measure of smoothness. It is less sophisticated than the modulus of smoothness (see below and section 1.2.5).

Symbols

For $\sqrt{-1}$, we will use the symbol i as is the convention in mathematics, rather than the symbol j commonly used in literature published in the context of electrical engineering. If z is a complex quantity (scalar or complex-valued function) then either $\overline{z}$ or $z^\star$ will stand for the complex conjugate.

In general, t will mean the time-domain variable, and ξ will be the frequency domain variable. The symbol ω will be taken to be $2\pi\xi$, a convention for expressing frequency as angular velocity, which is useful when writing complex exponentials. When working under discrete systems, we will consider $-\frac{1}{2} < \xi < \frac{1}{2}$ or $-\pi < \omega < \pi$, normalizing so the Nyquist frequency (sampling frequency) is $\xi = 1$.

The notation $\mathbb{T}$ will be used to indicate the space $\mathbb{R} \mod 2\pi$, also known as the circle. If we write $\mathbb{T}^d$, that is the d-dimensional torus, i.e., the product space of $\mathbb{T} \times \mathbb{T} \times ... \times \mathbb{T}$, where the total number of factors is d.

The notation $\omega_r(f, t)$ will mean the r-th modulus of smoothness (see section 1.2.5), and should not be confused with angular velocity. The meaning should be clear providing attention is paid to context.

The notation $A \asymp B$ should be read that there exist two positive constants C_1

and C_2 such that $C_1 A \leq B \leq C_2 A$, for some positive quantities A and B. Thus A and B are equivalent, up to constants.

1.2 Approximation Theory

Intuitively, approximation is finding a simple object that functions as a replacement for another object, in some sense. The objects that we wish to replace are functions f, which are "complicated" elements of some space X. We try to replace f by $\phi \in \Phi$, where $\Phi \subset X$. Usually Φ consists of "simple" functions, such as polynomials, trigonometric polynomials, splines (piecewise polynomials), wavelets, or some other elementary function. It is not only possible but usual that ϕ is a linear combination of functions: $\phi = \sum_k c_k \phi_k$.

Approximation theory, then, is the study of the distance between some set of functions and a set of approximating functions. The first definitions deal with sets containing a single function.

1.2.1 Approximation Error

We want to measure how "close" f is to ϕ by considering both to be elements of some metric space X. While noting that approximation is possible in even more general settings, we will take X to be a normed or quasi-normed space, which always induces a metric.

Therefore, the distance is given by $\|f - \phi\|_X$. Note that the choice of norm is quite significant, as it dictates the space in which approximation takes place; we are not necessarily limited to the induced metric. The norm chosen may depend on either

desirable mathematical properties, or properties of some physical problem, such as the human auditory system in perceptual coding, for example. Once we know our metric, we can define the *error of approximation* to be the distance from an arbitrary function f to the approximating set $\Phi \subset X$.

Definition 1.2.1 (Error of Approximation). Let f be an element of a space of functions X, and let $\Phi \subset X$. The "error of approximation" of f by functions from Φ is defined as

$$E(f) := E(f, \Phi)_X := \inf_{\phi \in \Phi} \|f - \phi\|_X. \tag{1.2.1}$$

1.2.2 Best Approximation

The error of approximation is an infimum, so we need to describe the case of a function that achieves the infimum:

Definition 1.2.2 (Best Approximant). Let f be an element of X, a Banach space, let $\Phi \subset X$, and let ϕ_0 be a function in Φ such that

$$\|f - \phi_0\|_X = \inf_{\phi \in \Phi} \|f - \phi\|.$$

Then ϕ_0 is a "*best approximant*" of f from Φ.

The best approximant may not exist, which is often the case. If the best approximant does exist, it may not be unique, which we will see contributes to the computational complexity of the algorithms in Chapter 2. If the best approximant can't be determined, then we must also find a way to say precisely what is "good enough."

1.2.3 (ε, m)-approximation

As mentioned, the approximants may be a linear combination of functions. We then talk about an approximation that falls within a given tolerance ε using some number of terms, usually m. Accordingly, we define a (ε, m)-approximation, noting that the coefficients c_i could just as easily be in $\mathbb{C}$.

Definition 1.2.3. Let f be an element of X, let $\Phi \subset X$ be an approximating set, and let c_i be in $\mathbb{R}$. For $\varepsilon > 0$, an approximant

$$\tilde{\phi} = \sum_{i=1}^{m} c_i \phi_i, \ \ \phi_i \in \Phi$$

such that $\|\tilde{\phi} - f\| < \varepsilon$ is called an (ε, m)-approximant.

The error of approximation is written

$$E_m(f)_X := E_m(f, \Phi) := \inf_{\tilde{\phi} \in \Phi} \|f - \tilde{\phi}\|_X. \tag{1.2.2}$$

The "rate of approximation" is the change in error with respect to the change in m, i.e., we want to see how E_m depends on m, such as $E_m(f) = Cm^{-\alpha}$, where C is some constant which usually depends on the target function and α may be some parameter, such as smoothness.

1.2.4 Approximation Classes

One of the central questions of Approximation Theory is to classify those functions that can be approximated equally well by the same method, or from the same set.

Definition 1.2.4. Let $\Phi \subset X$, and let $\mathcal{F} \subset X$ be another subset of X, where X is a Banach space. Let $E(f, \Phi)$ be the error of approximation from Φ to any function $f \in X$.

The "error of approximation of $\mathcal{F}$ by Φ" is

$$E(\mathcal{F})_X := E(\mathcal{F}, \Phi)_X := \sup_{f \in \mathcal{F}} E(f, \Phi)_X. \tag{1.2.3}$$

We have the analogous notation for m-term approximations of a class $\mathcal{F}$:

$$E_m(\mathcal{F}) := E_m(\mathcal{F}, \Phi)_X := \sup_{f \in \mathcal{F}} E_m(f, \Phi)_X. \tag{1.2.4}$$

While we are interested in a lower bound for individual functions, we define a class according to the upper bound of their error of approximation.

Approximation classes are usually denoted

$$\mathcal{A}^\alpha(\Phi) := \mathcal{A}^\alpha_q(\Phi, X) \tag{1.2.5}$$

where Φ is the approximating set, X the underlying space, and α and q are parameters of the definition. The space X and some parameters may be omitted when they are constant over several expressions.

As the notation can be cumbersome, below is an example following DeVore, Petrova, and Temlyakov [18], illustrating how to specify the class of functions with the same rate of m-term approximation from a basis B as the approximation class $\mathcal{A}^\alpha_q(B, X)$.

Example 1.2.5. Let X be a Banach space of functions and B a basis. Let $0 < q \leq \infty$

and $0 < \alpha$. The approximation class $\mathcal{A}_q^\alpha(B, X)$ is defined as the set of functions $f \in X$ such that the series in the following norm is convergent, and finite in the case $q = \infty$:

$$\|f\|_{\mathcal{A}_q^\alpha(B,X)} := \begin{cases} \left(\sum_{m \geq 0} (m+1)^{\alpha q - 1} \sigma_m(f, B)_X^q \right)^{\frac{1}{q}}, & 0 < q < \infty \\ \\ \sup_{m \geq 0} (m+1)^\alpha \sigma_m(f, B)_X, & q = \infty. \end{cases} \tag{1.2.6}$$

With apologies to the reader, this example anticipates a notation, but is highly instructive. Please read $\sigma_m(f, B)$ as $E_m(f, B)$, and refer to section 1.8.1 (equation 1.8.2) for the definition of $\sigma_m(f, B)$, which is a convention for nonlinear approximation error.

1.2.5 Moduli of Smoothness

The notion of the "smoothness" of a function is a particularly slippery one. The idea of measuring smoothness by the number of continuous derivatives a function possesses is useful, but we often need a finer measurement.

The first difference operator is defined by $\Delta_h := f(x+h) - f(x)$, for $h \in \mathbb{R}$. Higher difference operators are defined recursively:

Definition 1.2.6 (r-th difference)**.**

$$\Delta_h^r(f, x) := \Delta_h \left[\Delta_h^{r-1}(f, x) \right] \tag{1.2.7}$$

This can be expressed explicitly as

$$\Delta_h^r(f, x) = \sum_{k=0}^{r} \binom{r}{k} (-1)^{r-k} f(x + kh) \tag{1.2.8}$$

and we have $\Delta^r_{-h}(f,x) = (-1)^r \Delta^r_h(f, x - rh)$.

We need these difference operators in order to define the r-th modulus of smoothness:

Definition 1.2.7 (r-modulus of smoothness). For $f \in L_p$ (assuming it to be over the compact set $[a,b]$), with $1 \leq p \leq \infty$, then the r-th modulus of smoothness is defined as

$$\omega_r(f,t)_p := \sup_{0<h\leq t} \|\Delta^r_h(f,\cdot)\|_p(A_{rh}), \quad t \geq 0 \tag{1.2.9}$$

where $A_{rh} = [a, b - rh]$, where $a \leq b - rh$.

The first modulus of smoothness, for $r = 1$, is known as the "modulus of continuity," and is assumed when there is no subscript, i.e. $\omega(f,t)$. See [21, chapter 2, §6-7].

1.2.6 Besov Spaces

The classification of functions by smoothness spaces starts with considering spaces C^r, for $1 \leq r \leq \infty$ (if the reader will pardon the abuse of the notation to include the case for infinitely differentiable function $r = \infty$). However, we can work in other spaces of functions, such as L_p, the Sobolev spaces, Lip α, and others, with which we will assume familiarity.

Besov spaces are defined here following [37]. The idea is that they have a common smoothness, measured using the r-th modulus of smoothness, and are characterized later using wavelets [20].

Definition 1.2.8 (Besov Spaces). Let $0 < \alpha < r$, let $0 < q, p \leq \infty$, $d \in \mathbb{N}$, and let D be a domain in $\mathbb{R}^d$. Then define the Besov Space $B^\alpha_q(L_p(D))$ to be all functions such

that the following holds:

$$|f|_{B_q^\alpha(L_p(D))} := \left(\int_0^\infty [t^{-\alpha} \omega_r(f,t)_p]^q \right)^{\frac{1}{q}} < \infty \tag{1.2.10}$$

with the usual change to the supremum when $q = \infty$.

Besov spaces can also be characterized as functions that can be approximated by only a few non-zero wavelet coefficients, see [20].

1.2.7 Jackson-type Inequalities

A Jackson-type inequality, or Jackson theorem, is any result that uses the smoothness of the target function to place an upper bound on the error of approximation. This is to say that the rate of approximation increases as the target functions become smoother and smoother.

A Jackson theorem is also known as a "direct theorem" of approximation. They were originally introduced in 1911 by Dunham Jackson, specifically regarding the approximation of a function by trigonometric polynomials. Further information can be found in Cheney [7, p. 139] and in more depth in DeVore and Lorentz [21, Ch. 7]. The last summarizes the four usual forms given for the rate of approximation (in $L_p, p \geq 1$) from Jackson theorems:

$$E_m(f)_p \leq \begin{cases} C_r m^{-r} \|f^{(r)}\|_p \\ C_{\alpha,p} m^{-\alpha} \|f\|_{\mathrm{Lip}^*(\alpha,p)}, \\ C_r m^{-k} \omega_{r-k}(f^{(k)}, m-1)_p, \quad 0 \leq k \leq r \\ C_r \omega_r(f, m^{-1})_p \end{cases} \tag{1.2.11}$$

From the bottom to the top, each of these can be shown to imply the other, and if there are no side conditions, the uppermost statement implies the last statement. The uppermost statement is usually the easiest to show [21].

Jackson's original theorem is stated below, to show how it serves as a template for this type of result.

Theorem 1.2.9 (Jackson's Theorem (1911)). *For all 2π-periodic and continuously differentiable functions f,*

$$E_m(f) \leq \frac{\pi}{2(m+1)}\|f'\| \tag{1.2.12}$$

and the constant $\frac{\pi}{2(m+1)}$ is the best possible.

1.2.8 Bernstein-type Inequalities

As a complement to Jackson-type inequalities, we also have Bernstein-type inequalities. Where a Jackson-type inequality sets an upper bound on $E_m(f)$ due to the smoothness of the function, Bernstein-type inequalities show that the error of approximation says something about the smoothness of the function. Where Jackson-type inequalities are direct theorems, Bernstein-type inequalities are called "inverse theorems."

We offer a definition of the space Lip α in terms of the modulus of continuity:

Definition 1.2.10. Let $\alpha \in \mathbb{R}$, with $0 < \alpha \leq 1$, and let M be a constant. Then the space of functions Lip α is the set of functions such that $\omega(f,t) \leq Mt^{\alpha}$.

We show a Bernstein-type theorem as a template for other results.

Theorem 1.2.11 (Bernstein). *Let $0 < \alpha < 1$ and $E_m(f) \leq C_r n^{-r-\alpha}$ for $n \in \mathbb{N}$, then $f^{(r)} \in \text{Lip}\alpha$.*

1.2.9 References

While the study of Approximation Theory is a worthwhile pursuit, we have reached the point of diminishing returns. Further discussion will not clarify the topics we wish to explore, and so the interested reader is referred to Achieser [1], Cheney [7], DeVore and Lorentz [21], Lorentz [47], and Rivlin [57].

1.3 Fourier Transforms

Jean Baptiste Joseph Fourier and Napoleon Bonaparte knew each other, and worked together, both in government and, to some extent, in mathematics. Fourier was on the French expedition when Napoleon's fleet was destroyed by Lord Nelson in 1798, stranding the French in Egypt. At the Cairo Institute, which was founded by Bonaparte during that time, Napoleon and Fourier served together as two of the twelve members of the mathematical division [55].

Though Fourier was a good public administrator, a pioneering Egyptologist, and successful as a political animal, it is his idea of representing a function as a series of sines and cosines that make him a part of daily life, and not just for mathematicians. Through communications, engineering, physics, and mathematics, the modern world depends more on Fourier's ideas than on those of Bonaparte.

The Fourier transform is a tool of the trade for nonlinear approximation and greedy algorithms. This is not only because of the power of trigonometric approximation, but because many of the properties of wavelets are determined by studying them in the frequency domain. As the basic notions of the Fourier transform are found in undergraduate curricula, we will summarize the results that we need here, and refer

to results in [12, 71, 49].

1.3.1 Definitions

We adopt the following definitions in L_1. We will refer to the Continuous Fourier Transform as the CFT, the discrete transform as the DFT, and indicate the inverses by prepending "I" to the acronym.

Definition 1.3.1 (Continuous Fourier Transform). Let $f(t)$ be in $L_1(\mathbb{R})$. Define

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(t)e^{-it\xi}\, dt. \tag{1.3.1}$$

The Fourier transform is thus bounded and continuous [49].

Definition 1.3.2 (Inverse Continuous Fourier Transform). Let both $\hat{f}(\xi)$ and $f(t)$ be in $L_1(\mathbb{R})$. Then the following holds almost everywhere:

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \hat{f}(\xi)e^{it\xi}d\xi \tag{1.3.2}$$

We use the corresponding definition for the N-point Discrete Fourier Transform:

Definition 1.3.3 (Discrete Fourier Transform).

$$\hat{f}[k] = \sum_{n=0}^{N-1} f[n]e^{-i2\pi kn/N} \tag{1.3.3}$$

Definition 1.3.4 (Inverse Discrete Fourier Transform).

$$f[n] = \frac{1}{N}\sum_{k=0}^{N-1} \hat{f}[k]e^{i2\pi kn/N} \tag{1.3.4}$$

1.3.2 Trigonometric polynomials

Fourier died on Pafnuty Chebyshev's ninth birthday, 16-May-1830, but that is not the most interesting connection they share. Chebyshev's exploration of trigonometric polynomials, which was the birth of approximation theory, both complements and draws on Fourier's work.

Trigonometric approximation dates back to a trip to London Chebyshev made in 1853, when he became interested in the behaviour of linkages (connecting rods) on steam engines [6]. Chebyshev's motivation was to find a way, using hinged rods (straight lines) to convert circular motion into exact straight line motion [63]. The term "degrees of freedom," as in systems of equations, comes directly from the study of linkages for reasons that should be apparent. Most linkages, however, generate only an approximation to a straight line.

Over several decades, Chebyshev designed practical linkages in addition to the mathematics he developed. His theory of approximating by trigonometric polynomials was developed to quantify the error of approximation of the linkage.

A Trigonometric polynomial is defined as follows:

Definition 1.3.5. For $x \in \mathbb{T}$, and $k \in \mathbb{Z}$, we define the trigonometric polynomials of degree $\leq n$:

$$T_n(x) = \sum_{j=1}^{n} c_j e^{ikx} \tag{1.3.5}$$

Over the reals, where $d = 1$, this is considered to be the standard n-term Fourier expansion expressed in exponential form:

$$T_n(x) = \sum_{j=-n}^{n} c_j e^{ijx} = \frac{a_0}{2} + \sum_{k=1}^{n} (a_k \cos kx + b_k \sin kx).$$

So one can consider the Fourier transform as a method of decomposing a function into an approximating trigonometric polynomial. This powerful result leads us to consider some of the properties of the Fourier transform.

1.3.3 Fourier Properties

The useful properties of the Fourier transform are summed up nicely in Mallat [49, p.25]. The table is included (see table 1.1) for convenience.

Property	Function $f(t)$	Fourier Transform $\hat{f}(\xi)$
Inverse	$\hat{f}(t)$	$2\pi f(-\xi)$
Convolution	$\widehat{f \star g(t)}$	$\hat{f}(\xi)\hat{g}(\xi)$
Multiplication	$f(t)g(t)$	$\frac{1}{2\pi}\hat{f} \star \hat{g}(\xi)$
Translation	$f(t-t_0)$	$e^{-it_o\xi}\hat{f}(\xi)$
Modulation	$e^{i\xi_0 t}f(t)$	$\hat{f}(\xi-\xi_0)$
Scaling	$f\left(\frac{t}{s}\right)$	$\lvert s\rvert\hat{f}(s\xi)$
Time derivatives	$f^{(p)}(t)$	$(i\xi)^p\hat{f}(\xi)$
Frequency Derivatives	$(-it)^p f(t)$	$\hat{f}^{(p)}(\xi)$
Complex conjugate	$f^\star(t)$	$\hat{f}^\star(-\xi)$
Hermitian Symmetry	$f(t) \in \mathbb{R}$	$\hat{f}(-\xi) = \hat{f}^\star(\xi)$

Table 1.1: Properties of the Fourier Transform

Probably the two most important properties to note are convolution and the implication of Hermitian Symmetry, which implies that for real valued time-domain functions, half of the complex coefficients can be disregarded.

1.3.4 Parseval and Plancherel Identities

If we wish to be able to perform an operation that is invertible on a function, we must be able to recover the norm of the function. This is true of the Fourier transform,

and in fact for a transform to any orthonormal basis.

For $f, g \in L^2(\mathbb{R})$ (we want g to be the appropriate complex exponential, in the Fourier case), we have $\|f\|^2 = \|\hat{f}\|^2$, or

Theorem 1.3.6 (Plancherel Identity for $L^2(\mathbb{R})$).

$$\int |f(t)|^2 \, dt = \int |\hat{f}(\xi)|^2 \, d\xi. \tag{1.3.6}$$

If we are using an orthonormal basis, we would expect the norm of any function to be related to the size of the coefficients. This is true, for not just our Fourier basis, but again $f, g \in L^2(\mathbb{R})$:

Theorem 1.3.7 (Parseval Identity for $L^2(\mathbb{R})$).

$$\int f(t)\overline{g(t)} \, dt = \int \hat{f}(\xi)\overline{\hat{g}(\xi)} \, d\xi. \tag{1.3.7}$$

We recognize the preceding left-hand side as the integral expression for the Fourier coefficients, if we take $\overline{g(t)}$ to be the desired basis function again.

1.3.5 Fast Fourier Transform

The Fourier Transform is computable due to two technological advances: first, the transistor and thus the digital computer, and secondly the Fast Fourier Transform (FFT). The FFT allows the DFT to be computed efficiently, with computational complexity of $\mathcal{O}(N \log N)$, which makes it run very quickly, for an input of a give size. In many cases, it is possible to compute in real-time.

The details of the FFT are interesting, but also quite well-documented and soft-

ware libraries for performing the algorithm in efficient and flexible ways are widely available. We will not discuss the FFT here, but assume that the reader understands that it exists and works as advertised.

There are many books which may be consulted regarding the FFT. Two sources are Wickerhauser [76] or Oppenheim and Schafer [56]. The canonical paper is Cooley and Tukey [10]. Of interest is that it seems that Gauss anticipated the algorithm in 1805 (predating Fourier's work); an account is given in [36]. Of more recent interest, a good library used for research and commercial software alike is described in [26], and at `http://www.fftw.org` via internet.

1.3.6 Convolution Theorem

Convolution is a fundamental operation for us. We encounter it in both the continuous case,

$$f \star g(t) = \int_{-\infty}^{\infty} f(u)g(t-u)du \tag{1.3.8}$$

and the discrete case

$$f \star g[t] = \sum_{u=-\infty}^{\infty} f[u]g[t-u]. \tag{1.3.9}$$

The astute reader has already noticed the similarity between the definition of convolution and the usual inner product in an infinite dimensional Hilbert space, with the only difference being a translation.

Simply put, the convolution theorem says that convolution in the time-domain is the point-wise multiplication of functions in the frequency domain (and vice-versa). In practice, the convolution theorem gives us the flexibility to choose whichever domain

suits us. Multiplication tends to be an easier framework for human analysis, while convolution is very fast in numerical (machine) calculations. The theorem holds for both the continuous and discrete cases.

Theorem 1.3.8 (Convolution Theorem). *Let $f(t)$ and $g(t)$ be two functions over $\mathbb{R}$, and let $\hat{f}(\xi)$ and $\hat{g}(\xi)$ be their respective Fourier transforms. Then*

$$\widehat{f \star g(t)} = \hat{f}(\xi)\hat{g}(\xi). \tag{1.3.10}$$

1.4 Filters

We really wish to discuss wavelets, as they make up classes of functions used in non-linear approximation. In order to have that discussion, we need some understanding of numerical digital filters, simply "filters," hereafter.

In order to keep this section in some appropriate proportion relative to greedy algorithms, the particulars of realizing filters with the desired properties have been largely omitted. The use of pole-zero (complex) analysis of filters or the z-transform (discrete Laplace transform) are also eschewed. These are considered prerequisite for much of the engineering literature. The reader will find them in [56].

We do want to discuss what filtering is, how it works, and what the properties are of those filters which become wavelets. For many interesting wavelets, there is no closed form expression, but they are quickly generated by iterating a filter.

1.4.1 Application of the Convolution Theorem

Filtering a time-domain signal in the frequency domain is a straightforward application of the convolution theorem. Define a filter as follows:

Definition 1.4.1 (Filter). Let $\hat{h}(\xi)$ be a function with support in $[-\pi, \pi]$. We say that $\hat{h}$ is the "frequency response" of the filter h.

The procedure for filtering would then be as shown in Table 1.2.

1. Transform the signal $f(t) \rightarrow \hat{f}(\xi)$.
2. Multiply by the frequency response of the filter, $\hat{h}(\xi)$ to get $\hat{f}_h(\xi) = \hat{h}(\xi)\hat{f}(\xi)$.
3. Perform the inverse transform $\hat{f}_h(\xi) \rightarrow f_h(t)$ to yield the filtered signal in the time-domain.

Table 1.2: Frequency Domain Filtering

While the FFT is computationally efficient, and the convolution theorem is lovely theory, switching back and forth between time and frequency domains seems more awkward than doing computations in only the time-domain, especially in the case of a non-stationary signal. Instead of taking the transform of the signal, we take the inverse transform of the filter, thus bringing it into the time-domain.

The inverse transform of the frequency response is known as the "impulse response." Impulse response is more a property of discrete filters and is explained in section 1.4.2.

By the convolution theorem, we could perform the procedure above in the time-domain as follows:

$$f_h(t_0) = \int_{-\infty}^{\infty} h(t) f(t_0 - t)\, dt. \tag{1.4.1}$$

1.4.2 Discrete Filters

Secretly, we are always thinking about signal processing, so we are interested in functions that are discrete and regularly sampled. In this case, we write our time domain filter coefficients as $h[m]$, with $m = 0, ..., n-1$. We consider $h[m] \in \mathbb{R}$, but it is not required. For $k \in \mathbb{Z}$, our input signal is usually $x[k]$ and the output $y[k]$. So we define digital filters thus:

Definition 1.4.2 (Digital Filters). Let k be shorthand for t_k, and $h[k]$ be a sequence of n coefficients. Then a digital filter is one that produces the function $y[k]$ from the input $x[k]$ as follows.

$$y[k] = \sum_{m=0}^{n-1} h[m]x[k-m] \tag{1.4.2}$$

We will index our filters from 0 to some positive $n-1$ for an n tap filter. This is merely so that our indexing remains consistent. Many authors, such as Hamming [34], may index from $-m$ to $+m$. It is desired that $m = \frac{n}{2}$, with the appropriate adjustments depending on whether filters are of odd or even length. The differences should be only mechanical.

To understand convolution in the time domain, see figure 1.1. The real axis is oriented conventionally, which means that negative values of t are the past while positive values of t are the future. Thus a signal that is a function of time can be seen as flowing from right to left, and the filter stays fixed. (Imagine a seismograph or polygraph).

At each time step, each input from $x[k-(n-1)]$ through $x[k]$ is multiplied by the filter coefficient under which it is currently located, as indicated by the upward arrows. The n products are summed to produce $y[k]$. To advance things in real-time,

imagine that the block containing the filter $h[k]$ shifts one unit to the right (with $j = 0, ..., n-1$, $x[k-j]$ becomes $x[(k-j)+1]$), and we repeat the process to produce $y[k+1]$.

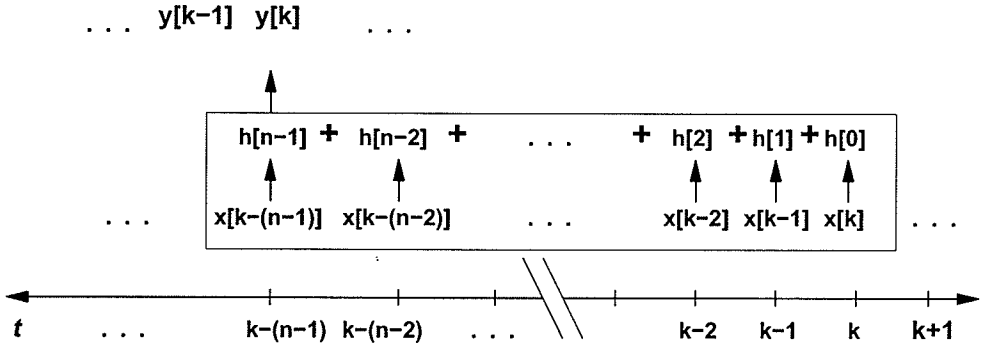

Figure 1.1: Discrete Convolution Illustrated

From this diagram, it is easy to see why people talk about filters as having delays. The output y will be delayed by a number of time steps equal to the number of filter coefficients. Each filter coefficient is sometimes known as a "tap," for reasons that should be obvious visually. It is the filter coefficients themselves that are the impulse response of the filter. In a discrete filter, if a single Dirac (unit) pulse was sent through, it would return exactly the sequence of filter coefficients.

The filter of equation 1.4.2 is known as a Finite Impulse Response (FIR) filter, because the number of coefficients which form the impulse response are finite. It is a "non-recursive" filter because y depends only on x, not on other values of y, and a "causal" filter because $y[k]$ depends on $x[j]$, with $j < k$.

