SOME NEW MULTISCALE METHODS FOR CURVE ESTIMATION AND BINOMIAL DATA

Matthew Alan Nunes

School of Mathematics

May 2006

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of Doctor of Philosophy in the Faculty of Science

Abstract

This thesis explores wavelet techniques for the problem of nonparametric regression in situations other than the classical model.

The first part of the thesis attempts to address function estimation for non-standard settings by introducing an adaptive wavelet *lifting scheme*. This construction enables the basis in the resulting wavelet transform to be automatically tailored to the specific characteristics of a function, thus offering better signal prediction as well as compression properties. The performance of this algorithm is assessed thoroughly, through extensive simulations and application to real data. The results show that the algorithm performs well compared to traditional techniques.

Another aspect of wavelet transforms generated through the lifting scheme is also considered, namely the primary resolution level or "stopping time" in the transform wavelet basis. The study aims to provide further improvements to the lifting algorithm through a judicious choice of this resolution level. However, our procedure for automatically choosing the stopping time does not produce good results for this difficult problem.

The second part of the thesis focusses on the area of wavelet methods for binomial data. We propose a transformation to Gaussianize and variance-stabilize a sequence of Bin(n,p) binomially distributed data, where the parameter p varies over the sequence. The transformation proves to show desired properties and good results are obtained through simulations.

Acknowledgements

Thanks to my supervisor, Professor Guy Nason for his enthusiasm and guidance over the last three years.

Special thanks has to go to my colleague and good friend, Marina Knight. It has been a sometimes difficult road, but we made it! I look forward to working with you more in the coming years.

I would like to thank my friends Chris, Geoff, Ian, Martin, Huw and Simon for making the office a fun place to work. I would also like to thank Jim for the constant supply of hearty meals and nights out over the years!

Lastly, I would like to thank my parents and family for their constant support and encouragement. It is impossible to say how much I appreciate it.

Thank you!

Author's Declaration

I declare that the work in this thesis was carried out in accordance with the Regulations of the University of Bristol. The work is original except where indicated by special reference in the text and no part of the dissertation has been submitted for any other degree. Any views expressed in the dissertation are those of the author and do not necessarily represent those of the University of Bristol. The thesis has not been presented to any other university for examination either in the United Kingdom or overseas.

Matthew Alan Nunes

Date: May 2006

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Wavelets and their applications have only started to be fully developed within the last twenty years, created in the mid eighties to solve geophysical problems, though their mathematical background has its origins further back in history. Their applications are now widespread, lying in the fields of science and technology, with data compression, image and signal denoising, time series and numerical analysis to name but a few. Wavelets can be viewed as building blocks for functions in $L^2(\mathbb{R})$. They are useful because of their simplicity and their ability to represent functions with sparse expansions.

Wavelets have been applied to the classical problem of nonparametric regression for many years. Early work in the area of thresholding the Discrete Wavelet Transform by Donoho and Johnstone [39] has now become a popular approach to take. The general idea is to wavelet transform the data, threshold the resulting wavelet coefficients and then invert the transform. This has been seen to be an effective way to estimate a signal from noisy input. However, the DWT assumes some restrictive conditions on the observations of an underlying signal, namely that the data is regularly-spaced and that the length of the dataset is a power of two. In addition, the data is often assumed to have a unique observation for each x-value. The case of multiple observations at x-values is often overlooked. In real-life situations, the assumptions made by the DWT are unrealistic; experimental data are often irregularly distributed and

of any length. Certain datasets, for example, the motorcycle data introduced by Silverman [89] do not have the unique correspondence between x-values and observations.

Chapter 2 reviews the literature in the area of wavelet shrinkage. We introduce basic wavelet ideas, including Mallat's Discrete Wavelet Transform algorithm. We give an overview of some techniques for nonparametric regression, and discuss the *lifting scheme*, a method of constructing wavelets and wavelet transforms suitable for situations involving departures from the assumptions of the DWT.

In Chapter 3, we propose an adaptive wavelet transform based on the "one coefficient at a time" lifting methodology of Jansen *et al.* [59, 60]. When using classical wavelet transforms, it is often necessary to specify the wavelet family which is used to decompose a signal. The best family to use is unknown, a priori. Our method attempts to overcome this disadvantage by tuning each individual wavelet to the data, thus automatically choosing the properties of the decomposing wavelets to suit the situation in hand. We also modify our lifting algorithm so that it is suitable for multiple point data.

An investigation into the sparsity and denoising capability of the adaptive lifting transform is provided in Chapter 4 in the form of a simulation study. The transform is compared to other wavelet and non-wavelet regression techniques for the now standard test functions of Donoho and Johnstone. We also apply our methodology to two real-life datasets.

The next chapter discusses the effect of the primary resolution level on our algorithm's performance. Simulations establish that there is indeed a variation in denoising ability at different resolution levels. A method of automatic choice of the primary resolution is tested to try to give improvements to the lifting transforms.

The work of Chapter 6 explores the problem of binomial proportion estimation over a space. The transformation of observed data to bring them closer to being Gaussian is one technique used in such problems. Fryźlewicz

and Nason [51] have introduced, for Poisson data, a successful Gaussianizing and variance-stabilizing algorithm based on an asymptotic result by Fisz [47] and the properties of the Haar wavelet transform. However, for binomial random variables, the Fisz result loses the ability to have constant variance, although interesting distributional features are discussed as an aside to the main focus of the chapter. Motivated by this observation, we propose an alternative transform to restore the constant variance property. The properties of this transform are compared with that of the traditional arcsine Gaussianizing transformation by Anscombe [6]. We introduce an equivalent preprocessing algorithm to the Haar-Fisz transform, and test the properties of the method against Anscombe; the algorithm for binomial proportion estimation is applied to a real dataset.

Motivated by the work in Chapter 6, a short examination of Fisz-transformed binomial random variables in given in Chapter 7. The discussion focusses on the exact probability mass function of the random variables ζ^1 and ζ^0 for fixed binomial sizes, and alternative forms for efficient computation of probabilities for different values of the random variables.

Chapter 8 summarizes the work of this thesis and outlines ideas for further research.

Chapter 2

Literature Review

Introduction

This chapter reviews aspects of the literature which are essential to the other chapters of this thesis. Section 2.1 gives an overview of wavelets from a theoretical point of view, discussing their background and general construction. Section 2.2 explores the lifting scheme, which forms the basis of work in Chapters 3 and 5. Section 2.3 reviews the ideas of thresholding and nonparametric regression, which are used throughout the thesis.

2.1 An Overview of Wavelet Theory

2.1.1 Fourier Series

Before introducing wavelets and wavelet bases, it is useful to give a few comments on their predecessors, *Fourier series*. In classical Fourier analysis, sine and cosine waves are used to form bases for functions in $L^2(\mathbb{R})$. In the discussion below, we will restrict our attention to functions in the Hilbert space $L^2([0, 2\pi))$, that is, the space of all *square integrable* functions over the interval $[0, 2\pi)$. In other words, if $f \in L^2([0, 2\pi))$, then $\int_0^{2\pi} f^2(t)\ dt$ is finite. For a more detailed discussion of Fourier series and their properties, refer to [101] or

[48]. We first recall the definition.

Definition 2.1. *Let f be square integrable over the interval $[0, 2\pi)$ and periodic with period 2π. Then the* **Fourier series** *representation of f is*

$$f(x) = \frac{a_0}{2} + \sum_n a_n \cos(nx) + b_n \sin(nx), \quad n \in \mathbb{Z}$$

where the Fourier coefficients are calculated from

$$a_n = \frac{1}{\pi} \int_0^{2\pi} f(x) \cos(nx)\ dx, \qquad b_n = \frac{1}{\pi} \int_0^{2\pi} f(x) \sin(nx)\ dx, \quad n \in \mathbb{Z}.$$

Note that the Fourier basis $\{\cos(nx), \sin(nx)\}_{n\in\mathbb{Z}}$ is orthonormal, and that the Fourier coefficients are calculated using the L^2 inner product. When using Fourier expansions, we would like to have efficient expansions, in the sense that only a few of the coefficients are non-zero. However, although trigonometric functions are localized in frequency, they are non-local in time. This means that, typically, many basis functions contribute to function reconstruction at any one point. As a result, they cannot approximate "sharp" parts of functions well: in order to represent singularities, we want the basis functions to have short support. One answer to this problem is *wavelets*.

Simply put, a wavelet is a small wave which decays rapidly, and whose translations and dilations form an orthonormal basis of $L^2(\mathbb{R})$. In this latter respect, they are similar to Fourier series, but locality can be achieved in both the time and scale domains simultaneously. Hence they can represent functions with discontinuities well.

2.1.2 Multiresolution Analysis

This section is devoted to the notion of a *multiresolution analysis* (MRA), first proposed by Mallat [72]. This mathematical structure is important in the construction of wavelets and wavelet bases, such as the Daubechies wavelets [35].

The MRA provides a way of analyzing signals by zooming in or out to examine the function in detail or in an overall sense. It brings out the ideas of filters associated with wavelet bases, and leads onto the *Discrete Wavelet Transform*, which will be useful later in this literature review and is fundamental to the work in the next chapters. Our aim is to construct an orthonormal wavelet basis for $L^2(\mathbb{R})$. We follow the method of working as in [35].

Definition 2.2. *A* **multiresolution analysis** *(MRA) of $L^2(\mathbb{R})$ is a system of closed subspaces $\{V_j\}$ of $L^2(\mathbb{R})$ such that:*

1. $\ldots V_{-1} \subset V_0 \subset V_1 \subset \ldots$
2. $\overline{\bigcup_{j\in\mathbb{Z}} V_j} = L^2(\mathbb{R})$, *i.e.* $\bigcup_{j\in\mathbb{Z}} V_j$ *is dense in* $L^2(\mathbb{R})$
3. $\bigcap_{j\in\mathbb{Z}} V_j = \{0\}$
4. $f(x) \in V_0 \iff f(2^j x) \in V_j$
5. $f \in V_0 \iff f(x-k) \in V_0, \forall\ k \in \mathbb{Z}$
6. $\exists\ \varphi \in V_0$, *called a* **scaling function** *or* **father wavelet**, *such that* $\{\varphi_{0,k} := \varphi(t-k)\}_{k\in\mathbb{Z}}$ *is an orthonormal basis in* V_0.

The last assertion means that any function $f \in V_0$ can be written as a linear combination $f = \sum_k c_k\, \varphi_{0,k}$, where the coefficient c_k is just the inner product $\langle f, \varphi\rangle_{L^2(\mathbb{R})} = \int_{-\infty}^{\infty} f(x)\, \varphi(x-k)\, dx$.

Definition 2.3. *A multiresolution analysis is said to be* **of order N** *if polynomials of degree up to degree $N-1$ can be written as a linear combination of scaling functions, i.e. for $0 \le p \le N-1$, $\exists\, c_k \in \mathbb{R}$ such that*

$$x^p = \sum_k c_k \varphi(x-k).$$

Note that the spaces V_j are just scaled versions of V_0 from 4. The statements 4 and 5 imply that for each j, the set $\{\varphi_{j,k} := 2^{j/2}\varphi(2^j x - k)\}_{k\in\mathbb{Z}}$ is

an orthonormal basis for the space V_j. The functions $\varphi_{j,k}$ are known as the *translations* and *dilations* of the function φ.

Let P_j denote the projection onto V_j. From the projection

$$(P_j f)(x) = \sum_k c_{j,k}\, \varphi_{j,k}(x),$$

where

$$c_{j,k} = \langle f, \varphi_{j,k} \rangle_{L^2(\mathbb{R})} = \int_{-\infty}^{\infty} f(x)\, \varphi_{j,k}(x)\ dx,$$

every function in $L^2(\mathbb{R})$ can be approximated by elements of the subspaces V_j, and as $j \to \infty$, the precision of this approximation increases (from conditions 2 and 3).

Let us return to our scaling function, φ. Since $\varphi \in V_0 \subset V_1$, φ can be written as $\varphi(x) = \sum_{k\in\mathbb{Z}} h_k \varphi_{1,k}(x)$ where $h_k = \langle \varphi, \varphi_{1,k} \rangle$, since $\{\varphi_{1,k}\}_{k\in\mathbb{Z}}$ is an orthonormal basis for V_1. In other words,

$$\varphi(x) = \sqrt{2} \sum_{k\in\mathbb{Z}} h_k \varphi(2x - k). \tag{2.1}$$

Equation (2.1) is known as the *scaling equation*. The coefficients $\{h_k\}_{k\in\mathbb{Z}}$ are referred to as the *(low-pass) filter associated with* φ.

We also obtain

$$\begin{aligned} \langle \varphi_{j-1,\ k}, \varphi_{j,\ n} \rangle &= \int \varphi_{j-1,k}(x)\ \varphi_{j,n}(x)\ dx \\ &= \int \sqrt{2}\ \varphi(t)\ \varphi(2t + 2k - n)\ dt \\ &= h_{n-2k}, \end{aligned} \tag{2.2}$$

by using the substitution $t = 2^{j-1}x - k$. Using $\{\varphi_{j,n}\}_{n\in\mathbb{Z}}$ as an orthonormal

basis for V_j, and by considering $\varphi_{j-1,k}$ as an element of V_j, we have

$$\begin{aligned} \varphi_{j-1,k}(x) &= \sum_{n\in\mathbb{Z}} \langle \varphi_{j-1,k}, \varphi_{j,n} \rangle \ \varphi_{j,n}(x) \\ &= \sum_{n\in\mathbb{Z}} h_{n-2k} \varphi_{j,n}(x). \end{aligned} \tag{2.3}$$

This equation is known as the scaling function *refinement relation.*

Wavelet Functions

For each j, consider the orthogonal complement of V_j in V_{j+1}, and denote it by W_j. Since the V_j are closed subspaces, we have $V_{j+1} = V_j \oplus W_j$. This is true for each j, so for $j > J$,

$$V_{j+1} = V_J \oplus \bigoplus_{k=0}^{j-J} W_{J-k}, \tag{2.4}$$

and hence from the properties 2 and 3 in the definition of a MRA, we obtain

$$L^2(\mathbb{R}) = \bigoplus_{j\in\mathbb{Z}} W_j. \tag{2.5}$$

So an L^2-function can be split up into mutually exclusive parts, each part being in one of the *detail subspaces* W_j. This is achieved by the orthogonal projection onto the subspace in question; we can define Q_j as the orthogonal projection onto the subspace W_j.

For each W_j, the scaling property 4 is inherited:

$$f(x) \in W_0 \iff f(2^j x) \in W_j.$$

Let $\psi \in W_0$. Since $\psi \in W_0 \subset V_1$, there is an expression corresponding to the scaling equation:

$$\psi(x) = \sqrt{2} \sum_k g_k \varphi(2x - k).$$

The coefficients $\{g_k\}_{k \in \mathbb{Z}}$ are called the *high-pass filter (associated with ψ).*

Define a family of functions $\{\psi_{j,k}\}_{j,k \in \mathbb{Z}}$ in the same way as $\{\varphi_{j,k}\}$ above. Then the large set $\{\psi_{j,k}\}_{j,k \in \mathbb{Z}}$ forms a basis of $L^2(\mathbb{R}) \iff \{\psi_{j,k}\}_{k \in \mathbb{Z}}$ forms an orthonormal basis for W_j for fixed $j \iff \exists\ \psi \in W_0$ (called a *mother wavelet*) such that $\{\psi(\cdot - k)\}_{k \in \mathbb{Z}}$ is an orthonormal basis of W_0. So a key development of wavelet theory was the need to find this mother wavelet, ψ.

From examining the structure of multiresolution analyses, for example the Fourier properties of father wavelet function expansions, one can prove the following theorem. For the proof, see [35].

Theorem 2.4. *Let $\{V_j\}$ be a multiresolution analysis of $L^2(\mathbb{R})$ with scaling function φ. Then $\exists$ an L^2 function ψ, called the* **mother wavelet** *such that $\{\psi_{j,k}\}$ is an orthonormal wavelet basis of W_j for fixed j. Furthermore, one such function is given by*

$$\psi(x) = \sqrt{2} \sum_{k \in \mathbb{Z}} h_{1-k} (-1)^k \varphi(2x - k).$$

Filters linked as above by the relation $g_k = (-1)^k h_{1-k}$ are called *quadrature mirror filters.* For a more detailed discussion of wavelet filters, see [94].

By following the same argument as with the scaling functions, we derive a similar refinement relation to equation (2.3),

$$\psi_{j-1,k}(x) = \sum_{n \in \mathbb{Z}} g_{n-2k}\ \varphi_{j,n}(x). \tag{2.6}$$

Vidakovic [100] shows that these filters satisfy the internal orthogonality

relations

$$\sum_{n\in\mathbb{Z}} h_n\, h_{n-2k} = \delta_{0,k} \quad \text{and} \quad \sum_{n\in\mathbb{Z}} g_n\, g_{n-2k} = \delta_{0,k}, \tag{2.7}$$

and the combined orthogonality relation

$$\sum_{n\in\mathbb{Z}} h_n\, g_{n-2k} = 0. \tag{2.8}$$

Remark. The conditions in Definition 2.2 require the integer translates of the scaling function φ to form an orthonormal basis for V_0. However, Property 5 can be relaxed. We start with a definition.

Definition 2.5. *Let $g \in L^2(\mathbb{R})$. Then the system of functions $\{g(x-k)\}_{k\in\mathbb{Z}}$ forms a* **Riesz basis** *if $\exists\ A, B > 0$ such that*

$$A\sum_{i\in\Lambda} \lambda_i^2 \leq \left\| \sum_{i\in\Lambda} \lambda_i g(x-i) \right\|^2 \leq B\sum_{i\in\Lambda} \lambda_i^2, \tag{2.9}$$

for any $\Lambda \subset \mathbb{Z}$ and any coefficient sequence $(\lambda)_k \in l^2(\mathbb{R})$ (i.e. with $\sum_{k\in\mathbb{Z}} |\lambda_k|^2 < \infty$).

For orthonormality of the functions $\{g(x-k)\}_{k\in\mathbb{Z}}$, the quantities A and B should be equal and one.

From a Riesz basis, it can be shown that a new basis can be constructed which is an orthonormal basis for V_0 (see [58], [35]). Using this basis, the rest of the MRA construction follows. So, provided that the integer translates $\{\varphi(\cdot - k)\}_{k\in\mathbb{Z}}$ (span the space V_0 and) form a Riesz basis for V_0, a wavelet basis for $L^2(\mathbb{R})$ can be found.

Example 2.6. Having seen how to obtain wavelets through the MRA process, we know that we can construct many wavelets, just depending on the scaling function used in the MRA which generated them. The wavelets generating

the orthonormal bases of $L^2(\mathbb{R})$ can be chosen to possess different properties, required for the situations in which they are to be used, for example:

- fast asymptotic decay
- vanishing moments (see below)
- compact support
- symmetry

We now describe some examples of wavelets. Both of these wavelets will appear later in this thesis. For details of other wavelet bases, see for example [100].

Haar wavelet

The Haar wavelet is the simplest example of a wavelet. The Haar scaling function is given by $\varphi^H(x) = \mathbb{I}_{[0,1)}$, the indicator function taking the value 1 on the interval [0,1) and zero elsewhere. It is clear that the set of integer translates $\{\varphi^H_{0,k}\}$ of φ^H form an orthonormal basis since they are indicator functions on disjoint unit intervals along the real line. If we define $\varphi^H_{j,k}$ as above, it can be shown that

$$\varphi^H(x) = 2^{-1/2}\varphi^H_{1,0}(x) + 2^{-1/2}\varphi^H_{1,1}(x),$$

so that the comparing this to the scaling equation in equation (2.1), we see that the Haar low pass filter is

$$h_k = \begin{cases} 2^{-1/2} & \text{for } k = 0, 1 \\ 0 & \text{otherwise.} \end{cases}$$

Using the quadrature mirror filter relation, we have

$$g_k = \begin{cases} 2^{-1/2} & \text{for } k = 0, \\ -2^{-1/2} & \text{for } k = 1, \\ 0 & \text{otherwise.} \end{cases}$$

This results in the Haar mother wavelet $\psi^H = \mathbb{I}_{[0,1/2)} - \mathbb{I}_{(0,1]}$, i.e.

$$\psi^H = \begin{cases} 1 & \text{on } [0,1/2) \\ -1 & \text{on } [1/2,1) \\ 0 & \text{otherwise.} \end{cases}$$

Figures 2.1 and 2.2 show the Haar scaling function and wavelet. The MRA structure for the Haar wavelet can be seen if we define the approximation subspaces V_j by

$$V_j = \{f \in L^2(\mathbb{R}) |\ f \text{ is constant on } [2^{-j}k, 2^{-j}(k+1)) \ \forall k \in \mathbb{Z}\}.$$

Any function in $L^2(\mathbb{R})$ can be approximated arbitrarily well by linear combinations of Haar wavelets. At each scale, there is a unique subinterval corresponding to each wavelet $\psi_{j,k}$, defined by its support. These intervals partition the real line. Due to the dyadic structure of the spaces V_j, the intervals nest as the scale increases.

Daubechies' Extremal Phase Wavelets

Definition 2.7. *We say that a wavelet ψ has $N+1$* **vanishing moments** *if*

$$\langle x^k, \psi(x) \rangle = \int x^k \, \psi(x) \, dx = 0 \quad \textit{for } k \in \{0, \ldots, N\}. \tag{2.10}$$

The vanishing moment property of a wavelet affects its smoothness: the

more vanishing moments it has, the smoother it is (see [35] for more details). Note that if the MRA associated with a wavelet is of order N (Definition 2.3), then the associated wavelet automatically possesses N vanishing moments due to orthogonality: for $0 \leq p \leq N-1$, we have

$$\langle x^p, \psi(x) \rangle = \left\langle \sum_k c_k \varphi(x-k), \psi(x) \right\rangle = \sum_k c_k \langle \varphi(x-k), \psi(x) \rangle = 0.$$

Vanishing moments lead to sparse function representations, since they imply that expansion coefficients will be small or zero on smooth (e.g. like higher order polynomial) parts of signals, and conversely, will be large at points of discontinuity. A wavelet with N vanishing moments is sometimes referred to as a wavelet *of order N*.