It is possible to define Infinite Impulse Response (IIR) filters which have a finite number of coefficients if they are non-causal or recursive; the output depends on other values of the output. Their study is tangential to the topics we are after and their pursuit is best left to the quiet of the reader's own study.

So, when we say "filter," we mean a causal, non-recursive, FIR filter with real coefficients. These filters have the properties of Linear Time-Invariant Operators. To be thorough, the following definition is given.

Definition 1.4.3 (Linear Time-Invariant Operator). Let $f(t)$ be a function, and let L be a linear operator on f. We say that L is a "linear time-invariant operator" whenever

$$g(t) = Lf(t) \implies g(t-u) = Lf(t-u). \tag{1.4.3}$$

1.4.3 Perfect Reconstruction Filter Banks

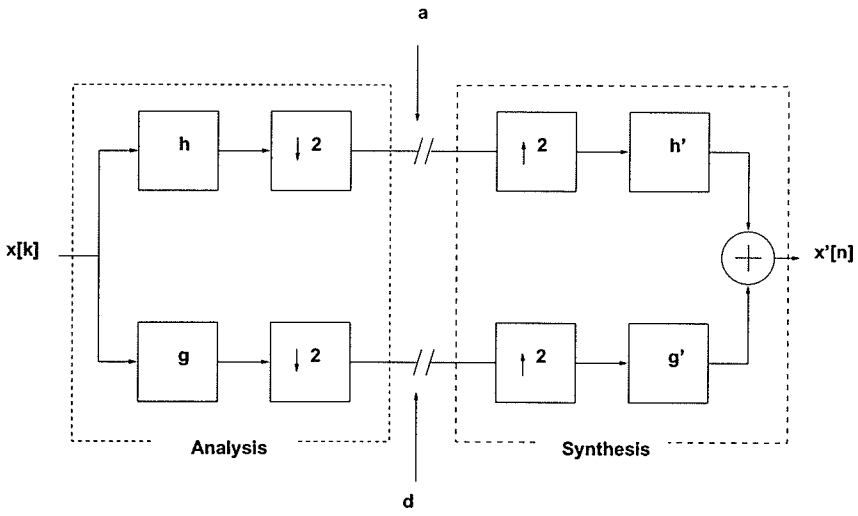

Figure 1.2: Perfect Reconstruction Filter Bank

A filter bank is just what it sounds like: a set of filters. Immediately, we are interested in the relationship between the filters and how they work together. What we wish to understand is the requirements for a filter bank that will divide the spectrum of a signal into two equal halves, and be able to put it back together again. This is

known as perfect reconstruction (PR), and it is necessary for the wavelet transforms of section 1.7.1 to have inverse transforms.

Figure 1.2 illustrates what we are trying to explain. We explain the diagram, then follow with explanations of each component. The left half, labelled "analysis," decomposes the signal. The right half, labelled "synthesis" reconstructs it.

The discrete signal $x[n]$ enters at the left, and is split. One copy of $x[n]$ goes through the box labelled h which is a low-pass filter, and is then down sampled. This results in the signal a. The other copy of $x[n]$ goes through a high-pass filter and is then down sampled, resulting in the signal d. The signals a and d may be transmitted, processed, or even run through another instance of the same filter bank.

The process is the reverse when going through the synthesis filter bank. The signal is up sampled, then filtered. When the two filtered halves are added again, they produce an output, $x'[n]$, that is identical to the input.

Low-Pass and High-Pass Filters

It is necessary, and probably sufficient, to understand the three filters in the title of this section. Before the reader protests the author's ability to count, note that the trivial filter, known as an "All-Pass" filter, is metaphorically included. The formula would simply be $y[t] = x[t]$, and all frequencies pass through unscathed.

The simplest non-trivial filter is a low-pass filter (LPF). As suggested, an LPF lets low frequency information through, and blocks high frequencies. The magnitude response ($|\hat{h}(\omega)|$) of an ideal LPF looks like a step function. Because we would like to end up with filter coefficients $h[k] \in \mathbb{R}$, the response should be symmetric about $\omega = 0$. See figure 1.3.

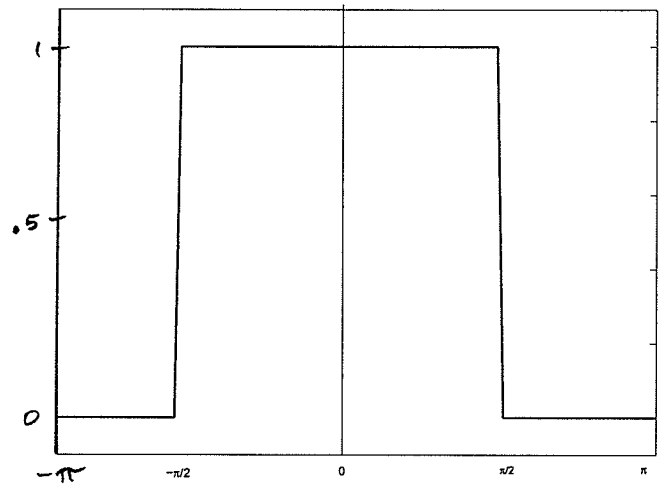

Figure 1.3: Ideal LPF Magnitude Response

Any averaging procedure is a low-pass filter; it smooths high-frequency variations in the data when viewed in the time domain. The moving average is the simplest non-trivial low-pass filter that we can consider. The coefficients would be $h[k] = \frac{1}{n}$ for each value of $k = 0, ..., n-1$. Hamming [34] has descriptions of a number of classical applications that look new again when seen from the point of view of low-pass filters, such as least-squares approximation and numerical integration [34, p.66].

The use of the word "ideal" filter is because the illustrated step function response is not something one can achieve in practice. This is due to the Gibbs Phenomenon, which sees the introduction of what is called "overshoot" and "ripple" to those in DSP, whenever we truncate a Fourier series [34]. Of interest, the use of Césaro sums (an averaging process) to improve convergence of Fourier series is really an application of an LPF to those in DSP. We reiterate their mantra: "Everything is a filter."

Another detail to observe when the Fourier series is truncated is that the rise from the frequency interval known as the "stop-band" to the interval known as the

"passband" is not actually a step. It has a slope, and the interval between the stop-band and passband is known as the transition band.

The final point to make about the LPF is the symmetry of the coefficients. If the frequency response is specified and an inverse FFT performed, the resulting coefficients will have the relationship

$$h[k] = h[N-k],\ k = 0, ..., N. \tag{1.4.4}$$

Is is assumed that N is odd. If not, padding with zeros usually removes any difficulties.

If the LPF smoothes a signal, a High-Pass filter (HPF) gets rid of the smoothness and keeps only the variations. If a LPF takes averages, an HPF keeps the differences; another way to think of an HPF is as a "moving difference." In the frequency domain, the magnitude response of an ideal HPF is also a mirrored step function, but this time inverted. See figure 1.4.

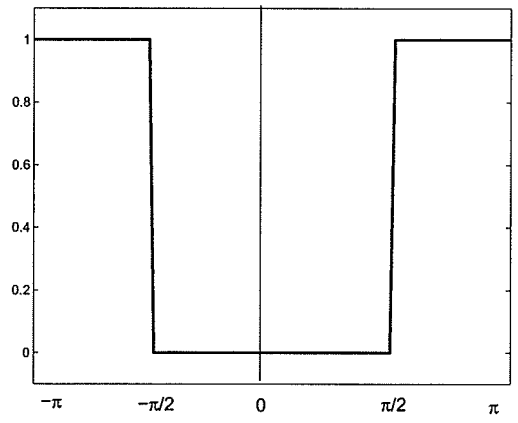

Figure 1.4: Ideal HPF Magnitude Response

In the frequency domain, the relationship between an HPF and an LPF can be

seen as a translation of the magnitude response. Let $|\hat{h}|$ be the magnitude of the LPF in figure 1.3 and $|\hat{g}|$ be the magnitude of the HPF in figure 1.4. Looking only at the region where $\omega > 0$, we have $|\hat{g}(\omega)| = |\hat{h}(\omega + \pi)|$.

Note that this holds for the region $\omega < 0$ as well. We assert that the magnitude response is periodic for ω over $\mathbb{R}$, but we are ignoring the regions outside $[-\pi, \pi]$ and assume that aliasing is properly handled as a matter of convenience. Not wishing to discuss all the details of sampling and aliasing in discrete-time systems, the reader is referred to any book on DSP such as [34], [56], or [48].

We are only interested in the construction of only one specific type of HPF, the component of a perfect reconstruction quadrature mirror filter. We continue with LP and HP filters in that context in the next section.

Quadrature Mirror Filters

One way to generate a high pass filter is to change the sign of every other coefficient of a LPF. So $h[k]$ gives us $g[k] = (-1)^k h[k]$. In the frequency domain this gives us that the magnitude response of h and g are symmetric on $[0, \pi]$ about the line $\omega = \frac{\pi}{2}$. See equation 1.4.5 and figure 1.5.

$$\left|\hat{g}\left(\frac{\pi}{2} - \omega\right)\right| = \left|\hat{h}\left(\frac{\pi}{2} + \omega\right)\right|. \tag{1.4.5}$$

This is known as a mirror image filter [2], but primarily known as a Quadrature Mirror Filter, and we will use QMF. There is some confusion in the literature; some people use QMF to mean the perfect reconstruction filter bank in its entirety, but in fact the definition above is sufficient.

The filters in the analysis portion of figure 1.2 are two portions of a QMF, as

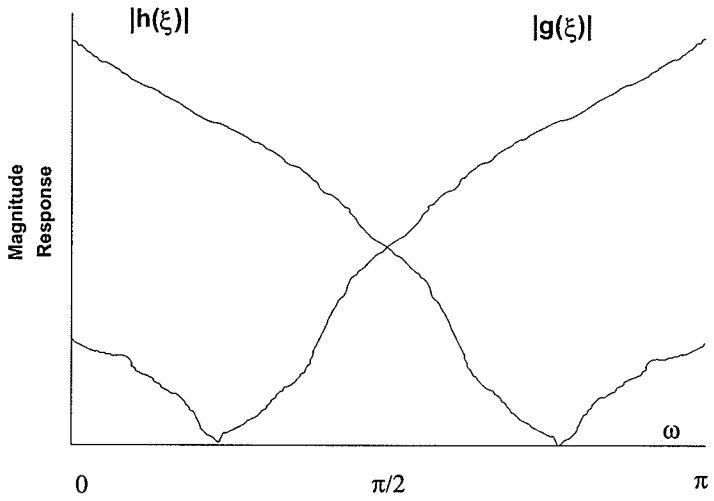

Figure 1.5: Magnitude Response of Mirror Filters

are the filters in the synthesis bank. The vertical relationship between them is now understood.

Perfect Reconstruction

In addition to the symmetry property, it will be necessary that we have what is known as "power complementary filters." This simple idea can be summed up in equation 1.4.6 and figure 1.6. The sum of the magnitude response squared of two power complementary filters must be unity at all frequencies.

$$|\hat{h}|^2 + |\hat{g}|^2 = 1. \tag{1.4.6}$$

Orthogonal Filters

We can actually do a little better in the construction of the HPF. We can have the same frequency response, but get filters that are orthogonal. Assume that the

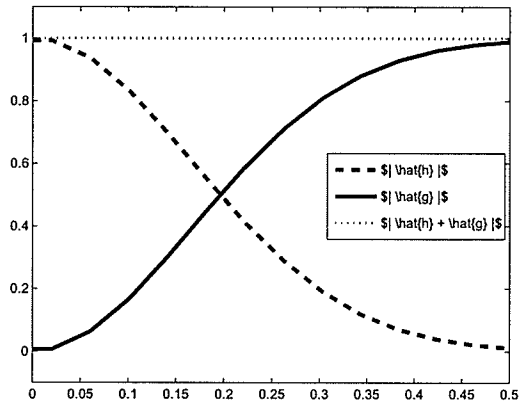

Figure 1.6: Power Complementary Magnitude Response

initial low-pass filter has N coefficients, with N odd. Because the coefficients of h are symmetric (see equations 1.4.4), we can reverse the sequence of coefficients before changing the signs to get the HPF. So now we can specify

$$g[k] = (-1)^k h[N-k]. \tag{1.4.7}$$

The filters are orthogonal in the sense that, as N-dimensional vectors, their inner product is zero. This is guaranteed by the flip of the coefficients.

Resampling

Sampling operators appear in a PR-QMF because telecommunications applications are limited in the amount of information they can transfer down a wire, or channel. The simplest application is to take one high-bandwidth signal and split it into two smaller bandwidth signals, possibly then being able to send the information on two channels. The fact is that they were developed for one reason, but borrowed for our

purposes later. It is a given that we need them.

The question arises – how many samples are necessary to represent a signal? The answer is easiest to understand in the frequency domain: the highest frequency that can be represented accurately is $\frac{1}{2}$ the sampling rate. Any frequencies that are higher than this are distorted and appear as their reflection about the sampling frequency π. The canonical example is when the wheels of a car appear to move backwards, because the camera's sampling rate of 24 frames per second is not fast enough. For a more complete discussion of aliasing, see [34, 56, 48].

It was Claude Shannon who brought forth the sampling theorem, introduced the word "bit" (for "binary digit", which he coined with Tukey), and provided a solid foundation for the wave of development in twentieth-century communications, all in a single paper in 1948 [60].

We digress for just a moment. Shannon worked not only on communications, but also mechanical computers, which were another application of Chebyshev's linkages. He was one of the most vivacious characters in mathematics, known to be quite studious and productive, yet often while riding his unicycle down the halls of MIT [54]. He was an accomplished juggler, built a mechanical robot that could juggle, and wrote the first paper on the mathematics of juggling [60]. But it is the use of downsampling and upsampling in the PR filter bank, thus the sampling theorem, that will tie us into the multiresolution of section 1.7.

After operating on the signal with the analysis filters, we downsample. This means that we throw away every second sample. Strang [61] devotes much attention to the details of performing this operation by shifting columns filters in a matrix. We omit the details here.

We want ideal half-band filters, but we will not be able to get them. Yet the need for power complementarity means that we cannot reduce the cutoff point below $\frac{\pi}{2}$. So when we downsample, there will be aliasing. We have seen that the output of a LPF has a support outside $[0, \frac{\pi}{2}]$. So the frequencies above $\frac{\pi}{2}$ are now above the new sampling rate, which is also $\frac{\pi}{2}$. Therefore, they are folded back into the new signal a as aliasing. When the signal is up sampled, the aliased components are unfolded. Then the synthesis low-pass filter makes sure the components that overlap the aliasing from the HPF cancel out.

Perfect Reconstruction Theorem

All of this discussion is summed up succinctly in a theorem of Vetterli [74, see also [49]].

Theorem 1.4.4. *Let h be an analysis LPF with down sampling by 2 and let g be an analysis HPF with down sampling by 2. Let h' be a corresponding synthesis LPF preceded by up sampling by 2, and let g' be a corresponding synthesis HPF preceded by up sampling by 2.*

These components form a filter bank F, which is an operator, such that $y[k] = Fx[k]$. Up to delay, $y[k] = x[k]$ if and only if two conditions hold:

$$\hat{\bar{h}}(\omega + \pi)\hat{h'}(\omega) + \hat{\bar{g}}(\omega + \pi)\hat{\bar{g}}(\omega) = 0; \tag{1.4.8}$$

$$\hat{\bar{h}}(\omega)\hat{h'}(\omega) + \hat{\bar{g}}(\omega)\hat{g'}(\omega) = 2. \tag{1.4.9}$$

We recognize equation 1.4.9 as the scaled condition of power complementation, and equation 1.4.8 as the anti-aliasing condition.

1.4.4 Filter References

Digital filters are an area with well developed results which apply many of the results in approximation theory. There are very interesting mathematics involved in digital filters, but they are beyond the scope of this work. For further information on digital filters, please consult [61, 71, 2], and any text on complex analysis, such as [52]. For an elementary introduction, see Lyons [48], a classic is [56], but perhaps the best reference to have when stranded on an island is Hamming [34].

1.5 Time-Frequency Analysis

While the Fourier transform is a powerful tool, it does not fill all our analytic needs. If we take only one frequency – a Dirac impulse in the frequency domain, clearly the time-domain representation has infinite support, being a sine or cosine wave of a single frequency. Likewise, if we take a Dirac impulse function in the time domain, we have a function with infinite support in the frequency domain.

The Fourier series representation of a function makes a very strong assumption: that the function being considered is 2π-periodic. It does not take much imagination to see that this is not the case for many functions of interest, and even less to see this is not the case for real-world (non-stationary) signals.

1.5.1 Gabor Transform

In 1946 Denis Gabor formally introduced the concept of an integral transform that would allow information about both time and frequency aspects of a function to be considered [27]. Gabor was a physicist, originally Hungarian. It was also around this

time that Gabor invented holography; decades later the invention of the laser saw the realization of his invention, for which he won the 1971 Nobel prize in physics. His contribution of the Gabor transform, or Short Time Fourier Transform (STFT), also called the Windowed Fourier Transform (WFT) was no less important.

Intuitively, the Gabor transform works like motion pictures, by "freezing" a signal in time, and looking at the Fourier transform of the frozen section. By repeating the process for another windowed section of the function, one can have "snapshots" of the spectrum in time.

Definition 1.5.1. Let $g(t) \in L_2(\mathbb{R})$ be real-valued, with $0 \leq g(t) \leq 1$, and $\|g\|_2 = 1$. Also, $\int_{\mathbb{R}} g(t)dt \neq 0$, and $g(0) \neq 0$. For a scale $s > 0$, a frequency ξ, and a time u, we define a Time-Frequency transformation as

$$(G_g f)(s, u, \xi) := \frac{1}{\sqrt{s}} \int_{\mathbb{R}} g\left(\frac{t-u}{s}\right) e^{i\xi t} f(t)\, dt. \tag{1.5.1}$$

Conventionally, the Gabor transform is taken to mean a transform using a parameterized Gaussian, i.e. $g(t) = e^{\sigma t^2}$, while a STFT may use any type of window.

Note that there are two possible ways of implementing a Gabor transform. The first is to literally perform the time-windowing and follow with an FFT on each slice. Using the duality of the time and frequency domains, it is also possible to create a filter bank that does time-domain convolution, with appropriate delays added to keep the support of the filters consistent. Mathematically, we can observe that there is a time-domain and frequency-domain operation required here, but their composition is commutative.

The translations of the window chosen must not fail to tile (or form a covering of)

the time-frequency plane. This could happen if the support of the window is less than half the distance between centres, in either the time or frequency domain. Also, the Gabor time-frequency decompositions are not orthonormal; in the continuous case, the time windows overlap. Gabor transforms are described in [71, 49]

Window Functions

In practice, there are many more choices of window functions available for consideration. Choosing windows to tile the time-frequency plane is not so difficult, but the shape of the window will have some effect on the spectrum. Again, it is time for the DSP mantra: "everything is a filter." Convolution with a time-domain window is to multiply the frequency response of the signal with that of the window function.

There are numerous choices of window functions in practice: Blackman, Hamming, Hanning, Chebyshev, Shannon, Harris, and more. For a taxonomy of windows with their corresponding spectral characteristics, please refer to Harris [35].

1.5.2 Heisenberg Uncertainty Principle

Gabor's efforts to develop time-frequency analysis bore some connection to quantum mechanics. The parallel is that in quantum mechanics it is impossible to know precisely both the position and momentum of a particle. Similarly, it is impossible to have completely precise information about a function in both the time and frequency domains at the same time.

For a function consisting of a single frequency in the Fourier domain, the time domain representation would be a sine wave of infinite duration. Likewise, the Fourier spectrum of a Dirac impulse, i.e., $f(t) = 0$ except for $f(0) = \infty$ with $\int_{\mathbb{R}} f(t)dt = 1$,

has equal energy at all frequencies. Precisely, $\hat{f}(\xi) = 1, \forall \xi \in \mathbb{R}$. The limit of the size of the support of a function in one domain is inversely proportional to the limit of the size of the support in the other domain.

To fully understand the last statement and the application of the Heisenberg Uncertainty Principle, we borrow only a little from statistics, after Mallat [49].

We take $\frac{|f(t)|^2}{\|f\|^2}$ to be the probability density that a signal takes a particular value at a time t. The average value of the function is then given by

$$u = \frac{1}{\|f\|^2} \int_{-\infty}^{\infty} t|f(t)|^2 \, dt. \tag{1.5.2}$$

The probability density in the frequency domain that the function $\hat{f}(\xi)$ has a certain amplitude at a frequency ξ is $\frac{|\hat{f}(\omega)|^2}{2\pi\|f\|^2}$, and the average frequency amplitude is similarly

$$\xi = \frac{1}{2\pi\|f\|^2} \int_{-\infty}^{\infty} \omega|\hat{f}(\omega)|^2 \, d\omega. \tag{1.5.3}$$

Then, noting that the statistical variance σ should not be confused with the error of m-term approximation σ_m, we have that the variances from these average values are

$$\begin{aligned} \sigma_t^2 &= \frac{1}{\|f\|^2} \int_{-\infty}^{\infty} (t-u)^2|f(t)|^2 \, dt \\ \sigma_\omega^2 &= \frac{1}{2\pi\|f\|^2} \int_{-\infty}^{\infty} (\omega-\xi)^2|\hat{f}(\omega)|^2 \, d\omega. \end{aligned}$$

With that machinery out of the way, we can state

Theorem 1.5.2 (Heisenberg Uncertainty Principle).

$$\sigma_t^2 \sigma_\omega^2 \geq \frac{1}{4} \tag{1.5.4}$$

with equality if and only if there exist $(u, \xi, a, b) \in \mathbb{R}^2 \times \mathbb{C}^2$ *such that*

$$f(t) = ae^{i\xi t} e^{-b(t-u)^2}. \tag{1.5.5}$$

The idea is that we cannot reduce the size of a time-frequency atom to a point in both time and frequency. Thus the time-frequency plane is divided into a "tiling", but the minimum area of a tile has a lower bound. The bound is exact only when the window function is the Gaussian of the Gabor transform. This is sometimes stated equivalently as $\sigma_t \sigma_\omega \geq \frac{1}{2}$.

1.6 An Historical Introduction to Wavelets

A wavelet is a "little wave." The derivation of the word is regarded in the wavelet community as a translation of the French term "ondelette," from the literature in geophysics of the 1980's. To be accurate, the English word "wavelet" appears earlier; it is found in print in a 1959 Kurt Vonnegut novel [75].

This author would have preferred the term "functino," due to the frequency with which that typographic error occurs, and because a wavelet can be thought of as an "elementary particle" of a function. When people say wavelet, the idea in mind is to decompose a function into smaller pieces. This is the important idea; as Shakespeare wrote, "What's in a name?"

We begin with Mallat's primitive, general definition of a wavelet as a starting point [49]. It is true, but not complete, and we will soon amend it.

A wavelet ψ is a function where

$$\int_{-\infty}^{\infty} \psi(t)dt = 0 \tag{1.6.1}$$

and that is translated by a parameter u and scaled (we say "dilated") by a parameter s such that

$$\psi_{u,s}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right). \tag{1.6.2}$$

So far, we only have a function that is translated and dilated. This is mathematically useless. It tells us very little about the elementary particles themselves, or their properties as a system. In fact, the term wavelet does not usually refer to what we have defined; it refers to the properties of systems of decomposition and (possibly) reconstruction. To study wavelets analysis is to study Fourier analysis in a more general sense.

Mallat immediately defines a wavelet transform, which we will continue to use.

Definition 1.6.1 (Wavelet Transform). For a function f, a wavelet ψ, and parameters s and u, we define a wavelet transform to be

$$(W_\psi f)(u,s) := \frac{1}{\sqrt{s}}\int_{-\infty}^{+\infty} f(t)\overline{\psi\left(\frac{t-u}{s}\right)}\,dt. \tag{1.6.3}$$

Retelling the story of wavelets is the canonical (and expected) way of explaining wavelet analysis. It is an interesting story, and one that has been told not just within the ivory tower, but also in public. There have been articles in the semi-popular

press, such as Strang's article in *American Scientist* (included in [61]), and even a book telling the story for a general audience [5], with accounts and interviews by many of the contemporary principal researchers. It is also an excellent way to cut a large swath through the field of mathematics.

This iteration of the story of wavelets is an amalgamation of various other versions of the story. Yves Meyer, Stéphane Jaffard, and Robert Ryan provided the main framework [38, chap. 2], but pieces and other viewpoints have been added from various other sources.

The search for wavelets begins as a way of mending Fourier analysis, picks up the notions of time-frequency analysis, and ends as being a search for orthonormal bases with good properties of smoothness, good localization in both domains, and fast algorithms. The results are not just a system of atomic decomposition of functions, but teach us the limitations of decompositions, and give us the tools to construct such systems.

1.6.1 Before Fourier

The mathematical history of wavelets dates back to Euler and Daniel Bernoulli in the 1750s, when they nearly solved the wave equation using infinite series of sine functions [62]. Bernoulli, however, did not have a formula for expressing the coefficients of the series, and while Euler did, he failed to connect it to Bernoulli's solution. Bernoulli was correct, but Euler derided him for his intuitive approach. With no defence, the lack of communication, due partly to stubbornness, caused a stalemate.

Fourier's investigations into heat allowed him to develop his theory of representing 2π-periodic functions, complete with formulæ for computing coefficients, in 1807.

However, this theory was for L_1 and there were certain formalities that were not considered; certainly the concepts of function spaces were not rigorously defined. Unfortunately, in 1873 DuBois-Raymond introduced a counter-example to the Fourier theorem, by finding a continuous, 2π-periodic function for which the Fourier series diverged at a point [44].

1.6.2 Covering the bases

There were three possible approaches to salvaging the beautiful and practical system of Fourier. Lebesgue took the approach of modifying the notion of a function. Another approach was to modify the notion of convergence, such as employing mean-square convergence like Césaro summability, due to Fejér [38].[1]

The other approach which leads to the development of wavelets was taken by Haar in 1910, where he chose to find another orthonormal basis in which to represent a function. His basis functions were not called wavelets until recently; now we use Haar basis and Haar wavelets interchangeably.

The Haar wavelet is given by

$$\psi(t) = \begin{cases} 0 & t \notin [0,1] \\ 1, & 0 \leq t < \frac{1}{2} \\ -1, & \frac{1}{2} \leq t < 1 \end{cases} \tag{1.6.4}$$

which is then dilated (scaled) and translated. That the Haar basis functions are orthonormal is elementary to prove, and the fact that they form a basis for the set of continuous functions on $[0,1]$ reduces to the proof of the Lebesgue integral using

[1]The strength of the Hungarian school of mathematics is often admired. Note that Von Neumann, Erdős, Egerváry, M. Riesz, Szegö, and Pólya, among others, shared a doctoral advisor: Lipót Fejér.

piecewise constants. The chief advantage of the Haar basis is that it is well-localized in time.

The Haar basis has another advantage over the Fourier basis (or even the windowed Fourier basis,) which is that it is an unconditional basis for L_p, when $1 < p < \infty$ [12]. We define an unconditional basis.

Definition 1.6.2 (Unconditional Basis). Let $B = \{b_k\}$ be a basis for X, such that every convergent series $\sum c_k b_k$ is unconditionally convergent. Then B is an unconditional basis.

Stated another way: an unconditional basis converges regardless of the order in which the terms are summed, for which absolute convergence is a necessary and sufficient condition, so the sign of the coefficients is not important for convergence.