Named after a significant contributor to wavelet theory, the *Daubechies' Extremal Phase wavelets*, D_N, have the minimum support possible for a given number of vanishing moments, N. The scaling functions have support $[0, 2N-1]$ and the (mother) wavelets have support $[1-N, N]$. They are very asymmetric in appearance, apart from when $N = 1$, when the Haar wavelet (above) is produced. A few of the mother and father wavelets from this family are shown in Figures 2.1 and 2.2. The compact support of the basis functions can be easily seen. Note also that the smoothness of the wavelets and scaling functions increases as N (the number of vanishing moments) does. The construction of these wavelets as well as computation of the associated filters can be found in [35] and [100].

2.1.3 The Discrete Wavelet Transform

From relation (2.5) and property 2 in the definition of a MRA, by projection onto the resolution subspace V_{j_0}, one can see that functions in $L^2(\mathbb{R})$ can be

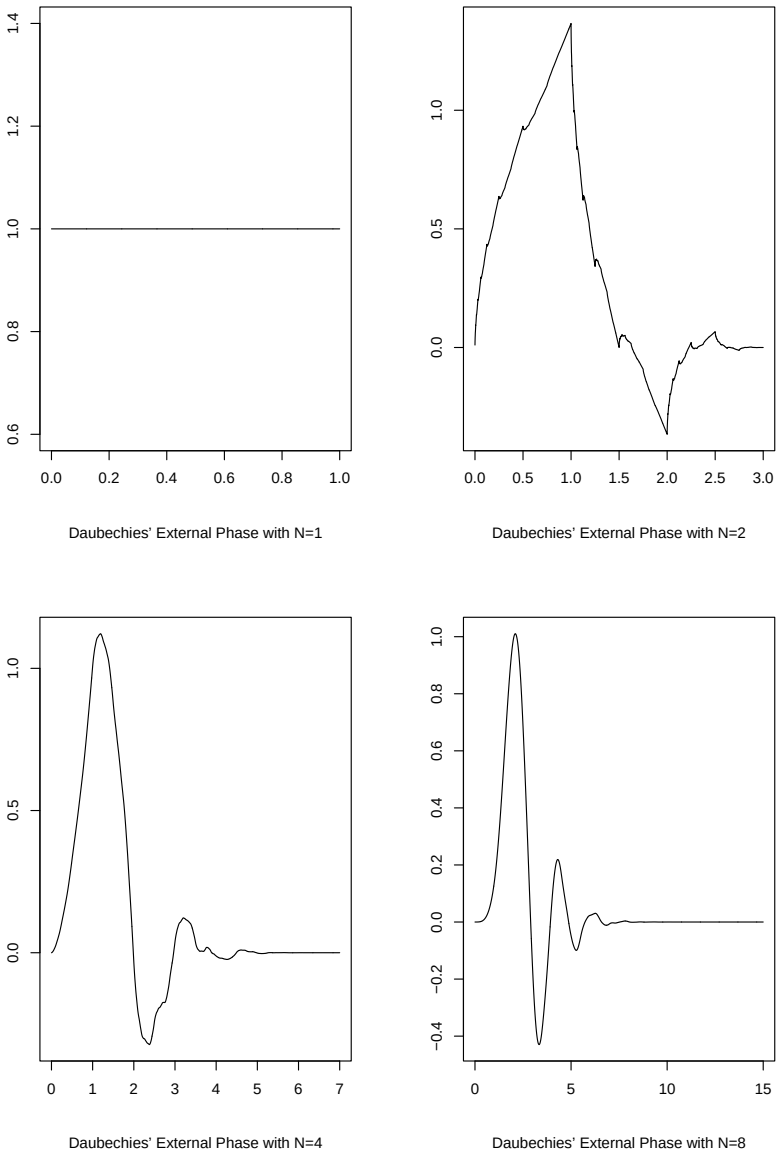

Figure 2.1: Examples of scaling functions (father wavelets) from the Daubechies' Extremal Phase wavelet family. Top left: Haar scaling function, D_1; top right: D_2; bottom left: D_4; bottom right: D_8.

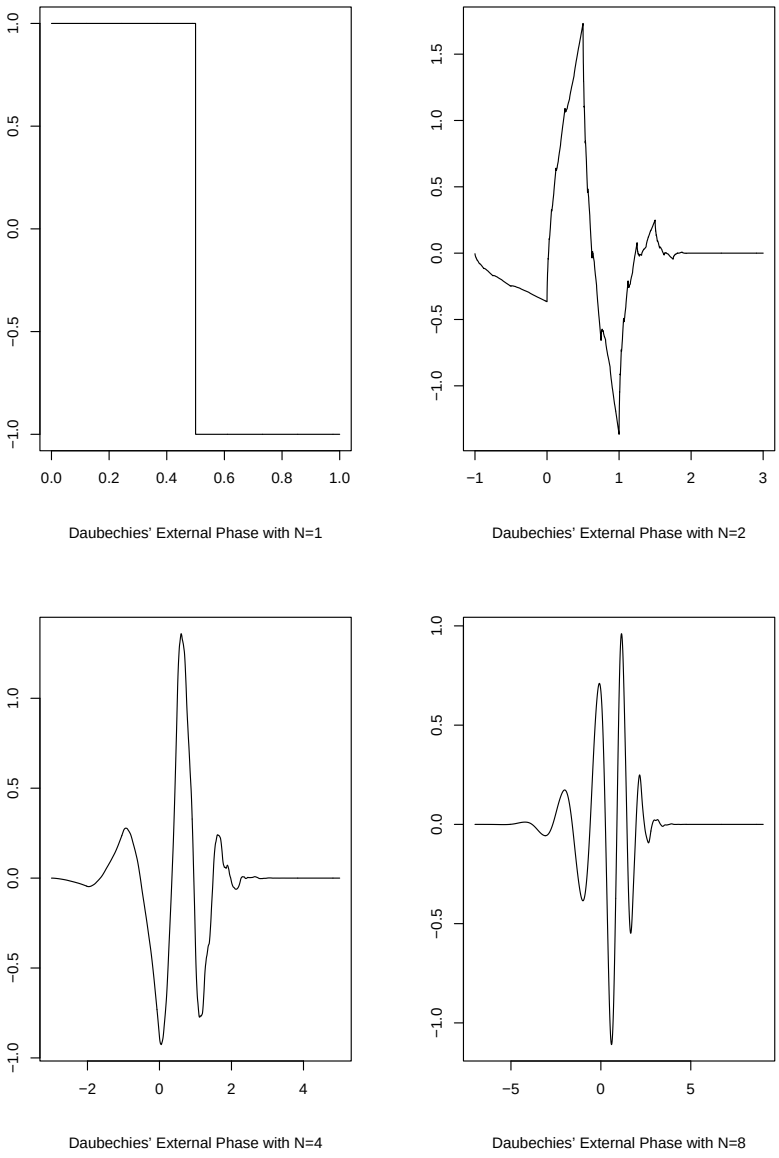

Figure 2.2: Examples of (mother) wavelets from the Daubechies' Extremal Phase wavelet family. Top left: Haar wavelet, D_1; top right: D_2; bottom left: D_4; bottom right: D_8.

represented as

$$f(x) = \sum_{j \geq j_0} \sum_{k \in \mathbb{Z}} d_{j,k} \psi_{j,k}(x) + \sum_{k \in \mathbb{Z}} c_{j_0,k} \phi_{j_0,k}(x), \tag{2.11}$$

where the coefficients $d_{j,k}$ and $c_{j_0,k}$ are calculated via the L^2 inner product. These coefficients are referred to as *detail or wavelet coefficients* and *scaling coefficients* respectively. The detail coefficients give local information about the function f at scale 2^j and location $2^{-j}k$. The number j_0 is called the *primary resolution level*. This decomposition parameter will be discussed in more detail later in this thesis.

For efficient representation, we want *sparse* wavelet expansions, in the sense that we want as many $d_{j,k}$ as possible to be zero, so that fewer wavelets are needed to approximate f. These representations are particularly useful in compression applications, where efficient data storage is a major consideration.

In most practical applications, the function we want to decompose is sampled, for example, from experimental data. In this section, we introduce Mallat's *discrete wavelet transform* (DWT), which is used to characterize sampled functions as a set of wavelet coefficients. It is similar to the Fast Fourier Transform for discrete data. The DWT comprises of two filters, namely the *low-pass* and *high-pass* filters, corresponding to the coefficients h_k and $g_k := (-1)^k h_{1-k}$ in the last section.

Equation (2.3) from the last section leads us to have the following recursive

relation between function expansion coefficients:

$$\begin{aligned} c_{j-1,k} &= \langle f, \varphi_{j-1,k} \rangle \\ &= \langle f, \sum_{n \in \mathbb{Z}} h_{n-2k} \varphi_{j,n} \rangle \\ &= \sum_{n \in \mathbb{Z}} h_{n-2k} \langle f, \varphi_{j,n} \rangle \\ &= \sum_{n \in \mathbb{Z}} h_{n-2k} c_{j,n}. \end{aligned} \tag{2.12}$$

A similar argument starting from equation (2.6) results in the relation

$$d_{j-1,k} = \sum_{n \in \mathbb{Z}} g_{n-2k} \; c_{j,n}. \tag{2.13}$$

In other words, if we have the scaling coefficients at a particular resolution level, we can obtain the detail and scaling coefficients at the next (coarser) level by implementation of formulae (2.12) and (2.13). These relations are the decomposition steps of the DWT. The coefficients can be seen as being *filtered* by h and g. Note that for compactly supported wavelets, for example Daubechies' Extremal Phase wavelets, the associated filters are finite, and so these sums are also finite. In fact for this wavelet family, the filter length is twice the number of vanishing moments.

From the definition of the detail subspace W_{j-1} and the projection operators P_{j-1} and Q_{j-1}, we can reverse the decomposition process and reconstruct the finer scaling coefficients:

$$\begin{aligned} (P_j f)(x) &= (P_{j-1} f)(x) + (Q_{j-1} f)(x) \\ &= \sum_{l \in \mathbb{Z}} c_{j-1,l} \; \varphi_{j-1,l}(x) + \sum_{l \in \mathbb{Z}} \; d_{j-1,l} \psi_{j-1,l}(x) \\ &= \sum_{l \in \mathbb{Z}} c_{j-1,l} \left(\sum_{n \in \mathbb{Z}} h_{n-2l} \varphi_{j,n}(x) \right) + \sum_{l \in \mathbb{Z}} d_{j-1,l} \left(\sum_{n \in \mathbb{Z}} g_{n-2l} \varphi_{j,n}(x) \right), \end{aligned}$$

And then by expressing the projection in the basis of V_j, i.e. $(P_j f)(x) =$

$\sum_{k\in\mathbb{Z}} c_{j,k}\ \varphi_{j,k}(x)$ and equating coefficients of $\varphi_{j,l}$, we have

$$c_{j,k} = \sum_{l\in\mathbb{Z}} c_{j-1,l} h_{k-2l} + \sum_{l\in\mathbb{Z}} d_{j-1,l} g_{k-2l}. \tag{2.14}$$

This is known as the reconstruction step of the DWT. It gives a method of inverting the DWT through repeated application of the equation (2.14).

Let us now assume that a function is sampled at $n = 2^J$ regularly-spaced sites, t_k. Let us denote the function values by the vector $\mathbf{f}$. As explained above, if we have the fine scale coefficients $c_{J,k}$ for fixed J, we can calculate all wavelet and scaling coefficients on resolution levels $j < J$. The algorithm for the computation of the wavelet and scaling coefficients is referred to as Mallat's *pyramidal* or *cascade* algorithm.

However, since we only know the function of interest at the sites t_k, in reality we will not be able to calculate the finest scaling coefficients $c_{J,k}$. Hence we need to approximate them in some way. We set the finest scaling coefficients to be the function values, so we have

$$c_{J,k} = f(t_k) = f_k \quad \text{for } k \in \{0, \ldots, 2^J - 1\}.$$

If we now define a function

$$\overline{f}(x) = \sum_k c_{J,k} \varphi_{J,k}(x),$$

since $\overline{f}$ is clearly in the approximation space V_J, the multiresolution analysis formalism is appropriate here. Note that this function essentially approximates the function $P_j f$ by approximating the inner products $\langle f, \varphi_{J,k}\rangle$ with the sampled function values $f(t_k)$; the functions $\varphi_{J,k}$ are compactly supported and localized around t_k, so for large J, the coefficients will be approximated by the function at the point t_k. We call these coefficients the *empirical scal-*

ing coefficients. The expansion coefficients can then be calculated recursively using the DWT decomposition relations (2.12) and (2.13). Denote the vectors of scaling coefficients and detail coefficients at each resolution level by $\mathbf{c_j}$ and $\mathbf{d_j}$ (respectively) for $j \leq J$.

To recap: each filter h and g produces a new vector of length $n/2$, which correspond to the "smooth" and "detail" (respectively) of the data, since c_{J-1} is a smoothed version of the original, whereas d_{J-1} can be viewed as the "detail lost" by the low-pass filter. This is a natural idea – every function in the present function space can be expressed as a combination of a mother wavelet (detail) basis expansion and a father wavelet (smooth) basis expansion. This process is repeated to only the smoothed data, until we have transformed the original sequence into the augmented data vector

$$\mathbf{d} := \mathrm{DWT}(\mathbf{f}) = (\mathbf{c_{j_0}}, \mathbf{d_{j_0}}, \mathbf{d_{j_0+1}}, \ldots, \mathbf{d_{J-1}}). \tag{2.15}$$

The DWT is implemented in S-Plus by the software package *WaveThresh* maintained by Nason[†].

A full decomposition of a vector of length n would have the first coefficient as $\mathbf{c_0}$ (so that j_0, the primary resolution level, is 0). Since the one element vector $\mathbf{c_0}$ is the result of repeated application of the low-pass filter h, it can be seen as a measure of "global mean" of the function f. Note that the new vector $\mathbf{d}$ also has length n. The total number of operations needed to transform the data is nn_f, where n_f is the length of the filter associated to a particular wavelet. In terms of computation, this is fast, compared to the Fast Fourier Transform equivalent of $n \log n$.

[†]Nason, G.P. (1998) WaveThresh3 Software. Department of Mathematics, University of Bristol, Bristol, UK. This software is available from http://www.stats.bris.ac.uk/~wavethresh/ .

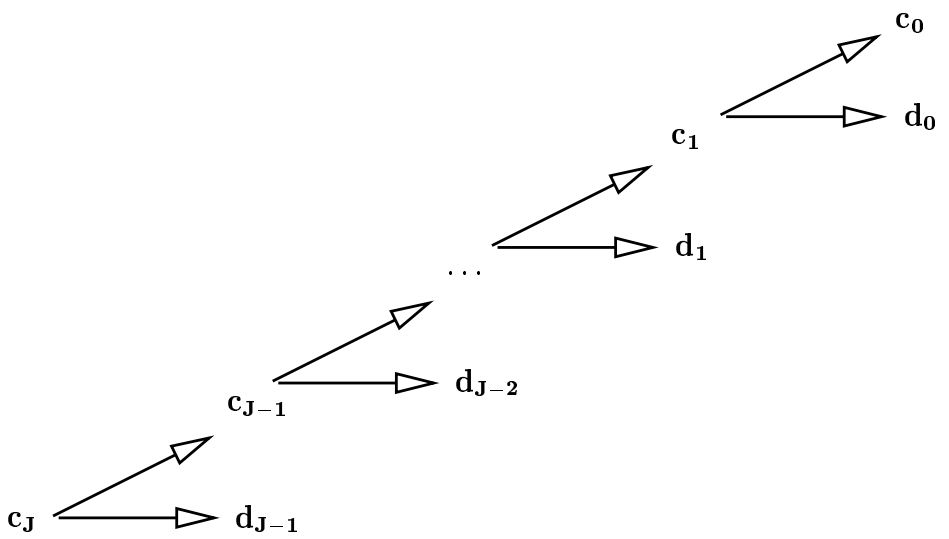

Figure 2.3: Schematic of Mallat's pyramidal algorithm. Each arrow represents a filter convolution followed by (even) dyadic decimation.

Operator notation and Decimation

The filter sequences h and g are sometimes written in operator form on the sequence space l^2. In other words, we define convolution functions $\mathcal{H}, \mathcal{G} : l^2 \longrightarrow l^2$, such that $\forall s \in l^2$,

$$(\mathcal{H}s)_k = \sum_n h_{n-k} s_n \tag{2.16}$$

and

$$(\mathcal{G}s)_k = \sum_n g_{n-k} s_n. \tag{2.17}$$

Consider also the operator $\mathcal{D}_0$, which takes every even element of a sequence s, i.e.

$$(\mathcal{D}_0 s)_k = s_{2k}. \tag{2.18}$$

This operation is known as *dyadic decimation*. One could also define $\mathcal{D}_1$ to keep every odd element of a sequence $s \in l^2$.

Note that if we compose the operators $\mathcal{D}_0$ with $\mathcal{G}$ or $\mathcal{H}$ and apply it to the

sequence c_j, this is equivalent to the decomposition relations (2.3) and (2.6):

$$\begin{aligned}(\mathcal{D}_0(\mathcal{H}\mathbf{c_j}))_k &= \left(\mathcal{D}_0\left(\sum_n h_{n-k}c_{j,n}\right)\right)_k \\ &= \sum_n h_{n-2k}c_{j,n} \\ &= c_{j-1,k},\end{aligned}$$

and

$$\begin{aligned}(\mathcal{D}_0(\mathcal{G}\mathbf{c_j}))_k &= \left(\mathcal{D}_0\left(\sum_n g_{n-k}c_{j,n}\right)\right)_k \\ &= \sum_n g_{n-2k}c_{j,n} \\ &= d_{j-1,k}.\end{aligned}$$

Thus the DWT filtering steps (2.3) and (2.6) can also be implemented by the filter convolutions (2.16) and (2.17) followed by decimation at each step of the transform.

Matrix representation

As well as the cascade algorithm, we can represent the DWT as a matrix equation

$$\mathbf{d} = W\mathbf{f}, \tag{2.19}$$

where W is an orthogonal matrix. Recall that in fact we define the coarse scaling coefficients to be the function values contained in $\mathbf{f}$, so the argument in the equation above can also be taken as $\mathbf{c_J}$. The matrix is orthogonal due to the orthonormality of the wavelet bases from the associated MRA. The matrix W can be seen as a change of basis matrix from the basis $\{\varphi_{J,k}\}_{k\in\{0,\ldots,2^J-1\}}$ of V_j to the basis $\{\varphi\}\cup\{\psi_{j,k}\}_{j\in\{0,\ldots,J-1\},k\in\{0,\ldots,2^j-1\}}$ of the full decomposition of the DWT.

DWT Boundary conditions

It often arises that the wavelet filter used in a DWT decomposition reaches outside the range of the data being decomposed. This does not happen with the Haar wavelet, but is likely with Daubechies' wavelets. Several methods have been proposed in the literature to deal with this issue. Nason and Silverman [76] suggest some choices for handling boundary problems:

symmetry. The function data is reflected at the endpoints, to extend further than the original sampled function vector.

periodic. The function to be decomposed is taken to be periodic on the range of the data, so that $\mathbf{f}_{-N+k} = \mathbf{f}_{N+k} = \mathbf{f}_k$ for $k \in \{0, \ldots, N-1\}$.

zero padding. The function values are assumed to be zero outside the range of the vector $\mathbf{f}$.

Other solutions to this issue could be used. Cohen *et al.* [27] designs wavelets specifically for the interval, which are *wrapped* outside the boundaries of the interval [0,1]. The *lifting scheme*, which is introduced in the next section and explored extensively in the following chapters of this thesis, can deal with boundary issues.

2.1.4 Other wavelet transforms

Non-decimated wavelet transform

Coifman and Donoho [28] and also Nason and Silverman [78] explore the *non-decimated* or *stationary* wavelet transform (NDWT). This transform uses the convolution operators above, but does not decimate at any step of the DWT. Since a vector of function values can be completely represented using only the augmented vector (2.15), the NDWT leads to an *overcomplete* wavelet basis for an input signal, due to not decimating. Recall that the decimation operators $\mathcal{D}_0$ and $\mathcal{D}_1$ are defined to take the even or odd elements of an input

sequence respectively. In the normal DWT, we use $\mathcal{D}_0$ to decimate at each stage. An alternate (and equally valid) way to decimate could be to use $\mathcal{D}_1$ instead. Nason and Silverman show that the non-decimated discrete wavelet transform is strongly linked to the *ε-decimated* discrete wavelet transform. In this variant of the DWT, a choice is chosen of how to decimate at each level of the transform. Let ε have the binary representation

$$\varepsilon = \varepsilon_0 \varepsilon_1 \dots \varepsilon_{J-1}.$$

Then for each coefficient vector $\mathbf{c_j}$, the number ε_{j-1} is used to determine whether to decimate with $\mathcal{D}_0$ or $\mathcal{D}_1$. It can be shown that every coefficient produced by an ε-decimated transform can be found in the coefficient sets resulting from the NDWT. For more details, see [78].

Wavelet packet transform

Wavelet packets are discussed in the papers by Coifman and Wickerhauser [29]. These transforms, as well as using the smooth vectors $\mathbf{c_j}$ to produce new decomposition coefficients, use the wavelet coefficient vectors $\mathbf{d_j}$. This also results in a redundant (overcomplete) set of decomposition coefficients. A subset of these coefficients are then selected to form a representation of the function being decomposed. Coifman [29] provides a method for searching the coefficient subsets for the "best basis".

Biorthogonal wavelets

Biorthogonal wavelets were first introduced in Cohen *et al.* [26]. Different wavelets are used for the decomposition and reconstruction transforms. These wavelets exhibit greater symmetry, whilst still keeping compact support. The price, however, is that orthogonality is lost, and hence the wavelets satisfy biorthogonality relations. The multiresolution analysis framework is reviewed in the biorthogonal setting in Section 2.2. The lifting scheme [95, 96], which

appears in Section 2.2, is a method of creating biorthogonal wavelet bases with desired properties.

Multidimensions

The multiresolution analysis framework described above in Section 2.1.2 is for L^2 functions on the real line. However, Mallat [72] shows the same ideas can be extended to d dimensions, by considering wavelets $\psi \in L^2(\mathbb{R}^d)$. However, for practical examples, it is easier to construct multidimensional MRA structures by using the tensor product of existing one-dimensional multiresolution analyses. For example in two dimensions, suppose we define a subspace $\mathbf{V_0} = V_0 \otimes V_0$, and the orthogonal subspace $\mathbf{W_0}$, similarly to above. Then through the properties of tensor products, the two dimensional space inherits the scalable and orthogonal structure of the original MRA. This results in orthonormal bases for orthogonal subspaces $\mathbf{W_j}$ in an analogous way to Theorem 2.4. This construction is often used in applications such as image analysis.

2.2 The Lifting Scheme

We have seen in the last section that wavelets provide a useful alternative to Fourier transforms for decomposing functions, and through the DWT we have a mathematical theory with which to construct wavelet bases for use on sampled signals. However, there are limitations of classical wavelet methods.

In practical settings, for example for finite length signals (on intervals, say), there is the problem of boundary handling. As suggested in the previous section, there are corrections which could remedy this. However, the methods described can be disadvantageous: if we pack out the signal at the boundary with zeros, this increases the number of coefficients to decompose using the transform; periodizing the signal over its range interval creates the possibility of boundary discontinuities (if the values of the signal at the beginning and

end of the interval do not match).

Also, the classical structure does not extend easily to more general settings, such as irregularly-sampled data. Irregular data methods have previously involved a modification of usual wavelet procedures, such as using pre-processed data, which we would ideally like to avoid.

There are other settings in which classical wavelet transforms are unsuitable or problematic. For example, recent work on bounded domains, curves and surfaces appears in the literature [87, 88]. Wavelets are normally adapted to these non-standard situations. Since the main focus of this thesis is to explore nonparametric regression techniques for non-equispaced data, these so-called *second generation wavelets*, used in general situations, will provide the mathematical tools for work in the later chapters.

In this section, the *lifting scheme* [95, 96] is introduced. First developed by Sweldens and others in the mid-nineties, it is a method now commonly used to build general wavelet transforms with desired properties, which are also computationally efficient.

Initially we present multiresolution analysis, considered in the last section, but from the more general second generation wavelet viewpoint. The lifting scheme will follow naturally from this theoretical background. A more intuitive and practical approach to lifting is then outlined.

2.2.1 Multiresolution analysis for second generation settings

The name *lifting scheme* comes from the fact that we first begin with a very simple multiresolution analysis and then add more and more wavelet properties until we have the desired structure for the situation in hand. Second generation wavelets are no longer generated by one single (mother) wavelet. Our approach follows [96].

We work on a general L^2 space. Let $L^2 = L^2(X, \Sigma, \mu)$ be a measure space,

with $X \subset \mathbb{R}$, Σ a σ-algebra, and mu a measure on $Sigma$.

Definition 2.8. *A* **multiresolution analysis** *(MRA) of L^2 is a system of closed subspaces $\{V_j | j \in \mathcal{J} \subset \mathbb{Z}\}$ of L^2 such that:*

- $V_j \subset V_{j+1}$
- $\overline{\bigcup_{j\in\mathcal{J}} V_j} = L^2$, *i.e* $\bigcup_{j\in\mathcal{J}} V_j$ *is dense in* L^2
- *for each $j \in \mathcal{J}$, V_j has a Riesz basis (see 2.9), given by functions $\{\varphi_{j,k} | k \in \mathcal{K}_j\}$ for some index set $\mathcal{K}_j$. Assume $\mathcal{K}_j \subset \mathcal{K}_{j+1}$.*

Sweldens considers the possible choices for $\mathcal{J}$ of $\mathbb{N}$ and $\mathbb{Z}$, but for the purposes of this exposition, and to relate ideas to the MRA wavelet construction in Section 2.1.2, we will impose $\mathcal{J} = \mathbb{Z}$. In this case, there is the additional condition corresponding to Property 3 of Definition 2.2:

$$\bigcap_{j\in\mathcal{J}} V_j = \{0\}.$$

We want to keep the properties of first generation wavelets, but also relax the conditions on the setting we work in. This is where second generation wavelets come in. However, this flexibility comes at a price; the cost of the extra flexibility to handle irregularity and bounded domains is in the loss of translation and dilation of a single wavelet function. In general, orthogonality of bases uses a lot of degrees of freedom and limits symmetric analysis and synthesis systems. Hence properties of an orthogonal MRA are hard to achieve - for example, the Haar basis forms the only orthogonal MRA for which the (real) wavelets are compactly supported and symmetric. Instead, we consider *biorthogonal* bases. As well as the *primal* MRA above, we now define a *dual* multiresolution analysis with corresponding subspaces $\widetilde{V}_j$ and scaling functions $\widetilde{\varphi}_{j,k}$. These dual scaling functions satisfy the biorthogonality relation

$$<\widetilde{\varphi}_{j,k}, \varphi_{j,k'}>= \delta_{k,k'} \quad \text{for } k, k' \in \mathcal{K}_j, \tag{2.20}$$

where $\delta_{k,k'}$ is the Kronecker delta symbol.

Similar to the first generation setting, we define the *order* of the primal and dual MRA to be N if any polynomial of degree at most $N-1$ can be written as a linear combination of $\{\varphi_{0,k}\}_{k\in\mathcal{K}_0}$ or $\{\widetilde{\varphi}_{0,k}\}_{k\in\mathcal{K}_0}$ respectively.

Since $\varphi_{j,k} \in V_j \subset V_{j+1}$ and $\{\varphi_{j+1,k}\}_{k\in\mathcal{K}_j}$ is a Riesz basis for V_{j+1}, we have the refinement relation

$$\varphi_{j,k}(x) = \sum_{l\in\mathcal{K}_{j+1}} h_{j,k,l}\ \varphi_{j+1,l}(x). \tag{2.21}$$

Note that now the filters have three subscripts; the coefficients are both scale ($j \in \mathcal{J}$) and position ($k \in \mathcal{K}_j$) dependent. We get a similar equation from the dual MRA

$$\widetilde{\varphi}_{j,k}(x) = \sum_{l\in\mathcal{K}_{j+1}} \widetilde{h}_{j,k,l}\ \widetilde{\varphi}_{j+1,l}(x). \tag{2.22}$$

In practice, the filters above are finite, and so the sets $\mathcal{L}_{j,k} = \{l \in \mathcal{K}_{j+1} |\ h_{j,k,l} \neq 0\}$ and $\mathcal{K}_{j,k} = \{k \in \mathcal{K}_j |\ h_{j,k,l} \neq 0\} = \{k \in \mathcal{K}_j | l \in \mathcal{L}_{j,k}\}$ are finite and uniformly bounded. For the rest of this discussion, we assume that the filters h (and later g) are finite.

Recall that in the first generation wavelet case, each scaling function and wavelet has a unique interval associated to it (for one-dimensional functions). Due to the scaling property of the MRA, each interval is split into two equal subintervals when going from the level j to the level $j+1$. We now define the analogue of this dyadic structure for the second generation setting. We follow Sweldens [96].

Definition 2.9. *a* **set of partitionings** *is a set of measurable subsets* $\{S_{j,k} |\ j \in \mathcal{J}, k \in \mathcal{K}(j)\}$ *such that*

1. $\forall j \in \mathcal{J},\ \overline{\bigcup_{k\in\mathcal{K}(j)} S_{j,k}} = X$

2. $\mathcal{K}(j) \subset \mathcal{K}(j+1)$

3. $S_{j+1,k} \subset S_{j,\tilde{k}}$ *for some* $\tilde{k} \in \mathcal{K}(j)$

4. *For fixed* $k \in \mathcal{K}(j_0)$, $\bigcap_{j>j_0} S_{j,k} = \{x_{j,k}\}$

In other words, every interval associated to a scaling function is wholly contained in an interval at the next (coarser) level. Further, the points $x_{j,k}$ defined by the last property are just the points which divide X into the disjoint partition in the first condition. These points are known as *interpolating points*. Note that the number of interpolating points increases as the resolution level increases. The method used to decide where the interpolating points occur is called a *subdivision scheme*. In the Haar wavelet case, the interpolating points divide the interval $[0,1]$ by successively splitting subintervals into equal parts, resulting in the intervals $[2^{-j}k, 2^{-j}(k+1))$ for fixed j and $k \in \mathbb{Z}$ (see Example 2.6). In the next chapter, we introduce algorithms which use sets of interpolating points to define scaling functions associated to subintervals of the real line. Sweldens [96] describes how to use a filter and a set of partitionings to construct scaling functions and dual scaling functions satisfying the refinement relations (2.21) and (2.22).

Wavelet functions

In the first generation MRA, the orthogonal complement of the approximation subspaces V_j was used to define wavelet bases for $L^2(\mathbb{R})$. From [96], we have the equivalent definition.

Definition 2.10. *A set of functions* $\{\psi_{j,m}\}_{j\in\mathcal{J},\ m\in\mathcal{M}_j}$, *where* $\mathcal{M}_j := \mathcal{K}_{j+1} \setminus \mathcal{K}_j$ *is a* **set of wavelet functions** *if*

1. *The space* $W_j := \overline{\text{span}\{\psi_{j,m}\}_{m\in\mathcal{M}_j}}$ *is the complement of* V_j *in* V_{j+1} *and* $W_j \perp \widetilde{V}_j$

2. *The set* $\left\{\frac{\psi_{j,m}}{\|\psi_{j,m}\|}\right\}_{j\in\mathcal{J},\ m\in\mathcal{M}_j}$ *is a Riesz basis for* L^2 ‡

We define the dual wavelet $\widetilde{\psi}_{j,m}$ similarly for the complement spaces $\widetilde{W}_j$ orthogonal to the primal approximation subspaces V_j. The orthogonality conditions mean that the wavelets form a biorthogonal system, in the sense that

$$\langle \psi_{j,m}, \psi_{j',m'} \rangle = \delta_{j,j'}\delta_{m,m'}. \tag{2.23}$$

From the filters, (primal and dual) wavelet refinement relations can be derived of the form

$$\begin{aligned} \psi_{j,m} &= \sum_l g_{j,m,l}\ \varphi_{j+1,l} \quad \text{and} \\ \widetilde{\psi}_{j,m} &= \sum_l \widetilde{g}_{j,m,l}\,\widetilde{\varphi}_{j+1,l}. \end{aligned} \tag{2.24}$$

Again, we assume that the filters are finite, which leads us to define the sets $\mathcal{L}_{j,m} = \{l \in \mathcal{K}_{j+1} |\ g_{j,m,l} \neq 0\}$ and $\mathcal{M}_{j,m} = \{m \in \mathcal{M}_j |\ g_{j,k,l} \neq 0\}$.

From the refinement relations, the filters $\{h, g, \widetilde{h}, \widetilde{g}\}$ can be seen to be a set of *biorthogonal filters* :

Definition 2.11. *A set of filters* $\{h, g, \widetilde{h}, \widetilde{g}\}$ *is said to be a* **set of biorthogonal filters** *if the following equations hold*

$$\begin{aligned} \sum_l h_{j,k,l}\,\widetilde{h}_{j,k',l} &= \delta_{k,k'} \qquad & \sum_l g_{j,m,l}\,\widetilde{h}_{j,k,l} &= 0, \\ \sum_l g_{j,m,l}\,\widetilde{g}_{j,m',l} &= \delta_{k,k'} \qquad & \sum_l h_{j,k,l}\,\widetilde{g}_{j,m,l} &= 0. \end{aligned}$$

Note that the relations in this definition are analogous to the equations (2.7) and (2.8).

‡This condition is for when $\mathcal{J} = \mathbb{Z}$. There is a similar condition for when $\mathcal{J} = \mathbb{N}$, which is that the set $\left\{\frac{\psi_{j,m}}{\|\psi_{j,m}\|}\right\}_{j\in\mathcal{J},\ m\in\mathcal{M}_j} \cup \left\{\frac{\varphi_{0,k}}{\|\varphi_{0,k}\|}\right\}_{k\in\mathcal{K}_0}$ is a Riesz basis for L^2.

Similar to the first generation case, we can define the wavelets and dual wavelets to have vanishing moments.

Definition 2.12. *The wavelets and dual wavelets have* **N (resp. $\widetilde{\mathrm{N}}$) vanishing moments** *if for any polynomial P_p of degree p with $0 \leq p \leq N-1$ (resp. $\widetilde{N}-1$) we have*

$$\begin{aligned} \langle P_p, \psi_{j,m} \rangle_{L^2} &= 0 \quad \textit{for } j \in \mathcal{J}, m \in \mathcal{M}_j \quad \textit{or} \\ \langle P_p, \widetilde{\psi}_{j,m} \rangle_{L^2} &= 0 \quad \textit{for } j \in \mathcal{J}, m \in \mathcal{M}_j. \end{aligned}$$

Following the considerations of Example 2.6, the order of the primal and dual MRA are linked with the number of vanishing moments of the primal and dual wavelets.

Now suppose $f \in L^2$. Then f can be expressed as

$$f(x) = \sum_{j,m} d_{j,m}\, \psi_{j,m}(x),$$

where the expansion coefficients are now $d_{j,m} = <f(x), \widetilde{\psi}_{j,m}(x)>$.

2.2.2 Fast wavelet transform

As in the normal DWT, there is an equivalent fast method for the computation of scaling and wavelet coefficients from the scaling coefficients at a finest chosen scale. From the filter refinement relations, we obtain the *forward* recursions

$$c_{j,k} = \sum_{l \in \widetilde{L}_{j,k}} \widetilde{h}_{j,k,l}\, c_{j+1,l}, \quad \text{and} \tag{2.25}$$

$$d_{j,m} = \sum_{l \in \widetilde{L}_{j,m}} \widetilde{g}_{j,m,l}\, c_{j+1,l}, \tag{2.26}$$

and the *inverse* transform recursion

$$c_{j+1,l} = \sum_{k \in \mathcal{K}_{j,l}} h_{j,k,l}\, c_{j,k} + \sum_{m \in \mathcal{M}_{j,l}} g_{j,m,l}\, c_{j,m} \tag{2.27}$$

for the finite filter sets defined in the text. As before, this allows us to start with the coefficients at one level, and calculate coefficients at all coarser levels by implementing the equations (2.25) and (2.26). Note that the filters $\widetilde{h}$ and $\widetilde{g}$ are for the decomposition of a function, and h and g for reconstruction. The four filters can be associated with the four operator functions $\{\mathcal{H}, \widetilde{\mathcal{H}}, \mathcal{G}, \widetilde{\mathcal{G}}\}$ as before.

Remark. As well as the method of working in this section, the second generation wavelets and MRA framework can be developed from an alternative viewpoint. Daubechies [35], and Cohen *et al.* [26] work through the construction of (dual) multiresolution analyses, but instead starting with four biorthogonal filters, subject to certain conditions (biorthogonal filters do not necessarily give rise to biorthogonal functions – see Theorem 3 in [26]).

2.2.3 The lifting scheme

We now introduce the lifting scheme, which will become very useful in later discussions. We restrict ourselves to finite filters, so as to ensure compactly supported wavelets. Our description follows that of Sweldens [96].

Theorem 2.13 (The Lifting Scheme). *Let* $\{h^{\text{old}}, \widetilde{h}^{\text{old}}, \widetilde{g}^{\text{old}}, g^{\text{old}}\}$ *be a set of (finite) biorthogonal filters. Then another set of biorthogonal filters* $\{h, \widetilde{h}, g, \widetilde{g}\}$

can be found from

$$\begin{aligned} h_{j,k,l} &= h^{\text{old}}_{j,k,l} \\ \widetilde{h}_{j,k,l} &= \widetilde{h}^{\text{old}}_{j,k,l} + \sum_m s_{j,k,m} \widetilde{g}^{\text{old}}_{j,m,l} \\ g_{j,m,l} &= g^{\text{old}}_{j,m,l} - \sum_k s_{j,k,m} h^{\text{old}}_{j,k,l} \\ \widetilde{g}_{j,m,l} &= \widetilde{g}^{\text{old}}_{j,m,l}. \end{aligned}$$

From the remark, we can use these new filters to construct a pair of new multiresolution analyses. Using these filters, the scaling functions and wavelets are modified in the following way:

$$\varphi_{j,k} = \varphi^{\text{old}}_{j,k} \tag{2.28}$$

$$\widetilde{\varphi}_{j,k} = \sum_l \widetilde{h}^{\text{old}}_{j,k,l} \widetilde{\varphi}_{j+1,l} + \sum_m s_{j,k,m} \widetilde{\psi}_{j,m} \tag{2.29}$$

$$\psi_{j,m} = \psi^{\text{old}}_{j,m} - \sum_k s_{j,k,m} \varphi^{\text{old}}_{j,k} \tag{2.30}$$

$$\widetilde{\psi}_{j,m} = \sum_l \widetilde{g}^{\text{old}}_{j,m,l} \widetilde{\varphi}_{j+1,l}. \tag{2.31}$$

This modification of the wavelet and scaling functions is called *primal lifting*, since the primal scaling function is left unchanged. Note that in primal lifting, instead of using scaling functions on the next finer level to express the new wavelet (as in equation (2.24)), scaling functions and a wavelet on the *same* level are used: we construct the new wavelet by simply subtracting translates of the scaling function from the old wavelet. Note also that the dual functions change. From the theorem, the filter $\widetilde{g}$ remains the same after primal lifting. However, the dual wavelet produced in equation (2.31) changes, since it is built from scaling functions which have been modified through equation (2.29).

Through appropriate choice of the coefficients $\{s_{j,k,m}\}_{j\in\mathcal{J},k\in\mathcal{K}_j,m\in\mathcal{M}_j}$, we can build wavelets with desired properties, with the equation (2.30). Recall

that a wavelet has N vanishing moments if equation (2.12) holds. Using equation (2.30), this translates into choosing the coefficients $\{s_{j,k,m}\}$ such that $\langle P_p, \psi_{j,m}^{\text{old}} \rangle = \sum_k s_{j,k,m} \langle P_p, \varphi_{j,k}^{\text{old}} \rangle$.

Sweldens notes that in imaging applications, for example other properties, such as the shape of the resulting wavelet, could also be chosen by specifying the coefficients.

As well as performing lifting on the primal MRA, the dual wavelets can be lifted also. In the *dual lifting scheme*, the dual low-pass filter $\widetilde{h}$ and the primal high-pass filter g remain the same. The filters $\{h^{\text{old}}, \widetilde{h}^{\text{old}}, \widetilde{g}^{\text{old}}, g^{\text{old}}\}$ are transformed into a set of new (finite) biorthogonal filters $\{h, \widetilde{h}, g, \widetilde{g}\}$ by the formulae

$$\begin{aligned} h_{j,k,l} &= h_{j,k,l}^{\text{old}} + \sum_m \widetilde{s}_{j,k,m} g_{j,m,l}^{\text{old}} \\ \widetilde{h}_{j,k,l} &= \widetilde{h}_{j,k,l}^{\text{old}} \\ g_{j,m,l} &= g_{j,m,l}^{\text{old}} \\ \widetilde{g}_{j,m,l} &= \widetilde{g}_{j,m,l}^{\text{old}} - \sum_k \widetilde{s}_{j,k,m} \widetilde{h}_{j,k,l}^{\text{old}}. \end{aligned}$$

and so the wavelet and scaling functions are changed by

$$\varphi_{j,k} = \sum_l h_{j,k,l}^{\text{old}} \varphi_{j+1,l} + \sum_m \widetilde{s}_{j,k,m} \psi_{j,m} \tag{2.32}$$

$$\widetilde{\varphi}_{j,k} = \widetilde{\varphi}_{j,k}^{\text{old}} \tag{2.33}$$

$$\psi_{j,m} = \sum_l g_{j,m,l}^{\text{old}} \varphi_{j+1,l} \tag{2.34}$$

$$\widetilde{\psi}_{j,m} = \widetilde{\psi}_{j,m}^{\text{old}} - \sum_k \widetilde{s}_{j,k,m} \widetilde{\varphi}_{j,k}^{\text{old}}. \tag{2.35}$$

We can specify the coefficients $\{\widetilde{s}_{j,k,m}\}$ in equation (2.35) according to the properties of the dual wavelet we want to add for the situation in hand, for example, vanishing moments.

2.2.4 Fast lifted wavelet transform

Using the formulation of the lifting scheme, it is possible to adapt the second generation fast wavelet transform described in equations (2.25), (2.26) and (2.27) to incorporate the new (biorthogonal) filters. By applying the primal lifting scheme to the fast wavelet transform, we obtain

$$\begin{aligned} c_{j,k} &= \sum_l \widetilde{h}^{\text{old}}_{j,k,l} c^{\text{old}}_{j+1,l} + \sum_m s_{j,k,m} d^{\text{old}}_{j,m} \\ &= c^{\text{old}}_{j,k} + \sum_m s_{j,k,m} d^{\text{old}}_{j,m}, \end{aligned} \tag{2.36}$$

where $c^{\text{old}}_{j,k}$ and $d^{\text{old}}_{j,m}$ are the coarse scaling and detail coefficient from the MRA with (unlifted) filters: we use the fast wavelet transform and then lift the coarse scale coefficients using the (old) detail coefficients.

The equation (2.36) is known as the *fast forward lifted wavelet transform.* The primal lifting scheme described above can be viewed as *updating* the coarse scaling coefficients with the detail coefficients. This notion of update is examined in more detail below.

Similarly, we obtain the *fast inverse lifted wavelet transform* recursion

$$c_{j+1,l} = \sum_k h^{\text{old}}_{j,k,l} \left(c_{j,k} - \sum_m s_{j,k,m} d_{j,m} \right) + \sum_m g^{\text{old}}_{j,m,l} d_{j,m}. \tag{2.37}$$

In other words, we first *unlift* the scaling coefficient and then perform the inverse transform using the old filters.