From 1910 to 1920, the work of Faber and Schauder concentrated on the fact that the Haar basis is not of the same class as the function it approximates, i.e., one is no longer approximating a continuous function by other continuous functions. Faber and Schauder constructed a basis over dyadic intervals and their translations, using "triangle" or "hat" functions, which are at least continuous.

$$\Delta(t) = \begin{cases} 0 & t \notin [0,1] \\ 2t, & 0 \leq t \leq \frac{1}{2} \\ 2(1-t) & \frac{1}{2} \leq t \leq 1 \end{cases} \tag{1.6.5}$$

Independently, the two of them considered sequences of hat functions $\Delta_n(t) = \Delta(2^j t - k)$ for $n = 2^j + k$ over dyadic intervals (see section 1.1). With the addition

of $\Delta_{-1} = 1$ and $\Delta_0 = t$, a continuous function on the unit interval can be written

$$f(t) = a + bt + \sum_{n=1}^{\infty} \alpha_n \Delta_n(t), \tag{1.6.6}$$

with uniform convergence and the coefficients α_n are unique. The last two properties define a Schauder basis.

1.6.3 The Franklin system

The work of Faber and Schauder was advanced by Franklin at MIT in the 1930s, when he had the idea to create an orthonormal basis from the Schauder basis. To accomplish this, he applied the Gram-Schmidt algorithm to the sequence $\Delta_n(t)$ of hat functions, which is then called the Franklin system.

The Franklin system has an advantage over the Schauder basis, in that it spans $L_2([0,1])$. As well, it has an advantage over the Haar basis, as it characterizes the spaces C^α for $0 < \alpha < 1$ with the following relation, where f_n is one of the orthonormalized Δ_n:

$$|\langle f, f_n \rangle| \leq C n^{-\frac{1}{2}-\alpha}. \tag{1.6.7}$$

But, it is the Franklin system that developed a feature common to modern wavelets. The functions of the Franklin system cannot be given by simple dilations and translations of a fixed function ψ. The function ψ had to be computed, which at the time meant that the system was impractical. And so the Franklin system gathered dust.

1.6.4 Norms and Locality

Meanwhile, an important piece of the wavelet (and time-frequency) puzzle was introduced in 1930 by Littlewood and Paley, which anticipated not only wavelet results but some of the results of the STFT 1.5.1. In essence, the Littlewood-Paley results show that we can know something about the p-norm of the function, if we look at the coefficients in dyadic blocks in the frequency domain.

Definition 1.6.3 (Dyadic blocks). Let a_k and b_k be the coefficients of the Fourier series of $f(t)$. Then we define a Dyadic Block by

$$\Delta_j f(t) := \sum_{2^j \le k < 2^{j+1}} (a_k \cos kt + b_k \sin kt). \tag{1.6.8}$$

Of note is that since the doubling in frequency is an octave, the Δ_j provide equivalent analysis to time-domain filtering in octave bands.The main result in this area follows, see [38]:

Theorem 1.6.4 (Littlewood-Paley). *Let $f \in L_p(\mathbb{T})$. For $1 < p < \infty$, there exist two constants $0 < c_p \le C_p$ such that*

$$c_p \|f\|_p \le \left\| \left(|a_0|^2 + \sum_{j=0}^{\infty} |\Delta_j f(t)|^2 \right)^{\frac{1}{2}} \right\|_p \le C_p \|f\|_p. \tag{1.6.9}$$

Thus the Littlewood-Paley basis has good frequency localization.

1.6.5 Calderón, Grossman, and Morlet

Working in the 1960s on ideas of atomic decomposition that were predicated by work by Lusin in the 1930s, Calderón showed that convolution operators with certain

functions could be inverted by a decomposition of the identity operator [38]. Let ψ be a function in $L_2(\mathbb{R}^n)$, such that for almost all $\xi \in \mathbb{R}^n$ the Fourier transform satisfies

$$\int_0^\infty \left|\hat{\psi}(t\xi)\right|^2 \frac{dt}{t} = 1. \tag{1.6.10}$$

Write $\tilde{\psi}(x) = \overline{\psi}(-x)$, $\psi_t(x) = t^{-n}\psi\left(\frac{x}{t}\right)$, and $\tilde{\psi}_t = t^{-n}\tilde{\psi}\left(\frac{x}{t}\right)$ (we can take the dimension n to be 1). Let the operator Q_t be convolution with ψ_t and let the operator $Q_t^\star(f)$ be convolution with $\tilde{\psi}_t$.

$$I = \int_0^\infty Q_t Q_t^\star \frac{dt}{t} \tag{1.6.11}$$

Then for all $f \in L_2(\mathbb{R})$,

$$f = \int_0^\infty Q_t[Q_t^\star(f)]\frac{dt}{t}.$$

Calderón's result gives conditions on a function ψ that can be used for both analysis and synthesis, under a convolution operator. The similarities to the Fourier transform should be clear, and the generalized identity is used as a foundation of wavelet analysis.

Grossman and Morlet, a physicist and geophysicist respectively, defined wavelets in the primitive sense, as at the beginning of section 1.6. They rediscovered the Calderón identities in 1980 and used them to define the inverse wavelet transform, or synthesis formula:

$$f(t) = \int_0^\infty \frac{1}{\sqrt{s}} \int_{\mathbb{R}} (W_\psi(f))(u,s)\psi_{u,s}(t)\, du\, ds. \tag{1.6.12}$$

However, they still had to trade off orthogonality for smooth wavelets and vice versa.

1.6.6 Strömberg Wavelets

The first orthonormal wavelets with better smoothness than the discontinuous Haar wavelets were constructed in 1982 by Strömberg [12], who got them by dusting off the Franklin system, and combining them with some results of Ciesielski, which incorporated the dyadic blocks of Littlewood and Paley. Ciesielski showed that the orthonormal Franklin functions f_n satisfied the following relationship with a Lipshitz function ψ that had exponential decay of it's Fourier coefficients as ξ increases.

$$f_n(t) = 2^{\frac{j}{2}}\psi(2^{\frac{j}{2}}t - k) \tag{1.6.13}$$

where j is now a scaling parameter and k a translation.

It was Strömberg who discovered the explicit expression for the ψ function of the Franklin system. It has three properties [38]:

1. The function ψ is continuous over $\mathbb{R}$, and it is linear on unit intervals for $t > 1$, and on half-unit intervals for $t < 1$:

$$\cdots, \left[\frac{-l+1}{2}, \frac{l}{2}\right], \cdots, \left[\frac{-1}{2}, 0\right], \left[0, \frac{1}{2}\right], \left[\frac{1}{2}, 1\right], [1,2], [2,3], [3,4], \cdots, [l, l+1], \cdots \tag{1.6.14}$$

2. $|\psi(t)| \leq C\left(2 - \sqrt{3}\right)^{|t|}$

3. For $j, k \in \mathbb{Z}$, the set of functions $2^{\frac{j}{2}}\psi(2^j t - k)$ is an orthonormal basis for $L_2(\mathbb{R})$.

So Strömberg had the first construction of an orthonormal wavelet basis that had good localization in both time and frequency. These wavelets are in C^k, for $k < \infty$,

so they are more regular than the Haar wavelets, yet enjoy the other advantages. His wavelets lived in relative obscurity, however. We will adopt property (3) as the common definition of wavelets below.

1.6.7 Wavelets Proliferate

The years between 1982 and 1986 saw the development of a number of wavelet families, each with some combination of the properties of orthonormality, regularity, ease of computation, ease of transformation, compact support, decay of frequency domain coefficients, and so forth. In many ways, choosing or designing a wavelet was like choosing a cellular telephone plan; it was possible to get any desired properties, but never at the same time.

Progress however, was made. Yves Meyer had become interested in wavelets through collaboration with Grossman [5], but was unaware of Strömbergs work. He found a family of orthogonal wavelets while trying to prove by contradiction that such wavelets couldn't exist [12, 5]. The Meyer wavelets lack the property of compact support, but they beat out Strömberg in that they were infinitely regular, i.e. they belong to C^∞. Tchamitchian developed the first example of what are known as bi-orthogonal wavelets [12], in which case the wavelet used for reconstruction is different from that used for analysis, which is necessary if symmetry of the wavelet is desired.

At this point, our interest in the details of all the various varieties of wavelets begins to wane. A concise and utilitarian summary of the properties of various families of wavelets can be found in the software MATLAB [53]. See [49] and [12], the references in section 1.7.3, and Appendix B.

1.6.8 Enter Signal Processing

Wavelets up until 1986 were interesting theoretically, but were not that practical. Without a fast algorithm to compute them, they were like Fourier transforms before 1965, when digital computers and the FFT made their calculation possible. This changed in 1986, when Stéphane Mallat observed the striking similarities between Quadrature Mirror Filters (section 1.4.3), the Laplacian pyramid algorithm of Burt and Adelson in image processing [76], and the orthonormal wavelet bases of Strömberg and Meyer [12].

Mallat and Meyer "spent three days holed up in a borrowed office" [5] in Chicago, and Mallat explained the connections between the mathematical ideas such as dyadic scaling and orthogonality were the same things being done in other fields under different names. The result was a paper that unified the branches of orthonormal wavelet theory coming from mathematics, physics, and engineering [51]. It has been acknowledged that while a joint work, Meyer generously insisted that the paper be published under only Mallat's name [5].

The notions of multiresolution analysis are sufficiently important that we postpone the details of wavelets, scaling functions, and the cascade algorithm. These will be found in section 1.7.

Informally, the connections Mallat found are easier to understand than to synthesize, once one has digested enough background. The wavelet is viewed as being the impulse response of one half of a PR-QMF. The roles of the high and low pass filters correspond respectively to the wavelet function and the scaling function, to be discussed in section 1.7.

The down sampling of a quadrature mirror filter results in an output that is

equivalent to the dyadic dilation between scales. The convolution of the filter with the signal is the realization of the inner product and the translation of the wavelet. If the filters are chosen correctly, they correspond to a multiresolution analysis.

1.6.9 Daubechies Wavelets

It was Ingrid Daubechies who finally completed the quest for control over orthonormal bases, through digital filters. Since Mallat had brought digital filters into the picture, Daubechies observed that the mathematics of filter design could be brought to bear on the problem of designing wavelets.

By designing appropriate filters, Daubechies [12] was able to develop wavelets, compactly supported in both the time and frequency domains, of arbitrary regularity (smoothness) and with control over the number of vanishing moments. The filters that Daubechies used were, of course, PR-QMF filters. The orthogonality and compact support were achieved by paying careful attention to the smoothness of the magnitude response of the filter.

The Daubechies filters are special cases of what are known as Bernstein QMF filters [2]. These filters use Bernstein polynomials, which derive from classic approximation results [21], to provide a magnitude response that is highly controlled, thus giving the filters the desired properties.

There is no free lunch – they still cannot beat the Heisenberg Uncertainty Principle – but they achieve the long sought for balance between orthonormality and locality. The wavelets have no closed form expression, but they can be calculated quickly using Mallat's "cascade algorithm," which we give in table 1.3.

The Daubechies wavelets are usually named $D6$, D_6, or something similar, with

the number indicating the number of taps or coefficients, and thus the regularity of the wavelet. Coefficients can be found in any of the sources on wavelets, and are often included in various software packages. We will not discuss them in any more detail here, but please see section 1.7.3 and appendix B.

The original Haar wavelets, sometimes called D_1, are considered the simplest case of the Daubechies wavelets. It is unlikely that Haar would have foreseen the generalization, and impossible that anyone could have synthesized them without going through the mathematics of numerical digital filters. The theory would not have been developed without impetus from concrete applications. Wavelets have again saved mathematics from the sterility feared by Chebyshev, and show why Joe Keller remarked that "pure mathematics is a branch of applied mathematics [45]."

1.7 Wavelets and Multiresolution Analysis

We now turn our attention to the particular details needed to make use of wavelets. We will understand the wavelets, the multiresolution analysis that determines orthonormal wavelets, and the scaling function that determines the multiresolution analysis.

The multiresolution analysis (MRA) is the key to understanding wavelets. It provides a means for dividing a space of functions (starting with L_2) into a sequence of subspaces of successive approximation. We need two small things first:

Definition 1.7.1 (Frame). Let H be a Hilbert space of functions, and let $\Phi = \{\phi_n\}$ be a collection of functions in H. Let A and B be two constants, $0 < A \leq B < \infty$

such that for every $f \in H$, we have

$$A\|f\|^2 \leq \sum_{n=-\infty}^{\infty} |\langle f, \phi_n \rangle|^2 \leq B\|f\|^2. \tag{1.7.1}$$

Then Φ is called a "frame", and A and B are called frame bounds. If $A = B$, then Φ is called a "tight frame".

It is noted in [76] that if Φ is an orthonormal basis, it is a tight frame (with $A = B = 1$,) but that the converse is not necessarily true of tight frames. Now we can state the definition of a Riesz basis:

Definition 1.7.2 (Riesz basis). Let H be a Hilbert space of functions and let Φ be a frame for H. If the functions ϕ_n are linearly independent, then Φ is called a "Riesz basis."

The following definition of an MRA is culled from [49], [61], and [76], each of which have their own typographic errors, outdated conventions, or omissions.

Definition 1.7.3 (MRA). Let $\{V_j\}$ be a sequence of closed subspaces in $L_2(\mathbb{R})$, with $j \in \mathbb{Z}$, such that the following six properties hold:

1. For each $j, k \in \mathbb{Z}$, we have

$$f(t) \in V_j \iff f(t-k) \in V_j; \tag{1.7.2}$$

2. For each $j \in \mathbb{Z}$, we have

$$V_{j+1} \subset V_j; \tag{1.7.3}$$

3. For each $j \in \mathbb{Z}$, we have

$$f(2t) \in V_j \iff f(t) \in V_j; \tag{1.7.4}$$

4.

$$\lim_{j\to\infty} V_j = \bigcap_{j=-\infty}^{+\infty} V_j = \{0\}; \tag{1.7.5}$$

5.

$$\lim_{j\to-\infty} V_j = \text{Closure}\left(\bigcup_{j=-\infty}^{+\infty} V_j\right) = L_2(\mathbb{R}); \tag{1.7.6}$$

6. There exists a function ϕ such that the translations $\{\phi(t-n)\}_{n\in\mathbb{Z}}$ are a Riesz basis for V_0.

Of minor interest is that between his paper [51] and book [49], Mallat changed to what is termed in [76] to be the "Daubechies convention" of notation, where the containment of the spaces is $V_j \subset V_{j-1}$. This was reversed in the original paper. It has no effect on the analysis, but is a convention that agrees more with the practice of using sampled signals, where the finest scale is determined by the sampling period.

Scaling Function

The scaling function is the function ϕ in the MRA. It corresponds to the low-pass filter in the QMF. Equation 1.7.3, equation 1.7.4, and the function ϕ lead to the following functional equation for the scaling function. It is also known as a two-scale equation, or simply the scaling equation.

$$\phi(t) = \sqrt{2}\sum_{k\in\mathbb{Z}} h(k)\phi(2t-k) =: H\phi(t). \tag{1.7.7}$$

In [76], $\{h(k)\}$ is a square-summable sequence, and H is a linear operator. In [61], the $h(k)$ are revealed to be the coefficients of our LPF. It is the scaling function, determined by the filter coefficients, that determines the multiresolution analysis.

The purpose of the scaling function is to cap off the decomposition. In the same way that a series expansion can be truncated leaving a remainder, a wavelet decomposition uses the scaling function to clean up everything that is left over.

Wavelet Functions

Despite the talk of scaling functions and MRAs, we have not lost track of our wavelets. If the scaling function corresponds to the LPF, then it is not difficult to see that the wavelet ψ should have a correspondence to the HPF. This is indeed the case.

In the context of an MRA, we know that the subspace V_j is included in V_{j-1}. If we are decomposing a function into the subspaces of the MRA, then V_{j-1} includes all the information about the function f, which we get by projecting it on V_j, plus some other information. This other information is the projection onto complementary subspaces (not necessarily orthogonal).

The complementary subspaces W_j are called "wavelet subspaces."

Now, instead of simply satisfying a vague definition of dilation and translation, for an orthonormal basis of wavelets, we can define them this way, after [76]:

Definition 1.7.4 (Wavelets). For $j \in \mathbb{Z}$, let V_j indicate spaces forming a multiresolution, and let the spaces W_j be the orthogonal complements of the spaces V_j. Let $h(k)$ be the coefficients that determine the scaling function ϕ. Let $g(k)$ be a set of coefficients given by

$$g(k) = (-1)^k \overline{h(1-k)}. \tag{1.7.8}$$

Then the mother wavelet ψ is a function satisfying

$$\psi(t) = \sqrt{2}\sum_{k\in\mathbb{Z}} g(k)\phi(2t-k) =: G\phi(t). \tag{1.7.9}$$

Numerical Wavelets

In order to generate the Daubechies wavelets, or any other given by filter coefficients rather than explicit formulæ there is a simple method known as the "cascade algorithm." It consists of iteration of the scaling equation, until a fixed point $\phi(t)$ is found. Of course the stopping criteria may be adjusted. A different norm, or tolerance (ε) may be specified. Finally the wavelet is found by a single application of the wavelet equation, as in table 1.3.

1. Let $\phi^{(0)}(t) = \chi_{[0,1]}$. Let $h(k)$ be the coefficients of the appropriate LPF.
2. $\phi^{(i+1)}(t) = \sum_{k\in\mathbb{Z}} h(k)\phi(2t-k)$
3. Continue step 2 until $\|\phi^{(i)}(t) - \phi^{(i-1)}(t)\|_\infty \leq \varepsilon$, for the desired $\varepsilon > 0$.
4. Create the wavelet $\psi(t)$ by application of the wavelet equation 1.7.9 to the final $\phi(t)$.

Table 1.3: Cascade Algorithm

MRA Orthogonality

If the scaling function provides an orthogonal MRA, the wavelet subspaces W_j are the orthogonal complement to the subspaces V_j. Then we can write that V_{j-1} is the

direct sum of the subspaces V_j and W_j:

$$V_{j-1} = V_j \oplus W_j \tag{1.7.10}$$

and by equation 1.7.6, we have that

$$L_2(\mathbb{R}) = \bigoplus_{j\in\mathbb{Z}} W_j. \tag{1.7.11}$$

In the same way that the scaling function provided $\phi(2^{-j}t-k)$ was a Riesz basis for the subspaces V_j, now we have a Riesz basis for the subspaces W_j which is generated by a function ψ, which is given by $\psi(2^{-j}t - k)$ [76] and our wavelets have finally returned.

Combining the last idea with equation 1.7.11, we have that for $j, k \in \mathbb{Z}$, the collection of functions given by

$$\left\{2^{-\frac{j}{2}}\psi(2^{-j}t - k)\right\} \tag{1.7.12}$$

are a Riesz basis for $L_2(\mathbb{R})$. This is what is usually meant when people talk about wavelets.

We have now made a long answer to a short question. There are some nice results in [49] that look at the frequency domain interpretation of the scaling and wavelet functions, which is a natural approach, given their relationships with filters. Not wishing to make our long answer even longer, we refer the reader to the references and section 1.7.3

Haar Basis

We reiterate one particular wavelet basis, however, as it will be referred to repeatedly (section 2.3.4). Recall that this is the Haar basis, mentioned in section 1.6.2.

Definition 1.7.5 (Haar basis for L_p). Let $\mathcal{H}_p := \{H_k^p\}_{k=1}^{\infty}$ be the Haar wavelet basis on $[0,1)$, normalized in L_p on $(0,1)$, and defined as follows:

$$\phi(t) = H_1^p := 1, \text{on } (0,1); \tag{1.7.13}$$

$$\begin{aligned} \text{for} \qquad & n = 0, 1, 2, ... \\ & l = 1, 2, ..., 2^n \\ & k = 2^n + l \end{aligned}$$

$$\psi_{k,l}(t) = H_k^p := \begin{cases} 2^{\frac{n}{p}}, & t \in [(2l-2)2^{-n-1}, (2l-1)2^{-n-1}), \\ -2^{\frac{n}{p}}, & t \in [(2l-1)2^{-n-1}, 2l2^{-n-1}) \\ 0, & \text{otherwise.} \end{cases}$$

1.7.1 Wavelet Transforms

Continuous Wavelet Transform

We have seen the wavelet transform $(W_\psi f)(u,s) = \langle f(t), \overline{\psi_{u,s}(t)} \rangle$, or

$$(W_\psi f)(u,s) := \frac{1}{\sqrt{s}} \int_{-\infty}^{\infty} f(t) \overline{\psi\left(\frac{t-u}{s}\right)}\, dt. \tag{1.7.14}$$

Both parameters can take all real values, hence the "continuous" part of the transform's name. We have already seen the inverse transform, in equation 1.6.12,

which we recall here.

$$f(t) = \int_0^\infty \frac{1}{\sqrt{s}} \int_{\mathbb{R}} W_\psi(f)(u,s)\psi_{u,s}(t)\, du\, ds. \tag{1.7.15}$$

Fast Orthogonal Wavelet Transforms

It is the Fast Orthogonal Wavelet Transform (FWT) for discrete signals that makes it easiest to both use and understand wavelets. The idea is quite simple: by taking any function it can be split into two parts, and this can be done repeatedly. One part is the detail at some level, which can be thought of as high-frequency information. The second part is the approximation, which is low-frequency information. For those who are musically inclined, this can be thought of as a crossover in a speaker, with some of the frequencies being sent to the tweeter, and some sent to the woofer. The difference is that it is a recursive process.

The wavelet decomposition is the way the signal is represented as a sum of approximations and details at different levels. It is not necessary to record the approximations, however, except for the last one. Therefore, the wavelet decomposition of a function could be written as in equation 1.7.16, see figure 1.7.

$$f(t) = d_1 + d_2 + ... + d_{n-1} + a_{n-1}. \tag{1.7.16}$$

We give the theorem of Mallat of the FWT [49]. We will develop a little notation first. For versions of the theorem that use a more concise but slightly more cryptic matrix notation, please see [61, p. 188].

To avoid ambiguity, the following should be observed: "wavelet coefficients" refers to the result of performing a wavelet transform, not the coefficients of the wavelet

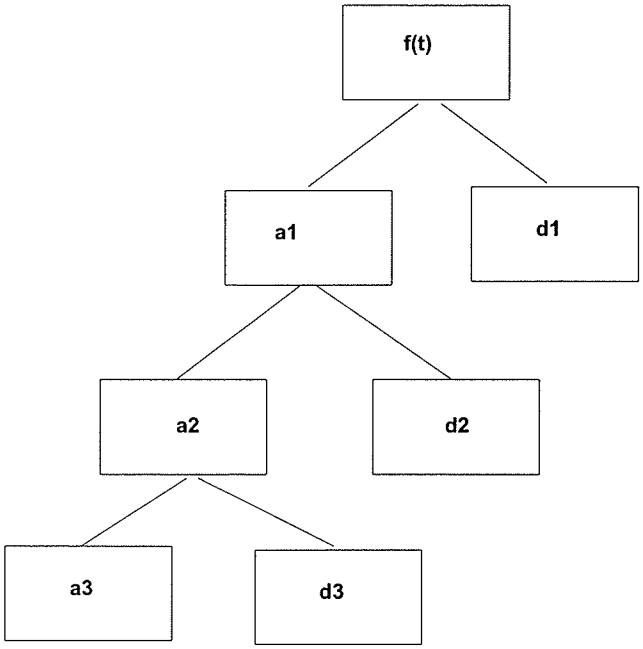

Figure 1.7: Fast Wavelet Transform Decomposition

filter $h[n]$ (or scaling function $g[n]$). The coefficients of the filters are usually referred to as "filter coefficients."

We use the notation $a_j[n]$ for the transform of a function into the scaling function basis at level j, and the notation $d_j[n]$ for the transform of a function into the wavelet basis at level j. To start things off, observe that $f = \sum_n a_0[n]\phi(t-n)$, since by definition, ϕ forms a basis for V_0.

Write $\bar{x}[n] = x[-n]$, and for the up sampling operation, we write

$$\check{x}[n] = \begin{cases} x[p], & n = 2p, \\ 0, & n = 2p+1 \end{cases} \tag{1.7.17}$$

We now give the FWT after [49].

Theorem 1.7.6 (Fast Orthogonal Wavelet Transform). *Let $f(t)$ be a real-valued*

function for $t \in \mathbb{R}$. Then suppose $f \in V_0$ for some multiresolution with a wavelet $\psi(t)$ and a scaling function $\phi(t)$. Let h be the LPF and g be the HPF for an MRA. Let $a_j[n]$ be the coefficients of the scaling function transform, and let $d_j[n]$ be the coefficients of the wavelet transform. Then the following are true:

$$a_{j+1}[p] = \sum_{n=-\infty}^{\infty} h[n-2p]a_j[n] = a_j \star \bar{h}[2p]; \tag{1.7.18}$$

$$d_{j+1}[p] = \sum_{n=-\infty}^{\infty} g[n-2p]a_j[n] = a_j \star \bar{g}[2p]; \tag{1.7.19}$$

$$a_j[p] = \sum_{n=-\infty}^{\infty} h[p-2n]a_{j+1}[n] + \sum_{n=-\infty}^{\infty} g[p-2n]d_{j+1}[n] \tag{1.7.20}$$

$$= \check{a}_{j+1} \star h[n] + \check{d}_{j+1} \star g[n]. \tag{1.7.21}$$

1.7.2 Wavelet Packets

The wavelet transform consists of iteratively splitting the target function into the approximation portion of the signal into a further approximation and detail. Once the coefficients of the detail level are recorded, no further work is done using it. In other words, the wavelet decomposition results in a binary tree with two nodes at each level.

For wavelet packets, on the other hand, both the coefficients for the approximation and the detail at each level are further split into approximation and detail portions. Therefore, a wavelet packet decomposition results in the full binary tree. From this tree, it is possible to choose several possible configurations which result in a basis for the space of functions to be approximated.

If there are any difficulties in using wavelet packets, they are in notation and keeping track of indices; strictly mechanical issues. For further description and helpful

notions of dealing with these mechanics, please see [76].

1.7.3 Wavelet References

The amount of literature on wavelets that has appeared in the last 20 years is truly staggering. For further reading, some of the central references would be [12, 49, 61, 76]. Two important collections are [4, 58]. There are any number of good books on wavelets, mathematics and DSP, but the author finds [71] useful.

The Wavelet Digest maintains an online resource `http://www.wavelet.org/`, with information on all ripples in the wavelet community – conferences, publications, jobs, materials, and software. And of course Ms. Hubbard's book [5] is as close to a "ripping yarn" as one is likely to find in mathematics.

1.8 Nonlinear Approximation

1.8.1 Basic Nonlinear Approximation

Historically, solutions to problems of approximation began by finding the first m terms of a series expansion of the target function, such as Taylor series, or Fourier series. It is difficult to avoid seeing that if we truncate the series after m terms, the error of this type of approximation will be given by the norm of the tail of the series.

But suppose that the first m terms are not the terms that have the largest coefficients. Then there would be the possibility of some m-term approximation obtained by selecting the terms "out of order." In fact, for certain functions, this is the case, and now we are considering nonlinear approximation. The sum of two m-term approximants which are not composed of the same basis functions will not be an m-term

approximant; they will have no fewer than $2m$-terms, with the exception being when the two approximants share components with the same basis, such as the same frequency in the case of a Fourier basis.