Now considering the dual lifting scheme, again by combining the fast wavelet transform and the lifting equations for the new biorthogonal filters we get the *fast dual lifted wavelet transform.* Essentially, this is performed by obtaining the coarse coefficients from those at the finer resolution (using the fast wavelet transform), and then using the scaling coefficients produced to *predict* (see

below) the detail coefficient for the new MRA.

$$\begin{aligned} d_{j,m} &= \sum_l \widetilde{g}^{\text{old}}_{j,m,l} c^{\text{old}}_{j+1,k} - \sum_k \widetilde{s}_{j,k,m} c^{\text{old}}_{j,k} \\ &= d^{\text{old}}_{j,m} - \sum_k \widetilde{s}_{j,k,m} c^{\text{old}}_{j,k}, \end{aligned} \tag{2.38}$$

since the expression $\sum_l \widetilde{g}^{\text{old}}_{j,m,l} c^{\text{old}}_{j+1,k}$ is just the old detail coefficient from the fast forward transform equation (2.26).

The *inverse dual lifted transform* is given by the equation

$$c_{j+1,l} = \sum_k h^{\text{old}}_{j,k,l} c^{\text{old}}_{j,k} + \sum_m g^{\text{old}}_{j,m,l} \left(d_{j,m} + \sum_m \widetilde{s}_{j,k,m} c^{\text{old}}_{j,k} \right). \tag{2.39}$$

Note that the construction of the new biorthogonal filters using equations (2.28)–(2.31) and (2.32)–(2.35) is unnecessary when implementing the fast lifted wavelet transform: all computation is incorporated into the equations (2.36)–(2.39).

Alternating primal and dual lifting steps can be used on a pair of dual multiresolution analyses. This procedure thus builds up desired properties of the MRA by first modifying the primal wavelets, and then the dual wavelets. This is called the *cakewalk construction*. It can be shown (see Sweldens [96]) that lifting the old wavelet to the new wavelet does not change the number of vanishing moments of the dual wavelets, and hence the vanishing moments of the primal and dual wavelets can be increased in turn.

2.2.5 Stability of wavelet transforms

Now we make a slight digression about the stability of wavelet decompositions produced by lifting transforms.

Classical wavelet transforms like the DWT use orthonormal wavelet bases,

which result in stable transforms. The stability of wavelet transforms has been well investigated, and it has been shown that orthogonality of wavelet bases are strongly linked with transform stability: non-orthogonality of a basis will introduce instability of the associated transform.

Instability in wavelet transforms basically means that there will be a decrease in reliability of thresholding of sparse wavelet coefficient sequences in regression and smoothing applications: small changes in wavelet coefficients in sparse expansions will have larger effects in resulting function estimates.

When constructing second generation wavelet bases, for example with the lifting scheme, the wavelets in function representations are *biorthogonal Riesz bases* from the underlying multiresolution analysis structure. Recall from equation (2.9) earlier, that a Riesz basis is a system of functions $\{g(x-k)\}_{k\in\mathbb{Z}}$ such that $\exists\ A, B > 0$ with

$$A\|\lambda\|_{l^2}^2 \leq \left\| \sum_{i\in\Lambda} \lambda_i g(x-i) \right\|^2 \leq B\|\lambda\|_{l^2}^2, \tag{2.40}$$

for any $\Lambda \subset \mathbb{Z}$ and any coefficient sequence $(\lambda)_k \in l^2(\mathbb{R})$.

Definition 2.14. *The* **condition number** *of a (Riesz) basis is the quantity* $\kappa = \frac{A}{B}$*, where* A *and* B *are the orthogonality constants as in the definition of a Riesz basis.*

The condition number gives us a measure of instability of a wavelet basis. For stability (and orthogonality) of a wavelet transform, we would want the condition number of a wavelet basis to be close to one. High values of condition numbers indicate instability. An alternative definition of the condition number of a (wavelet) basis is from the formula

$$\kappa = \|\widetilde{W}\|_{l^2}\|\widetilde{W}^{-1}\|_{l^2}, \tag{2.41}$$

where $\widetilde{W}$ is the transform matrix obtained from the wavelet decomposition.

Note that we use $\widetilde{W}$ to denote the transform matrix, as opposed to W in the case of the DWT. This is due to the decomposition matrix being constructed using the *dual* filters. The matrix associated to the inverse transform (denoted W) is simply the inverse of the matrix $\widetilde{W}$. Here, the norm $\|\cdot\|_{l^2}$ is taken to mean $\|\widetilde{W}\| = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}\widetilde{W}_{i,j}^2}$ for $\widetilde{W}$ an $n \times n$ matrix.

The paper [91] features a recursive method of constructing a transform matrix using the decomposition filters $\widetilde{\mathcal{H}}$ and $\widetilde{\mathcal{G}}$.

Stability of lifting schemes

Unfortunately, for irregularly spaced data, it is not guaranteed that wavelet bases produced through lifting are Riesz, and hence orthogonality (and therefore stability) does not necessarily hold.

The stability of lifting transforms was investigated in Simoens and Vandewalle [91]. The authors conclude that for stability of second generation wavelet transforms using the odd/even split (described below in Example 2.15), the weights in the prediction and update lifting steps need to be uniformly bounded. A local semi-orthogonalization of the prediction operator is suggested to try to improve stability. The wavelets are imposed to be orthogonal to the scaling functions at the same level. This removes any vanishing moments that the dual wavelets possess, so to counteract this, an update stage is performed afterwards to build the vanishing moments again.

Vanraes *et al.* [98] study the effects of stability. Their study discovers that the update weights in the primal lifting step are affected by the scaling functions from prediction. Highly irregular data can cause problems due to using information in data that is relatively far apart in prediction and update steps. This could result in undesirable scaling function properties. This is exhibited by high transform matrix condition numbers.

In the next chapter, we will investigate the stability of our adaptive lifting transforms.

2.2.6 The lifting scheme: computational approach

Let us now introduce a more computational approach to the lifting scheme. The viewpoint which follows demonstrates the thinking behind what has been described above, and clarifies choices for dual and primal lifting.

The basic idea is as follows. Suppose we have a data vector, $\mathbf{f}$, representing an $L^2(\mathbb{R})$ function, f. Our motivation, as usual in signal processing and wavelet basis constructions, is to try and find a more efficient way of representing the data vector. We accept that this may result in a loss of information about the signal, or approximation of the original signal, provided that this loss is "small". This is an obvious benefit in applications such as data compression. Our aim is to try and take advantage of the correlations between different parts of the signal. If we do this, then we can approximate some parts with others, thus making the storage and reconstruction process more efficient.

As in Section 2.1.3, suppose the data vector is of length $n = 2^J$ and is sampled regularly. Again, define $c_{J,k}$ to be the function values, i.e. $c_{J,k} = f(t_k) = f_k$ for sites t_k and $k \in \{0, \ldots, 2^J - 1\}$. Here J denotes the finest scale we are working from.

First we subsample the data, that is, split the vector into two subsets. Let us denote the two indexing sets of the data components by $\mathcal{K}$ and $\mathcal{M}$, so that $\mathcal{K} \cup \mathcal{M} = \{0, \ldots, n-1\}$. In the notation of below, $\mathbf{f} = \text{Merge}(\mathbf{f}^{\mathcal{K}}, \mathbf{f}^{\mathcal{M}})$. If the two subsets of the signal are strongly correlated, then we will be able to 'predict' the whole signal fairly accurately from one of the subsets only.

However, it is normally not the case that a signal can be split into two subsets to achieve perfect reconstruction from using one of them. Hence we predict one subset using the other, and encode the difference between the prediction and the original part of the data vector (dual lifting). In this way, we can still reproduce the signal without any loss of information. With this

operation, we create the two vectors $\{c_{J-1,k}\}_{k\in\mathcal{K}}$ and $\{d_{J-1,k}\}_{k\in\mathcal{M}}$. The vector $\mathbf{d_{J-1}}$ contains the coded difference of the prediction. These form the scaling coefficients and detail coefficients as in the discrete wavelet transform case. So a good prediction will produce small detail coefficients and thus a sparser representation of the function.

We now update the scaling coefficient vector, using the two vectors we have at our disposal (primal lifting). The motivation for using an updating step is that in many applications, we would want to preserve certain properties of the low pass coefficients at each decomposition level. For example, in image processing, a constant overall (average) brightness of an image is usually desired at each level.

Computationally, the new scaling coefficients are stored in the same place as the old scaling coefficients so the algorithm has efficient storage. This is then repeated.

In operator notation, this can be formulated as

Split. $(\mathbf{c_J}^{\mathcal{K}_J}, \mathbf{c_J}^{M_J}) =: \text{Split}(\mathbf{c_J})$

Prediction. $\mathbf{d_{J-1}} := \mathbf{c_J}^{\mathcal{M}_J} - P(\mathbf{c_J}^{\mathcal{K}_J})$, where $P(\cdot)$ is a prediction operator. Typically, we choose P so that it makes use of the correlation between $\mathbf{c_J}^{\mathcal{M}_J}$ and $\mathbf{c_J}^{\mathcal{K}_J}$.

Update. $\mathbf{c_{J-1}} := \mathbf{c_J}^{\mathcal{K}_J} + U(\mathbf{d_{J-1}})$, where $U(\cdot)$ is an update operator based on the wavelet coefficients.

Repetition of this procedure will completely decompose the signal as a normal multiresolution analysis would. All classical filter banks can be decomposed into an alternating sequence of predict and update lifting steps with methods involving Euclid's Algorithm and Laurent polynomials, as shown in [36].

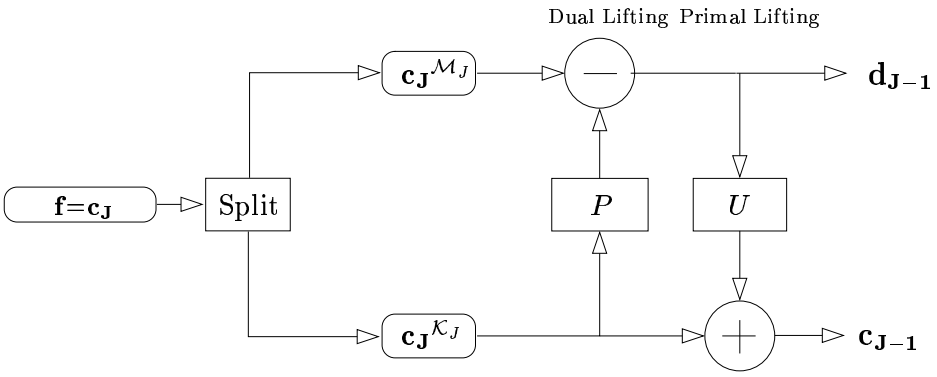

Figure 2.4: A diagrammatical view of the wavelet lifting scheme. The lifting steps defined by the operators P and U can be combined for a cakewalk construction of wavelet properties.

Note that this lifting scheme procedure is easily inverted as the steps:

Remove update: $\mathbf{c_J}^{\mathcal{K}_J} = \mathbf{c_{J-1}} - U(\mathbf{d_{J-1}})$

Remove prediction: $\mathbf{c_J}^{\mathcal{M}_J} = \mathbf{d_{J-1}} + P(\mathbf{c_J}^{\mathcal{K}_J})$

Merge: $\mathbf{c_J} =: \text{Merge}(\mathbf{c_J}^{\mathcal{K}_J}, \mathbf{c_J}^{\mathcal{K}_J})$.

Note that the same operators are used in the forward and inverse transforms, only with a change of sign at each step.

This computational version of the lifting scheme agrees with the MRA version introduced earlier. The lifting scheme is superior to the classical wavelet transform in the following ways:

1. By an appropriate choice of P and U, one can build wavelets with desired properties, for example vanishing moments, or resemblance to specific features of functions (used in feature extraction and recognition).

2. There is an in-built way of calculating the wavelet transform: the wavelet coefficients are just stored in the place of the function values $f^{\mathcal{M}}$ at each stage, and the scaling coefficients are stored in the function positions $f^{\mathcal{K}}$. In this way, the new filters do not have to be worked out explicitly,and so no extra memory is required to preform the transform.

3. It has a fast reconstruction and it can be easily inverted.

4. It can produce wavelets which are needed in a more general setting than traditional transforms, for example on bounded domains, curves and surfaces, and irregular grids, thus having greater flexibility. We can adapt the wavelet transform produced by lifting to a specific situation by choosing the prediction and update operators carefully. In the next chapter, we demonstrate an adaptive lifting scheme used for nonparametric regression, in which the predict and update steps are designed for efficiency.

Example 2.15. We will now briefly outline some simple lifting transforms.

The Haar transform

Firstly, we need to split the data into two groups. A trivial choice is to take one set as the even coefficients and the other as the odd ones. These are known as the *polyphase components*. For the purposes of this discussion, assume that the data has length a power of two, so that $\mathcal{K}$ and $\mathcal{M}$ are the same size. If we take the even coefficients as the scaling coefficient representation of the original data, that is, $\mathbf{c_J}^{\mathcal{K}} = \{f_{J,2k}\}$, then the other information is then contained in the odd coefficients. If two consecutive function values are similar, then their difference will be small on average. We predict an odd coefficient with its (right) even neighbour (the next data point), encoding the difference between the approximation and the original data vector in the wavelet coefficients

$$d_{J-1,k} := f_{J,2k-1} - f_{J,2k}.$$

We now update the even coefficients by

$$c_{J-1,k} := c_{J,k} + d_{J-1,k}/2.$$

This choice of prediction and update corresponds to the Haar wavelet. Note

that with this update, the average is preserved, since, retracing the lifting steps,

$$c_{J-1,k} + d_{J-1,k}/2 = f_{J,2k} + \frac{f_{J,2k-1} - f_{J,2k}}{2} = \frac{f_{J,2k} + f_{J,2k-1}}{2}.$$

The prediction step ensures that the wavelet coefficients will be zero for constant parts of the function, i.e. the dual wavelet has one vanishing moment. Delouille *et al.* [37] and Delouille *et al.* [38] discuss an (unbalanced) Haar lifting transform which generalizes the Haar wavelet for irregularly-spaced data, and explores adapting the technique for higher order wavelets.

The Linear Wavelet

The linear wavelet uses a slightly more complex prediction step:

$$d_{J-1,k} := f_{J,2k-1} - \frac{1}{2}(f_{J,2k} + f_{J,2k-2}), \tag{2.42}$$

that is, the odd coefficients are estimated by the average of the two neighbouring even coefficients. For the moment, we will assume that this sequence is infinite, and return to the finite case a little later. This prediction has the obvious property that if the original signal is (piecewise) linear between the even coefficients, the corresponding wavelet coefficients will be zero. Hence the coefficients can be seen to encode how much the signal fails to be linear. Similar to above, if in the update stage the aim is to preserve the average of the signal (level independently), we want to maintain

$$\sum_k c_{J-1,k} = \frac{1}{2}\sum_k f_k. \tag{2.43}$$

Using the same form as the prediction stage, we have an update

$$c_{J-1,k} := f_{J,2k} + \alpha(d_{J-1,k} + d_{J-1,k+1}) \tag{2.44}$$

and using (2.42) and (2.44) to solve the average equation (2.43)

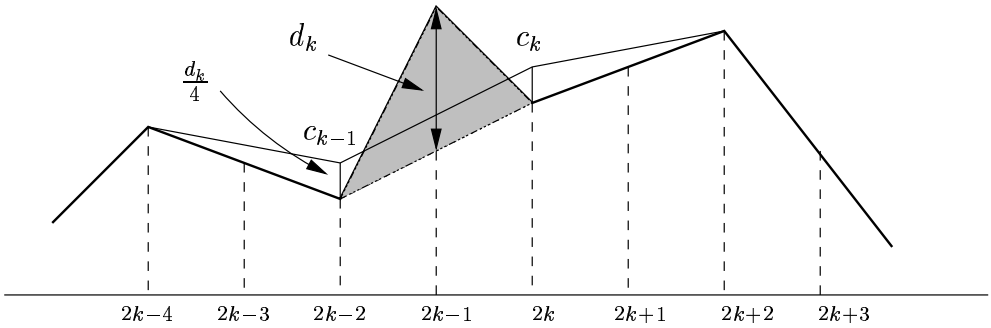

Figure 2.5: A linear prediction scheme. The diagram shows that for the next (coarser) scale, linear prediction (dotted line) results in the shaded area being redistributed to maintain the overall level average. This has the effect of increasing f_{2k-2} and f_{2k} by $d_k/4$ from the original signal (heavy line) to the lifted approximation (thin line). This is captured in the scaling coefficients c_{k-1} and c_k. Here the previous and next wavelet coefficients are zero since the function is linear over the even samples.

$$\begin{aligned}
\frac{1}{2}\sum_k f_{J,k} = \sum_k c_{J-1,k} &= \sum_k \{f_{J,2k} + \alpha(d_{J-1,k} + d_{J-1,k+1})\} \\
&= \sum_k f_{J,2k} + 2\alpha \sum_k d_{J-1,k} \\
&= \sum_k f_{J,2k} + 2\alpha \sum_k f_{J,2k-1} - \frac{2\alpha.2}{2}\sum_k f_{J,2k} \\
&= 2\alpha \sum_k f_{J,2k-1} + (1-2\alpha)\sum_k f_{J,2k}
\end{aligned}$$

we get $\alpha = 1/4$. Hence to preserve the average, the update step is

$$c_{J-1,k} := f_{J,2k} + \frac{1}{4}(d_{J-1,k} + d_{J-1,k+1}).$$

From equation (2.44), the (dual) wavelet is also ensured to have one vanishing moment.

The lifting scheme is clearly not restricted to these cases. One could use more complex prediction steps, such as a cubic approximation of the odd scal-

ing coefficient $c_{J,2k-1}$ using the even coefficients $c_{J,2k-4}$,c_{2k-2},c_{2k} and c_{2k+2} (two on either side). This would have the effect of making the wavelet coefficients zero on cubic pieces of the signals, hence giving the wavelets four vanishing moments. We are not limited to splitting the data into even and odd subsets either; moreover, the subsets do not have to be the same size. In the next chapter, we describe "one coefficient at a time" lifting [59, 60], where only one wavelet coefficient is predicted at each lifting step.

Remark. At the start of this section, we remarked that first generation wavelets do not generalize to non-standard situations easily. It was noted that the classical DWT needs boundary considerations when used on finite length signals. When lifting, if a high order prediction scheme is used on a finite signal, there may be insufficient data points when lifting near or on the signal boundary. However, where no value exists, the prediction can be made using copies of neighbouring data points where needed and then using extrapolation. In this way, the correlation structure of the data is used wherever possible in prediction.

In the exposition above, the data was assumed to be on a regular grid, that is, sampled at equally-spaced sites. On irregular data grids, we could still use the prediction and update operators as above, but the prediction coefficients could vary depending on the irregularity of the samples.

These issues are considered in the next chapter when we introduce work on adaptive lifting schemes.

2.3 Nonparametric regression

In this section, we review the standard statistical problem of *nonparametric regression*, that is, estimating a function g from noisy observations, assuming no particular functional form for the signal. The model used extensively in the

literature is as follows. Suppose we have n noisy samples of the form

$$f_i = g(t_i) + \varepsilon_i \quad \text{for } i \in \{1, \ldots, n\}, \tag{2.45}$$

where the sites are usually assumed to be regularly-spaced on the unit interval ($t_i = i/n$) and the noise is taken to be independently identically distributed and typically Gaussian: $\varepsilon_i \sim N(0, \sigma^2)$. Classical wavelet methods also assume that n is a power of two.

In usual estimation techniques, we assess the performance of an estimator $\widehat{g}$ by its mean integrated squared error

$$\mathrm{MISE}(\widehat{g}, g) = \mathbb{E} \int \{\widehat{g}(x) - g(x)\}^2 \, dx. \tag{2.46}$$

In many applications, however, we do not know the function g, and since the data is sampled, we usually compute the estimate $\widehat{g}$ at the sampled points t_i. Using the l^2 error, we then assess the estimator by the mean squared error

$$\mathrm{MSE}(\widehat{g}, g) = \frac{1}{n} \sum_{i=1}^{n} (\widehat{g}(t_i) - g(t_i))^2. \tag{2.47}$$

We now review some popular methods for estimating the function g. Initially, we outline a few non-wavelet smoothing techniques, before concentrating on some aspects of wavelet estimators. For a fuller overview, see for example, [100, 1, 83].

2.3.1 Non-wavelet regression techniques

Local polynomial regression

Local polynomial regression estimators use weight functions W over a window to fit polynomials of order p according to the (weighted) least squares criterion

$$\sum_{i=1}^{n}[f_i - \beta_0 - \cdots - \beta_p(x - x_i)^p]^2 W\left(\frac{x - x_i}{h}\right). \tag{2.48}$$

The minimizer of this equation is an estimate for the nonparametric regression model (2.45). The bandwidth parameter h specifies the width of the window, and affects the smoothness of the resulting estimate of g. This method assumes that the underlying signal g has a certain degree of smoothness. The software *Locfit* [69, 70] used in the next chapter is a variant of a local polynomial regression estimator.

Other basis functions can be used as the weight functions in the linear superpositions above, some with similar local minimization criteria. Popular choices are, for example, sines and cosines (Fourier expansions) and kernels (kernel smoothers).

Smoothing spline estimators

Another type of regression estimator uses smoothing splines. A smoothing spline method finds the function $\mu(x)$ which minimizes

$$\sum_i \{f_i - \mu(x_i)\}^2 + \lambda \int \mu''(x)^2\, dx.$$

The estimate produced is a piecewise cubic polynomial on subintervals of the real line defined by adjacent regression ordinates $\{x_i\}$ (see for example, [92]). These are also called *knots*. There is a trade-off between how close the spline estimate fits the data, and how "wiggly" the estimate is, which is controlled by the parameter λ. The Comte-Rozenholc method [30] used in later simulations is similar to a smoothing spline estimator. This is described fully in the next chapters.