Definition 1.8.1. Let Φ be a set of functions in a space X, let $m \in \mathbb{N}$, and let Λ be any set of indices of $\phi_\lambda \in \Phi$, with $|\Lambda| \leq m$. For all Λ,

$$\Sigma_m := \Sigma_m(\Phi) := \bigcup_{\Lambda} \left\{ s \mid s = \sum_{\lambda \in \Lambda} c_\lambda \ \phi_\lambda, c_\lambda \in \mathbb{R} \text{ (or } \mathbb{C}), \phi_\lambda \in \Phi \right\} \tag{1.8.1}$$

Thus Σ_m is the set of all combinations of at most m. Analogous to the m-term linear error E_m, we have the

Definition 1.8.2.

$$\sigma_m(f, \Phi)_X := \inf_{s \in \Sigma_m} \|f - s\|_X. \tag{1.8.2}$$

For a function class $\mathcal{F}$, we have

$$\sigma_m(\mathcal{F}, \Phi)_X := \sup_{f \in \mathcal{F}} \sigma_m(f, \Phi)_X. \tag{1.8.3}$$

Two complementary papers provide a fairly comprehensive survey of nonlinear approximation to date. DeVore [19] and Temlyakov [69]. We, however, are interested in nonlinear approximation only as a preface to the highly nonlinear approximation studied in chapter 2, and to provide a point of reference to see if any advantage can be gained by moving to the highly nonlinear setting.

Nonlinear Trigonometric Approximation

DeVore and Temlyakov [15] show that it is possible to approximate a function to a given accuracy by using nonlinear approximation over trigonometric polynomials, i.e., using a Fourier transform decomposition and selecting the most significant m-terms. They also show that it is possible to have functions that are less smooth than in the linear case, and still achieve the same approximation error.

In this section, we take $\mathcal{T}_n$ to be the set of all trigonometric polynomials of degree less than or equal to n, with Λ being a set of at most n frequencies $k \in \mathbb{Z}$ chosen arbitrarily and distinctly. For a given set of indices Λ, a member $\mathcal{T} \in \mathcal{T}_n$ has the form:

$$T = \sum_{k \in \Lambda} c_k e^{ikx}$$

Then for $0 < q \leq \infty$, the class $\mathcal{A}_q(\mathcal{T}_n)$ is all trigonometric polynomials $T \in \mathcal{T}_n$ such that

$$\|T\|_{\mathcal{A}_q(\mathcal{T}_n)} := \|(\hat{T}(k))\|_{l_q} \leq 1,$$

or we can say the set of all trigonometric polynomials of degree less than or equal to n such that the l_q norm of the coefficients is less than or equal to 1.

The direct theorem comes as a corollary to a more general result, derived from approximation in finite dimensional geometries by more generalized functions than the complex exponentials.

Corollary 1.8.3 (DeVore, Temlyakov). *For each $0 < q \leq \infty$, each $n \in \mathbb{N}$, and each $1 \leq m \leq (2n+1)$, we have*

$$\sigma_m(\mathcal{A}_q(\mathcal{T}_n))_\infty \leq C m^{\frac{1}{2}-\frac{1}{q}} L\left(\frac{n}{m}\right), \quad 0 < q \leq 1 \tag{1.8.4}$$

and

$$\sigma_m(\mathcal{A}_q(\mathcal{T}_n))_\infty \leq Cn^{1-\frac{1}{q}}m^{-\frac{1}{2}}L\left(\frac{n}{m}\right), \quad 1 < q \leq \infty \tag{1.8.5}$$

with C depending only on q.

In these results, the term $L\left(\frac{n}{m}\right)$ is a technical device. It is defined in [15, p. 31] as

$$\begin{aligned} L(x) &:= (1+\ln^+ x)^{\frac{1}{2}},\ x \in \mathbb{R}_+ \\ \ln^+ x &:= \begin{cases} \ln x, & x \geq 1 \\ 0, & 0 \leq x \leq 1 \end{cases} \end{aligned}$$

though the remark is made that there exist cases where this term L can be deleted, if a suitable adjustment is made to the constant C.

There is a theorem for the lower bound. Using $\mathcal{A}_{n,q}$ to be sequences which correspond to the coefficient sequences of the trigonometric polynomials, the inequality below is established [15]. The term $\bar{\sigma}_m(\mathcal{A}_{n,q}(\mathcal{T}_n))_1$ is used to indicate that the approximation should have frequencies $|k| \leq n$ in order to build the chain of inequalities.

Theorem 1.8.4. *For each $0 < q \leq \infty$, each $1 \leq p \leq \infty$, each $n = 1, 2, ..., m = \dfrac{N-1}{2}$, and $N := (2n+1)$ we have*

$$\sigma_m(\mathcal{A}_q(\mathcal{T}_n))_p \geq \sigma_m(\mathcal{A}_q(\mathcal{T}_n)_1 \geq \bar{\sigma}_m(\mathcal{A}_{n,q}(\mathcal{T}_n))_1 \geq Cm^{\frac{1}{2}-\frac{1}{q}}. \tag{1.8.6}$$

Nonlinear Wavelet Approximation

The case of nonlinear approximation using wavelets has been examined by DeVore, Jawerth, and Popov in [20]. The results are thorough and hold in all p-norms with

$0 < p < \infty$. They also apply to the case where f is a function of d variables; we are satisfied by setting $d = 1$. The results are stronger than the case for trigonometric approximation.

What is proved is that the rate of nonlinear approximation of f by wavelets can be determined by which Besov spaces f is a member of, and likewise if we know f to be a member of a certain Besov space, we can determine the rate of approximation of f by nonlinear wavelets.

There are some conditions on the wavelets employed, and certain ranges for the parameter α. The notation used is slightly cumbersome. B^α are Besov spaces $B^\alpha := B^\alpha_\tau(L_\tau)$, where $\tau := \dfrac{1}{\alpha + \frac{1}{p}}$. So we can write:

$$\sum_{n=1}^{\infty} \left[n^\alpha \sigma_n(f)_{L_p(\mathbb{R})}\right]^\tau \frac{1}{n} < \infty \iff f \in B^\alpha. \tag{1.8.7}$$

This is done by proving the appropriate Jackson and Bernstein inequalities hold. S is the nonlinear approximant, $S = \sum_k c_k \psi_k$, or an element of all possible n term wavelet approximants, and $\beta > \alpha$ is used as a device in the proofs. So the Jackson inequality for nonlinear wavelet approximation is

$$\sigma_n(f)_p \leq C n^{-\beta} |f|_{B^\beta}, \quad f \in B^\beta \tag{1.8.8}$$

and the Bernstein inequality, if S is a nonlinear wavelet approximant, is

$$|S|_{B^\beta} \leq C n^\beta \|S\|_{L_p(\mathbb{R})}. \tag{1.8.9}$$

1.8.2 Highly Nonlinear Approximation

Once we have seen the desirability of nonlinear approximants, it doesn't take long before the charms of highly nonlinear approximation start to tempt us. There are two general approaches that have been investigated.

Best-Basis Approximation

The first approach is to "stack the deck" by first choosing the basis that will give us the best N-term approximation, and then constructing the approximation itself. Approximation results for best-basis approximation were first given by Kashin [40], are described for the L_2 case by Donoho [24], and are extended to L_p, where $p \neq 2$ by DeVore, Petrova, and Temlyakov in 2003 [18].

In order to quantify the notion of best basis, the following definition is adopted. In this definition, $\mathcal{F}_0$ signifies the unit ball of the function class $\mathcal{F}$.

Definition 1.8.5 (Best Basis). Let X be a Banach space, and let $\mathcal{B}$ be a collection of bases. Consider a function class $\mathcal{F}$ and a basis $B \in \mathcal{B}$, with $\alpha \in \mathbb{R}$, such that

$$\sigma_m(\mathcal{F}_0, B) = \mathcal{O}(n^{-\alpha}), \ n \to \infty. \tag{1.8.10}$$

If no other basis $B' \in \mathcal{B}$ satisfies

$$\sigma_m(\mathcal{F}_0, B) = \mathcal{O}(n^{-\beta}), \ n \to \infty. \tag{1.8.11}$$

for $\alpha < \beta$, then B is the "best basis" for $\mathcal{F}$.

So the best basis for a particular function class is the one with the fastest rate of

approximation. The term "best" is not used in any other sense.

Kashin [40, see also [18]], showed asymptotic results for functions in Lipα.

Theorem 1.8.6. *Let B be any orthonormal basis, and let $0 < \alpha \leq 1$. Then*

$$\sigma_m(\text{Lip}\alpha, B)_{L_2} \geq cn^{-\alpha} \tag{1.8.12}$$

with c depending only on α.

Since this is the best basis, any orthonormal basis (e.g. Fourier basis, Haar or Daubechies bases) is a best basis for Lipschitz classes – and also that a best basis is not necessarily unique.

The generalization to other normed spaces is a welcome development, but a purist might chafe a little. It is claimed that Best Basis approximation is one approach to highly nonlinear approximation, yet in order to generalize the results to other than L_2, the orthonormal bases are replaced with greedy bases (see section 2.3.4) which are developed in studying greedy algorithms.

In short, the best basis result tells how to find a best basis by stating the properties that it needs in regards to the function class being approximated. A few words of explanation are included, as we may find it convenient in discussing greedy bases in Chapter 2, but they are not necessary to observe that the result is a generalization of Kashin's result.

We will recall an unconditional basis, and also require the notion of a democratic basis:

Definition 1.8.7 (Democratic Basis). Let $B = \{b_k\}$ be a basis for a Banach space X and let Λ and Λ' be two subsets of indices of the same cardinality. Let C be an

absolute constant. When

$$\left\| \sum_{k\in\Lambda} b_k \right\|_X \leq C \left\| \sum_{k\in\Lambda'} b_k \right\|_X , \tag{1.8.13}$$

we say that B is a "democratic basis" for X.

Intuitively, and according to [43], a democratic basis is one where the norm of a sum of basis elements is determined (up to a constant) by the number of elements.

The last notion is that of alignment [18].

Definition 1.8.8. Let $B = \{b_k\}$ be a basis for X, and the $\mathcal{F} \subset X$ be a function class. Let $\left(f = \sum_k \alpha_k b_k \right) \in \mathcal{F}$ and $\left(g = \sum_k \gamma_k b_k \right) \in X$. Let $c_0 > 0$ be an absolute constant depending only on $\mathcal{F}$. The we say that B is aligned with $\mathcal{F}$ when

$$|\gamma_k| \leq |\alpha_k|, \ k = 1, 2, ... \tag{1.8.14}$$

and

$$c_0 \|g\|_{\mathcal{F}} \leq \|f\|_{\mathcal{F}}. \tag{1.8.15}$$

In other words, the coefficients of the expansion of a function in $\mathcal{F}$ are larger (up to a constant) than the coefficients of expansions in B of functions that may be in $\mathcal{F}$ or in the complement of $\mathcal{F}$.

Using a definition of best that is slightly more technical, but completely analogous to Donoho's, we can state the main result of DeVore, Petrova, and Temlyakov.

Theorem 1.8.9 (Best-Basis in L_p). *Let $\mathcal{F}$ be a function class in L_p and let $\bar{B}$ be a democratic, unconditional basis to which $\mathcal{F}$ is aligned. Then $\bar{B}$ is a best coarse order basis for $\mathcal{F}$.*

On the numerical front, Coifman and Wickerhauser [9] describe an algorithmic method for building a best-basis from a library of several orthonormal basis, specifically wavelet packets. Their work concentrates on defining the cost functional that measure which basis is the best, and develop strategies for minimizing it.

The cost functional measures which decomposition has the fewest non-zero coefficients. For this, they use Shannon entropy, which differs only in specifics from the entropy (Kolmogorov entropy) used in the approximation theoretic approach [18]. The reader interested in entropy is referred to the papers cited above, and the work of Shannon in Information Theory [60].

Approximation by Redundant Dictionaries

If we relax the requirement that the set of approximating functions form a basis for the space of approximation, we are speculating that we can find a really efficient approximation. After all, if the target function happens to be a member of the approximating set, only one coefficient will suffice to represent it.

To improve our chances at finding such a beast, the aim is to try and make the approximating set as large as possible, so that an arbitrary function in the space of approximation will never be very far from some approximant. There is no such thing as a free lunch, however, and the big payoff we strive for by increasing the size of our dictionary runs the risk of having to search through a larger pile of rubble to find a building block that fits.

A dictionary is general and straightforward: it is a set of elements that span a given space. A basis generates a dictionary, but the more general dictionary does not necessarily form a basis – it is redundant or "overcomplete." Any dictionary that is

not a basis is often referred to as a "redundant dictionary."

In practice, we put a few other restrictions on membership in a dictionary, mainly for convenience, such as normalizing elements. The usual definition follows:

Definition 1.8.10. Let X be a Banach space. Let $\mathcal{D} \subset X$ be made up of functions g such that $\|g\|_X = 1$ and $g \in \mathcal{D} \implies -g \in \mathcal{D}$.

Then we say that $\mathcal{D}$ is a "dictionary."

We are interested in the loosest pile of rubble possible – we want to know how well we can work with an arbitrary dictionary that may have little if any structure. It may contain Fourier, Gabor, wavelet, spline, Padé, Dirac, or heuristically developed functions. If we have no structure to offer us a guide, we will need some sort of algorithm to construct an approximant. The class of algorithms known as greedy algorithms has been the method most widely studied, and the next chapter contains an exposition of greedy approximation.

Chapter 2

Greedy Algorithms

"Until now, man has been up against Nature; from now on he will be up against his own nature."

—D. Gabor

One of the most satisfying aspects of approximation theory is that it is often constructive rather than existential in nature. We have many step-by-step algorithms for constructing approximants. Greedy algorithms are not particularly new; a classic algorithm that is greedy is the second algorithm of Remez [7][1], which constructs the polynomial of best-approximation to a given function.

The property that admits an algorithm to the class known as greedy algorithms is that there is a step requiring some optimal choice, and that choice is made without regard to the overall solution or output [11]. Local optimization at every step is the hallmark of a greedy algorithm.

[1]Interestingly, this algorithm is used in approximating frequency response in filter design. Published in 1934, it probably predates the use of the term greedy.

When a greedy algorithm is correct in the algorithmic sense of the word, i.e., when the algorithm produces an output possessing the desired properties for any input, all of the locally optimal choices that have been made will result in a final output that is also globally optimal for the problem at hand. Therefore, a problem that will fall to a greedy algorithm has a property called "recursive suboptimality." In the case of greedy approximation, we rely on the fact that the coefficients of an expansion can be arranged into a decreasing sequence (or at least non-increasing) to establish that it is recursively suboptimal.

2.1 Pure Greedy Algorithm

We begin with the Pure Greedy Algorithm, henceforth PGA. Both the name and acronym can be used without ambiguity only in our context of approximation. The PGA is a mind-numbingly simple scheme, provided we can retain some measure of the advertised purity. We can't, but the beauty of mathematics is that we can still find worthwhile relationships by making such otherwise ridiculous assumptions. Later variations on the PGA are attempts at finding ways to bridge the gap between pure theory and the messy real world of implementation.

Initially, we consider $f \in \mathcal{H}$, and a dictionary $\mathcal{D} \in \mathcal{H}$, where $\mathcal{H}$ is a Hilbert (inner product) space, equipped with a norm. The PGA takes the target function as the input and somehow finds the single element $g \in \mathcal{D}$ that agrees the best with the function. That is, g has the largest inner product with f of all functions from $\mathcal{D}$. Finding the best g is the greedy step; assuming we can do so is the "pure" part.

The PGA then adds g (multiplied by the appropriate coefficient) to the approximant, and subtracts it from the function leaving a remainder, completing one itera-

tion. In the next step, the remainder becomes the function to be approximated, and the algorithm continues until the error is sufficiently small or the desired number of terms has been calculated.

The output may be stopped when $\|f-G\| < \varepsilon$, or when G is a combination no more than m terms, depending on the application. When the algorithm terminates, the approximant will be the combination of the dictionary functions chosen at each step. The norm of the last remainder will be the error of that particular approximation. We are interested in rate of approximation to answer questions of convergence and error of approximation.

The PGA is known by several different names, in different areas of research. In signal processing [50], it is known as "Matching Pursuit," though it incorporates a weakness parameter (see section 2.2.) It originated in statistics, where it has been known as "Projection Pursuit Regression," and the canonical results are due to Jones [39, 1987] and Huber in 1985, after Friedman and Stuetzle in 1981. It is not impossible that the essential algorithm appears in some other field as well.

To begin our examination more precisely, we can adopt the definition given in [16]:

Definition 2.1.1 (PGA). Let g be an element from $\mathcal{D}$ that maximizes the inner product $\langle f, g\rangle$.

Define the greedy element $G(f)$ and the remainder $R(f)$ as follows:

$$\begin{aligned} G(f) := G(f,\mathcal{D}) &:= \langle f, g\rangle g, \\ R(f) := R(f,\mathcal{D}) &:= f - G(f). \end{aligned} \tag{2.1.1}$$

Let $R_0(f) := R_0(f,\mathcal{D}) := f$, and $G_0(f) := 0$. Then for each $m \geq 1$, define $G_m(f)$

and $R_m(f)$ recursively:

$$\begin{aligned} G_m(f) := G_m(f, \mathcal{D}) &:= G_{m-1}(f) + G(R_{m-1}(f)), \\ R_m(f) := R_m(f, \mathcal{D}) &:= f - G_m(f) = R(R_{m-1}(f)). \end{aligned} \tag{2.1.2}$$

Does the PGA converge? The original results of Jones [39] show that it does. Intuitively, it seems that if we are given an infinite number of terms, we should get what Jones proves:

$$\lim_{m \to \infty} \|R_m(f, \mathcal{D})\|_H = 0. \tag{2.1.3}$$

Then we can expand $f \in \mathcal{H}$ as

$$f = \sum_{m=0}^{\infty} \langle R_m(f, \mathcal{D}), g_m \rangle \, g_m. \tag{2.1.4}$$

Since $\|g_m\| = 1$, we have

$$\|f\|^2 = \sum_{m=0}^{\infty} |\langle R_m(f, \mathcal{D}), g_m \rangle|^2. \tag{2.1.5}$$

When $\mathcal{D} = B$ is an orthonormal basis for $\mathcal{H}$, since the representation of f in terms of B is unique, the greedy approximant $G_m(f, B)$ will be a best approximant,

$$\|f - G_m(f, B)\| = \sigma_m(f, B). \tag{2.1.6}$$

2.1.1 Greedy Nonlinear Trigonometric Approximation

Before moving to the case of an arbitrary dictionary, it is worthwhile to see what happens when we apply the PGA to an orthonormal basis, such as the trigonometric

polynomials, to illustrate the approximation for classes of functions.

In [15], Temlyakov and DeVore study the L_p-error of m-term approximation by trigonometric polynomials (see section 1.8.1.) They showed that nonlinear methods achieved good results. For the same rate of approximation, nonlinear methods will approximate a larger class of functions, in particular they do not have to be as smooth.

In [64], Temlyakov wanted to find an algorithm that did well with constructing nonlinear approximants. He studied application of the PGA to nonlinear m-term trigonometric approximation. By viewing the Greedy Algorithm as an operator, G_m provides the optimal error of approximation for several classes of functions.

In order to do this, he generalizes a result that is true for trigonometric polynomials to any orthonormal system.

Theorem 2.1.2. *For any orthonormal system $\Phi = \{\phi_j\}_{j=1}^{\infty}$ of uniformly bounded functions $\|\phi_j\|_\infty \leq M$, there exists a constant C depending on M such that*

$$\|f - G_m(f,\Phi)\|_p \leq C(M) m^{h(p)} \sigma_m(f,\Phi)_p, \quad 1 \leq p \leq \infty.$$

where $h(p) := \left|\frac{1}{2} - \frac{1}{p}\right|$.

So we know that for orthonormal bases, things are fairly well behaved.

Now, to look at the behaviour of greedy approximation, we recall the class of functions $\mathcal{F}_q^r$.

Definition 2.1.3. Let $0 < r < \infty$ and let $0 < q \leq \infty$. Let f be a function in L_1 on $\mathbb{T}$ (i.e., a 2π-periodic function). Define

$$|f|_{\mathcal{F}_q^r} := \|(|k|^r |\hat{f}(k)|)_{k\in\mathbb{Z}}\|_{l_q}, \tag{2.1.7}$$

and $\mathcal{F}_q^r := \{f \mid |f|_{\mathcal{F}_q^r} \leq 1\}$.

This is the class of functions whose r-th time-derivatives, expressed in the frequency domain, are in the unit ball of l_q. This is characterizing smoothness via the Fourier coefficients of the function.

DeVore and Temlyakov showed in [15] that the rate of nonlinear approximation by trigonometric polynomials for the function class $\mathcal{F}_q^r$ is

$$\sigma_m(\mathcal{F}_q^r) \asymp m^{-\lambda}, \tag{2.1.8}$$

where $\lambda := r + \frac{1}{q} - \frac{1}{2}$.

Temlyakov shows that for the function class $\mathcal{F}_q^r$, the rate of approximation for the greedy algorithm is optimal for a given class of functions, many times [64]. For $0 < q < \infty$ and $1 \leq p \leq 2$,

$$G_m(\mathcal{F}_q^r)_p \asymp \sigma_m(\mathcal{F}_q^r)_p. \tag{2.1.9}$$

Theorem 2.1.4. *For any $0 < q < \infty$ and $r > (1 - 1/q)_+$, we have*

$$\begin{aligned} G_m(\mathcal{F}_q^r)_p &\asymp m^{r-\frac{1}{q}+\frac{1}{2}}, \quad 1 \leq p \leq 2, \\ G_m(\mathcal{F}_q^r)_p &\asymp m^{r-\frac{1}{q}+1-\frac{1}{p}}, \quad 2 \leq p \leq \infty. \end{aligned} \tag{2.1.10}$$

Therefore, the algorithm is deemed good for this class.

2.1.2 PGA Using General Dictionaries

Now suppose that we want to consider the case of an arbitrary dictionary $\mathcal{D}$. Alas, there is a slight irony here, in that the development of wavelets was motivated by

a desire to find orthonormal bases. In greedy approximation where we may employ wavelets, we don't necessarily require that they be orthogonal.

There is an initial result due to Davis, Mallat, and Avellaneda [14], that in infinite dimensional Hilbert spaces, the rate of convergence can be extremely slow, but that for finite dimensional spaces, the convergence is exponential, with the rate being given as $-\frac{1}{2}\log(1-\lambda_{\min}^2)$, where

$$\lambda(f) := \sup_{g\in\mathcal{D}} \left| \left\langle \frac{f}{\|f\|}, g \right\rangle \right|. \tag{2.1.11}$$

and $\lambda_{\min} \leq \lambda(f)$, for any $f \in \mathcal{H}$.

At about the same time, DeVore and Temlyakov produced other results for the rate of convergence in [16]. The approximation classes used are defined first.

Definition 2.1.5. Let $\mathcal{D}$ be a general dictionary and let $\tau > 0$. Let $\Lambda \subset \mathcal{D}$ be of finite cardinality, and let $f \in \mathcal{H}$ be a function in the span of $\mathcal{D}$, i.e. for $w_k \in \mathcal{D}$, we can write $f = \sum_{k\in\Lambda} c_k w_k$. Then we define

$$A_\tau^0(\mathcal{D}, M) := \left\{ f \mid \sum_{k\in\Lambda} |c_k|^\tau \leq M^\tau \right\}. \tag{2.1.12}$$

Now define $A_\tau(\mathcal{D}, M)$ to be the closure of $A_\tau^0(\mathcal{D}, M)$ in $\mathcal{H}$, and let

$$A_\tau(\mathcal{D}) := \bigcup_{M>0} A_\tau(\mathcal{D}, M). \tag{2.1.13}$$

Finally, define $|f|_{A_\tau(\mathcal{D})}$ to be the smallest M such that $f \in A_\tau(\mathcal{D}, M)$.

Perhaps it would be easier, for a first reading, to restate that in order for f to

be in $\mathcal{A}_\tau(\mathcal{D})$, the coefficients to the power τ must be absolutely summable, with M being the measure of "how finite." The quantity M defines the norm of f.

So the estimate for the rate of approximation using a general dictionary is stated.

Theorem 2.1.6. *Let $\mathcal{D}$ be an arbitrary dictionary in $\mathcal{H}$. Then for each $f \in A_1(\mathcal{D})$ we have*

$$\|f - G_m(f, \mathcal{D})\| \leq |f|_{A_1(\mathcal{D})} m^{-\frac{1}{6}} \tag{2.1.14}$$

In [42], Konyagin and Temlyakov improve this estimate slightly.

$$\|f - G_m(f, \mathcal{D})\| \leq 4|f|_{A_1(\mathcal{D})} m^{-11/62}. \tag{2.1.15}$$

Choice of Dictionary

There are two drawbacks to the PGA shown by studying the rate of convergence. For one, these estimates are asymptotic, as shown in figure 2.1. Additionally, there is an example in [16] where the rate of convergence is bounded below. In order to upset the efficiency, $\pm g$ is added to an otherwise well-behaved orthonormal basis, where g is a specialized linear combination of basis elements. This result shows that it is possible, through a poor choice of dictionary, to nullify the advantages of greedy approximation.

Theorem 2.1.7. *Let $B = \{h_k\}_{k=1}^{\infty}$ be an orthonormal basis for $\mathcal{H}$ and let $\mathcal{B} = \{\pm h_k\}_{k=1}^{\infty}$ be the dictionary generated by B. Consider the function*

$$g := Ah_1 + Ah_2 + aA \sum_{k \geq 3} \frac{h_k}{\sqrt{k^2 + k}} \tag{2.1.16}$$

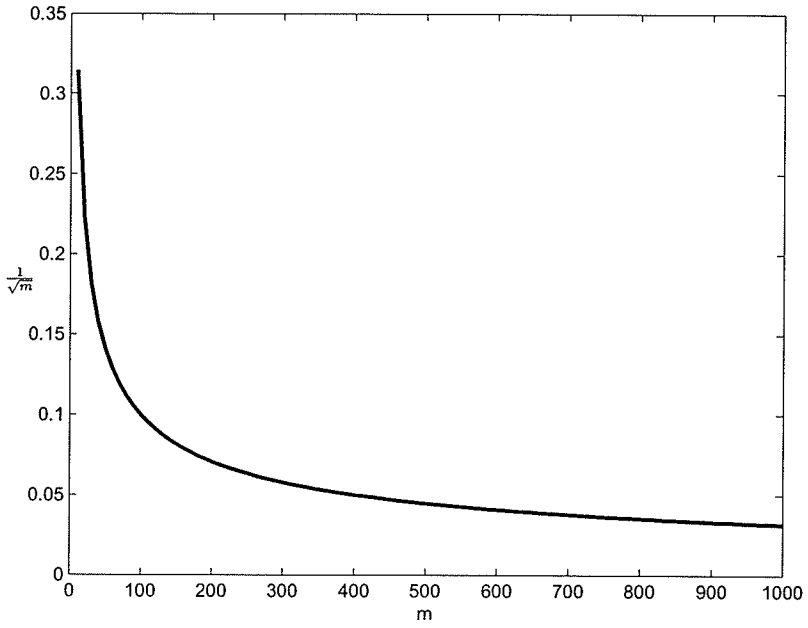

Figure 2.1: Asymptotic Convergence of PGA

with $A = \sqrt{\frac{33}{89}}$ *and* $a = \sqrt{\frac{23}{11}}$. *Now let the dictionary of greedy approximation be* $\mathcal{D} = \mathcal{B} \cup \{\pm g\}$.

The function $f = h_1 + h_2$ *is in each space* $A_\tau(\mathcal{D})$ *for* $0 < \tau \leq 2$. *For* $m \geq 4$,

$$\|f - G_m\| \geq \frac{1}{\sqrt{m}} \tag{2.1.17}$$

2.1.3 Relaxed Greedy Algorithm

The Relaxed Greedy Algorithm, RGA, is a variant of the PGA. Where G and R indicated the greedy approximant and remainder at a given step, we use G^r and R^r to indicate the relaxed greedy choice and relaxed remainder, respectively.

Definition 2.1.8. Let $g(h) \in \mathcal{D}$ be the function that maximizes $\langle h, g \rangle$. Define

$R_0^r(f) := R_0^r(f, \mathcal{D}) := f$ and $G_0^r := 0$. For $m = 1$, use the first step of the PGA, so that $G_1^r(f) := G_1(f)$, and $R_1^r(f) := R_1^f$. Then for $m \geq 2$, define

$$\begin{aligned} G_m^r(f) := G_m^r(f, \mathcal{D}) &:= \left(1 - \frac{1}{m}\right) G_{m-1}^r(f) + \frac{1}{m}\, g\left(R_m^r(f)\right), \\ R_m^r(f) := R_m^r(f, \mathcal{D}) &:= f - G_m^r(f). \end{aligned} \tag{2.1.18}$$

The RGA still attempts to find the dictionary element with the best correlation to the remainder of the function at each step. However, as stated in [65], the approximant forms a co-convex approximant, i.e., it preserves the convexity properties of the target function.

RGA Approximation

DeVore and Temlyakov give us the following upper bound for the RGA [16], and they mention that there is a similar result in L_p from research in neural networks.

Theorem 2.1.9. *Let $f \in \mathcal{A}_1(\mathcal{D}, 1)$. Then*

$$\|f - G_m^r\| \leq \frac{2}{\sqrt{m}}, \tag{2.1.19}$$

Note that in [65], Temlyakov uses the more general relaxation parameter $0 < \alpha < 1$. The RGA is not paid much attention in the literature, and this is explained by a remark: the form of the m-th approximant $G_m^r(f)$ is a co-convex approximant. This assures that inductively, $G_{m-1}^r(f)$ and $G_m^r(f)$ are in the same approximation class $\mathcal{A}_1(\mathcal{D})$, and also that there is an upper bound on the improvement in approximation

between any two iterations, δ_m. This is given to be

$$\delta_m^2 \leq \|R_{m-1}^r\|^2 - \frac{1}{4}\sup_{g\in\mathcal{D}}\left\langle R_{m-1}^r, g - G_{m-1}^r\right\rangle^2. \tag{2.1.20}$$

2.1.4 Orthogonal Greedy Algorithm

The second major variation of the PGA is the Orthogonal Greedy Algorithm, or OGA. The term "orthogonal" is applied because the remainder at each iteration is guaranteed to be orthogonal to the atoms that have been chosen thus far. This is accomplished by abandoning simple addition of a new dictionary element to the approximant, but by improving the approximant by projecting the remainder onto the subspace that is the span of the previous atoms.

Definition 2.1.10. Let $\mathcal{H}$ be a Hilbert space. Let $H_k := \{\phi_1, ..., \phi_k\}$ be an orthonormal subset of $\mathcal{H}$. Let P_{H_k}

$$P_{H_k} := \sum_{j=1}^{k}\langle f, \phi_j\rangle\phi_j \tag{2.1.21}$$

be the orthogonal projector from $\mathcal{H}$ to H_k. For $f \in \mathcal{H}$ let $g(f) \in \mathcal{D}$ be the function that maximizes $\langle f, g\rangle$.

$$\begin{aligned} H_m := \quad H_m(f) \quad &:= \text{span}\{g(R_0^o(f)), ..., g(R_{m-1}^o(f))\}, \\ G_m^o(f) := \quad G_m^o(f,\mathcal{D}) \quad &:= P_{H_m}(f), \\ R_m^o(f) := \quad R_m^o(f,\mathcal{D}) \quad &:= f - G_m^o(f). \end{aligned} \tag{2.1.22}$$

Lest the notation confuse the fact that H_m is a set of orthonormal functions, we recall that $\|g\| = 1$ by the definition of a dictionary. That it is orthogonal comes from

the fact that the procedure is really a Gram-Schmidt procedure.

OGA Approximation Results

The OGA has the benefit of providing the best approximation from H_m [16], and has better properties of convergence than the PGA/WGA, though it requires significantly more computational expense for a given number of iterations [14].

Theorem 2.1.11. *Let $\mathcal{D}$ be an arbitrary dictionary in $\mathcal{H}$. Then for each $f \in \mathcal{A}_1(\mathcal{D}, M)$ we have*

$$\|f - G_m^o(f, \mathcal{D})\| \leq |f|_{\mathcal{A}_1(\mathcal{D})} m^{-1/2}. \tag{2.1.23}$$

2.2 Weak Greedy Algorithms

Temlyakov has studied a family of greedy algorithms which relaxes the assumption that we know the optimal dictionary element at each iteration. It is possible to take functions that are not optimal, but "good enough," with the hope of rendering the computations more tractable. This is done by choosing g in the following way.

$$g_m := t_m \sup_{g \in \mathcal{D}} \langle f, g \rangle, \ 0 < t_m < 1. \tag{2.2.1}$$

At each step m, we use t_m, the m-th term of the sequence τ, as the parameter indicating the amount of suboptimality we are willing to accept. The case where the t_m are constant was studied in [39], and is included in the results of the previous section.

The notation τ is the "name" of a particular weakness sequence. It should not be confused with the parameter τ of the approximation classes $A_\tau(\mathcal{D})$. The WGA

is defined in [65], and restated below. Note the shift in notation such that $f_m^\tau := R_m^\tau(f, \mathcal{D})$.

Definition 2.2.1 (WGA). Let a sequence $\tau = \{t_k\}_{k=1}^\infty$, with $0 < t_k < 1$ be given. Define $f_0^\tau = f$. For $m \geq 1$, $\phi_m^\tau \in \mathcal{D}$ is any function such that

$$|\langle f_{m-1}^\tau, \phi_m^\tau \rangle| \geq t_m \sup_{g \in \mathcal{D}} |\langle f_{m-1}^\tau, g \rangle|. \tag{2.2.2}$$

Then we define the m-th iteration of the WGA as follows:

$$\begin{aligned} f_m^\tau &:= f_{m-1}^\tau - \langle f_m^\tau, \phi_m^\tau \rangle \phi_m^\tau, \\ G_m^\tau(f, \mathcal{D}) &:= \sum_{j=1}^m \langle f_{j-1}^\tau, \phi_j^\tau \rangle \phi_j^\tau. \end{aligned} \tag{2.2.3}$$

2.2.1 WGA Variants

For the PGA, RGA, and OGA, there are corresponding weak variants, respectively WGA (above), WRGA, and WOGA. The reader is advised that care should be paid to superscripts, which indicate the algorithm being used. The presence of τ in the superscript indicates the presence of a weakness parameter.

Definition 2.2.2 (WOGA). Let a sequence $\tau = \{t_k\}_{k=1}^\infty$, with $0 < t_k < 1$ be given. Let $f_0^{o,\tau} = f$. Let $f_1^{0,\tau}$ and $\phi_1^{o,\tau}$ be the results from the first iteration of the WGA. Then for $m \geq 2$, $\phi_m^\tau \in \mathcal{D}$ is any function such that

$$|\langle f_{m-1}^\tau, \phi_m^\tau \rangle| \geq t_m \sup_{g \in \mathcal{D}} |\langle f_{m-1}^\tau, g \rangle|. \tag{2.2.4}$$

We define the m-th iteration of the WOGA as follows:

$$\begin{aligned} H_m^{o,\tau} := \quad H_m^{o,\tau}(f) \quad &:= \text{span}\{\phi_1^{o,\tau}, ...\phi_m^{o,\tau}\}, \\ G_m^{o,\tau}(f) := \quad G_m^{o,\tau}(f,\mathcal{D}) \quad &:= P_{H_m}(f), \\ R_m^{o,\tau}(f) := \quad R_m^{o,\tau}(f,\mathcal{D}) \quad &:= f - G_m^{o,\tau}(f). \end{aligned} \tag{2.2.5}$$

For the WRGA, it is possible to specify the relaxation parameter in different ways. There are two different variants for which we have results, so to keep them straight, we'll use the superscript notation τ, i to indicate variant i of the WRGA with weakness sequence τ, and the relaxation parameter will be noted α_m^i.

Definition 2.2.3 (WRGA1 and WRGA2). Let a sequence $\tau = \{t_k\}_{k=1}^{\infty}$, with $0 < t_k < 1$ be given. Let $f_0^{\tau,1} = f$. Let $\phi_m^{\tau,1} \in \mathcal{D}$ be any function such that both of the following are true:

$$|\langle f_{m-1}^{\tau,1}, \phi_m^{\tau,1} - G_{m-1}^{\tau,1}\rangle| \geq t_m \|f_{m-1}^{\tau,1}\|^2 \tag{2.2.6}$$

$$\|\phi_m^{\tau,1} - G_{m-1}^{\tau,1}\| \leq \|f_{m-1}^{\tau,1}\|. \tag{2.2.7}$$

Let $\phi_m^{\tau,2} \in \mathcal{D}$ be any function such that the following is true:

$$|\langle f_{m-1}^{\tau,2}, \phi_m^{\tau,2} - G_{m-1}^{\tau,2}\rangle| \geq t_m \|f_{m-1}^{\tau,2}\|^2 \tag{2.2.8}$$

Let α_m^1 be defined as

$$\alpha_m^1 := \frac{\left\langle f_{m-1}^{\tau,1}, \phi_m^{\tau,1} - G_{m-1}^{\tau,1}\right\rangle}{\left\|\phi_m^{\tau,1} - G_{m-1}^{\tau,1}\right\|^2}. \tag{2.2.9}$$

Let α_m^2 be defined for $m \geq 1$ as

$$\alpha_m^2 := \frac{t_m}{(1 + \sum_{k=1}^{m} t_k^2)} \tag{2.2.10}$$

Then for $m \geq 1$, define the m-th iteration of either version of the WRGA ($i = 1$ or $i = 2$) as follows:

$$G_m^{\tau,i}(f, \mathcal{D}) := (1 - \alpha_m) G_{m-1}^{\tau,i} + \alpha_m^i \phi_m^{\tau,i}, \tag{2.2.11}$$

$$f_m^{\tau,i} := f_{m-1}^{\tau,i} - G_m^{\tau,i}, \ i = 1, 2. \tag{2.2.12}$$

2.2.2 Weakness Sequences and Convergence

Imagine that the target function is a picnic basket full of food, and that the WGA is a line of ants carrying food away from it. Each ant is one ϕ_m^τ, chosen to be able to carry away the next scrap of food, but the ants aren't very good at sorting themselves by size. Our t_k determine the size of the next available scrap. For a fast rate of convergence, we want to choose big t_k, but it's easier to get an ant if we're not too fussy about size.

Abandoning the ant analogy, we need to answer the question: how small can the t_m be and still guarantee convergence. Again, [65] has some results:

Theorem 2.2.4. *Let $\mathcal{D}$ be an arbitrary dictionary in $\mathcal{H}$. Let f be a function in $\mathcal{H}$. Let $\tau = \{t_k\}_{k=1}^{\infty}$, and let*

$$\sum_{k=1}^{\infty} \frac{t_k}{k} = \infty. \tag{2.2.13}$$

Then

$$\lim_{m \to \infty} \|f - G_m^\tau(f, \mathcal{D})\| = 0. \tag{2.2.14}$$

Also, there is a covering theorem:

Theorem 2.2.5. *Let $\mathcal{D}$ be an arbitrary dictionary in $\mathcal{H}$. Let f be a function in $\mathcal{H}$. Let $\tau = \{t_k\}_{k=1}^{\infty}$, and let*

$$\sum_{s=0}^{\infty} \left(\frac{1}{2^s} \sum_{k=2^s}^{2^{s+1}-1} t_k^2 \right)^{\frac{1}{2}} = \infty. \tag{2.2.15}$$

Then

$$\lim_{m \to \infty} \|f - G_m^{\tau}(f, \mathcal{D})\| = 0. \tag{2.2.16}$$

For the WOGA, there is a similar result.

Theorem 2.2.6. *Let $\mathcal{D}$ be an arbitrary dictionary in $\mathcal{H}$. Let f be a function in $\mathcal{H}$. Let $\tau = \{t_k\}_{k=1}^{\infty}$, and let*

$$\sum_{k=1}^{\infty} t_k^2 = \infty. \tag{2.2.17}$$

Then

$$\lim_{m \to \infty} \|f - G_m^{o,\tau}(f, \mathcal{D})\| = 0.$$

WGA Convergence Criterion

The strongest results, a criterion for convergence, is found in [69, 67]. It is characterised by the complement of a class of sequences ν, i.e., whenever the weakness sequence τ is not in ν, the WGA will converge. It is formulated here.

The difference between successive terms of a sequence $\{q_s\}$ will be given by $\Delta q_s := q_s - q_{s-1}$.

Definition 2.2.7. Let $x := \{x_k\}_{k=1}^{\infty}$ be a sequence such that $x_k \geq 0, k \geq 1$. Let $q := \{q_j\}_{j=1}^{\infty}$ be a sequence where $0 = q_0 < q_1 < \ldots$. If q and x satisfy the two

properties

$$\sum_{s=1}^{\infty} \frac{2^s}{\Delta q_s} < \infty \tag{2.2.18}$$

$$\sum_{s=1}^{\infty} 2^{-s} \sum_{k=1}^{q_s} x_k^2 < \infty, \tag{2.2.19}$$

then x is a member of the class of sequences ν.

This leads to the theorem on convergence, restated:

Theorem 2.2.8. *Let $\mathcal{H}$ be an arbitrary Hilbert space, let $\mathcal{D}$ be an arbitrary dictionary in $\mathcal{H}$, and let τ be a weakness sequence for a WGA. Then*

$$\lim_{m\to\infty} \|f - G_m^{\tau}(f, \mathcal{D})\| = 0 \tag{2.2.20}$$

if and only if $\tau \notin \nu$.

2.2.3 WGA Approximation Results

Jackson theorems for convergence for approximation classes are proved in [65] in terms of the weakness sequence, for the WOGA and the WRGA. First, we give the rate of approximation for the WGA.

Theorem 2.2.9. *Let $\mathcal{D}$ be an arbitrary dictionary in $\mathcal{H}$. Let $\tau := \{t_k\}_{k-1}^{\infty}$ be a non-increasing sequence. Let $f \in \mathcal{A}_1(\mathcal{D})$, and let $\beta_m := \dfrac{t_m}{2(2+t_m)}$. Then*

$$\|f - G_m^{\tau}(f, \mathcal{D})\| \leq \frac{|f|_{\mathcal{A}_1(\mathcal{D},M)}}{(1+\sum_{k=1}^{m} t_k^2)^{\beta_m}}. \tag{2.2.21}$$

Next we have results for the WOGA.

Theorem 2.2.10. *Let $\mathcal{D}$ be an arbitrary dictionary in $\mathcal{H}$, and let f be a function in $\mathcal{A}_1(\mathcal{D}, M)$. Then*

$$\|f - G_m^{o,\tau}(f, \mathcal{D})\| \leq \frac{|f|_{\mathcal{A}_1(\mathcal{D})}}{\sqrt{(1 + \sum_{k=1}^m t_k^2)}}. \tag{2.2.22}$$

Similarly, for the two variations of the WRGA defined in section 2.2.1, we have the following results:

Theorem 2.2.11. *Let $\mathcal{D}$ be an arbitrary dictionary in $\mathcal{H}$, and let f be a function in $\mathcal{A}_1(\mathcal{D}, M)$. Then*

$$\|f - G_m^{\tau,1}(f, \mathcal{D})\| \leq \frac{2}{\sqrt{(1 + \sum_{k=1}^m t_k^2)}}. \tag{2.2.23}$$

$$\|f - G_m^{\tau,2}(f, \mathcal{D})\| \leq \frac{2}{\sqrt{(1 + \sum_{k=1}^m t_k^2)}}. \tag{2.2.24}$$

2.2.4 Additional Variants

There are some additional variants suggested by the WGA, which appear in [69]. They are presented here as they may prove to be of use in implementation, yet they fall slightly outside the neighbourhood of the WGA; in fact, the Co-Convex Algorithm that follows is not greedy in the same sense as previous algorithms we have seen.

Modified WGA Variants

The Modified variants come from replacing the standard condition for choosing a dictionary atom,

$$|\langle f_{m-1}^\tau, \phi_m^\tau \rangle| \geq t_m \sup_{g \in \mathcal{D}} |\langle f_{m-1}^\tau, g \rangle| \tag{2.2.25}$$

with the condition

$$|\langle f_{m-1}^\tau, \phi_m^\tau \rangle| \geq t_m \|f_{m-1}^\tau\|^2. \tag{2.2.26}$$

We have seen something similar in the definition of the WRGA. Following the suggestion of Temlyakov, we replace equation 2.2.4 in the initial step of the WGA with

$$|\langle f_{m-1}^{o,\tau}, \phi_m^{o,\tau}\rangle| \geq t_m \|f_{m-1}^{o,\tau}\|^2 \tag{2.2.27}$$

and get a rate of convergence that is analogous to that given in equation 2.2.22.

And for the final WGA variant, a similar procedure allows us to define the MWGA and state a result for the rate of convergence that Temlyakov derives from the results in [65]. The superscript M indicates the Modified algorithm, and the omission of the approximant from the definition in [69, p. 62] is corrected here.

Definition 2.2.12 (MWGA). Let a sequence $\tau = \{t_k\}_{k=1}^{\infty}$, with $0 < t_k < 1$ be given. Define $f_0^{M,\tau} = f$. For $m = 1$, $\phi_m^{M,\tau} \in \mathcal{D}$ is any function such that equation 2.2.26 holds. Set

$$f_1^{M,\tau} := f_0^{M,\tau} - \left\langle f_0^{M,\tau}, \phi_1^{M,\tau} \right\rangle \phi_1^{M,\tau} \tag{2.2.28}$$

Then we define the m-th iteration of the MWGA as follows:

Let $\phi_m^{M,\tau} \in \mathcal{D}$ be any function satisfying

$$\left|\left\langle f_{m-1}^{M,\tau}, \phi_m^{M,\tau} \right\rangle\right| \geq \frac{t_m \left\| f_{m-1}^{M,\tau} \right\|^2}{1 + \sum_{k=1}^{m-1} \left|\left\langle f_{m-1}^{M,\tau}, \phi_k^{M,\tau} \right\rangle\right|}, \tag{2.2.29}$$

then update approximant and remainder

$$\begin{aligned} f_m^{M,\tau} &:= f_{m-1}^{M,\tau} - \left\langle f_m^{M,\tau}, \phi_m^{M,\tau} \right\rangle \phi_m^{M,\tau}, \\ G_m^{M,\tau}(f, \mathcal{D}) &:= G_{m-1}^{M,\tau} + \phi_m^{M,\tau}. \end{aligned} \tag{2.2.30}$$

The following result concerns the norm of the residual.

Proposition 2.2.13. *Let τ be a weakness sequence, with $0 \leq t_k < 1$, such that τ is non-increasing. Let $\beta_m := \dfrac{t_m}{2(2+t_m)}$. For any $f \in \mathcal{A}_1(\mathcal{D})$, there exists a realization of an MWGA. Then for any such realization*

$$\left\|f_m^{M,\tau}\right\| \leq \frac{1}{(1+\sum_{k=1}^m t_k^2)^{\beta_m}}. \tag{2.2.31}$$

Co-convex Approximation Algorithm

There is another algorithm inspired by the WRGA. Called the Co-convex Algorithm, Temlyakov presents it as a remark in [65], observing that it does not require the solution of any optimization problem in the construction of the m-th term. In fact, the choice of the function $\phi \in \mathcal{D}$ is not a greedy choice, and it can be argued that this algorithm is not even a member of the greedy class.

Furthermore, the weakness sequence $\tau = \{t_k\}_{k=1}^{\infty}$ is not fixed, but constructed as the algorithm increments. Otherwise, it corresponds to the WRGA2 in form. The error bound for the Co-convex Algorithm is exactly that of the WRGA,

$$\|f_m\| \leq \frac{2}{\sqrt{(1+\sum_{k=1}^m t_k^2)}}. \tag{2.2.32}$$

Definition 2.2.14 (Co-Convex Algorithm). Let $f \in \mathcal{A}_1(\mathcal{D})$.

1. Find $\phi_1 \in \mathcal{D}$ such that $\langle f, \phi_1 \rangle > 0$.

   Define:

$$t_1 := \min\left\{1, \tfrac{\langle f,\phi_1\rangle}{\|f\|^2}\right\}; \qquad \beta_1 := \tfrac{t_1}{1+t_1^2};$$
$$G_1 := \beta_1\phi_1; \qquad f_1 := f - G_1.$$

2. Find $\phi_2 \in \mathcal{D}$ such that $\langle f_1, \phi_2 - G_1 \rangle > 0$.

   Define:

$$t_2 := \min\left\{1, \tfrac{\langle f_1, \phi_2 - G_1\rangle}{\|f_1\|^2}\right\}; \qquad \beta_2 := \tfrac{t_2}{\left(1+t_1^2+t_2^2\right)};$$
$$G_2 := (1-\beta_2)\, G_1 + \beta_2 \phi_2; \qquad f_2 := f - G_2.$$

... Continue ...

m Find $\phi_m \in \mathcal{D}$ such that $\langle f_m - 1, \phi_m - G_1 \rangle > 0$.

   Define:

$$t_m := \min\left\{1, \tfrac{\langle f_{m-1}, \phi_m - G_{m-1}\rangle}{\|f_{m-1}\|^2}\right\}; \qquad \beta_m := \tfrac{t_m}{\left(1+\sum_{k=1}^{m} t_k^2\right)};$$
$$G_m := (1-\beta_m)\, G_{m-1} + \beta_m \phi_m; \qquad f_m := f - G_m.$$

2.3 Banach Space Algorithms

Most signal processing approximation takes place in L_2, and results have been generalized to L_p and Hilbert spaces, but is there more that can be said about greedy approximation? The obvious question is to ask what if we move from a Hilbert space to the more general Banach space. Theory developed in this context has helped refine results in Hilbert spaces, as mentioned in section 1.8.2.

These algorithms are not only theoretical in nature. Some recent modifications of greedy-type Banach algorithms are motivated by numerical applications. It has been shown that in numerical cubature, the numerical approximation of integration (area in $\mathbb{R}^d$, $d > 2$,) Banach algorithms perform very well. In fact, they offer a deterministic method that performs as well as Monte Carlo methods. (The interested reader is referred to [68].)

2.3.1 Banach Space Preliminaries

This section will provide a review of a few results in the context of Banach spaces. We need a few definitions, taken from [69, 23], and beginning with that for a peak functional.

Definition 2.3.1 (Peak functional). Let X be a Banach space, and $f \neq 0$ be in X. Let F be a functional in the dual space X' such that

$$\begin{aligned} \|F\|_{X'} &= 1 \\ F(f) &= \|f\|_X. \end{aligned} \tag{2.3.1}$$

Then we say F is a "peak functional."

To be clear, $\|F\|_{X'}$ indicates the operator norm of F. Each $f \in X$ has at least one peak functional, provided $f \neq 0$, as a consequence of the Hahn-Banach theorem. Unicity of the peak functional provides the following definition.

Definition 2.3.2 (Smooth Banach space). Let X be a Banach space. For each $f \in X$, there exists a unique peak functional. Then we say that X is "smooth."

It is remarked in [23] that in a Hilbert space, the functional $\frac{f}{\|f\|}$ is unique, where $f : \mathcal{H} \to \mathbb{R}$ is the inner product, therefore every Hilbert space is a smooth Banach space. This also illustrates why the term "norming functional" is sometimes used, and probably a better label, though "peak" will be primarily used.

We need to define the modulus of smoothness of a Banach space.

Definition 2.3.3 (Modulus of smoothness of a Banach space). Let X be a Banach space. Let $x, y \in X$ be any two elements with $\|x\| = \|y\| = 1$. For $u \in \mathbb{R}$, the

modulus of smoothness $\rho(u)$ is defined by

$$\rho(u) := \sup_{x,y} \left(\frac{\|x + uy\| + \|x - uy\|}{2} - 1 \right). \tag{2.3.2}$$

In application, remember that $\rho(u)$ is an even and convex function, and for $u \in (0, \infty)$

$$\max(0, u - 1) \leq \rho(u) \leq u. \tag{2.3.3}$$

Then the definition of a uniformly smooth Banach space is short.

Definition 2.3.4 (Uniformly Smooth Banach Space). Let X be a Banach space with modulus of smoothness $\rho(u)$, such that

$$\lim_{u \to 0} \frac{\rho(u)}{u} = 0. \tag{2.3.4}$$

Then we say that X is a "uniformly smooth" Banach space.

Several consequences are noted in [23]. First, uniformly smooth spaces are also smooth spaces, as we would hope. For a Hilbert space, $\rho(r) = \sqrt{1 + r^2} - 1$. And not only are Hilbert spaces uniformly smooth, but so are the spaces $L_p, 1 < p < \infty$.

2.3.2 Basic Algorithms in Banach Spaces

Since the starting point for studying greedy approximation is the PGA, it seems that the first thing to do is generalize that algorithm to Banach spaces. This is the way most results have been approached – by generalizing the Hilbert space algorithms. We begin with the X-Greedy Algorithm [69]. Where the PGA assumes we can find

the optimal inner product, we now assume that we can find the coefficient functional $\alpha(f)$ and dictionary element $g(f)$ in the greedy step.

Definition 2.3.5 (XGA). Let X be a Banach space and let $\mathcal{D} \subset X$ be a dictionary, and let f be a function in X. Let $\alpha(f)$ be a functional from X to $\mathbb{R}$, and let $g(f) \in \mathcal{D}$, such that

$$\min_{\substack{\alpha \in \mathbb{R}, \\ g \in \mathcal{D}}} \|f - \alpha g\| = \|f - \alpha(f) g(f)\|. \tag{2.3.5}$$

Define $G(f, \mathcal{D}, X) := \alpha(f) g(f)$. Define

$$R_0(f, \mathcal{D}, X) := f \tag{2.3.6}$$

$$G_0(f, \mathcal{D}, X) := 0. \tag{2.3.7}$$

Then define the m-th iteration to be

$$R_m(f) := R_m(f, \mathcal{D}, X) := R_{m-1}(f) - G\left(R_{m-1}(f), \mathcal{D}, X\right), \tag{2.3.8}$$

$$G_m(f, \mathcal{D}, X) := G_{m-1}(f, \mathcal{D}, X) + G\left(R_{m-1}(f), \mathcal{D}, X\right). \tag{2.3.9}$$

There is another generalization of the PGA as well, called the Dual Greedy Algorithm (DGA). Again, we use a functional to find the coefficient of a particular element. It is pointed out that the existence of the desired ("peak") functional is a consequence of the Hahn-Banach theorem. The superscript D indicates we are in the context of the DGA.