It is possible to use local weights for smoothing splines as well as local polynomial estimators. For more information on these smoothing techniques, see for example [102], [92] or [15].

2.3.2 Non-linear wavelet shrinkage

We now explain how wavelet shrinkage works. For a review of the ideas presented below, the reader is referred to [76], and also to the important papers [39, 40, 41].

In 1994, Donoho and Johnstone [39] proposed the idea of *thresholding*, which has now become the norm to remove noise in wavelet shrinkage. To estimate the function vector $\mathbf{g}$ from the noisy observations $\mathbf{f}$ we proceed as follows:

1. Decompose the vector with the DWT down to some primary resolution level.

2. *Threshold* the wavelet coefficients to remove the noise.

3. Invert the DWT to obtain the estimate of $\mathbf{f}$, denoted $\widehat{\mathbf{g}}$.

The motivation behind this procedure is that thresholding in the wavelet domain leads to smoothing in the time domain. We mentioned earlier that wavelet representations are often sparse, i.e. most wavelet coefficients in the expansion of a signal are zero, with a few large coefficients.

With this in mind, if we apply the DWT to the noisy observations $\mathbf{f}$, due to orthogonality of the transform, we have

$$d_{j,k} = d^*_{j,k} + e_{j,k}, \tag{2.49}$$

where $\mathbf{d} = \mathbf{W}\mathbf{f}$ is the discrete wavelet transform of $\mathbf{f}$ as in (2.15) and $\mathbf{e} = W\boldsymbol{\varepsilon}$ is the DWT of the noise $\{\varepsilon_{j,k}\}$. Similarly, $\mathbf{d}^* = W\mathbf{g}$.

The regression problem now is how to find the true wavelet coefficients $\mathbf{d}^*$ from the noisy wavelet coefficients $\mathbf{d}$. The i.i.d. Gaussian noise random variables from the original model (2.45) are transformed via the DWT into i.i.d.

Gaussian noise random variables $\mathbf{e}$. Since we expect that the true wavelet coefficients to be sparse, the aim of thresholding is to decide which of the noisy coefficients in $\mathbf{d}$ are purely noise (near zero) or signal information (large).

Thresholding schemes shrink values in the coefficient vector towards zero, or replace the values with zero if their size is below the threshold level, τ. There are various thresholding methods, but two main rules are commonly used. Suppose d is the coefficient to be thresholded. Then

Soft thresholding:

$$\widehat{d} = T^s(d;\tau) = \mathrm{sgn}(d)\,(|d| - \tau)\,\mathbb{I}(|d| - \tau)$$

Hard thresholding:

$$\widehat{d} = T^h(d;\tau) = d\,\mathbb{I}(|d| - \tau),$$

where $\tau > 0$ is the threshold level, and $\mathbb{I}$ is the indicator function.

Soft thresholding shrinks the coefficient d or sets it to zero, whereas hard thresholding is a decision rule of whether to keep the coefficient or set it to zero.

Whichever threshold level is used, the extreme effects are the same: if too low a threshold is used, the function estimate will still be noisy, whereas if too high a threshold is used, the function estimate will lead to "oversmoothing". This leads us onto the question of how to choose the threshold, τ. A few frequently used threshold selection methods are outlined below.

The Universal Threshold

It has been well documented and proposed in the now well-known paper by Donoho and Johnstone [39] that the *universal threshold* performs well in many

cases of denoising. Here the level is set to be

$$\tau^u = \sqrt{2\log(n)}\,\sigma, \tag{2.50}$$

where n is the number of datapoints (as above), and σ^2 is the noise variance of the wavelet coefficients.

This threshold choice is motivated by the following result. If $\{Z_i\}_{i\in\{1,\ldots,n\}}$ are i.i.d. $N(0,1)$ (white noise), then as $n \to \infty$,

$$\mathbb{P}(\max_i |Z_i| \leq \sqrt{2\log n}) \to 1.$$

This means that using the universal threshold, any noise will be set to zero with probability approaching one (as n increases).

To use this threshold, we need to know the variance of the noise, which is rarely possible. The variance is usually estimated from the data, and [39] suggests using

$$\widehat{\sigma} = \mathrm{MAD}(\mathbf{d_J})/0.6745,$$

where $\mathrm{MAD}(\mathbf{x}) = \mathrm{median}(|\,x_i - \mathrm{median}(\mathbf{x})|)$. This estimate of the variance uses the median absolute deviation of the finest level wavelet coefficients. The rationale behind this quantity is that it is expected that the wavelet coefficients at this level will be mostly noise, and the non-zero coefficients will not affect the median of the absolute values. In practice, using the universal threshold tends to oversmooth data, as shown by [75].

Donoho and Johnstone [39] implement soft thresholding of wavelet coefficients with the universal threshold under the name *VisuShrink*.

Thresholding via Stein's Unbiased Risk Estimate

Let $\boldsymbol{\mu}$ be the mean vector of a p-dimensional multivariate normal distribution with observations x_i, i.e. $x_i \sim N(\mu_i, 1)$. A result proved by Stein gives a method of estimating the risk of a nonlinear estimator of the mean vector.

Theorem 2.16. *Suppose an estimator of μ, $\widehat{\boldsymbol{\mu}}(\mathbf{x})$, is an estimator that can be written as*

$$\widehat{\boldsymbol{\mu}}(\mathbf{x}) = \mathbf{x} + \mathbf{g}(\mathbf{x}),$$

where $\mathbf{g} = \{g_i\}_{i\in\{1,\dots,p\}}$ is a function from $\mathbb{R}^p$ to $\mathbb{R}^p$. If $\mathbf{g}$ is weakly differentiable, then

$$\mathbb{E}_\mu \|\widehat{\boldsymbol{\mu}}(\mathbf{x}) - \boldsymbol{\mu}\|^2 = p + \mathbb{E}_\mu\{\|\mathbf{g}(\mathbf{x})\|^2 + 2\nabla\cdot\mathbf{g}(\mathbf{x})\} \tag{2.51}$$

is an unbiased estimate of the risk, where

$$\nabla\cdot\mathbf{g} \equiv \sum_i \frac{\partial}{\partial x_i} g_i.$$

At the beginning of this section, we introduced the soft thresholding function

$$T^s(\mathbf{x};\tau) = \operatorname{sgn}(\mathbf{x})\,(|\mathbf{x}| - \tau)\,\mathbb{I}(|\mathbf{x}| - \tau).$$

By considering the cases when $\tau > |\mathbf{x}|$ and $\tau \le |\mathbf{x}|$ separately, one can easily derive the alternative form

$$T^s(\mathbf{x};\tau) = \mathbf{x} - \operatorname{sgn}(\mathbf{x})\min(|\mathbf{x}|,\tau). \tag{2.52}$$

Since this function is weakly differentiable, substituting this into (2.51) gives that

$$\mathrm{SURE}(\mathbf{x};\tau) = p - 2\cdot\#\{i\ |\ |x_i| \le \tau\} + \sum_{i=1}^{p}\min(|x_i|,\tau)^2 \tag{2.53}$$

is an unbiased estimate of the risk $\mathbb{E}_\mu \|\widehat{\boldsymbol{\mu}}^s(\mathbf{x}) - \boldsymbol{\mu}\|^2$.

The natural choice to select the threshold, τ, is to choose the parameter value

$$\tau^{\mathrm{SURE}} = \mathrm{argmin}_{0 \le \tau \le \sqrt{2 \log p}} \ \mathrm{SURE}(\mathbf{x}; \tau). \tag{2.54}$$

Donoho and Johnstone [40] show that the threshold τ^{SURE} will be at one of the normal sample magnitudes $|x_i|$.

Donoho and Johnstone [40] also advocate a level-dependent thresholding decision rule. Their *Sureshrink* procedure uses variance computations based on wavelet coefficient sequences to determine whether to set the threshold for the wavelet coefficient level j as $\tau_j^{\mathrm{SURE}} = \mathrm{argmin}_{0 \le \tau \le \sqrt{2 \log n_j}} \ \mathrm{SURE}(\mathbf{d_j}; \tau)$ or the universal threshold. Here n_j denotes the number of wavelet coefficients in level j. In cases when it is suspected that the coefficients only represent noise, that is, for sparse coefficient sequences, the universal threshold τ^u is used. This choice is motivated by the observation in Donoho and Johnstone [40] that the noise content in the coefficient sequences tends to dominate the SURE threshold computation, therefore resulting in oversmoothing due to a low threshold value. The *Sureshrink* thresholding rule is implemented in some of our simulation comparisons in later chapters.

Empirical Bayes thresholding

As mentioned before, the discrete wavelet transform produces sparse wavelet coefficient sequences. There is a lot of work in the literature on Bayesian thresholding, in which a positive prior is placed on the wavelet coefficients in an expansion being zero. This prior is designed to capture the sparseness of the wavelet coefficients. For summarizing texts, see for example [99], [74] or [24].

Clyde *et al.* [25], Abramovich *et al.* [3] and Silverman [90] explore a nor-

mal mixture form for the choice of the prior. Chipman *et al.* [21] discusses thresholding methods arising from a mixture of two normals as a prior. The latest incarnation of Bayesian thresholding comes from Johnstone and Silverman [61, 63]. Their method places a prior on the true wavelet coefficients of the form

$$d^*_{j,k} \sim (1-\pi)\delta_0 + \pi\gamma, \tag{2.55}$$

where γ is the density of the wavelet coefficient, conditional on it being non-zero. Here π represents the (prior) probability of a wavelet coefficient being non-zero. The authors suggest a heavy-tailed choice for γ, such as the Laplace distribution, or the "quasi-Cauchy" density described in [61, 63].

Recall from (2.49) that $d_{j,k} \sim N(d^*_{j,k}, \sigma^2)$. For each wavelet coefficient $d_{j,k}$, we can find the posterior distribution of $d^*_{j,k}$ (independently) conditional on this observation, $f(d^*_{j,k}|d_{j,k})$. Based on this, the true wavelet coefficients $d^*_{j,k}$ can be estimated using the median of the posterior distribution. Effectively, this statistic acts as a thresholding method, since with a fixed π, there exists a value τ_π with the median equal to zero whenever $|d^*_{j,k}| \leq \tau_\pi$ (see [63]). Johnstone and Silverman [61, 63] also consider different thresholding rules such as the posterior mean, and soft or hard thresholding using τ_π.

Since it is expected that coarse resolution levels will result in important signal information and sparse coefficient sequences at fine levels, this provides motivation for assuming the mixture probabilities to be equal across each level, and choosing them level-dependently. Johnstone and Silverman suggest that π_j are found with marginal maximum likelihood (MML) by assuming the observations are independent, and the noise level is estimated (if not known) by the median of the absolute values of the wavelet coefficients at the finest resolution level, as in [39]. The true wavelet coefficients are then estimated with the Bayesian model above.

This thresholding technique is proved to possess good properties in a variety of situations. Hence it is used heavily in the following chapters.

Cross-validatory threshold selection

Cross-validation is a technique used in general statistical methods to obtain optimal values of parameters, from minimizing errors in approximation. In particular, it is often used in estimating the optimal smoothing or bandwidth parameter in density estimation. Here, though, we want to estimate the optimal threshold level, based on the data. There are various norms which can be used to measure the approximation error from cross-validation, but for purposes of regression, minimizing the mean integrated square error (MISE) is a popular approach. We therefore want to minimize

$$\mathrm{MISE}(\tau) = \mathbb{E} \int \left\{ \widehat{f}_\tau(x) - f(x) \right\}^2 dx \tag{2.56}$$

which results from minimizing the integrated square error (ISE)

$$\int \left\{ \widehat{f}(x) - f(x) \right\}^2 dx = \int f(x)^2 \, dx + \int \widehat{f}_\tau(x)^2 \, dx - 2 \int \widehat{f}_\tau(x) f(x) \, dx. \tag{2.57}$$

There are different cross-validation methods. A popular type is the so-called *leave one out* (LOO) cross-validation method. All datapoints except one are used to obtain an estimate of the function in question, therefore predicting the one point left out.

Since there is often no natural choice as to which particular datapoint to leave out, the approximation is performed, leaving out each datapoint in turn, and then the n different approximations are averaged to find the overall approximation. This is consistent with the approximation being an unbiased estimator. Since the first term in (2.57) does not depend on the minimizing parameter τ, minimizing the MISE requires us to minimize the *cross-validation*

condition

$$CV(\tau) = \int \widehat{f}_\tau(x)^2\,dx - \frac{2}{n}\sum_{i=1}^{n} \widehat{f}_{\tau,-i}(x_i), \tag{2.58}$$

where $\widehat{f}_{\tau,-i}$ is the estimate of f when leaving out the ith sample point.

Whilst this particular type of cross-validation is suitable for wavelet techniques applicable on datasets of arbitrary length (such as the lifting scheme), it will not work for the classical DWT, since it can only handle data with length a power of two. Nason [75] adapts this technique for the DWT. A point is removed, and then the remaining data is split into two groups, corresponding to the datapoints before and after the removed point. These are extended to a dyadic length and used to form two estimates for the "left" and "right" function values. These estimates are then augmented and used to predict the removed point.

The author also proposes a *two-fold* cross-validation variant, in which half the data points are used to predict the others. This method is then suitable for the DWT, since the subset used for the prediction is of length 2^J, for some J.

In later chapters, we use the general method of cross-validation for smoothing parameter selection in some regression methods. For a more general exploration of cross-validation, refer to [93].

False Discovery Rate thresholding

Another area gaining interest recently is thresholding based on False Discovery Rate (FDR). The FDR of a set is the expected proportion of false predictions from a set of predictions through, for example, a sequence of hypothesis tests. For multiple hypotheses, Benjamini and Hochberg [12] propose a technique for controlling the FDR.

In the context of wavelet coefficient thresholding, the problem takes the form of a multiple hypothesis test that the detail coefficients are zero, that is the hypotheses $H_{j,k} : d_{j,k} = 0$. As a method of analyzing a thresholding procedure, Abramovich and Benjamini [2] suggest using a FDR setup to control the proportion of wavelet coefficients which are erroneously included in the wavelet representation by the procedure. With this in mind, a thresholding procedure is designed to maximize the number of coefficients included a wavelet representation of a function, subject to the constraint that the FDR of erroneous coefficients is at most q, for some control quantity q.

2.3.3 Other regression parameters

We have seen that there is already a choice between using linear and non-linear (wavelet) smoothing techniques for regression, which will affect one's ability to recover signal information accurately. As well as the choice of threshold as a smoothing parameter, varying other aspects of wavelet shrinkage will change our estimates obtained through the methods above.

Firstly, an important issue to consider is the choice of wavelet used in the decomposition–threshold–reconstruction procedure. As a general rule, it is generally thought that it is good practice to use a wavelet that is suitable (in terms of vanishing moments) according to the smoothness of the signal g. However, in all regression problems, the true nature of g is unknown, and often we have no prior information to use in choosing a wavelet basis.

Secondly, the primary resolution level above which thresholding is performed can affect smoothing performance. If the primary resolution level is set too low (coarse), then some important signal information could be lost; if it is set too high, then thresholding could lead to undersmoothing, thus being left with a noisy function estimate after wavelet shrinkage.

This issue is discussed in more detail in Chapter 5, where the primary resolution of adaptive lifting algorithms (introduced in the next chapter) is

studied to try and improve smoothing accuracy.

Chapter 3

Adaptive Lifting

Introduction

This chapter explores adaptive lifting algorithms and their ability to be used in smoothing in a variety of situations. The work of this chapter, together with Chapter 4 is joint with my colleague, Marina Knight. The algorithms described here are featured in the paper [79], accepted for publication.

In the previous chapter, we discussed general wavelet shrinkage methods, and detailed the lifting scheme as a tool for constructing wavelet transforms. Let us return to the problem of nonparametric regression described in equation (2.45). Our aim is to recover a sampled function vector $\mathbf{g}$ from noisy observations $\mathbf{f}$, where the noise is assumed to be Gaussian. We have

$$f_i = g(x_i) + \varepsilon_i \quad \text{for } i \in \{1, \ldots, n\}.$$

Classical wavelet transforms also usually assume that

1. The sample sites x_i are regularly spaced. Furthermore, the data is taken to be sampled from the interval. In other words, $x_i = i/n$ for i in the range $\{1, \ldots, n\}$;

2. the number of observations is a power of two, i.e. $n = 2^J$ for some J;

3. Each observation has one (and only one) function value f_i.

The general procedure in wavelet shrinkage (Section 2.3) is to wavelet transform the data into a set of detail coefficients; modify the wavelet coefficients (by thresholding) to remove the noise; and then perform the inverse transform on the modified coefficients to obtain an estimate of the true signal at the sample sites.

Unfortunately, the assumptions listed above lead to severe limitations in using some wavelet transforms for real-world data. There are many applications which produce datasets from real-life situations, such as industrial or experimental data, which do not satisfy these criteria: they are often irregularly-spaced, for example, because of being observed at certain time points; or are datasets of unspecified length. Moreover, certain applications have more than one observation at each regression point. An example of this type of data is the well-known `mcycle` motorcycle data, introduced by Silverman [89].

To perform the above wavelet shrinkage on irregularly-spaced data, classical wavelet transforms often need to be heavily modified. This sometimes involves, for example, a transformation of the data to the equispaced situation, which can introduce unwanted error and possibly other problems. This also applies when trying to deal with datasets of general length.

In addition to these issues, when performing the classical DWT, there is a matter of how to decompose the signal samples in the model above. This leads us to two main questions when considering wavelet shrinkage with the DWT: which wavelet basis should be used to decompose the data, and in particular, how many vanishing moments should the wavelets possess? Some examples of wavelet bases were introduced in Chapter 2. As noted before, the general recommendation is to use wavelets with smoothness characteristics similar to the true signal g. However, the regularity of g is unknown in most situations.

Nason [77] has investigated the choice of (first generation) wavelet basis for signal decomposition, but in this case, we want the smoothness and general characteristics of the wavelet to be allowed to change at each regression ordinate x_i, to reflect the function behaviour around that point.

In this chapter, we propose a new regression technique for irregularly-spaced data, based on the lifting scheme, as introduced in the last section. We employ "one coefficient at a time" lifting schemes in the algorithms, motivated by being able to use the local properties of the signal data to represent the function more accurately. Our methods exploit the features of the lifting scheme, taking advantage of it being applicable to irregular design datasets of any length.

Furthermore, one feature of our algorithms is that it is not necessary to specify the wavelet functions to be used for different parts of the signal. The signal properties are used to select from a range of different dual lifting (prediction) steps, and thus we are able to introduce an automatic choice of wavelet (and number of vanishing moments) into the design of the procedure, without any user input.

Our methodology also allows for more than one f value at each x value, to be able to handle data such as the `mcycle` dataset mentioned above, which is not generically possible with classical wavelet shrinkage methodology.

The chapter is organized as follows. We first give a brief overview of wavelet and non-wavelet methods for handling irregularly-spaced data. We then describe lifting "one coefficient at a time", used to capture the local features of the data at hand. Section 3.3.1 explains how we introduce adaptivity into this lifting framework, and we discuss a few different issues resulting from this. Section 3.3.3 deals with the modifications to the original adaptive lifting algorithms so that they can cope with multiple observation data. Sections 3.4 and 3.5 show how the sparsity and denoising capability of our algorithms

are assessed, and how well they perform on simulated data compared to both wavelet and non-wavelet methods. We compare our technique against *Locfit* [69, 70], smoothing splines and the Comte-Rozenholc method [30]. In addition, we perform thorough comparisons with the Kovac-Silverman wavelet method for irregularly-spaced data [67]. We also give the examples of the motorcycle and real inductance plethysmography datasets to illustrate how well the methods work compared to other popular smoothing techniques.

3.1 Wavelet methods for irregular designs

As mentioned above, the classical DWT cannot handle irregularly-spaced data. Many techniques have been developed to remove this restriction to regular designs. In this section, we outline some of the popular methods.

Cai *et al.* [19] makes a correspondence between the irregular regression ordinates $\{x_i\}_{i\in\{1,\ldots,n\}}$ and the regular grid $\{i/n\}_{i\in\{1,\ldots,n\}}$ using a strictly increasing mapping H such that $x_i = H^{-1}(i/n)$. Thus if we know the function H, through the composition $g \circ H^{-1}$, the function can be estimated on the regular grid, and then mapped back using H to obtain an estimate of g. If H is unknown, then this function is estimated. However this does not produce a good estimator of g, especially when g is smoother than H. To remedy this, for piecewise Hölder functions, the projection $\mathrm{Proj}_{V_J} n^{-1/2} \sum_{i=1}^{n} f_i \varphi_{J,i}^{per}(x)$ is used to form a new estimator for g, which exhibits good properties for piecewise Hölder functions. The thresholding procedure *VisuShrink* [39] is generalized to select the smoothing parameter.

Sardy *et al.* [86] introduce four simple extensions of the Haar wavelet basis to irregularly sampled data. Piecewise constant interpolation of the irregular datapoints is used to generate new Haar bases. Examples of these bases are the isometric Haar and asymmetric wavelets. For each extension, the Donoho and Johnstone *Visushrink* [39] wavelet shrinkage procedure is adapted

to take advantage of the wavelet structure. The isometric Haar basis can be generalized to wavelets with higher order.

The authors of [20] consider the model (2.45), but where the ordinates x_i are independently uniformly distributed on the interval [0,1]. Considering the order statistics $x_{(i)}$, it is well-known that their expectations have values which are equispaced on the interval, that is $\mathbb{E}(x_{(i)}) = \frac{i}{n+1}$. Thus the datapoints $(\frac{i}{n+1}, f_i)$ are treated as regularly-spaced, and the usual shrinkage procedure is followed to produce an estimate of the regression function. This estimate is shown to have good convergence properties.