Definition 2.3.6 (DGA). Let X be a Banach space and let $\mathcal{D} \subset X$ be a dictionary, and let f be a function in X. Given f, let F_f be a peak functional. Let $g_f \in \mathcal{D}$ be

any function such that

$$|F_f(g_f)| = \max_{g \in \mathcal{D}} |F_f(g)|. \tag{2.3.10}$$

Now find $a \in \mathbb{R}$ such that

$$\|f - ag_f\|_X = \min_{b \in \mathcal{R}} \|f - bg_f\|_X. \tag{2.3.11}$$

Each term of the approximant is $G^D(f, \mathcal{D}) := ag_f$, so at the m-th step, we have

$$G_m^D(f, \mathcal{D}) := G_{m-1}^D(f, \mathcal{D}) + G^D(R_{m-1}^D(f, \mathcal{D})), \tag{2.3.12}$$

$$R_m^D(f, \mathcal{D}) := f - ag_f = R_{m-1}^D - ag_f. \tag{2.3.13}$$

Temlyakov notes in [69] a convergence result for DGA from [23], restated here.

Theorem 2.3.7. *Let X be a Banach space and $\mathcal{D} \subset X$ be an arbitrary dictionary. For each $f \in X$, and for each $m > 1$, let $R_m^D(f, \mathcal{D})$ be the residual of the DGA.*

$$\left\|R_m^D(f, \mathcal{D})\right\|_X < \left\|R_{m-1}^D(f, \mathcal{D})\right\|_X \tag{2.3.14}$$

if and only if X is a smooth Banach space.

As of 2003, there were no general results for either of these algorithms, and as far as can be ascertained, there are still none as of this writing. There are, however, some results for the XGA that can be formulated for a finite-dimensional space. In many cases, greedy approximation or nonlinear approximation reduces to a case of this approximation, but it is beyond the scope of this work. The interested reader is referred to [69, sec. 4.2] and [17].

2.3.3 Weak Banach Algorithms

We can define a weak version of the DGA by choosing the dictionary element ϕ_m^D suboptimally, just as we did to derive the WGA from the PGA. Again, there are no results for this algorithm, but we include the definition of the weak DGA as a model.

Definition 2.3.8 (WDGA). Let X be a Banach space and let $\mathcal{D} \subset X$ be a dictionary. Let τ be a weakness sequence, with $0 \leq t_k \leq 1$, and let f be a function in X. Given f, let F_f be a peak functional. Define $f_0^{D,\tau} := f$. Then for each $m \geq 1$, we make the following definitions:

Let $\phi_m^{D,\tau} \in \mathcal{D}$ be any element such that

$$F_{f_{m-1}^{D,\tau}}(\phi_m^{D,\tau}) \geq t_m \sup_{g \in \mathcal{D}} F_{f_{m-1}^{D,\tau}}(g). \tag{2.3.15}$$

Now find $a_m \in \mathbb{R}$ such that

$$\|f_{m-1}^{D,\tau} - a_m \phi_m^{D,\tau}\| = \min_{a \in \mathbb{R}} \|f_{m-1}^{D,\tau} - a\phi_m^{D,\tau}\|. \tag{2.3.16}$$

Each term of the approximant is $G^{D,\tau}(f, \mathcal{D}) := a_m \phi_m^{D,\tau}$, so at the m-th step, we have

$$G_m^{D,\tau}(f, \mathcal{D}) := G_{m-1}^{D,\tau}(f, \mathcal{D}) + G^{D,\tau}(f_{m-1}^{D,\tau}, \mathcal{D}), \tag{2.3.17}$$

$$f_m^{D,\tau} := f_{m-1}^{D,\tau} - a_m \phi_m^{D,\tau}. \tag{2.3.18}$$

Weak Chebyshev Greedy Algorithm

The Weak Chebyshev Greedy Algorithm (WCGA) was first defined as a generalization of the WOGA in [66]. The generalization is straightforward, with the choice

of dictionary element $\phi^{C,\tau}$ being a weak choice using a functional, and the (best) approximant being selected as the projection over the space of atoms.

Definition 2.3.9 (WCGA). Let X be a Banach space, let τ be a weakness sequence, and let $F_{fC,\tau}$ be a peak functional. Define $f_0^{C,\tau} := f$. For $m \geq 1$, define $\phi_m^{C,\tau} \in \mathcal{D}$ to be any element in D such that

$$F_{f_m^{C,\tau}}(\phi_m^{C,\tau}) \geq t_m \sup_{g \in \mathcal{D}} F_{f_m^{C,\tau}}(g). \tag{2.3.19}$$

Define

$$\Phi_m^{C,\tau} := \text{span}\{\phi_j^{C,\tau}\}_{j=1}^m. \tag{2.3.20}$$

Now $G_m^{C,\tau} := P_{\Phi_m^{C,\tau}}$, the best approximant from $\Phi_m^{C,\tau}$, and

$$f_m^{C,\tau} := f_{m-1}^{C,\tau} - G_m^{C,\tau}. \tag{2.3.21}$$

There are three main results concerning the WCGA. The first is, by observation, that the sequence of the residuals $\left\{f_m^{C,\tau}\right\}_{m=1}^{\infty}$ should be non-increasing as a result of the Gram-Schmidt process. The second result is a Jackson type theorem [66] for the approximation class $\mathcal{A}_1(\mathcal{D})$. It concerns a class of Banach moduli of smoothness known as power-type moduli [66] for non-trivial Banach spaces, and proves that in the following theorem, there are limits placed on q. This is applicable to L_p spaces.

Theorem 2.3.10. *Let X be a uniformly smooth Banach space with the modulus of smoothness $\rho(u) \leq \gamma u^q$, for $1 < q \leq 2$. Let τ be a weakness sequence with $t_k \leq 1$ for $k \in \mathbb{N}$. Let $C(q,\gamma)$ be a constant depending on only q and γ, and let $p := \dfrac{q}{q-1}$. For*

any $f \in \mathcal{A}_1(\mathcal{D})$,

$$\|f_m^{C,\tau}\| \leq \frac{C(q,\gamma)}{(1+\sum_{k=1}^m t_k^p)^{\frac{1}{p}}} \tag{2.3.22}$$

The third result gives a condition for convergence on the weakness sequence. This condition involves an auxiliary sequence, $\{\xi_m(\rho,\tau,\theta)\}$, defined here for a given modulus of smoothness. This keeps us in the context of uniformly smooth Banach spaces.

Definition 2.3.11. Let $\rho(u)$ be an even convex function on $\mathbb{R}$ with $\rho(2) \geq 1$ and $\lim_{u\to 0} \frac{\rho(u)}{u} = 0$. For any weakness sequence τ with $0 < t_k \leq 1$, and a parameter θ such that $0 < \theta \leq \frac{1}{2}$, we define $\xi_m := \xi_m(\rho,\tau,\theta) := u$ where u is a solution of the equation

$$\rho(u) = \theta t_m u. \tag{2.3.23}$$

The theorem, then, can be formulated.

Theorem 2.3.12. *Let X be a uniformly smooth Banach space with the modulus of smoothness $\rho(u)$. For any $\theta > 0$, and auxiliary sequence $\{\xi\}$, let the weakness sequence τ be one such that*

$$\sum_{m=1}^{\infty} t_m \xi_m(\rho,\tau,\theta) = \infty. \tag{2.3.24}$$

Then for any $f \in X$, the limit of the WCGA residual is zero:

$$\lim_{m\to\infty} \|f_m^{C,\tau}\| = 0. \tag{2.3.25}$$

Note that this holds for all $f \in X$, but that is when given an infinite expansion into $\mathcal{D}$.

Weak Relaxed Greedy Algorithm

By this point, the reader has become adept at defining greedy algorithms. The only remark here is that the superscript denoting the Banach space Weak Relaxed Greedy Algorithm is r, τ, as opposed to τ, i for the i-th variation on the Hilbert WRGA.

Definition 2.3.13 (Banach WRGA). Let X be a Banach space, let $F_{f_m^{r,\tau}}$ be a peak functional, and let τ be a weakness sequence. Define $f_0^{r,\tau} := f$ and $G_0^{r,\tau} := 0$. For each $m \geq 1$, let $\phi_m^{r,\tau} \in \mathcal{D}$ be any element satisfying

$$F_{f_{m-1}^{r,\tau}}\left(\phi_m^{r,\tau} - G_{m-1}^{r,\tau}\right) \geq t_m \sup_{g \in \mathcal{D}} F_{f_{m-1}^{r,\tau}}\left(g - G_{m-1}^{r,\tau}\right). \tag{2.3.26}$$

Find λ_m, where $0 \leq \lambda_m \leq 1$ such that

$$\left\|f - \left((1-\lambda_m)G_{m-1}^{r,\tau} + \lambda_m \phi_m^{r,\tau}\right)\right\| = \inf_{0 \leq \lambda \leq 1} \left\|f - \left((1-\lambda)G_{m-1}^{r,\tau} + \lambda \phi_m^{r,\tau}\right)\right\| \tag{2.3.27}$$

and define

$$G_m^{r,\tau} := (1-\lambda_m)G_{m-1}^{r,\tau} + \lambda_m \phi_m^{r,\tau}. \tag{2.3.28}$$

Finally,

$$f_m^{r,\tau} := f - G_m^{r,\tau}. \tag{2.3.29}$$

The rate of approximation results (Theorem 2.3.10) and convergence (Theorem 2.3.12) for the Banach WRGA are exactly the same as those for the WCGA. There are also results for a modification of the Banach WRGA and a Banach space version of the Co-convex Algorithm discussed in section 2.2.4. They are not included here; please refer to [66].

Other Lines of Investigation

Recent results in [70] concern one last modification of the greedy step. For the peak functional $F_{f_{m-1}}$, the conditions are weakened slightly, and the selection of the dictionary atom is also allowed a little variance from optimality. Greedy algorithms of this class are known as Approximate Weak Greedy Algorithms. There are the usual cast of characters, with Approximate variants of the WCGA and WRGA being the stars of the show.

The results for this last class of algorithm show that further relaxing the approximation method does not have a negative effect on either convergence or the rate of approximation. Other recent results by Gribonval, Nielsen, Temlyakov, et al, are exploring generalizations of the greedy algorithm that retain the name for mnemonic purposes only, and so we turn our attention to another topic.

2.3.4 Greedy Banach Approximation over Bases

Having considered m-term approximation in Hilbert spaces in bases and in dictionaries, then considering dictionaries in Banach spaces, it is left to consider approximation in Banach spaces over bases. The initial techniques and results in this area are due to Konyagin and Temlyakov [43].

Greedy Bases

A greedy basis is a bit like a Best Basis, in the Banach space context; it is defined in terms of the properties in which we are interested, namely that a greedy algorithm over a greedy basis will achieve the order of best approximation.

Let X be a Banach space, and we denote an infinite dimensional normalized basis

by Ψ, i.e. $\{\psi_k\}_{k=1}^{\infty}$, and $\|\psi_k\| = 1$. Then for any $f \in X$, it can be expanded as

$$f = \sum_{k=1}^{\infty} c_k(f)\psi_k. \tag{2.3.30}$$

Now for $j \in \mathbb{N}$, we write $\rho := \rho(j) := k_j$ and say that ρ is a permutation of the positive integers. When used as an index into the coefficients of the expansion of f in Ψ, and when the following property holds,

$$|c_{k_1}(f)| \geq |c_{k_2}(f)| \geq ..., \tag{2.3.31}$$

we can write $\rho \in D(f)$. If all the inequalities in the last expression are strict, there is exactly one permutation in $D(f)$.

Thresholding Greedy Algorithm

In effect, the permutation ρ is used to sort the terms of the expansion by the size of the coefficients, like our original notion of nonlinear approximation. So we can define the Thresholding Greedy Algorithm (TGA) by the m-term expansion in Ψ:

Definition 2.3.14 (TGA). Let f be an element of a Banach space X. Let Ψ be a normal basis, and $\rho \in D(f)$ be a permutation. Then define

$$G_m(f, \Psi) := G_m^T(f, \Psi) := G_m^T(f, \Psi, \rho) := \sum_{j=1}^{m} c_{k_j}(f)\psi_{k_j}. \tag{2.3.32}$$

Letting G be a constant that depends on X and Ψ, but not on f or m. Then we define a greedy basis.

Definition 2.3.15 (Greedy Basis). A basis Ψ of a Banach space X is a greedy basis

if for every $f \in X$, there exists a permutation $\rho \in D(f)$ such that

$$\|f - G_m(f, \Psi, \rho)\| \leq G\sigma_m(f, \Psi). \tag{2.3.33}$$

Recall the definition of an unconditional basis (Definition 1.6.2) and the definition of a democratic basis (Definition 1.8.7), we have the main result of [43]:

Theorem 2.3.16. *A basis Ψ for a Banach space X is greedy if and only if it is both unconditional and democratic.*

For a Hilbert space, each orthonormal basis is a greedy basis with the constant $G = 1$, corroborating that we can get near-optimal m-term approximation using an orthonormal basis.

L_p-equivalence

We need the notion of L_p equivalence of bases. In the same way that two norms X_1 and X_2 for a space X are defined to be equivalent if $\forall x \in X$, $\|x\|_{X_1} \asymp \|x\|_{X_2}$, we say that two bases are L_p equivalent if any two simultaneous m-term expansions into them have equivalent L_p norms. In particular, there are results concerning when a basis of a Banach space is L_p equivalent to the Haar wavelet basis, Definition 1.7.5 of section 1.7. We define L_p equivalence following [43]:

Definition 2.3.17 (L_p equivalence). Let $H_p = \{H_k^p\}_{k=1}^{\infty}$ be the Haar basis in L_p, and let $\Psi = \{\psi_k\}_{k=1}^{\infty}$ be another basis for the same space. Let Λ be any finite set of indices k, with c_k coefficients of the expansion into either basis. Let $C_1(p, \Psi)$ and

$C_2(p, \Psi)$ be two constants dependent on only p and Ψ. If H_p and Ψ satisfy

$$C_1(p, \Psi) \left\| \sum_{k \in \Lambda} c_k H_k^p \right\| \leq \left\| \sum_{k \in \Lambda} c_k \psi_k \right\| \leq C_2(p, \Psi) \left\| \sum_{k \in \Lambda} c_k H_k^p \right\| \tag{2.3.34}$$

then we say that Ψ is L_p-equivalent to H_p.

The existence of greedy bases for many L_p spaces is established in the following theorem:

Theorem 2.3.18. *Let $1 < p < \infty$, let X be a Banach space with the L_p norm, and let Ψ be a basis for X that is L_p equivalent to the Haar basis H_p. Let $\rho \in D(f)$ be a permutation of any function $f \in L_p(0,1)$, and let $C(p, \Psi)$ be a constant depending on only p and Ψ. $G_m(f, \Psi, \rho)$ denotes the m-term TGA approximant. Then*

$$\|f - G_m(f, \Psi, \rho)\|_{L_p} \leq C(p, \Psi)\ \sigma_m(f, \Psi)_{L_p}. \tag{2.3.35}$$

Jackson and Bernstein Theorems

Finally, we are nearly prepared to understand that we have results like Jackson and Bernstein theorems for L_p-equivalent bases. In keeping with the literature, we write our expansion in terms of a basis Ψ as $f = \sum_{n=1}^{\infty} f_n \psi_n$. We need two more things to formulate the results.

Let $\mathcal{E} = \{\varepsilon_k\}_{k=0}^{\infty}$ be a sequence, monotonically decreasing to zero, of positive numbers. Call this class of sequences MDP. Given some $\mathcal{E} \in MDP$, define another sequence of non-negative integers $\{N_s\}_{s=0}^{\infty}$ inductively:

Let $N_0 = 0$. Let N_s be the smallest non-negative integer such that $\varepsilon_{N_s} < 2^{-s}$. Also define $n_s := \max(N_{s+1} - N_s, 1)$.

Now let $f \in L_p$, rearrange the sequence $\|f_n\psi_n\|$ in decreasing order,

$$\|f_{n_1}\psi_{n_1}\|_p \geq |f_{n_2}\psi_{n_2}\|_p \geq \ldots \tag{2.3.36}$$

and let $a_k(f,p) := \|f_{n_k}\psi_{n_k}\|_p$.

We are now equipped to write the characterization of this approximation.

Theorem 2.3.19. *Let $\mathcal{E} \in MDP$ be a sequence such that the following are true for $s = 0, 1, 2, \ldots$:*

$$\varepsilon_{N_s} \geq C_1 2^{-s} \tag{2.3.37}$$

$$n_{s+1} \leq C_2 n_s. \tag{2.3.38}$$

Then we have

$$\sigma_n(f,\Psi)_p \ll \varepsilon_n \iff a_{N_s}(f,p) \ll 2^{-s} n_s^{-1/p}. \tag{2.3.39}$$

Applying this theorem and two lemmas to two examples in [69, pp. 88-89], we get the following two corollaries for bases that are L_p-equivalent to H_p.

Corollary 2.3.20. *Given $f \in L_p$, and a basis Ψ that is L_p-equivalent to the Haar basis H_p. Define $\mathcal{E}$ such that $\varepsilon_0 = 1$ and for $r > 0$ and $k = 1, 2, \ldots$, we have $\varepsilon_k = k^{-r}$. Then $N_s \asymp 2^{s/r}$ and $n_s \asymp 2^{s/r}$, and*

$$\sigma_m(f,\Psi)_p \asymp (m+1)^{-r} \iff a_n(f,p) \asymp n^{-r-1/p}. \tag{2.3.40}$$

Corollary 2.3.21. *Given $f \in L_p$, and a basis Ψ that is L_p-equivalent to the Haar basis H_p. Fix $0 < b < 1$ and for $k = 0, 1, 2, \ldots$ define $\varepsilon_k = \dfrac{1}{2^{k^b}}$. Then $N_s = s^{1/b} + \mathcal{O}(1)$*

and $n_s \asymp 2^{1/b-1}$, *and*

$$\sigma_m(f,\Psi)_p \asymp \frac{1}{2^{m^b}} \iff a_n(f,p) \asymp \frac{n^{\frac{(1-1/b)}{p}}}{2^{n^b}}. \tag{2.3.41}$$

And for the conditions used in the first of the two corollaries above, another characterization is possible.

Theorem 2.3.22. *Let* $1 < p < \infty$ *and* $0 < q < \infty$. *Let* $r > 0$, *arbitrarily. Define* $\mathcal{E}$ *such that* $\varepsilon_0 = 1$ *and we have* $\varepsilon_k = k^{-r}$. *Thus* $N_s \asymp 2^{s/r}$ *and* $n_s \asymp 2^{s/r}$. *Then, letting* $\aleph := rq - 1$, *we have the equivalence relation*

$$\sum_m \sigma_m(f,\Psi)_p^q m^{\aleph} < \infty \iff \sum_n a_n(f,p)^q n^{\aleph+\frac{q}{p}} < \infty \tag{2.3.42}$$

The observation is also made that if $\Psi = H_p$ and

$$q = \beta := \frac{1}{r + \frac{1}{p}}, \tag{2.3.43}$$

then the right hand statement of equation 2.3.42 becomes the membership criteria for the Besov space $B^r_\beta(L_\beta)$.

Stability and Implementation

The final results about greedy algorithms in Banach spaces in [69] concern stability, where there are classical results showing that for an unconditional basis, coefficients will be numerically stable. However, it is advertised that employing the technique of thresholding is usually preferable to employing greedy algorithms when implementing nonlinear m-term approximation. That may be for some applications, but we will still examine numerical implementation of greedy algorithms in Chapter 3.

Chapter 3

Numerical Implementation

"The difference between theory and practice is that
in theory, there is none."
—Various

In the literature, it has been repeatedly said that the algorithms of greedy approximation are "theoretical" algorithms. There exist, however, various software implementations of the Matching Pursuit Algorithm, which is the term we will favour in this chapter, dating back to Mallat and Zhang's paper [50] in 1993. How can this be? In a sense, it is by cheating.

When discussing theory, we are interested in an algorithm in the most general sense. When we set out to realize a given algorithm, however, we do not hesitate to use information specific to our particular dictionaries. So when Temlyakov says "these algorithms are not yet ready for implementation," he is not incorrect, merely speaking from the catholic viewpoint of the theoretician. Any software that actually runs only works for limited dictionaries and specific algorithms, in prescribed spaces

of functions.

The problem of implementation, then, is one of specification. There are five primary areas to specify:

1. the type of dictionary to be used and methods of analysis,

2. algorithms to search and sort analysis results,

3. structures and methods of recording the data and results,

4. what to update at each iteration and how to do so,

5. housekeeping.

Housekeeping, in proper implementation, specifies stopping criteria, range and type checking, error trapping, and so forth. Like some modern computer science curricula, we will discuss housekeeping theoretically or superficially, if at all, despite the fact that good housekeeping is the difference between theory and practice.

As a prelude to specific issues in writing programs, there is one result concerning complexity that begs examination. It has been shown that greedy algorithms belong to a class of problems that are known as NP-hard. A relatively concise discussion of computational complexity as it pertains to approximation follows.

Not only does this give us the proper context in which to consider implementation, but makes us aware that working on fast MP algorithms buys the industrious programmer a ticket in the "P vs. NP" lottery [8].

3.1 Greedy Choice is NP-hard

As stated in Chapter 2, the PGA, RGA, and OGA are considered "pure" because they make the assumption that at each iteration, the function $g \in \mathcal{D}$ that maximizes the inner product with the remainder is known. When $\mathcal{D}$ is an orthonormal basis, we know that this is possible. In the case where $\mathcal{D}$ is a redundant dictionary, the time required to find this g increases far faster than the rate at which the size of the dictionary increases.

3.1.1 Computational Complexity Defined

A few words about computational complexity are necessary, but it should not be forgotten that approximation is the focus of this work. The computational complexity of an algorithm is the number of steps that an algorithm must take to generate the solution on an input of length n. Formally, we are talking about the input as a string in the context of a Turing machine. Further detail is unnecessary for the current subject; a less rigorous description than that found in [59] will suffice.

Definition 3.1.1. An algorithm L is a member of the class P if it takes $\mathcal{O}(n^k)$ steps to run on an input of size n, for some constant k.

From a theoretical standpoint, class-P algorithms are feasible. For a particular problem, they may be slow, but in general are able to be solved.

If an algorithm takes something like $\mathcal{O}(2^n)$ steps, the complexity increases exponentially with the size of the Turing input. This means that the algorithm runs in exponential time, not polynomial time, and is usually too complex to be useful.

For some algorithms, it is possible to *verify* their results in polynomial time.

Definition 3.1.2. An algorithm L is a member of the class NP if it can be *verified* in polynomial time.

This means that there is some algorithm that will take the results and another input, called a certificate, and verify that the results are a solution to the original problem. For example, we might like to know whether $\tilde{f} = \sum_i c_i \phi_i$ is an (ε, M)-approximant.

A verifier V in this case would be an algorithm that takes the input for the PGA: some function f, the norm, the desired ε, and a dictionary, with a certificate as well. The certificate would in this case just be an approximant: the set $\{c_i\}$ and $\{\phi_i\}$, each of cardinality M.

The verifier V would compute $\|\tilde{f} - f\|$. If $\|\tilde{f} - f\| < \varepsilon$, then we have verified that $\tilde{f}$ is an (ε, M)-approximant.

It is relatively intuitive that, in this case, the verification can be done in polynomial time. We presume that there are a finite number of computations to multiply a function by a coefficient, and a finite number to add a function to the rest, which should vary directly with the length of the input. Therefore, verifying an (ε, M)-approximation is in the class NP.

Note that being a member of the class NP tells us that the program can be verified in polynomial time, but note as well that we do not know whether or not a solution can be found in polynomial time. One of the most important questions in mathematics and the theory of computation is whether or not P = NP. Any algorithm in P must be in NP as well, but there are some problems in NP for which no polynomial time algorithm has been found. Generally, it is considered that P $\subset$ NP, but it has not been proved.

The former assumption is more pragmatic because there is a class of problems known as NP-complete. We will call a problem L, for language, as a problem in this sense can be regarded as a "formal language."

Definition 3.1.3. A problem L is *NP-complete* if two properties hold:

1. $L \in \text{NP}$,

2. $L' \subset_P L, \quad \forall L' \in \text{NP}$.

The notation $L_1 \subset_P L_2$ means that L_1 can be *reduced* to L_2, by means of a polynomial-time algorithm. The notation $L_1 \subset_P L_2$ indicates that the problem L_1 is a special case of the problem L_2. Often $L_1 \leq_P L_2$ is written, but due to the ambiguity introduced by the use of the word "reduces" to indicate membership in a larger set, the notation above will be used.

In other words, if a problem is in class NP, and all other problems in NP are special cases of the first, then the first is considered NP complete. An important consequence is that if there exists a polynomial-time algorithm for *any* problem in NP, then P = NP. For further discussion of reducibility and complexity classes, refer to [11, 59].

If the second condition holds, i.e., if $L' \subset_p L$ for all L' in NP, but we do not necessarily have that L itself is in NP, then the problem L is considered to be NP-hard.

3.1.2 Complexity of Optimal Approximation

Davis, Mallat, and Avellaneda in [14] showed the computational complexity of greedy approximations.

Theorem 3.1.4. *Let $\mathcal{H}$ be an N-dimensional Hilbert space. Let $\mathcal{D}_N$ be the set of all dictionaries for $\mathcal{H}$ that contain $\mathcal{O}(N^k)$ vectors, where $k \geq 1$. Let $0 \leq \alpha_1 \leq \alpha_2 < 1$ and $M \leq N$ such that $\alpha_1 N \leq M \leq \alpha_2 N$. The finite input* (ε, M)-approximation problem, *determining for any given $\varepsilon > 0$, $\mathcal{D} \in \mathcal{D}_N$, and $f \in \mathcal{H}$, whether an (ε, M)-approximation exists, is NP-complete. The finite-input M-optimal approximation problem, finding the optimal M-approximation, is NP-hard.*

The proof that the M-optimal approximation problem is NP-hard consists of showing that the solution to the (ε, M) problem reduces to finding M-optimal approximation, then checking that the optimal approximation is better than ε. But, it is shown that the (ε, M)-approximation problem is NP-complete by showing that a known NP-complete problem reduces to solving the (ε, M)-approximation problem. The NP-complete problem used is known as the *exact-cover by 3-sets* problem [29]. See also [14].

Definition 3.1.5 (Exact cover by 3-sets). Let X be a set containing $N = 3M$ elements and let $\mathcal{C}$ be a collection of 3-element subsets of X. The Exact cover by 3-sets problem is to decide whether $\mathcal{C}$ contains a subcollection $\mathcal{C}'$ such that every member of X occurs in exactly one member of $\mathcal{C}'$.

We are then given the following lemma:

Lemma 3.1.6. *In polynomial time, we can transform any instance $(X, \mathcal{C})$ of the exact cover by 3-sets problem having size $|X| = 3M$ into an equivalent instance of the (ε, M)-approximation problem using a dictionary of size $\mathcal{O}(N^3)$ in an N-dimensional Hilbert space.*

There are two important consequences. While the problem may be intractable in

the general case, for a *specific* dictionary, there may be an efficient solution. The other consequence is that if we have an algorithm which solves the optimal approximation problem, then the algorithm solves an NP-hard problem. The reader should, however, remain skeptical of any claims of existence for such an algorithm and firmly resist any attempts to generalize the implications above. The odds of answering the P vs. NP question are long.

3.1.3 Measuring Efficiency

There is one word of caution to be made here about some poor practices commonly found in numerical research involving computers. In order to write results that are verifiable and have some degree of scientific rigour, some papers try to measure elapsed time, and qualify the results by including information about the make and model of computer being used, the operating system, or other such specifications. These attempts are misguided at best.

When measuring the efficiency of an algorithm, reference to elapsed time in seconds, minutes, or any other unit of time is mostly useless when making comparisons between algorithms on the same computer system. When comparing between different computer systems, these measurements are completely useless.

Nearly all modern computers use time-sharing operating systems. During the time during which the performance of a program is measured, the computer may well undertake a number of other operations, ranging from internal tasks like swapping and paging memory to external tasks such as dealing with peripherals, or in the case of a multi-user or multi-threaded system, even running entirely different programs or virtual computer systems. Even on a single-user system, there are interrupts which

cause program execution to be suspended from time to time. With the exception of running a program directly on a processor with no operating system, it is impossible to know the load and scheduling conditions that existed in the kernel of the operating system during execution, and thus the load on the system at that time. Moreover, even if they were known, there is no guarantee they can be reproduced.

The preferred method for indicating efficiency of algorithms in publication is to use asymptotic notation, i.e. $\mathcal{O}(N)$. No matter how carefully the system is described, any results for runtime given in units smaller than fortnights should be ignored.

3.2 Existing Software

In order to keep from reinventing the wheel, it is necessary to understand previous work. Here, then, are descriptions of known implementations of the matching pursuit algorithm. Note that all of them work in the context of a discretized Hilbert space, in fact $L^2(\mathbb{R})$, which is the most natural setting for the time-frequency results desired in signal processing (and other) applications.

3.2.1 Classic Matching Pursuits

The first implementation of greedy algorithms was Mallat and Zhang's software [50], which we will identify as MP-93. This software implements a fast matching pursuit algorithm. The algorithm in table 3.1 is extremely similar to the mathematical algorithms of Chapter 2, with α being a static parameter of suboptimality, akin to the weakness parameter of the WGA and its cousins. Discussion and clarification follows.

1. Set $m = 0$, compute all inner products $\{\langle f, g_\gamma \rangle\}_{\gamma \in \Gamma}$
2. Find $g_{\gamma_m} \in \mathcal{D}$ such that
$$|\langle R^m f, g_{\gamma_m} \rangle| \geq \alpha \sup_{\gamma \in \Gamma} |\langle R^m f, g_\gamma \rangle|$$
3. For all $g_\gamma \in \mathcal{D}$ with $\langle g_{\gamma_m}, g_\gamma \rangle \neq 0$
$$\langle R^{m+1} f, g_\gamma \rangle = \langle R^m f, g_\gamma \rangle - \langle R^m f, g_\gamma \rangle \langle g_{\gamma_m}, g_\gamma \rangle$$
4. Stop if
$$\left\| R^{m+1} f \right\|^2 = \|R^m f\|^2 - |\langle R^m f, g_\gamma \rangle|^2 \leq \varepsilon^2 \|f\|^2.$$
Else $m := m + 1$. Return to step 2.

Table 3.1: Fast Matching Pursuit Algorithm

The notation γ is an indexing parameter into the dictionary, and Γ is the set of all indices. In practice, these are assumed to be time-frequency parameters, such as $\gamma = (s, u, \xi)$ for Gabor dictionaries.

Real vs. Complex atoms

In the MP-93 software, there are three possible dictionaries that can be chosen: a Gabor dictionary, a wavelet packet dictionary using QMF generated wavelets, and a Fourier basis with Dirac (unit) impulses added. While this software is capable of iterating a pursuit using any one of the three dictionaries, it is not capable of conducting a pursuit across multiple dictionaries. The exception would be if the user were to take manual control by limiting each pursuit to a single iteration.

The Gabor dictionary is the one that is described the most completely. It performs the best in most situations, and is the one that best illustrates issues found in implementation and their solutions. Fourier atoms can be considered as a special

case of Gabor atoms.

Recall the Gabor transform introduced in section 1.5.1, and the expression given in equation 1.5.1. From this, it requires a small deduction to determine that an atom in a Gabor dictionary is as related in [50, 49]. Let the indexing parameter mentioned earlier be $\gamma = (s, u, \xi)$, so that

$$g_\gamma(t) := \frac{1}{\sqrt{s}} g\left(\frac{t-u}{s}\right) e^{i\xi t} \tag{3.2.1}$$

Since we are working with real-valued signals, we need real-valued Gabor atoms. The most efficient way to compute inner products is using the FFT, but for practical reasons, that implies that we are working with complex exponentials. Summing the sine and cosine portions of the atom can be thought of as an implicit expression of phase for a single cosine function.

Using the phase parameter $\phi \in [0, 2\pi)$ the atoms become

$$g_{(\gamma,\phi)} = \frac{K_{(\gamma,\phi)}}{\sqrt{s}}\, g\left(\frac{t-u}{s}\right) \cos(\xi t + \phi). \tag{3.2.2}$$

The value $K_{(\gamma,\phi)}$ is a constant, because we require that $\|g_{(\gamma,\phi)}\| = 1$. Bear in mind that without the subscript (γ, ϕ), the function g indicates the window function. The window used in this implementation is

$$g(t) = 2^{1/4} e^{-\pi t^2}. \tag{3.2.3}$$

Because of the Hermitian symmetry of real-valued atoms in the frequency domain,

we have $\gamma^- = (s, u, -\xi)$, so from the Fourier transform, we get

$$g_{(\gamma,\phi)} = K_{(\gamma,\phi)} \frac{e^{i\phi} g_\gamma(t) + e^{-it} g_{\gamma^-}}{2} \tag{3.2.4}$$

Since in practice we are interested in sampled (discrete) signals, we need discrete atoms, but these are easily generated by taking t at N discrete values (regularly spaced,) which means that the scale parameter s and the translation parameter u should also be taken at the same discrete values. The frequencies are also taken at the discrete values $\frac{2\pi k}{N}$.

Sub-dictionaries

There are two key ideas, both related to avoiding the computation of inner products, that make matching pursuits possible. The first is that for some dictionaries, it is possible to search through them on a coarse level by taking only the appropriate discrete subset of indexing parameters. This subset will be determined by the optimality parameter α.

Theorem 3.2.1. *Let Δu and $\Delta\xi$ be discrete intervals of time and frequency (respectively) that satisfy*

$$\Delta u = \frac{\delta\xi}{2\pi} < 1. \tag{3.2.5}$$

Let $a > 1$ be an elementary dilation factor. Let $\Gamma = \mathbb{R}^+ \times \mathbb{R}^2$ be the set of all indices into a Gabor dictionary. Let $\Gamma_\alpha \subset \Gamma$ be the discrete subset made up of $\gamma = (a^j, pa^j\Delta u, ka^{-j}\Delta\xi)$, for $(j, p, k) \in \mathbb{Z}^3$.

There exists a constant $\alpha > 0$ *such that for all* $f \in L^2(\mathbb{R})$,

$$\sup_{\gamma \in \Gamma_\alpha} |\langle f, g_\gamma \rangle| \geq \alpha \sup_{\gamma \in \Gamma} |\langle f, g_\gamma \rangle| \tag{3.2.6}$$

This theorem guarantees that even for an infinite (continuous) dictionary, we will be able to make a finite table of inner products, which will allow us to eliminate vast portions of the input when searching for the greedy choice.

Incremental updates

The second idea which makes fast matching pursuit possible is that there is no need to compute every inner product at each step. Beginning with the residual at step $m+1$ and taking inner products with the whole equation, we can write

$$R^m f = \langle R^m f, g_{\gamma_m} \rangle g_{\gamma_m} + R^{m+1} f \tag{3.2.7}$$

$$\Longleftrightarrow \langle R^{m+1} f, g_\gamma \rangle = \langle R^m f, g_\gamma \rangle - \langle R^m f, g_{\gamma_m} \rangle \langle g_{\gamma_n}, g_\gamma \rangle . \tag{3.2.8}$$

This is the consequence of observing that there will be no interaction between atoms when the intersection of their intervals of support is empty. (The last inner product in equation 3.2.8 is zero.) Thus step 3 in table 3.1 appears in the software algorithm, but not the mathematical algorithm.

Back Projection

The software for MP-93 incorporates the method of the OGA. It is termed "back projection," but is clearly the same method of expressing the residual as an orthogonal projection over the m dictionary elements that have already been chosen.

Implementing the orthogonalization is an ordinary application of numerical analysis. The coefficients of the linear combination of functions is computed as the solution to a system of linear equations.

This system is expressed as $Y = GX$. G is the Gram matrix of all possible inner products between the vectors of the subspace H_m of section 2.1.4. Here X is the vector of coefficients that are sought, and Y is the vector of inner products of a given element from H_m with R^m. The result is a sparse matrix system, which is solved using the conjugate-gradient method.

When the statement is made that the OGA has better accuracy with more computational expense than the PGA, it is solving this system of equations that is meant. If p is the number of non-zero entries in the matrix G and n is the number of conjugate-gradient iterations, then the additional expense is given to be $\mathcal{O}(np)$ operations more than a non-orthogonal pursuit.

Wavelet Packets

3.2.2 MPTK

A recent revision of the Matching Pursuit algorithm, called the Matching Pursuit Tool Kit (MPTK) is under development [31], with initial results being presented within days of this writing [32]. While not complete, MPTK provides a framework similar to MP-93, but with some algorithmic improvements, and some additional features for the user.

The foremost improvement is that of speed. Where matching pursuit is usually found to be of complexity $\mathcal{O}(n^2)$, MPTK uses a refinement of the notion that most atoms do not interact with each other in order to reduce the computational complexity

to the order $\mathcal{O}(N \log N)$. The bulk of the improvement comes from arranging the coefficients so that inner products may be found using a tree based search (see [11]).

Without ignoring earlier warnings about quantitatively measuring efficiency, we will point out a qualitative improvement. On the same system, a matching pursuit on a large sample that took about 20 hours using classic algorithms was shrunk to about 15 minutes of run-time using the tree-search of MPTK [31].

One other interesting thing about MPTK is that it is designed to support compound dictionaries. The dictionary that it uses for a pursuit is specified by a text file. At this time, it does not, however, implement wavelet dictionaries. The infrastructure exists for them to be added in the future, however.

3.2.3 Other Packages

There is other software available that implements either matching pursuits, or provides tools from which matching pursuits could be implemented. For the most part, it is all being developed within the mathematical signal processing community.

The package WaveLab [25] by Donoho and others implements matching pursuit in a local cosine (Gabor) basis, as well as a wavelet packet basis, following MP-93. WaveLab is a set of routines that run under MATLAB. Many illustrations and graphs were produced, such as those for Mallat's book [49], under a philosophy known as Reproducible Results. Development of WaveLab reached a zenith several years ago, however, and has been quiet for some time.

A package called LastWave [3] has been made available by Emmanuel Bacry. The basic package concentrates on wavelet analysis, but Rémi Gribonval wrote the precursor to MPTK as a module for LastWave. The package is written in C.

There is a library of C++ functions (software "routines," not mathematical functions) for wavelet analysis and other signal analysis tools called wave++ [41], which also provides an implementation of matching pursuit over Gabor dictionaries. The documentation for this package includes detailed explanations of some of the numerical computations employed, which due to some clever arithmetic, eliminate the phase parameter in finding a real Gabor atom.

The software mentioned up to this point are the major packages that are available for matching pursuits. Many textbooks have accompanying software, such as Teolis [71] and Strang [61]. Pierre Vandergheynst has some nice tutorial time-frequency exercises for those that read French [73]. Any developments regarding matching pursuit software would likely appear on the web site for the Wavelet Digest [72].

3.3 Adaptations

After evaluating all the various implementations, certain refinements and adaptations have become apparent. Some have been implemented, others have not. No software that is currently available incorporates or considers all of them.

3.3.1 Weakness Sequences and Other Norms

Since the optimality parameter α is employed in MP-93, then it would seem possible to have it vary and thus implement a version of the WGA. The difficulty lies in the way that the optimality parameter is used. By letting $\alpha = t_m$, it would need to be changed at every iteration, which implies that it is now necessary to recompute the tables of inner products, which is the thing to avoid.

Another adaptation that could prove useful is to measure in norms other than L_2.

3.3.2 Compound Dictionaries

By a compound dictionary, we mean one that is made up of more than one type of atom. If the greedy approximant were made up of a sum of wavelets, Gabor atoms, Fourier components, splines, and so forth, in a non-homogeneous manner, that would imply that the dictionary was compounded from each of these types. None of the existing software to date can search across a dictionary made up of different types of atoms. The MPTK software [31] is designed to be able to incorporate this ability in the future, but cannot as of this writing.

The compound nature of the dictionary suggests a structure that can be used to implement it. At each step of the pursuit, the best atom could be found within each type of dictionary. Each dictionary would be used to yield a greedy choice. Once all the dictionaries have reported, then the best out of the various types of dictionaries would be chosen.

The parameters for an atom would necessarily depend on the type of atom chosen, so an object-oriented approach is suggested. Each type of atom would be a class, and would need a method to add itself to a function, and to subtract itself from a residual.

In order to be effective, it would also need to know how to update tables of inner products, so that the fast computational aspects can be preserved. As long as these concerns are addressed, it appears that there is room in the future for implementing functions other than those mentioned.

3.3.3 Parallel Processing

Since the search for the best inner product is probably conducted independently over each family of atom, using multiple processors in parallel is a possibility that demands consideration. This author had this thought, and found that Griboval and Krstulović had it independently [31]. As of this writing, no parallel algorithms have been published. For entertainment, the reader may find the song "Lobachevsky," by Tom Lehrer [46] either apt or amusing.

In a parallel algorithm, there would be a Master node, whether a separate machine, processor, or thread. Each type of dictionary to be implemented would run on its own node. Each node would maintain a copy of the residual and inner products for that particular type of dictionary.

Initially, the target function would be passed to each node as the first residual, and zero as the first atom. Each node would update the residual, and the local table of inner products, then nominate a candidate atom by conducting the MP search, and send that atom and it's coefficient back to the Master node.

The Master node would elect the best atom from all the atoms nominated, add it to the approximant, and subtract it from the remainder. It would track the number of iterations and current error, and compare these to the stopping criteria that have been specified. In this way, the greedy algorithm could scale to a large and highly redundant dictionary, provided it is built from various types of atoms for which inner products can be computed.

At the present time, the primary toolset for parallel scientific computing is the Message Passing Interface (MPI) [33]. This is a standardized software library which has interfaces (API) for a number of languages (C, C++, FORTRAN, Python, Lisp,

etc.). It is supported in MATLAB with the Distributed Computing Toolbox, and can be used on a wide variety of computing platforms, from personal computers to supercomputers.

Furthermore, should the reader find it useful to process FFT calculations in parallel, the FFTW library [26] mentioned in section 1.3.5 has a version that supports MPI. Also, it is possible that the wavelet transform could be implemented in parallel due to the local properties of wavelets, though translation across the boundaries of the support of wavelets (or scaling functions) could negate any potential benefits.

Chapter 4

Omissions and Conclusion

4.1 Omissions

In order to produce a comprehensible thesis, it is necessary to choose those results which correlate the best with the central topic. It follows that some omissions must also be made. Some concern fascinating results, but they may be either too complex or too tangentially related to include. This section is the list of intentional omissions; it is hoped that no omissions are of any other type.

There are recent results concerning convergence of the Weak Chebyshev Banach algorithm in unweighted Bergman Spaces [22]. There are convergence results for the Weak Dual Greedy Algorithm where X is a subspace of a quotient of L_p, for $1 < p < \infty$ [28]. Both of these have been omitted because of the onus of defining prerequisite material: Bergman spaces, Frechet derivatives, quotient spaces, etc.

Any mention of ergodic theory has been omitted. It arises in the statistical study of the asymptotic properties of the residual in matching pursuits [14]. Also omitted are the properties of chaos in the same context.

There are results on a class of algorithms known as Approximate Greedy Algorithms, where a further approximation is made. The normalizing functional used to make the greedy choice in a Banach space is replaced by an approximation of the ideal functional.

Of interest to this author are the use of Harmonic Atoms, which are Gabor atoms with multiple frequency centres, as per Gribonval and Bacry [30]. The September 2006 publication of a paper by L. Daudet [13] on Molecular Matching Pursuit is also anticipated.

Finally, many researchers including DeVore and Temlyakov, have found greedy approximation, through connections to pattern matching and heuristics, to be a bridge to research in computational learning theory. It is there that we must draw a line, and leave the discovery of further results to the ingenuity of the reader.

4.2 Conclusion

We have seen that approximation can be done in a greedy way, but that the success of such approximation is dependent on the functions available to us. Wavelets provide a means for developing functions, which are, due to their orthonormal nature, easy to calculate. Relaxing the amount of structure on the approximating set allows great flexibility, and quick initial convergence, though approximation rates tend toward zero asymptotically. The permutations of working in inner product spaces, Banach spaces, bases, and redundant dictionaries have been thoroughly, but not completely researched. There are still many open questions, and the potential benefit to numerical applications suggests that there are still a few more fertile years left in Chebyshev's cow.

Appendix A

Notation

$:=$	[expr. A] $:=$ [expr. B]	A defined (non-commutatively) as B
t		time-domain variable
ξ		frequency-domain variable
α		parameter of smoothness, unless otherwise indicated
X		A linear space of functions. Safe to assume Banach, possibly Hilbert.
X'		The dual space of X; all bounded linear functionals from $X \longmapsto \mathbb{R}$.

$f[k]$	$:=$	$f(t_k)$	Function taken at discrete values, usually regularly spaced.
ω	$:=$	$2\pi\xi$	radian angular velocity
$\chi_{[a,b]}$	$:=$	1 on $[a,b]$, 0 elsewhere	Characteristic function on $[a,b]$.
$(f(\cdot))_+$	$:=$	$\max(0,f)$.	
$A \asymp B$	$:=$	$C_1 A \leq B \leq C_2 A$	Equivalent up to constants
$\Delta_h^1(f,h)$	$:=$	$f(x+h) - f(x)$	First difference operator
$\Delta_h^r(f,h)$	$:=$	$\Delta_h(f,h)[\Delta_h^{r-1}]$	r-th Difference
$\omega_r(f,t)_p$	$:=$	$\sup_{t \leq h} \|\Delta_h^r\|_p$	r-th Modulus of Smoothness, $r=1$ is Modulus of Continuity
$\sigma_m(f)$	$:=$	$\sigma_m(f,\Phi) := \inf_{\phi \in \Phi} \|f - \phi\|$	error of m-term approximation from Φ in X.
$\sigma_m(\mathcal{F})$	$:=$	$\sigma_m(\mathcal{F},\Phi) := \sup_{f \in \mathcal{F}} \|f - \Phi\|$	error of m-term approx. for class $\mathcal{F}$.
L_p	$:=$	$\left(\int_{\mathbb{R}} \|f(t)\|^p \, dt \right)^{\frac{1}{p}} < \infty$	Membership criteria: Lebesgue p-integrable spaces.
Lipα	$:=$	$\|f(x+t) - f(x)\| \leq Mt^\alpha$	Membership criteria:
	$=$	$\omega(f,t) \leq Mt^\alpha$	Lipschitz (Hölder) space, for $0 < \alpha \leq 1$.

$\mathcal{A}_\tau^o(\mathcal{D}, M)$	$:= \left\{ f \mid \sum_{k\in\Lambda} \lvert c_k\rvert^\tau \le M^\tau \right\}$ $f = \sum_k c_k g,\ g \in \Lambda \subset \mathcal{D}$	Approximation class of τ-summable coefficients.
$B_q^\alpha(L_p(D))$	$:= \left(\int_0^\infty \left[t^{-\alpha} \omega_r(f,t)_p \right]^q \right)^{\frac{1}{q}} < \infty$	Membership criteria: Besov function spaces
Σ_m		The set of m-term approximants, usually a nonlinear manifold.
$L_1 \subset_P L_2$		L_1 is a special case of (*reduces to*) L_2 by a polynomial time algorithm.

Appendix B

Families of Wavelets

Wavelet theory is not really about any particular system of analysis and synthesis of functions. Instead, it is about understanding the requirements and consequences of constructing such a system. By asking "What happens if we require symmetry," or "how can I balance smoothness and time-localization," it has been found that different properties dictate different wavelets.

There are different classes of wavelets, each containing a few families of wavelets. The major wavelets arising through the history of section 1.6 are briefly described here. We work backwards, beginning with the most structured wavelets, which were long sought, and move to the earlier, less structured wavelets. The reason is that the more crude wavelets lack of nice properties is a shortcoming in many settings, but in greedy approximation, we may be able make some use of these wavelets.

B.1 Orthogonal Wavelets with Compact Support

All orthogonal wavelets with compact support are useful outside greedy approximation as they admit the possibility of forming an orthonormal basis for a given space without losing local analysis properties. The generation of such wavelets comes from using techniques found in filter design, namely defining a frequency-domain magnitude response and letting the wavelet be the time-domain impulse response.

The particular techniques that result in compact support in both domains are due to Daubechies, and involved spectral factorization of the magnitude response polynomial, typically a Bernstein polynomial, which results in the desired number of zeros at $\xi = \pi$. For further details, please refer to [12, 61, 2].

B.1.1 Haar / Daubechies wavelets

These wavelets are often considered the "standard" wavelets, because they are the equilibrium point in balancing two key conflicting properties. They are described as orthogonal wavelets with compact support by Daubechies [12, Ch. 6]

With the exception of the Haar wavelet, which is in some ways a degenerate case of a Daubechies wavelet, these wavelets are described by a functional equation, which leads to a FIR filter. There is no explicit expression for the wavelet, but through the iteration of the numerical cascade algorithm, it is possible to generate them quickly.

The Haar/Daubechies wavelets are usually denoted D_N (sometimes "dbN"), where N is a positive integer parameter describing the number of filter coefficients. The Haar wavelet is D_1. Each wavelet has $2N$ filter coefficients, and an interval of support of length $2N - 1$. The multiple $2N$ instead of N is a consequence of FIR filter design conventions. It is possible to do both Discrete and Continuous Wavelet Transforms

using the D_N family wavelets.

Each Daubechies wavelet has $N+1$ vanishing moments, which is to say that

$$\int_{\mathbb{R}} t^j \psi(t)\, dt = 0, \quad j = 0..N-1. \tag{B.1.1}$$

The implication of the number of vanishing moments is that since the equation above looks like the inner-product wavelet transform, all polynomial signals of degree $\leq N-1$ have wavelet coefficients that are zero, and thus such signals are suppressed [53].

Though D_1 is discontinuous, the Daubechies wavelets increase in regularity, in an asymptotic fashion as N grows large. Regularity is a measure of the smoothness of the wavelet, denoted s, which indicates the number of times a signal is continuously differentiable at a point. The regularity over an interval is $\min_{x\in(a,b)} s$. There are further subtle points of regularity, for which the reader is referred to [12, 53, 61]. The Daubechies wavelets approach a regularity of $s = 0.2N$ for large N, so this is not their strongest feature.

Excepting the Haar wavelet, the Daubechies wavelets tend to be highly asymmetric. For some applications, such as numerical analysis, this is not a problem, while for applications such as image coding, the D_N family of wavelets may be unsuitable [12]. Graphs of D_1, D_2, D_4, D_8, and D_{16} are illustrated in figure B.1.

B.1.2 Symlets

Symmetry is desirable for greater compression for a fixed error in a perceptual coding context. It is impossible to have all the nice features of the Daubechies wavelets

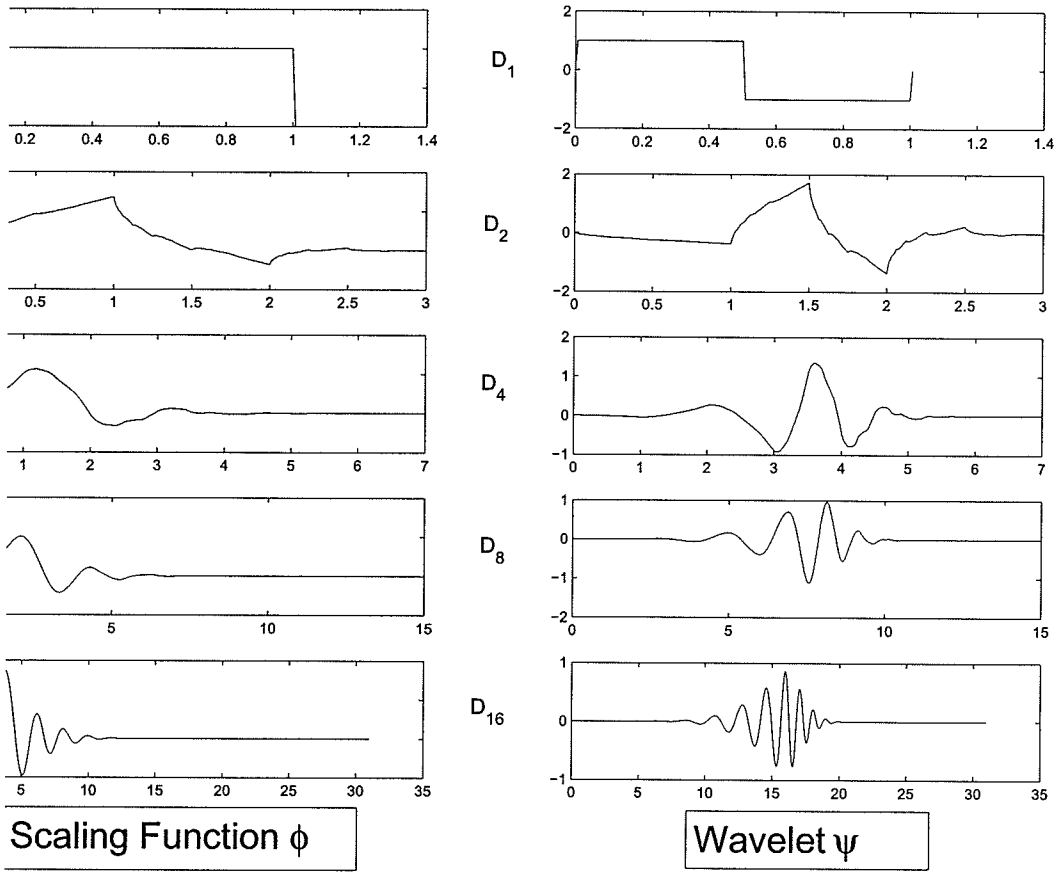

Figure B.1: Daubechies Wavelet Family

and incorporate symmetry as well, but it is possible to make wavelets that are "less asymmetric." Daubechies discusses this and the particulars of finding a spectral factorization of the magnitude response polynomial in order to generate filters (and thus wavelets) that are still orthogonal, yet near to being "linear-phase," which is what engineers often call symmetric filters [12].

Symlets retain most of the other properties of the Daubechies wavelets – orthogonality, compact support, a defined number of vanishing moments. Both the DWT and CWT are possible, the cascade algorithm replaces an explicit expression, and fast

transforms are possible [53]. They are illustrated in figure B.2 for $N = 4, 8, 12$.