Kovac and Silverman [67] use linear interpolation of the given (irregular) data to a regular grid, that is, $\widetilde{\mathbf{f}} := R\mathbf{f}$ is defined, where R is the interpolation matrix. Through interpolation, the method is suitable for datasets of arbitrary length. The usual DWT is applied to the interpolated data. The effects of interpolation on the covariance structure of the resulting detail coefficients is investigated and taken into account when thresholding. Various thresholding techniques are discussed. Nason [77] uses cross-validation to develop an algorithm in this situation to choose the primary resolution level, threshold and wavelet simultaneously. This fast algorithm is applied to the Kovac-Silverman (KS) procedure.

A penalized least squares method for estimating the wavelet coefficients $d^*_{j,k}$ is presented in the paper by Antoniadis and Fan [7]. This is initially applied to the equispaced grid model. The authors prove that if the penalty function has certain characteristics, then there exists a unique solution to the least squares minimization problem. The universal threshold is adapted to the situation in hand to form a new thresholding technique. The resulting estimators are shown to outperform classical DWT methods when considering the L^2 risk. Estimators for irregular grid data were then constructed using non-linear regularized Sobolev interpolators, that is, estimators which minimise a penalisation criterion based on their Sobolev norm. These estimators were then improved forming a regularized one-step estimator. Other penalty functions

are discussed, which lead to other thresholding rules.

The nonparametric regression problem is approached in [82] by imposing a probabilistic model on the x-values. Let X be a random variable with density h. The underlying multiresolution analysis structure is used to obtain an estimate for the regression function $E(f|X = x)$ by projection onto the subspace V_j, i.e. it is expressed as the linear combination $\sum_k \hat{c}_{J,k}\varphi_{J,k}$, where the expansion coefficients $\hat{c}_{J,k}$ are estimates of the scaling coefficients $c_{J,k}$ using the density h and the scaling functions $\varphi_{J,k}$. The asymptotic properties of the estimator are investigated, and provided that h is reasonably smooth, it is shown that the estimator has good performance.

Whilst the methods outlined can handle irregularly-spaced data, they often impose assumptions which may not necessarily hold, for example on the regularity of g or on the distribution of the x_i. Furthermore, some of the procedures need other functions to be estimated, for example, H in Cai *et al.* [19], or h in [82]. Using interpolation to adapt wavelet methods to irregular designs often introduces unwanted error, depending on the choice of how the data is mapped to a regular grid.

In later sections, we compare the sparsity and denoising performance of our adaptive lifting algorithms to non-wavelet methods as well.

The first of our comparisons is to the local polynomial regression implementation *Locfit* [70, 69]. *Locfit* fits a low order (linear or quadratic) polynomial to the data in a sliding window. *Locfit* can handle multiple observations at x-values.

Secondly, we compare the adaptive lifting algorithms to the S-Plus function `smooth.spline()`, which performs cubic smoothing spline regression. General estimators of this type are discussed in Section 2.3.

We also see how our procedures perform against the Comte-Rozenholc method in [30]. This algorithm fits piecewise standard and trigonometric poly-

nomials to a collection of *knots* to estimate the signal, in a fashion similar to smoothing splines. The authors add automatic adaptivity to the algorithm by allowing a large set of knot configurations and polynomial degrees to be chosen. Using the two user-specified inputs of maximum number of knots and maximum polynomial degree, the data-driven procedure decides which knot sites to use and selects an optimum (mixture) set of polynomials with different degrees to fit to the data.

3.2 Lifting "one coefficient at a time"

In Section 2.2 we introduced the lifting scheme, a method of constructing wavelet expansions with an underlying biorthogonal MRA structure. In the general irregular setting, it is not unreasonable to expect that function values in a (small) region will be correlated, so to take advantage of the local properties of the function, a logical step is to change our aim to predict only one coefficient at a time. Before focussing on our adaptive lifting algorithms in more detail, we explore lifting schemes in which predictions of *only one* coefficient are made using all other datapoints. This approach was first discussed by Jansen *et al.* [59, 60]. In particular, in what follows, we give a brief description of the idea of single coefficient lifting in one dimension. Although this has a slightly computational emphasis, we should point out that the MRA structure of Section 2.2 holds here. We will support our heuristic approach in the next section and next chapter with extensive simulations.

Suppose we have an irregularly sampled signal of length n, with sample sites x_i. Since we have n points, we index the levels by $k \in \{1 \dots n\}$, in other words, so that the level describes how many data points are yet to be removed.

Suppose we also have constructed initial scaling functions, $\varphi_{n,k}$, satisfying the constraint that $\varphi_{n,k}(x_i) = \delta_{i,k}$ for $i, k \in \{1 \dots n\}$. Initially, we have that the function f is expressed as

$$f(x) = \sum_{k=1}^{n} c_{n,k}\varphi_{n,k}(x),$$

since at this stage, no coefficients have been lifted.

Since our regression problem is on the real line, we can construct intervals associated to each gridpoint by ordering the x-values, and using the midpoints between successive gridpoints as the interval endpoints. In this way, we can take the initial scaling functions to be the characteristic functions of the intervals associated to the sample sites, so that the scaling functions satisfy the constraint above. From the constraint, we have

$$f_i := f(x_i) = \sum c_{n,k}\varphi_{n,k}(x_i) = \sum c_{n,k}\delta_{i,k} = c_{n,i}.$$

Now the motivation for the constraint on the initial scaling function is demonstrated: we can use the observed function values as the initial scaling coefficients.

Since we are lifting one coefficient at a time, there is still the issue of which order we use to estimate the wavelet coefficients. For lifting individual coefficients, one way of choosing the order in which the points are removed is to proceed in order of increasing support of the scaling functions (associated interval length), since then in some sense we start by encoding the finest detail areas and continue to the coarsest areas with most effect. If we denote the scaling function integrals by $\mathcal{I}_{n,k}$, this choice corresponds to finding j_n, with $\mathcal{I}_{n,j_n} = \min_{k\in\{1,\ldots,n\}} \mathcal{I}_{n,k}$. Note that in the notation of Section 2.2, we take $\mathcal{M}_n = \{j_n\}$ and $\mathcal{K}_n = \{1,\ldots,n\} \setminus \{j_n\}$.

Next, we choose a set of neighbours counted by our indexing set I_n. Note that here, we also index the neighbourhood by the stage n, since there is a

correspondence between neighbourhoods and the removal stage. We use a particular coefficient's "neighbours" in the prediction of the coefficient, using linear regression. The choice of which set of neighbours we use in the prediction of each point is important to the description of introducing adaptivity into the algorithms, so this issue is postponed until later. As described in Section 2.2, the detail coefficient is calculated as the difference between the (old) scaling coefficient and the prediction:

$$d_{j_n} := c_{n,j_n} - \sum_{i \in I_n} a_i^n c_{n,i}, \tag{3.1}$$

where the coefficients a^n are the regression weights from performing the regression over the neighbourhood I_n. Where there is only one neighbour in I_n, then the point j_n is predicted by $f(x_i)$, and the wavelet coefficient is obtained from

$$d_{j_n} := c_{n,j_n} - c_{n,i}. \tag{3.2}$$

For a function which is constant over the prediction neighbourhood, the wavelet coefficient should be zero; it is easily seen that prediction with weights satisfying $\sum_j \alpha_j^r = 1$ has this property.

We then update the neighbourhood I_n using the single detail coefficient produced. The update stage has the form

$$c_{n-1,i} := c_{n,i} + b_i^n d_{j_n}, \ \forall i \in I_n, i \neq j_n. \tag{3.3}$$

Note that the only scaling coefficients to be affected by the update lifting step are the neighbours, so we also have

$$c_{n-1,i} := c_{n,i} \quad \text{for any } i \notin I_n \ (i \neq j_n).$$

In the update stage, we want to keep the quantity $\sum c_{n,k} \int \varphi_{n,k}(x)\, dx$ con-

stant from level to level (after we remove a point). This translates into the equation

$$\sum_k c_{n,k}\mathcal{I}_{n,k} = \sum_k c_{n-1,k}\mathcal{I}_{n-1,k}. \tag{3.4}$$

Since the only integrals to change after lifting a datapoint are those in the prediction neighbourhood, which take a proportion of the lifted point's scaling function integral, this reduces to the equation

$$c_{n,j_n}\mathcal{I}_{n,j_n} + \sum_{i\in I_n} c_{n,i}\mathcal{I}_{n,i} = \sum_{i\in I_n} c_{n-1,i}\mathcal{I}_{n-1,i}. \tag{3.5}$$

By using the relations for the update lifting step, and rearranging this, we get

$$\sum_{i\in I_n} b_i^n \mathcal{I}_{n-1,i} = \mathcal{I}_{n,j_n}.$$

This equation is used to calculate the update weights b^n. Lifting does not guarantee stability of the transform. To find the update weights, the minimum norm solution $b_i^n = \mathcal{I}_{n,j_n}\mathcal{I}_{n-1,i}/\sum_{i\in I_n}\mathcal{I}_{n-1,i}^2$ to this equation is suggested by Jansen *et al.* in [60] to try and minimize instability of the transform. This is the formula for the update weights which we use in our adaptive lifting algorithms detailed in the next section. Note that the computation of the update weights needs no more information than the present function integrals.

Note that the operations can be inverted in the normal way as the mirror of the lifted transform to get the inverse transform.

In the update lifting step, the lifted datapoint is removed, and the interval associated to that point is then divided up and distributed among the neighbour intervals. In other words, when the detail coefficient replaces the predicted scaling coefficient, the vertex is removed from the dataset, which leads to the coarser grid. This decomposes f into the following expansion:

$$f(x) = d_{j_n}\psi_{j_n}(x) + \sum_{i\in\{1,\ldots,n\}\setminus\{j_n\}} c_{n-1,i}\varphi_{n-1,i}(x), \tag{3.6}$$

where ψ_{j_n} and $\varphi_{n-1,i}$ are wavelet and scaling functions.

We have now lifted one wavelet coefficient. To recap, the procedure for this is to identify the coefficient to be lifted, j_n, through examination of the scaling function integrals; perform prediction and update lifting steps and change the scaling function φ_{n,j_n} into the wavelet ψ_{j_n} with associated wavelet coefficient d_{j_n}; by doing this, we obtain the representation in (3.6).

The algorithm proceeds by using the minimum updated scaling function integral to choose the next coefficient to be lifted and repeating the steps above. For further clarity, suppose we have already performed the transform until stage r. In this situation, we have lifted the coefficients $j_n, j_{n-1}, \ldots, j_r$. We then have the representation

$$f(x) = \sum_{k\in\{n,n-1,\ldots,r\}} d_{j_k}\psi_{j_k}(x) + \sum_{i\in\{1,\ldots,n\}\setminus\{j_n,j_{n-1},\ldots,j_r\}} c_{r-1,i}\varphi_{r-1,i}(x), \tag{3.7}$$

where the d_{j_k} are the detail coefficients corresponding to the $n-r+1$ wavelets ψ_{j_k}. Therefore the whole procedure at each stage can be performed using the objects j_r, I_r (which is a subset of the unlifted coefficients), together with the prediction and update weight vectors $\mathbf{a^r}$ and $\mathbf{b^r}$. Using the interval construction, a full decomposition of a dataset will consist of $n-1$ detail coefficients and one scaling coefficient.

To see the effect of the lifting transform on the scaling and wavelet functions, we set the data function equal to a specific scaling function or wavelet, and then invert the transform. For example, suppose we want to find how the

scaling function $\varphi_{r-1,l}$ ($l \in I_r$) changes. Then we set all d_{j_k} and $c_{r-1,i}$, $i \neq l$ to zero, but with $c_{r-1,l} = 1$. By inverting the lifting steps (3.3) and then (3.1), $c_{r,l} = 1$ and $c_{r,i} = 0$ for $i \neq l$ at the previous level, which implies $c_{r,j_r} = a_l^r$. Hence using the function expansion equation (3.7),

$$\varphi_{r-1,l} = \varphi_{r,l} + a_l^r \varphi_{r,j_r}. \tag{3.8}$$

By integrating this equation, we get formulae for finding the updated scaling function integrals. In this way, do not need to find the actual scaling functions at each stage, but just use the resulting equation

$$\mathcal{I}_{r-1,l} = \mathcal{I}_{r,l} + a_l^r \mathcal{I}_{r,j_r} \tag{3.9}$$

with the vector of finer scaling function integrals to choose the next coefficient to be lifted. Obviously, the other integrals outside the neighbourhood I_r are not affected and so do not change.

A similar calculation with $d_{j_r} = 1$, $d_{j_k} = 0$ ($k \neq j$) and $c_{r-1,i} = 0\ \forall i$ leads to the coefficients $c_{r,i} = -b_i^r$ in the neighbourhood I_r. Then $c_{r,j_r} = 1 - \sum_{i \in I_r} a_i^r b_i^r$. Whence the updated wavelet is computed

$$\begin{aligned} \psi_{j_r} &= \left(1 - \sum_{i \in I_r} a_i^r b_i^r\right) \varphi_{r,j_r} - \sum_{i \in I_r} b_i^r \varphi_{r,i} \\ &= \varphi_{r,j_r} - \sum_{i \in I_r} b_i^r \left(\varphi_{r,i} + a_i^r \varphi_{r,j_r}\right) \\ &= \varphi_{r,j_r} - \sum_{i \in I_r} b_{j_r}^r \varphi_{r-1,i}. \end{aligned} \tag{3.10}$$

Note that after one step of the transform, the new scaling functions are a combination of the present scaling function and the old scaling function, and in the case of the wavelet, it is a linear combination of the unlifted wavelet and the new *coarser* scaling functions. This is different from the standard wavelet multiresolution analysis, where the wavelet is a combination of scaling

functions on the previous (finer) level.

Notice also, that integrating the equation (3.10) leads to the equation (3.6), but with the left-hand side equal to zero. This shows that the update stage assumption (3.4) is equivalent to requiring that the resulting wavelet produced after lifting has integral zero.

Similarly, the dual scaling functions and wavelets change as follows

$$\begin{aligned} \widetilde{\varphi}_{r-1,l} &= \widetilde{\varphi}_{r,l} + b_l^r \widetilde{\psi}_{j_r} \quad \text{for } l \in I_r, \\ \widetilde{\psi}_{j_r} &= \widetilde{\varphi}_{r,j_r} - \sum_{i \in I_r} a_i^r \widetilde{\varphi}_{r,i}. \end{aligned}$$

Remark. In the description above, we construct the intervals partitioning the support of f by using the midpoints between adjacent x-values as the interval endpoints. With this construction, there is the issue of where to set the endpoints of the first and last intervals. There are different possibilities to resolve this matter, but in our computations, we have reflected the end intervals, so that the first and last datapoints are in fact the midpoints of their associated intervals.

Based on the literature on transforms on the real line [38], another method of interval construction is to create the intervals with the datapoints being the endpoints of the intervals. With this construction, we are faced with the problem of defining the last interval, including assigning its length (since the only information we have about this interval is its left endpoint).

3.2.1 Single coefficient lifting in more than one dimension

One can generalize the single coefficient lifting into higher dimensions, as proposed in [59, 60].

For example, in two dimensions, we create a triangulation on the data grid plane by joining the datapoints with non-overlapping edges, under the constraint that the circumcircle of the triangles (the circle passing through the three points of the triangle) does not contain any other point of the triangulation. This is the *Delaunay triangulation.*

Onto this grid, a tessellation of cells is overlaid. For a given vertex of the triangulation, mark out cells such that the cells mark out the area closer to a given triangulation vertex than any other vertex. This cell tessellation is called the *Voronoi diagram.*

The cells of the Voronoi diagram are associated with scaling functions, their areas being the support of the scaling functions. As with before, the scaling coefficients at the start of the transform are just the data points (for convenience we choose the initial scaling functions as the characteristic functions of the Voronoi polygons). At each step of the transform, the datapoint corresponding to the smallest scaling function integral is removed, and the grid and triangulation becomes coarser through retriangulation. Jansen *et al.* [59, 60] suggest prediction through least squares plane minimization, and an update stage of Sibson's Natural Neighbour method.

In the same papers, a multidimensional one coefficient at a time lifting scheme is introduced using minimal spanning trees. With this notion, a weight is associated to each node of the tree, which represents the scaling function integrals above. The neighbouring nodes are used to predict the value at a particular node. This tree-based lifting is suitable for any higher dimensional space.

3.3 Adaptive lifting schemes

In this section, we introduce adaptive single coefficient lifting schemes, which we will show to perform well against other nonparametric techniques. We first review other existing adaptive lifting schemes in this area.

Claypoole *et al.* [23] introduce adaptivity into the prediction stage of a lifting scheme for image compression. The $(1, N)$ Cohen-Daubechies-Feauveau family (see [26]) is used as a set of predictors, and when lifting, a suitable wavelet is chosen so that if an edge is detected in an image, the wavelet's support does not overlap the edge. When using the algorithm for lossy coding, the update step is applied first. The rationale behind this is so that information about the chosen predictor is not sent, and also so that the update stage preserves frequency localization. A quantiser is applied to all the scaling coefficients (computed from the update stage). These quantised coefficients are then used in the prediction step. The detail coefficients produced from this are then quantised themselves and transmitted.

The algorithm's "update first" policy was first adopted in [22]. This paper introduces two adaptive transforms. The first adapts the wavelet according to the signal features at each scale. The second selects the wavelet which minimizes the detail coefficient at each stage. The algorithms are compared to classical wavelet denoisers on the Donoho and Johnstone test functions. The simulations show that the adaptive transform perform similarly, and slightly better in some cases.

The "update first" approach is also employed by [84]. However, adaptivity is added to the *update* stage of the lifting transform, not the predict stage as in [23]. The performance of the transform is compared to its non-adaptive version on a few signals.

Adaptiveness is also introduced into the prediction lifting step in [97], using Wiener filtering to minimize the l^2-norm of the signal being decomposed. The transform was applied to an AR(2) process to decorrelate its low-pass and high-pass subbands. The algorithm was used again to smooth a noisy version of the same process.

Boulgouris *et al.* [14] develop an adaptive lifting scheme for the still image lossless compression.

The adaptive transforms above all use the odd/even split as described in Example 2.15. However, our approach is to modify the "one coefficient at a time" lifting scheme outlined above and featured in [59, 60]. Our motivation being that the wavelets will be individually tuned to the signal particularities. In addition, our algorithms will be able to handle multiple observations for each x-value.

3.3.1 Adaptive 1D single coefficient lifting transforms

Most work on adaptive lifting has been on images (2D), so introducing adaptivity to a 1D lifting transform could be beneficial. As mentioned before, the main aim of our lifting transforms is to take advantage of the *local* properties of a signal to create the sparsest wavelet representation. Lifting one coefficient at a time lends itself particularly to this, since we can add features to every wavelet in the representation one at a time. We now describe how we propose to do this, through adaptive prediction lifting steps.

Prediction schemes using linear regression

In the last section, we deferred the question of prediction neighbourhood choice. We now return to it. When deciding on how to obtain a detail coefficient, we could use *symmetrical* neighbours, that is, using an equal number of neighbours on either side of the point to be lifted, and perform (linear) regression to get a least squares prediction of the removed point. Alternatively, we could choose the neighbouring regression points according to being the closest datapoints to the lifted point (recall that we assume that our x-values are irregularly-spaced). Both choices are catered for in our software implementation of the 1D lifting transform, with the possibility of any number of total neighbours being used for the prediction. This choice can be made according to some prior knowledge about the signal structure.

The prediction of a lifted point is done through regression (commonly used

in statistical data modelling). We use orders of up to three in our algorithms. This corresponds to linear, quadratic and cubic regression using the neighbour x-values as explanatory variables. This adds vanishing moments to the wavelet in question, which will aid in sparsity where the signal is locally smooth.

Recall that computationally, least squares regression can be formulated as a matrix equation

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{3.11}$$

where $\mathbf{y}$ is a vector of responses, $\mathbf{X}$ is the design matrix for the problem, $\boldsymbol{\beta}$ is the vector of fitting polynomial parameters, and $\boldsymbol{\varepsilon}$ is a (zero mean) vector of errors with constant variance.

In our context, suppose we are performing a prediction of point j_r with m_r total neighbours (irrespective of being symmetrical or nearest) in I_r. Then if the order of regression is p, the equation (3.11) will be

$$\begin{pmatrix} c_{r,1} \\ c_{r,2} \\ \vdots \\ c_{r,m_r} \end{pmatrix} = \begin{pmatrix} 1 & x_1 & x_1^2 & \dots & x_1^p \\ 1 & x_2 & x_2^2 & \dots & x_2^p \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{m_r} & x_{m_r}^2 & \dots & x_{m_r}^p \end{pmatrix} \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_p \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_{m_r} \end{pmatrix}. \tag{3.12}$$

Here, $\{c_{r,i}\}_{i\in I_r}$ are the scaling coefficients of the neighbours, and $\{x_i\}_{i\in I_r}$ are the x-values of the neighbours. This equation corresponds to the case when an intercept is used. If no regression intercept is required, then the design matrix will not have the initial column of ones, and $\boldsymbol{\beta}$ will be the $p \times 1$ vector without β_0. Note that with our lifting schemes, we only consider the cases when $p = 1, 2$ and 3 corresponding to linear, quadratic and cubic regression respectively. For example, suppose we were using a prediction with two symmetrical neighbours. If quadratic prediction is used with intercept, $\mathbf{X} = \begin{pmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \end{pmatrix}$, and $\boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix}$; if cubic regression without intercept is used,

$\mathbf{X} = \begin{pmatrix} x_1 & x_1^2 & x_1^3 \\ x_2 & x_2^2 & x_2^3 \end{pmatrix}$, and $\boldsymbol{\beta} = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{pmatrix}$.

Note that at each stage, the matrix equation (3.12) will be specific to the neighbourhood used for prediction.

The usual least squares solution to the parameter problem (3.12) is given by

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}, \tag{3.13}$$

where $\mathbf{X^T}$ is the transpose of $\mathbf{X}$. This gives us the coefficients of the regression curve fitted according to the neighbourhood around the lifted point, I_r, using equation (3.11).