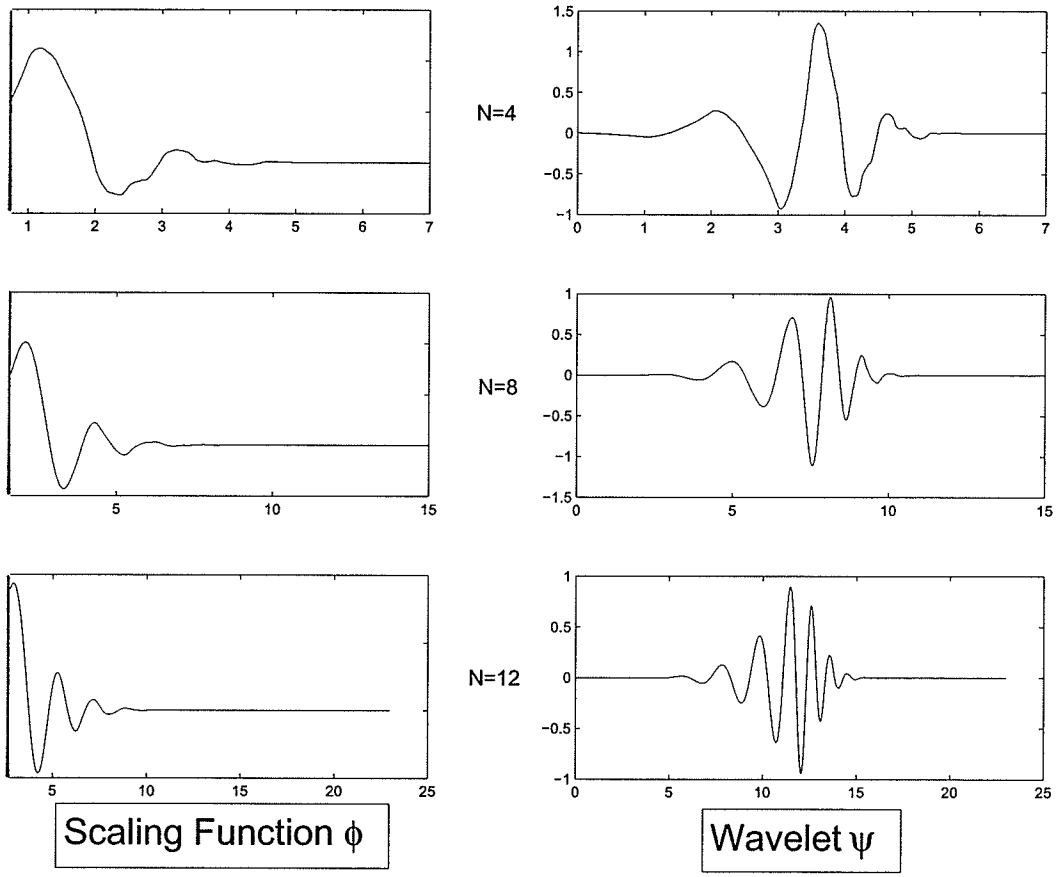

Figure B.2: Symlets Wavelet Family

B.1.3 Coiflets

The family of wavelets known as Coiflets was created by Ingrid Daubechies in pursuing a line of research suggested by Ronald Coifman. His idea was that it might be desirable to have wavelets where not only the wavelet ψ had a certain number of

vanishing moments, but the scaling function ϕ also had a defined number of vanishing moments.

The identifying parameter is often given as K, rather than N. The number of vanishing moments is $2K$, and there is even better symmetry than the Symlets family (figure B.3), but it comes at the expense of a wider support: $6K - 1$, with filters of length $6K$. Again the reader is referred to [12] for details of the construction and [53] for a nice summary of the properties.

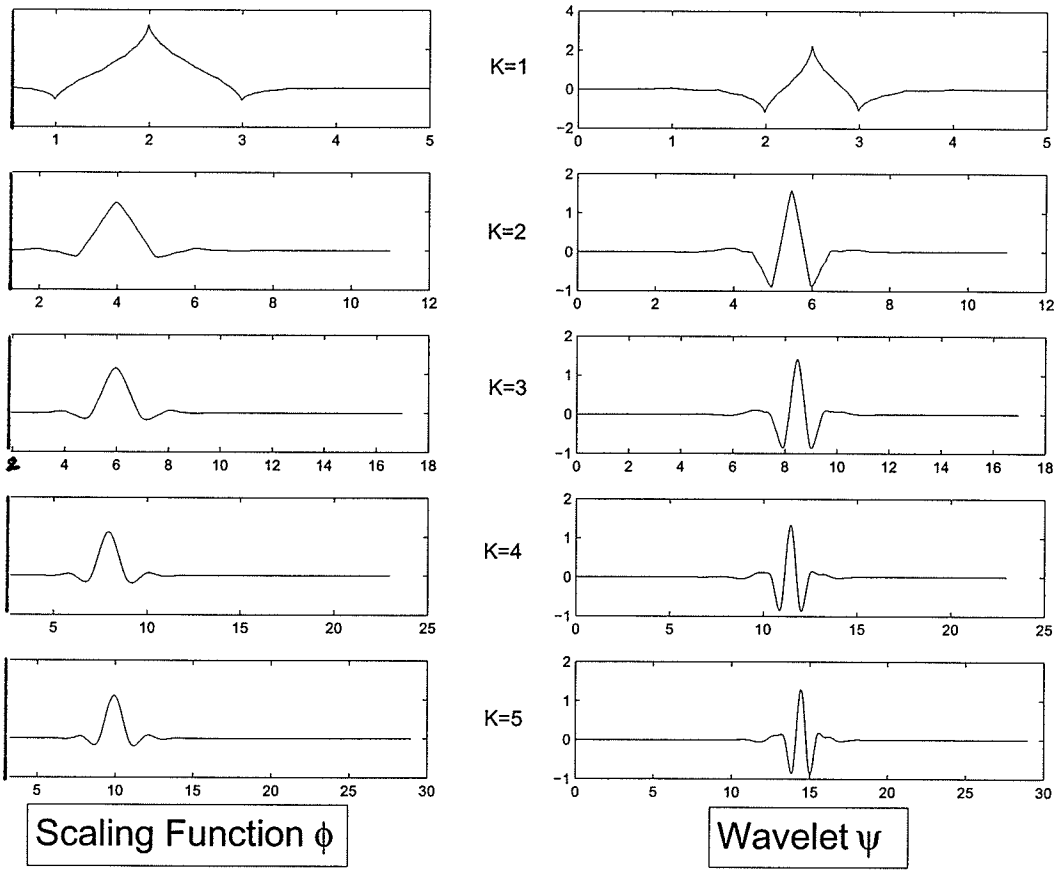

Figure B.3: Coiflets Wavelet Family

B.2 Regular Wavelets

B.2.1 Meyer Wavelets

Predating the Daubechies family of wavelets, Yves Meyer developed a family of wavelets where both ϕ and ψ are in C^∞. They are both symmetric and orthogonal. Unfortunately, they do not have a compact support, thus there is no fast algorithm for doing transforms. The effective support is considered to be $t \in [-8, 8]$.

There is an explicit expression for Meyer wavelets, but it is given in the frequency domain. This expression is found in equation B.2.2. In that equation, ν is a C^k or C^∞ function such that

$$\nu(x) = \begin{cases} 0, & x \leq 0, \\ 1, & x \geq 1 \end{cases} \tag{B.2.1}$$

and where $\nu(t) + \nu(1-t) = 1$ [12].

$$\hat{\psi}(\xi) = \begin{cases} \frac{1}{\sqrt{2\pi}} e^{i\xi/2} \sin\left(\frac{\pi}{2}\nu(\frac{3}{2\pi}|\xi| - 1)\right), & \frac{2\pi}{3} \leq |\xi| \leq \frac{4\pi}{3}, \\ \frac{1}{\sqrt{2\pi}} e^{i\xi/2} \cos\left(\frac{\pi}{2}\nu(\frac{3}{4\pi}|\xi| - 1)\right), & \frac{4\pi}{3} \leq |\xi| \leq \frac{8\pi}{3}, \\ 0 & \text{otherwise} \end{cases} \tag{B.2.2}$$

Daubechies remarks that the proof of orthonormality for the Meyer wavelets relies on some "quasi-miraculous cancellations", which are later explained using a Multiresolution Analysis [12, p. 119].

The wavelet and scaling functions are plotted in figure B.4. In this case, $\nu(t) = x^4(35 - 84x + 70x^2 - 20x^3)$, and different choices of $\nu(t)$ will result in different

wavelets [53].

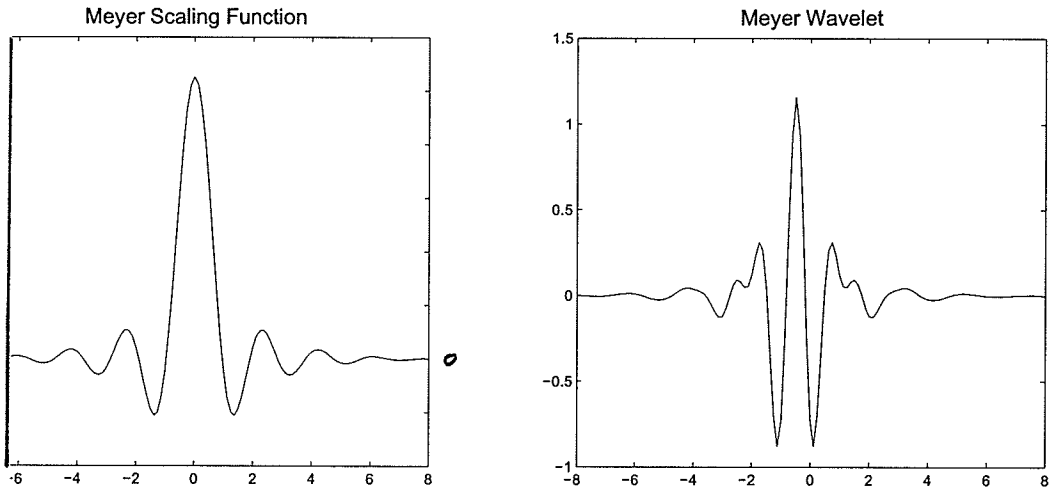

Figure B.4: Meyer Infinitely Regular Wavelets

As mentioned at the beginning of this section, there is no fast transform for analysis by Meyer wavelets. There exists, however, a good approximation of the Meyer wavelet which does allow for the construction of FIR filters, and thus fast transforms. Such a wavelet is called a Discrete Meyer wavelet [53].

B.3 Crude Wavelets

As mentioned in section 1.6, the original definition of a wavelet is a function with an average value of zero, with translations and dilations. While other functions may fit this criteria, there are three wavelets that have been used to some extent: the Gaussian wavelets, the Morlet wavelets, and the Mexican Hat wavelets.

For each of the families, the list of advantages are short. The wavelets are symmetric, and the wavelet ψ can be given explicitly. Continuous variation of the wavelet parameters is possible, and thus a CWT can be performed.

In contrast, the list of disadvantages is somewhat longer.

- There is no scaling function ϕ.
- The wavelets ψ are not orthogonal.
- The wavelet ψ has poor localization (support not compact.)
- There is no fast numerical algorithm for a wavelet transform.

In the theoretical context of greedy approximation, these are not drawbacks at all – any functions will serve as atoms. It is only in practical use that the lack of fast numerical algorithms presents some difficulty. As with the Discrete Meyer Wavelets, it might be possible to use some approximation to the function in order to provide a reasonable candidate atom.

B.3.1 Gaussian Wavelets

The Gaussian function, $f(t) = e^{-t^2}$ is not a wavelet by itself, as $\int_{-\infty}^{\infty} e^{-t^2}\, dt \neq 0$, but for each of derivative $f^{(p)}(t)$, this does hold. The explicit expression for the wavelet is given by

$$f(t) := C_0 e^{-t^2} \tag{B.3.1}$$

$$G_p(t) := C_p f^{(p)}(t),\ p \in \mathbb{N}, \tag{B.3.2}$$

where C_p is a normalizing constant such that $\|G_p(t)\|^2 = 1$.

The support of the Gaussian wavelets is infinite, though for practical purposes the wavelet is truncated and used inside an interval of effective support, e.g. $[-5, 5]$. It is readily apparent that this may result in errors which may be undesirable in some applications.

The second derivative of the Gaussian wavelet, i.e., $G_2(t)$, is also called the "Mexican Hat Wavelet," due to the resemblance it bears to a sombrero. See figure B.5.

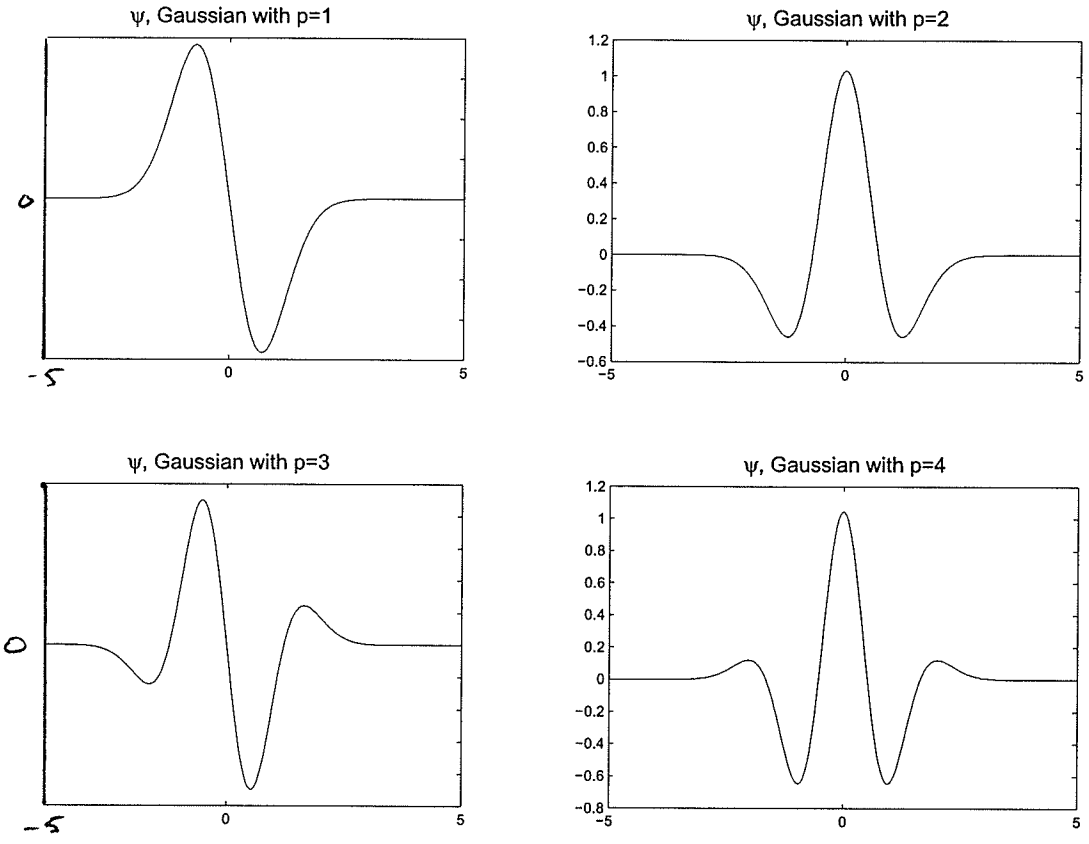

Figure B.5: Wavelets of Gaussian p-th Derivatives

B.3.2 Morlet Wavelets

The Morlet wavelet is, in fact, only nearly a wavelet. It is much closer to a Gabor atom – a modulated Gaussian. It's frequency transform is shifted such that $\hat{\psi}(0) = 0$, and is defined in Daubechies [12] as

$$\begin{aligned} \hat{\psi}(\xi) &= \frac{1}{\sqrt[4]{\pi}} \left[e^{-\frac{(\xi-\xi_0)^2}{2}} - e^{-\frac{\xi^2}{2}} e^{-\frac{\xi_0^2}{2}} \right], \\ \psi(t) &= \frac{1}{\sqrt[4]{\pi}} \left(e^{-i\xi_0 t} - e^{-\frac{\xi_0^2}{2}} \right) e^{-\frac{t^2}{2}}. \end{aligned} \tag{B.3.3}$$

She remarks that in practice it is usual to let $\xi_0 = 5$, which is why the expression for the wavelet is given in [53] as

$$\psi(x) = Ce^{-\frac{x^2}{2}} \cos 5x. \tag{B.3.4}$$

B.4 Omitted wavelets

There are several more types of wavelets which we have not included here, as our interest in wavelets has been secondary to our interest in greedy approximation. Biorthogonal wavelets, where the analysis wavelet and scaling function differ from the synthesis wavelet and scaling functions, have been omitted, as they are not immediately useful in our context. Spline wavelets have also been ignored, as we have been less particular about wavelet properties than some. In addition, we have limited ourselves to single variable, one-dimensional signals, so have not included any discussion of complex wavelets or multi-dimensional wavelets. These are widely used in image processing applications, and beyond the scope of this work.

Bibliography

[1] N.I. Achieser. *Theory of Approximation.* Frederick Ungar, New York, english edition, 1956.

[2] Ali N. Akansu and Richard A. Haddad. *Multiresolution Signal Decomposition.* Academic Press, San Diego, 2nd edition, 2001.

[3] Emmanuel Bacry, et al. LastWave software package. Available via internet at `http://www.cmap.polytechnique.fr/~bacry/LastWave/index.html`, 1997-2004.

[4] John J. Benedetto and Michael W. Frazier, editors. *Wavelets: Mathematics and Applications.* CRC Press, Boca Raton, 1994.

[5] Barbara Burke Hubbard. *The World According to Wavelets.* A.K. Peters, Wellesly, MA, 1996.

[6] N.L. Carothers. A short course on approximation theory. Course notes available on web: `http://personal.bgsu.edu/~carother/Approx.html`, 1998.

[7] E.W. Cheney. *Introduction to Approximation Theory.* McGraw-Hill, New York, 1966.

[8] Clay mathematics institute millenium prize. Web: `http://www.claymath.org/millennium/`, 2000.

[9] R. R. Coifman and M.V. Wickerhauser. Entropy based algorithms for best-basis selection. *I.E.E.E. Trans. Inform. Theory*, 38:713–718, 1992.

[10] J.W. Cooley and J.W. Tukey. An algorithm for the machine calculation of complex fourier series. *Math. Comput.*, 19:297–301, 1965.

[11] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms.* The MIT Press, Cambridge, MA, 2001.

[12] Ingrid Daubechies. *Ten Lectures on Wavelets.* Society for Industrial and Applied Mathematics, Philadelphia, 1992.

[13] L. Daudet. Sparse and structured decompositions of signals with the molecular matching pursuit. To appear, IEEE Trans. on Speech and Audio Proc., Sept. 2006.

[14] G. Davis, S. Mallat, and M. Avellaneda. Adaptive greedy approximations. *Constructive Approximation*, 13:57–98, 1997.

[15] R.A. DeVore and V.N. Temlyakov. Nonlinear approximation by trigonometric sums. *Journal of Fourier Analysis and Applications*, 2(1):29–48, 1995.

[16] R.A. DeVore and V.N. Temlyakov. Some remarks on greedy algorithms. *Advances in Computational Mathematics*, 5:173–187, 1996.

[17] R.A. DeVore and V.N. Temlyakov. Nonlinear approximation in finite-dimensional spaces. *Journal of Complexity*, 13(4):489–508, Dec 1997.

[18] Ronald DeVore, Guergana Petrova, and Vladimir Temlyakov. Best basis selection for approximation in L_p. *Foundations of Computational Mathematics*, 3:161–185, 2003.

[19] Ronald A. DeVore. Nonlinear approximation. In *Acta Numerica*, pages 51–150. Cambridge University Press, 1998.

[20] Ronald A DeVore, Bjorn Jawerth, and Vasil Popov. Compression of wavelet decompositions. *American Journal of Mathematics*, 114(4):737–785, Aug. 1992.

[21] Ronald A. DeVore and George G. Lorentz. *Constructive Approximation.* Springer-Verlag, Berlin, 1993.

[22] S.J. Dilworth, Denka Kutzarova, and Karen L. Shuman. The weak Chebyshev X-greedy algorithm in the unweighted Bergman space. *Journal of Mathmatical Analysis and Applications*, 318:692–706, 2006.

[23] M.J. and Donahue. Rates of convex approximation in non-Hilbert spaces. *Constructive Approximation*, 13(2):187–220, July 1997.

[24] D. Donoho. Unconditional bases are optimal bases for data compression and for statistical estimation. *Applied Computational Harmonic Analysis*, 1:100–115, 1993.

[25] David Donoho, Mark Reynold Duncan, Xiaoming Huo, Ofer Levi, and et al. WaveLab. `http://www-stat.stanford.edu/~wavelab/`, 1999.

[26] Matteo Frigo and Steven G. Johnson. The design and implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231, 2005. special issue on "Program Generation, Optimization, and Platform Adaptation".

[27] D. Gabor. Theory of communication. *JIEE*, 93:429–457, 1946.

[28] M. Ganichev and N.J. Kalton. Convergence of the weak dual greedy algorithm in L_p spaces. *Journal of Approximation Theory*, 124:89–95, 2003.

[29] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman, New York, 1979.

[30] Rémi Gribonval and Emmanuel Bacry. Harmonic decomposition of audio signals with matching pursuit. *I.E.E.E. Transactions on Signal Processing*, 51(1):101–111, Jan. 2003.

[31] Rémi Gribonval and Sacha Krstulović. Matching pursuits tool kit (mptk). Available at `http://mptk.gforge.inria.fr/`, Nov. 2005.

[32] Rémi Gribonval and Sacha Krstulović. MPTK: Matching pursuit made tractable. Poster, I.E.E.E. Conference on Acoustics, Speech, and Signal Processing, May 2006.

[33] William Gropp, Ewing Lusk, and et al. Message passing interface web site. `http://www-unix.mcs.anl.gov/mpi/`, 1995 - present.

[34] R.W. Hamming. *Digital Filters.* Dover, Mineola, NY, 3rd edition, 1989.

[35] Frederick J. Harris. On the use of windows for harmonic analysis with the discrete Fourier transform. *Proceedings of the I.E.E.E.*, 66(1):51–84, January 1978.

[36] M.T. Heideman, D.H. Johnson, and C.S. Burrus. Gauss and the history of the fast Fourier transform. *Archive for History of Exact Sciences*, 34(3):265–277, 1985.

[37] Y.K. Hu, K.A. Kopotun, and X.M. Yu. On multivariate adaptive approximation. *Constructive Approximation*, 16:449–474, 2000.

[38] Stéphane Jaffard, Yves Meyer, and Robert D. Ryan. *Wavelets, Tools for Science and Technology.* SIAM, Philadelphia, 2001.

[39] Lee K. Jones. On a conjecture of Huber concerning the convergence of projection pursuit regression. *Annals of Statistics*, 15(2):365–379, June 1987.

[40] B.S. Kashin. Approximation properties of complete orthonormal systems. *Proc. Steklov Inst. Math*, 1987(3):207–211, 1987.

[41] L. Kolasa, S. Ferrando, and N. Kovačević. wave++ software library (C++). `http://www.scs.ryerson.ca/~lkolasa/CppWavelets.html`, 2000.

[42] S.V. Konyagin and V.N. Temlyakov. Rate of convergence of pure greedy algorithm. *East Journal on Approximation*, 5(4):493–499, 1999.

[43] S.V. Konyagin and V.N. Temlyakov. A remark on greedy approximation in Banach spaces. *East Journal on Approximation*, 5(3):365–379, 1999.

[44] Christopher Lance. Theorem of du Bois-Reymond (1873). Course materials for Fourier Analysis, University of Leeds, `http://www.maths.leeds.ac.uk/~pmt6ecl/MATH3214/duBoisReymond.pdf`.

[45] Peter D. Lax. Abel Prize Acceptance Speech, Trondheim, Norway, `http://www.abelprisen.no/en/nyheter/nyhet.html?id=87`, March 17 2005.

[46] Tom Lehrer. "Lobachevsky". Audio CD: Tom Lehrer Revisited, Reprise, 1990.

[47] G.G. Lorentz. *Approximation of Functions.* Holt, Reinhart, and Winston, New York, 1966.

[48] Richard G. Lyons. *Understanding Digital Signal Processing.* Addison-Wesley, Reading, MA, 1997.

[49] Stéphane Mallat. *A Wavelet Tour of Signal Processing.* Academic Press, San Diego, 1998.

[50] Stéphane Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *I.E.E.E. Transactions on Signal Processing*, 41(12):3397–3415, December 1993.

[51] Stéphane G. Mallat. Multiresolution approximations and wavelet orthonormal bases of $L_2(\mathbb{R})$. *Transactions of the American Mathematical Society*, 315(1):69–87, September 1989.

[52] Jerrold E. Marsden and Michael J. Hoffman. *Basic Complex Analysis.* W.H. Freeman and Co., New York, 1999.

[53] Michel Misiti, Georges Oppenheim, Jean-Michel Poggi, and Yves Misiti. MATLAB Wavelet Toolbox. Run the `waveinfo` command, or see `http://www.mathworks.com/access/helpdesk/help/toolbox/wavelet/`.

[54] J. J. O'Connor and E. F. Robertson. Mactutor history of mathematics archive. `http://www-gap.dcs.st-and.ac.uk/~history/Biographies/Shannon.html`, October 2003.

[55] J.J. O'Connor and E.F. Robertson. Mactutor history of mathematics archive. `http://www-gap.dcs.st-and.ac.uk/~history/Mathematicians/Fourier.html`, Jan. 1997.

[56] Alan V. Oppenheim and Ronald W. Schafer. *Digital Signal Processing.* Prentice-Hall, Englewood, NJ, 1975.

[57] Theodore J. Rivlin. *An Introduction to the Approximation of Functions.* Dover, New York, 1981.

[58] M.B. Ruskai, G. Beylkin, R. Coifman, I. Daubechies, S. Mallat, Y. Meyer, and L. Raphael, editors. *Wavelets and their Applications.* Jones and Bartless, Boston, 1992.

[59] Michael Sipser. *Introduction to the Theory of Computation.* PWS Publishing Co., Boston, 1997.

[60] N.J. Sloane and A.D. Wyner, editors. *Claude Elwood Shannon: collected papers.* I.E.E.E. Press, New York, 1993.

[61] Gilbert Strang and Truong Nguyen. *Wavelets and Filter Banks.* Wellesley-Cambridge Press, Wellesley, MA, 1997.

[62] Robert S. Strichartz. *The Way of Analysis.* Jones and Bartlett, Boston, revised edition, 2000.

[63] Daina Taimina. How to draw a straight line. `http://kmoddl.library.cornell.edu/tutorials/04/`.

[64] V.M. Temlyakov. Greedy algorithm and m-term trigonometric approximation. *Constructive Approximation*, 14:569–587, 1998.

[65] V.N. Temlyakov. Weak greedy algorithms. *Advances in Computational Mathematics*, 12:213–227, 2000.

[66] V.N. Temlyakov. Greedy algorithms in Banach spaces. *Advances in Comp. Mathematics*, 14:277–292, 2001.

[67] V.N. Temlyakov. A criterion for convergence of Weak Greedy Algorithms. *Adv. in Comp. Math.*, 17:269–280, 2002.

[68] V.N. Temlyakov. Cubature formulas and related questions. *Journal of Complexity*, 19:352–391, 2003.

[69] V.N. Temlyakov. Nonlinear methods of approximation. *Foundations of Computational Mathematics*, 3:33–107, 2003.

[70] V.N. Temlyakov. Greedy-type approximation in Banach spaces and applications. *Constructive Approximation*, 21:257–292, 2005.

[71] Anthony Teolis. *Computational Signal Processing with Wavelets.* Birkhäuser, Boston, 1998.

[72] Michael Unser, ed. The wavelet digest (online newsletter). `http://www.wavelet.org/`.

[73] Pierre Vandergheynst. Time-frequency exercises with MATLAB code. `http://ltspc89.epfl.ch/~vandergh/Download/download.html`, 2000.

[74] Martin Vetterli. Filter banks allowing perfect reconstruction. *Signal Processing*, 10(3):219–244, April 1986.

[75] Kurt Vonnegut, Jr. *The Sirens of Titan.* Delacorte Press, New York, 1959.

[76] M.V. Wickerhauser. *Adapted Wavelet Analysis from Theory to Software.* A.K. Peters, 1994.