Our prediction is then obtained by projecting the lifted point onto this regression curve. If the regression is with intercept, this is

$$\widehat{c}_{r,j_r} = (1\ x_{j_r} \ldots\ x_{j_r}^p)\,\widehat{\boldsymbol{\beta}}.$$

Thus the detail coefficient d_r is calculated from $c_{r,j_r} - \widehat{c}_{r,j_r}$. This is the same as the prediction of the form (3.1), since the prediction weights $\mathbf{a^r}$ are obtained from

$$\mathbf{a^r} = (1\ x_{j_r} \ldots\ x_{j_r}^p)(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T.$$

Adding adaptivity to prediction schemes

In the single coefficient algorithm described, the user has the choice as to how many neighbours to use, and also which order of regression to use in the prediction step. In addition, the neighbourhood configuration (symmetrical or nearest neighbours) needs to be specified. There are few situations when the user would know *a priori* or have an intuition how to decide between these options.

Let us now describe our two adaptive algorithms. These allow for automatic

selection of some of the choices above. In this way the wavelets are tuned to the local features of the signal.

AdaptPred. After a point to be removed has been chosen, the detail coefficients are computed for all possible combinations of predictions with linear, quadratic, or cubic regression; the regression is also considered with or without an intercept. The prediction scheme corresponding to the minimum detail coefficient in absolute value is chosen at each step. The user inputs the number of neighbours to use, and also the neighbourhood configuration (symmetrical or nearest neighbours). For example, suppose *AdaptPred* was chosen with three closest neighbours. Then the algorithm would compute the detail coefficients resulting from linear prediction with three neighbours; quadratic prediction using three neighbours, and cubic prediction using three neighbours. This would be done using an intercept in the regression, and then also not using an intercept, and the smallest detail coefficient in absolute value would be selected and the combination of prediction and intercept recorded which produced it.

AdaptNeigh. As well as the *AdaptPred* algorithm, we introduce more flexibility by allowing the neighbourhood size and configuration to change at each step. In this way, the wavelets' support and smoothness are local and particular to the lifted point. The neighbourhoods under consideration at each step are symmetric configurations up to and including a specified number of neighbours and neighbourhoods with neighbours up to twice the specified number (for nearest neighbour configurations). Again, the prediction scheme is chosen according to the minimum detail coefficient in absolute value. Essentially, *AdaptNeigh* performs the *AdaptPred* procedure for the different neighbourhood choices, and minimizes the detail coefficient from all implementations of *AdaptPred*.

The linear and adaptive algorithms described above show $\mathcal{O}(n)$ computa-

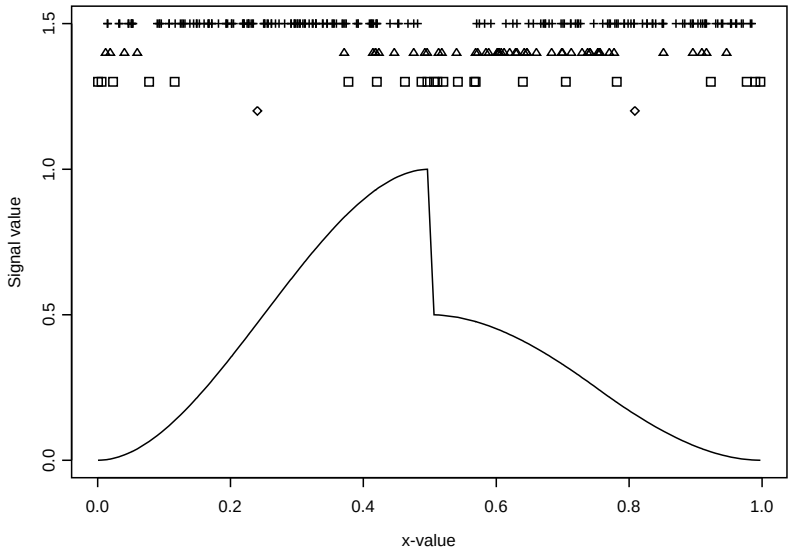

Figure 3.1: Plot showing choice of prediction scheme for the *Ppoly* test signal decomposed with AN2 on an irregular grid. Horizontal placement of symbol indicates location of following kinds of prediction: linear (□); quadratic (△); cubic (+); scaling functions (◇).

tional efficiency.

Example 3.1. The procedure AdaptNeigh with two neighbours was used to decompose the Donoho and Johnstone test functions *Ppoly* and *Bumps*. Figures 3.1 and 3.2 show how the order of regression changes for different parts of the signals.

The plots show that the adaptive lifting scheme is choosing the basis function as expected. For the *Ppoly* signal, cubic regression is chosen most of the time, apart from when linear prediction steps are applied around the point of discontinuity. With the *Bumps* signal, the regression scheme chosen across the signal is more variable than for the *Ppoly* signal. However, near the discontinuities linear regression is performed, since the function is similar to high gradient lines at the spikes.

Figure 3.3 shows examples of the basis functions occurring when the *Bumps*

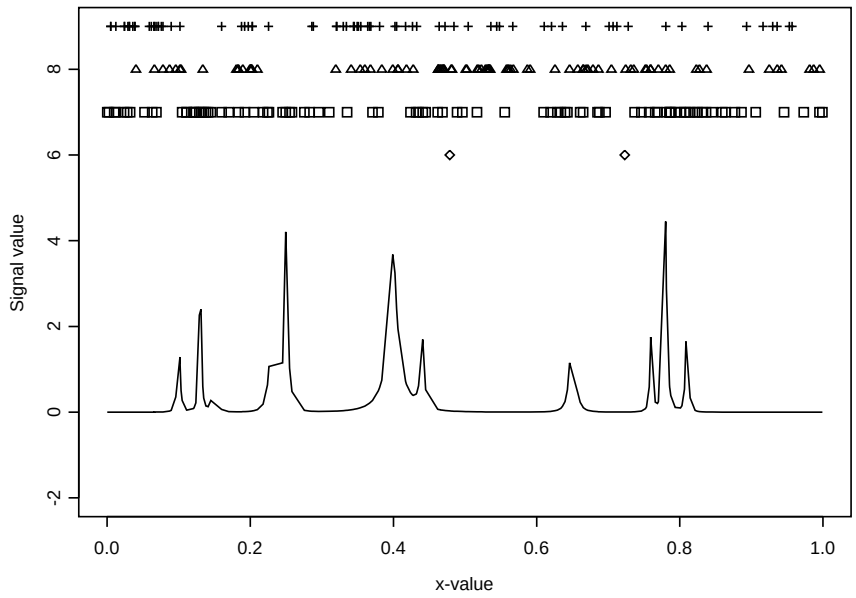

Figure 3.2: Plot showing choice of prediction scheme for the *Bumps* test signal decomposed with AN2 on an irregular grid. Horizontal placement of symbol indicates location of following kinds of prediction: linear (□); quadratic (△); cubic (+); scaling functions (◇).

signal was decomposed on an irregular grid with AdaptNeigh and two neighbours. The plots demonstrate that the decomposition basis functions can vary in magnitude and support. They can also exhibit different smoothness and frequency behaviour.

Implications of the adaptive lifting scheme construction

There are a few issues which arise from our adaptive lifting scheme construction which should be addressed.

Prediction weights. As noted in the description of lifting "one coefficient at a time", following Jansen *et al.* [59], the prediction weights $\mathbf{a^r}$ sum to one. Signals which are piecewise constant over the prediction neighbourhood I_r will result in zero detail coefficients. When using least squares prediction schemes as outlined above, this will occur if a regression intercept is included as one of the parameters. If an intercept is not used, then it is not guaranteed that the

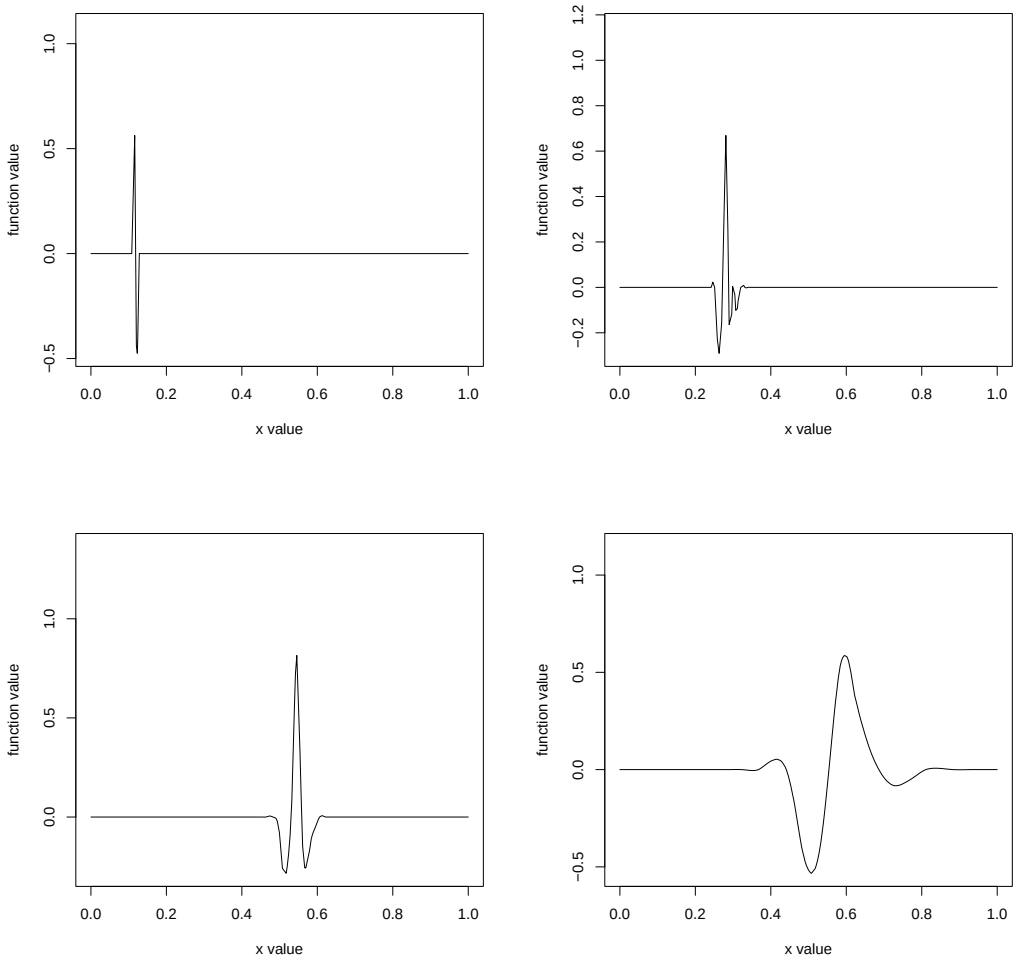

Figure 3.3: Plot showing examples of basis functions produced by the AdaptNeigh adaptive algorithm with two neighbours. The appearance of the basis functions can differ dramatically, depending on the grid structure and prediction schemes chosen.

weights $\mathbf{a^r}$ will sum to one.

When using closest neighbours, it is possible that (some of) the prediction weights will take negative values. As a consequence, the neighbouring intervals corresponding to the negative prediction weights will decrease in size, instead of increasing. This disagrees with the notion that all neighbouring associated scaling functions will have greater support when the interval corresponding to the lifted point is redistributed over the neighbourhood. Jansen *et al.* [60] notes that the scale of wavelet functions (measured by the scaling function integral $\mathcal{I}_{r,j_r}$) is a monotonic function of the index r. In the case when we have negative prediction weights, this does not hold.

Insufficient neighbours. The user-specified choice of the neighbourhood configuration to be used in the prediction stage may not be possible in some cases. When the removed point is on the boundary, to avoid using artificial boundary neighbours (for example, if symmetrical neighbours are required), the nearest (single) neighbour is used for prediction. Also, when many points of a dataset have already been lifted, there may not be enough unlifted coefficients for the neighbourhood configuration required by the user. In this situation, the number of neighbours is decreased so that the maximum available can be used.

Regression order reduction. Non-degenerate curves ensure transform stability. For higher order prediction schemes, namely quadratic and cubic regression neighbourhoods, more neighbours are needed for non-degenerate regression curves: for linear prediction, at least two neighbours are needed; for constructing a parabola, at least three, and for cubic prediction schemes, at least four neighbours are needed. At some steps of the lifting transform, we will have too few neighbours to use the required prediction regression order. In these steps, the regression order is decreased to be able to perform the lifting. This means that even though a fixed (higher) prediction order is chosen by

the user, the transform in reality, can be a mixture of orders (depending on when the transform is stopped). Hence the order of the underlying MRA for the transform will not be exactly 2, 3 or 4.

Nonlinearity of adaptive algorithms. Recall that for prediction schemes which use least squares regression, the prediction weights are computed using the grid values only. This means that for a fixed prediction order (and neighbourhood configuration), the transform is linear. However, when we introduce adaptivity into the transform design, the type of prediction used adapts to the local signal structure. Hence the associated prediction *weights* will be signal dependent. Since the function integrals and update weights are changed from this, the point chosen for removal next is also influenced by the signal, f. The transform operator associated to the adaptive transforms is therefore not linear, and is a function of the signal. This means that any measure of stability, for example the matrix condition number defined in earlier sections, will change according to the function values $\mathbf{f}$. Hence the characterization of the adaptive lifting algorithms by condition numbers is unsuitable, though some overall measure of stability of some set Ω of function vectors $\{\mathbf{f}\}_{\mathbf{\Omega}}$ could be formulated.

Stability. We studied the stability of the adaptive lifting transforms, by constructing the transform matrices for a range of test functions, grid irregularities and (adaptive) lifting algorithms. We then computed the matrix condition numbers as described in Section 2.2.5.

The results of the study were revealing. They suggest that higher order prediction schemes were more unstable than when using linear regression. Moreover, larger neighbourhoods (using more neighbours) introduced more instability as well. This agrees with the conclusions of [91] and [98] pointed out earlier. However, there was not an appreciable difference in stability when increasing the irregularity of sample grids.

Due to these findings, and in light of the sparsity and denoising simulations in Sections 3.4 and 3.5, the adaptive lifting algorithms with two neighbours (AP2T, AP1S and AN1 – abbreviations explained later) are recommended in practice.

3.3.2 The inverse single coefficient lifting transform

When inverting a fixed prediction transform, the transform simply reverses the steps performed during the forward transform, using inverse lifting steps as made explicit in Section 2.2. However the order of removal of lifted points and the removed scaling integral lengths is required.

For AdaptPred, the inversion needs the prediction scheme used at each step of the transform (regression order and whether an intercept was used or not).

When an adaptive lifting scheme is inverted, in addition the information mentioned above, the neighbourhood configuration of each lifted coefficient is also needed by the inverse transform. This basically means the (indices of the) neighbours of each lifted point, since otherwise the removed point's position relative to its neighbours could not be determined.

3.3.3 Single coefficient lifting transform for multiple observations

We have not yet explored the extension of lifting "one coefficient at a time" for data with more than one observation at each x-value, for use with datasets such as the motorcycle data. The algorithms work the same for multiple point data, but with some important differences.

If the multiple observation datapoints were just treated as separate individual datapoints, then the algorithm would cause these multiple points to have zero integrals, since the interval construction would cause their intervals to coincide. As a consequence, the multiple points would be chosen to be

removed first, according to the minimal scaling function integral condition of removal point selection. Hence the datapoints with more than one observation are treated to be of the form $(x_i, \mathbf{f_i})$, i.e. one x-value but with a vector of function values.

When predicting the value of a removed point, if the neighbours are multiple points, no modification is necessary, since the linear regression curve fitting computations will still hold with these extra points.

When the removed point is itself a multiple point, then this will result in possibly more than one detail coefficient for that point, due to the vector of observations/scaling coefficients at that point. Cases of multiple detail coefficients would lead to problems when updating neighbour scaling coefficients, so for these points we compute one detail coefficient, by averaging all the detail coefficients corresponding to the multiple observations. Thus the lifted point is replaced by one value. Another interesting treatment of multiple detail coefficients would be to choose the minimum detail coefficient in absolute value as the associated detail coefficient, but this possibility is not treated here.

In the update step of the transform, when the neighbours have multiple scaling coefficients, all coefficients are updated using the detail coefficient computed from prediction.

If the forward transform is performed until a resolution level which leaves (unlifted) multiple points, then before inversion, these points are averaged to form a vector of single coefficients from which to invert. Also, similar to the inverse transform in the normal situation, the neighbourhood information for each lifted coefficient (including whether the neighbours were multiple or not) is used.

3.4 Sparsity performance of lifting algorithms

In the discussion below, we summarize the results of our sparsity simulation study. The simulations were carried out on five test signals and three grid jitters. Neighbourhoods of sizes up to four neighbours were considered. The full study is detailed in the next chapter.

We use abbreviations for the single coefficient lifting schemes according to their prediction scheme as follows: LP=Linear Prediction, QP=Quadratic Prediction, CP=Cubic Prediction. For the adaptive algorithms, the codes AN=AdaptNeigh and AP=AdaptPred are used. The methods are also coded by a number, N, which gives the number of neighbours used in the prediction. The neighbourhood configuration is also described by an extra letter, which is denoted by S in the case of symmetrical neighbours (regression is performed using an equal number of datapoints on either side of lifted points) or N in the case of when the neighbours were chosen according to being the closest to the removed point. It should be noted that when symmetrical neighbours are used, then the actual total number of neighbours is twice the number given by N. For example, LP1S denotes (fixed) linear prediction, with the regression being performed with one neighbour on either side of the removed points. Note that the AdaptNeigh methods do not need a characterization by either N or S, since they are adaptive over *all* configurations with a given number of neighbours.

3.4.1 Sparsity results

Out of the non-adaptive lifting transforms, the fixed linear prediction schemes with two neighbours (LP2N, LP1S) shows the best sparsity. The LP methods also have small matrix condition numbers across all three grid jitters (d_1, d_2 and d_3). Increasing the grid irregularity did not affect either the magnitude of the condition numbers or the compression performance for these two algorithms. The adaptive algorithms AP2N, AP1S and AN1 all exhibit small condition numbers, irrespective of the grid jitter, thus showing the algorithms' stabil-

ity. The adaptive algorithms demonstrate better compression than the linear transforms, with AdaptNeigh methods being the best in terms of sparsity. As justified earlier though, the algorithms with larger neighbourhoods have the potential of being unstable, and hence AN1 is recommended as the algorithm of choice, since it achieves a good balance between stability and sparsity. The other adaptive methods AP2N and AP1S are also good. These results hold for all the test functions, even though on *HeaviSine*, there was less difference between the three algorithms AN1, AP2N and AP1S. The grid irregularity did not alter the sparsity performance of the adaptive algorithms. The adaptive algorithms are also competitive when compared with the classical Daubechies Extremal Phase wavelets and KS algorithms.

3.5 Adaptive lifting and wavelet shrinkage

We now explore the method for signal smoothing with our adaptive lifting schemes. Again we refer to the model (2.45)

$$f_i = g(t_i) + \varepsilon_i \quad \text{for } i \in \{1, \ldots, n\},$$

where the vector $\mathbf{f}$ are our observations. $\boldsymbol{\varepsilon}$ is taken here to be independently identically distributed with distribution $N(0, \sigma^2)$. This means that $f_i \sim N(g_i, \sigma^2)$. In Section 2.3, we noted that using the DWT, this equation transforms into (2.49)

$$d_{j,k} = d^*_{j,k} + e_{j,k},$$

where $\mathbf{d}$ is the DWT of $\mathbf{f}$, $\mathbf{d}^*$ is the DWT of $\mathbf{g}$ and $\mathbf{e}$ is the DWT of $\boldsymbol{\varepsilon}$. In other words, if $\widetilde{W}$ is the forward transform matrix, the equation above is equivalent to

$$\widetilde{W}\mathbf{f} = \widetilde{\mathbf{W}}\mathbf{g} + \widetilde{\mathbf{W}}\boldsymbol{\varepsilon}. \tag{3.14}$$

Note that here, we use $\widetilde{W}$ for the forward transform matrix, for generality (even though when talking about the DWT in earlier sections we used W). Due to the orthogonality of the discrete wavelet transform, the transformed noise $\mathbf{e}$ is also i.i.d. Gaussian $N(0, \sigma^2)$ vector. With the DWT, there is no correlation between the coefficients, i.e. the variance-covariance matrix is just the noise σ^2 multiplied by the $(n \times n)$ matrix identity. The noisy detail coefficients can be seen to be distributed as $d_{j,k} \sim N(d^*_{j,k}, \sigma^2)$.

For our linear transforms (LP, QP and CP), the equation (3.14) also holds, but with the alteration that the noise vector after performing the wavelet transform is now $\{e_{j_k}\}_{k \in \{n, n-1, \ldots, l\}}$ (for some l), the equivalent vector for single coefficient lifting. For the 1D single lifting schemes in Section 3.2, a full decomposition would be with $l = 2$. The stopping time of our lifting transforms is explored extensively in Chapter 5.

For the adaptive lifting algorithms, the transform matrices depend on the observations $\mathbf{f}$ as well as the grid locations $\{x_i\}$. Since they are nonlinear, we cannot write the transform of $\mathbf{f}$ as a combination of the form of (3.14). However, we still assume the detail coefficients to satisfy equivalent form of equation (2.49):

$$d_{j_k} = d^*_{j_k} + e_{j_k} \quad k \in \{n, n-1, \ldots, l\}. \tag{3.15}$$

In other words, when we transform the vector $\mathbf{f}$, the resulting coefficients are assumed to be noisy and of the form (3.15), where the vector $\mathbf{e} = \{e_{j_k}\}$ is the noise. Conditioning on the local structure of the signal $\mathbf{f}$, though, the noise can be written $\mathbf{e} = \widetilde{\mathbf{W}}\boldsymbol{\varepsilon}$, and since the original noise vector $\boldsymbol{\varepsilon}$ is i.i.d. normal noise, the vector $\mathbf{e}$ will also be normal noise with zero mean. Consequently, the detail coefficients produced by the transform are also normally distributed.

However, since out transforms are not orthogonal, the transformation induces a correlation structure between wavelet coefficients, and so the coefficients will have different variances. The correlation structure of the noise can

be described as $\Sigma_{\mathbf{e}} = \sigma^2 \widetilde{W}\widetilde{W}^T$. This variance-covariance structure transfers to the detail coefficients to give $\Sigma_{\mathbf{d}} = \sigma^2 \widetilde{W}\widetilde{W}^T$, so that the variance of the detail coefficient d_{j_k} is $\sigma^2 \operatorname{diag}(\widetilde{W}\widetilde{W}^T)_k$, where $\operatorname{diag}(A)_k = a_{kk}$ for a square matrix $A = [a_{ij}]$.

3.5.1 Correlation induced from lifting steps

What follows explains the correlation structure inherent in our lifting transforms. This section is a development of considerations in Jansen *et al.* [60]. Since the adaptive algorithms are not linear (and so dependent on the signal $\mathbf{f}$), the working below should be taken to be conditional on the local structure of the signal, $\mathbf{f}$.

At the first step of the lifting transform, the coefficient j_n is predicted by (3.1) $d_{j_n} = c_{n,j_n} - \sum_{i \in I_n} a_i^n c_{n,i}$. Due to the assumption that the initial data $\mathbf{f}$ is i.i.d.,

$$\begin{aligned} \operatorname{cov}(c_{n,i}, d_{j_n}) &= \operatorname{cov}(c_{n,i}, c_{n,j_n}) - \sum_{i \in I_n} \operatorname{cov}(c_{n,i}, c_{n,i}) \\ &= -a_i^n \sigma^2 \quad \text{for } i \in I_n. \end{aligned} \tag{3.16}$$

For $i \notin I_n$, $i \neq j_n$,

$$\operatorname{cov}(c_{n,i}, d_{j_n}) = 0,$$

and for $i = j_n$,

$$\operatorname{cov}(c_{n,i}, d_{j_n}) = \sigma^2.$$

We also obtain

$$\operatorname{var}(d_n) = \sigma^2 \left\{ 1 + \sum_{i \in I_n} (a_i^n)^2 \right\}. \tag{3.17}$$

Also, using update step (3.3), we have, for all $i \in I_n$, $i \neq j_n$,

$$\operatorname{var}(c_{n-1,i}) = \operatorname{var}(c_{n,i}) + (b_i^n)^2 \operatorname{var}(d_{j_n}) + 2 b_i^n \operatorname{cov}(c_{n,i}, d_{j_n}). \tag{3.18}$$

We also have, for $i, j \in I_n, i \neq j, \quad i, j \neq j_n$,

$$\begin{aligned}
\operatorname{cov}(c_{n-1,i}, c_{n-1,j}) &= \operatorname{cov}(c_{n,i}, c_{n-1,j}) + b_i^n \operatorname{cov}(d_{j_n}, c_{n-1,j}) \\
&= (0 + b_j^n \operatorname{cov}(c_{n,i}, d_{j_n})) + (b_i^n \operatorname{cov}(d_{j_n}, c_{n,j}) + b_i^n b_j^n \operatorname{cov}(d_{j_n}, d_{j_n})) \\
&= (-a_i^n b_j^n - a_j^n b_i^n)\sigma^2 + b_i^n b_j^n \operatorname{var}(d_{j_n}). \qquad (3.19)
\end{aligned}$$

For when $i \in I_n$, $j \notin I_n$, $i, j \neq j_n$ we have

$$\begin{aligned}
\operatorname{cov}(c_{n-1,i}, c_{n-1,j}) &= \operatorname{cov}(c_{n-1,i}, c_{n,j}) \\
&= 0. \qquad (3.20)
\end{aligned}$$

Since the update step does not affect coefficients outside the prediction neighbourhood, for any $i, j \notin I_n$, $\operatorname{cov}(c_{n-1,i}, c_{n-1,j}) = 0$.

All of this shows that the lifting steps create correlations between the coarser level coefficients produced, and that these correlations will transfer down through later levels of the transform.

3.5.2 Empirical Bayes shrinkage with adaptive lifting

Recall from Section 2.3 that the general approach to wavelet shrinkage is

- transform the noisy data $\mathbf{f}$ into detail coefficients
- threshold the detail coefficients to remove the noise
- invert the transform to obtain an estimate of the underlying signal.

In our denoising simulations, we have used a modified version of the empirical Bayes thresholding technique as proposed in [61, 62, 63]. These techniques adapt well different situations. The authors of these papers exploit the properties of sparse coefficient sequences produced by the DWT by imposing a prior

on the true wavelet coefficients

$$d^*_{j,k} \sim (1-\pi)\delta_0 + \pi\gamma,$$

where the prior probability that a detail coefficient is nonzero is given by π, and the conditional density of the coefficient is the function γ. Section 2.3.2 gives a review of this thresholding method in more detail, and describes how to compute the probabilities π and associated thresholds. Since from Section 3.4, our lifting wavelet methods are shown to transform signal data to sparse coefficients sets, the *EbayesThresh* methodology is appropriate for our transforms. In our denoising procedures, the "quasi-Cauchy" prior (see [61, 63]) is taken for γ, and the posterior median is chosen as the thresholding method.

We now give some adjustments to this Bayesian technique to make it suitable for our lifting algorithms.

Artificial levels

In previous sections, we have associated the scale of a wavelet coefficient d_{j_r} as the integral $\mathcal{I}_{r,j_r}$, following Jansen *et al.* [60], since the "scale" in "one coefficient at a time" lifting is no longer a purely dyadic notion. However, to mimic the dyadic scale structure of the DWT, we can partition the wavelet coefficients into *artificial levels* as suggested in [59, 60]. Arranging the detail coefficients in ascending order of their scale, we can define the "finest" level detail coefficients by the first half of the scales (those with scale $\mathcal{I}_{r,j_r}$ less than the median of all scales). If we then find the upper quartile of the scales, we have the next finest level as those detail coefficients corresponding to the scales $\mathcal{I}_{r,j_r}$ between the median and this upper quartile. For coarser levels we proceed in this way. The next level would be defined by the 87.5th quantile, and so on.

We can now use the level-dependent empirical Bayesian thresholding procedure. We assume each coefficient in a specific artificial level has the same

probability of being non-zero (π_j), and so we compute a threshold for each artificial level.

Correlation treatment

In Section 3.5.1, we saw that propagating correlations between decomposition coefficients are introduced by the lifting steps during a single coefficient lifting wavelet transform, and the non-orthogonality of our lifting transforms leads to the detail coefficients with different variances, even though the initial observations all have equal variance, σ^2. The correlation structure of the coefficients is ignored, as in [59, 60]. However, to compensate for the different coefficient variances whilst using the *EbayesThresh* procedure, we make the following modifications.

We noted before that the variance of the coefficient d_{j_k} can be described as $\sigma^2 \operatorname{diag}(\widetilde{W}\widetilde{W}^T)_k$. Since *EbayesThresh* is designed to work on equal variance coefficient vectors, Jansen *et al.* suggest preprocessing the detail coefficients before thresholding. The coefficients defined by $d_{j_k} \left\{\operatorname{diag}(\widetilde{W}\widetilde{W}^T)_k\right\}^{-1/2}$ will all have equal variance of σ^2, and hence we apply the thresholding procedure to these coefficients. We then renormalize the thresholded coefficients by multiplying by $\left\{\operatorname{diag}(\widetilde{W}\widetilde{W}^T)_k\right\}^{1/2}$. These coefficients can then be inverted to form an estimate of $\mathbf{g}$.

Other modifications: heteroscedastic coefficient variances

Our work has also considered unequal coefficient variances. This has lead us to also adapt the empirical Bayes technique for when the coefficient variances are heteroscedastic, but known up to proportionality, i.e. $\operatorname{var}(f_i) = \sigma^2\gamma_i^2$, where the variance is unknown, but the proportionality constants γ_i are known. After the lifting transform the variance of the detail coefficients are then calculated as $\operatorname{var}(d_{j_k}) = \sigma^2 \sum_{i\in\{1,\ldots,n\}} \gamma_i^2 \widetilde{W}_{j,i}^2$, where $\widetilde{W}_{j,i}$ is the (j,i)th entry in $\widetilde{W}$. As with the previous situation with different coefficient variances, we divide the

wavelet coefficient sequence so that it becomes homoscedastic, and suitable for empirical Bayes' thresholding using *EbayesThresh*. In this case, the normalizing factor is $\sqrt{\sum_{i\in\{1,\ldots,n\}} \gamma_i^2 \widetilde{W}_{j,i}^2}$. Since the quantity σ is unknown, we estimate it using the MAD of the coefficients from the first artificial level, as suggested in [67]. After thresholding and renormalization, the transform is inverted.

If we do not have any prior knowledge about the heteroscedastic variances, we use the approach in [67] to estimate the variance $\sigma_{d_{j_k}}$. They propose using a window centered on the datapoint x_{j_k} to identify any detail coefficients in the finest artificial level which correspond to x-values which lie close to x_{j_k}. The detail coefficients which occur in this window and are in the finest artificial level are then used to estimate the variance through the MAD of these coefficients. After thresholding, inverting the transform will lead to an estimate of the true signal.

To summarize, to use *EbayesThresh* for our lifting transforms, we use the following steps

1. Transform the data $\mathbf{f}$ with a lifting algorithm;
2. Use the modifications above to deal with heteroscedastic variances;
3. Partition the detail coefficients into artificial levels, and if necessary use the finest level coefficients to estimate the overall noise variance σ^2;
4. Threshold the modified detail coefficients level-dependently according to the artificial levels, using *EbayesThresh*;
5. Invert the lifting transform to obtain an estimate of the signal.

3.5.3 Denoising simulation results

In this section we give a summary of the conclusions from the simulation study we performed to test the denoising capability of the lifting transforms. Full

details of the results obtained can be found in the next chapter.

We compared our smoothing techniques against wavelet and non-wavelet function estimators: the local polynomial fitting estimator *Locfit* [69, 70]; the S-Plus function `smooth.spline()`, which fits smoothing splines to the data; the Comte-Rozenholc [30] (CR) method, a denoiser based on adaptive polynomial basis selection; and the irregular interpolation wavelet algorithm by Kovac and Silverman (KS) [67]. Descriptions of these algorithms can be found at the start of this chapter.

The study was performed on the five test signals *Doppler*, *Bumps*, *Blocks*, *HeaviSine*, and *Ppoly*, over irregular grids with three levels of jitter. We also used three different additive noise levels in the denoising simulations. Several different variants of our linear and adaptive lifting schemes were compared to the denoising methods listed above, with different neighbourhood configurations up to and including four neighbours. The algorithms' performance was also tested on the real examples of the inductance plethysmography data, and also the motorcycle data (described in the next chapter).

Overall, our methods were very competitive, with AN1 being significantly better on the signals with discontinuities over all grid irregularities and noise levels. For smoother signals, AdaptPred worked best, outperforming the other smoothers over all degrees of grid jitters and noise level, apart from one of the signals, where the Kovac-Silverman algorithm was best. These conclusions are expanded in the next chapter. The application of the lifting algorithms to the two real datasets showed that the curves produced are competitive with other data smoothers.

3.6 Conclusions and further work

This chapter proposes a lifting wavelet transform suitable for nonparametric regression on irregularly-sampled data, which is hard to achieve with classical wavelet transforms, since it often involves serious modification to the regular

grid transform framework. Furthermore, we are able to introduce adaptivity into the technique, so that through the use of "one coefficient at a time" lifting, the algorithm can tune every wavelet in the decomposing basis to automatically adapt to the local features of a signal. This facilitates sparse wavelet representations, which can be exploited in denoising applications. The algorithms can be modified for use with datasets with multiple observations for each x-value, a situation which is often overlooked in regression techniques.

Results from sparsity and denoising simulations show that our methods perform very well against other current wavelet and non-wavelet denoising techniques.

Further work on adaptive lifting algorithms could involve trying to use the wavelet coefficient correlation induced by lifting to improve shrinkage techniques. The treatment of multiple point data is still fairly open: other ideas could be incorporated into the lifting algorithms for these datasets. In addition, the issue of transform stability could be addressed more fully, perhaps by building on the suggestions in [91, 98].

Chapter 4

Adaptive lifting simulations

Introduction

When assessing the merits of a wavelet transform, two main qualities are judged to be important: sparsity and denoising capability, and in some sense, these two aspects are strongly linked. Good sparsity indicates efficient representation of functions and good data compression. In many real life situations, noisy data is common, for example, due to experimental or measurement error. It is therefore desirable to be able to "estimate the truth" from the given data, so that this estimate can be a base from which to perform data prediction, forecasting or modelling. Hence the ability to denoise functions well is also essential for a wavelet transform. In this chapter, we detail the results of the simulation study on sparsity and denoising performance of our adaptive lifting algorithms explored in the last chapter.

This chapter is organized as follows. Firstly, we explain the general setup of our simulations, common to both the sparsity and denoising sections. This includes a description of the test functions and the irregular grids used in the study, and also a summary of the notations used in the later sections.

The sparsity section will compare our lifting algorithms against the other nonparametric regression techniques mentioned in the last chapter. We mea-

sure the sparsity of a smoothing method through a *sparsity plot*. This construction is fully described below. The improved sparsity of our lifting transforms over its competitors is supported with plots for different test signals.

The section on the denoising performance of the lifting transforms will show how well our transforms smooth noisy data input. In the extensive study, we compare our methods against the other competitors, on all test signals and grids. We use the numerical error measure of the AMSE, as well as visual evidence. The algorithms are then applied to real examples, to demonstrate their ability to produce suitable function estimates.

4.1 Simulation preliminaries

We now give some initial details and remarks on the nature of our simulations.

4.1.1 Test signals

We will demonstrate the sparsity and denoising capability of our transforms on the Donoho and Johnstone test signals *Doppler*, *Bumps*, *Blocks*, and *HeaviSine*, first used in simulations in [39]. These functions are commonly used as standard test signals for simulations, because they exhibit a range of smoothness properties and also different numbers of discontinuities. Donoho and Johnstone [39] state that these features are typical of functions appearing in scientific applications such as imaging and spectroscopy. As well as these functions, we use the *Ppoly* piecewise polynomial function of Nason and Silverman [76]. All functions are defined on the interval [0,1]. We now give a brief description of these functions.

Blocks. This is a piecewise constant function. There are jumps at the sites $\{0.1, 0.13, 0.15, 0.23, 0.25, 0.4, 0.44, 0.65, 0.76, 0.78, 0.81\}$, which makes the signal difficult to denoise. The blocks have the different heights of $\{0, 4, -1, 2, -2, 3, -1.2, 0.9, 5.2, 4.2, 0\}$ respectively.

Bumps. This function is a combination of "bumps" described by the function $1/(1+|t|)^4$ at the same sites as for *Blocks* above. The heights and widths of the bumps vary.

HeaviSine. A sinusoid with two discontinuities created by jumps at t=0.3 and t=0.72. The function has formula $g(t) = 4\sin(4\pi t) - \text{sgn}(t - 0.3) - \text{sgn}(0.72 - t)$.

Doppler. The well-known signal from sound modelling, described by $g(t) = \sqrt{t(1-t)}\sin(2\pi \cdot 1.15/t + 0.05)$.

PPoly. A piecewise polynomial with a jump discontinuity at t=0.5. The function equation given by

$$g(t) = \begin{cases} 4t^2(3-4t) & [0, \frac{1}{2}) \\ \frac{4}{3}(4t^2 - 10t + 7) - \frac{3}{2} & [\frac{1}{2}, \frac{3}{4}] \\ \frac{16}{3}t(t-1)^2 & [\frac{3}{4}, 1]. \end{cases}$$

4.1.2 Construction of irregular (jittered) grids

To demonstrate the applicability of our transforms to irregular data, we simulated irregularly-sampled x-values from which to sample the test functions.

To generate the irregular grids on which to sample the test signals, we *jittered* a regular grid uniformly with varying degrees of jitter. More precisely, suppose we have a regular grid of n points in the interval [0,1]. Each point is moved from its original location by adding a value randomly taken from a uniform distribution on the interval $(-d/(n+1), d/(n+1))$, where d signifies the amount of *jitter*. We use three values of d: $d_1 = 0.01$; $d_2 = 0.1$; and $d_3 = 1$. The rationale behind generating irregular grids this way is that we can use different values of d to tune the irregularity of the grid: small values of d will be irregular, but will resemble the regular grid, so we can compare

the sparsity of our methods with classical wavelet methods on regular grids. As we increase d, the grids will become "more irregular", until with $d = d_3$, the grid will resemble a grid sampled uniformly from the interval [0,1].

For both the sparsity and denoising simulations, grids of length $n = 256$ were used. This enables us to compare our methods with classical wavelet transforms, even though with our algorithms there is no restriction of grid length.

4.1.3 Lifting algorithm notation

The different prediction schemes are abbreviated by a two letter code: LP for linear prediction; QP for quadratic prediction; CP for cubic prediction; AP for the AdaptPred adaptive lifting scheme and AN for the fully adaptive AdaptNeigh algorithm.

This code is then augmented with a number, denoting the number of neighbours used in the prediction, and then a choice of the letter N or S, describing the neighbourhood configuration. For symmetrical neighbours, the letter S is used, with the number of neighbours, N, being the number of neighbours on one side of the removed point at each lifting step. In other words, when symmetrical neighbours is chosen, the total number of neighbours used in the prediction steps will be twice N. When using closest neighbours, that is, abbreviated by the letter N, the number N will be the total number of neighbours nearest the removed point throughout the transform.

For the adaptive algorithm, we only need to use its code (AN) with the number of neighbours (we take the number N according to the symmetrical neighbourhood specification, so that AN1 will denote the AdaptNeigh algorithms which searches for the minimum detail coefficient from the results of AP1S and AP2T).

4.2 Sparsity investigation

Throughout this thesis, the advantages of sparsity of wavelet decompositions has been stressed, the main benefits of which is the ability to achieve efficient data compression. In this section, the investigation into the sparsity of our lifting transforms summarized in Section 3.4 is detailed.

There are many variants of the lifting scheme which we could use in our investigation of sparsity and denoising, based on different prediction schemes, neighbourhood configuration and number of neighbours used in the prediction. As outlined in Section 3.4, we have devised a notation for our lifting schemes, which we will refer to in this section.

4.2.1 Sparsity plots

To examine the sparsity of the wavelet transforms, we want to measure how error changes with sparsity of wavelet coefficient sequences. To this end, we use a tool called a *sparsity plot*. This is constructed as follows.

First of all, we decompose the sampled signal down to two scaling coefficients*. Then we put the detail coefficients in increasing order of magnitude (in absolute value). Next, we invert the transform, but with the smallest wavelet coefficient replaced with zero. We compute the error between the original signal and this estimate by the ISE. For the next ISE calculation, we compute the error between the sampled signal and the estimate produced from setting the two smallest detail coefficients to zero. We proceed like this until the last time, when we invert the transform with only the two scaling coefficients (all the detail coefficients are zero).

The number of non-zero wavelet coefficients is then plotted against the corresponding ISE value. In other words, if i denotes the number of non-zero

*Even though one more lifting step can be performed, we consider this last step as somewhat forced, so we view $n-2$ as the maximum number of detail coefficients for the interval single coefficient lifting construction.

wavelet coefficients, $i = 0$ corresponds to the estimate with only the scaling coefficients, and we have $\text{ISE}(i) = \sum_j (f_j - \widehat{f}_j^{\,i})^2$, where $\widehat{f}^{\,i}$ is the reconstruction based on i non-zero wavelet coefficients.

For each jitter value, the results illustrated will be the average performance over 50 jittered grids, i.e. the plots show the average of the 50 ISE sparsity curves.

The motivation for this construction is that when i is low, there should be only a small difference between $\text{ISE}(i)$ and when $i = 0$, with this difference increasing dramatically as i approaches the grid length.

Remark. We expect that the sparsity plot will show a decreasing trend in the error of reconstruction as i increases. However, because of possible instability of our lifting algorithms, due to aspects mentioned before in Section 3.3.1 (e.g. higher order prediction schemes and large neighbourhoods), the ISE may not necessarily be a decreasing function of i.

We compare our results with the Kovac-Silverman method for irregular data [67], and also the classical DWT. We denote the Kovac-Silverman method by KS. A few remarks should be made about the sparsity constructions for these wavelet transforms.

KS method. The Kovac-Silverman method works on irregular grids, so is suitable for comparison with the lifting transforms. With this algorithm, there is a choice of wavelet basis to use in the wavelet decomposition. We use KS with Daubechies Extremal Phase wavelets, featured in Section 2.6. For the choice of decomposition wavelet, we used the best wavelets as judged by the denoising study in [80]. This choice is discussed fully in the next section, where we assess the denoising performance of wavelet transforms.

Moreover, as described in Section 3.1, the KS algorithm works by transforming the irregular grid data to a regular one by linear interpolation. A drawback to the KS method is that after the signal is decomposed and inverted, the resulting estimate is given on the *regular* interpolated grid. With this in mind, to enable ISE computations, we sample the (original) signals on the interpolated grids, and compare these values with the estimate values. So all ISE values for this method actually correspond to values for the regular interpolated grids.

Classical DWT construction. We have seen before that the DWT is only suitable for regular grids. To enable sparsity comparisons with classical wavelet transforms, a similar construction to the sparsity plot diagnostic tool is made. Unlike the methods for irregular grids, where we show the transform behaviour averaged over 50 grids, there is no source of variation for the DWT because we apply the transform to *one* regular grid. Hence the ISE values only have to be computed once for each signal.

4.2.2 Sparsity results

The sparsity of the linear lifting algorithms was best with fixed linear prediction with two neighbours (LP2N and LP1S). The nonlinear transforms had better sparsity than the fixed regression transforms, with the fully adaptive algorithms showing better compression than the AdaptPred methods. Figures 4.1 and 4.2 show examples of the sparsity behaviour of our algorithms for two test signals.

In general, there was no significant difference in compression when increasing the irregularity of the grid. This is shown, for example, in Figure 4.3, where the *HeaviSine* test function was decomposed with varying number of non-zero detail coefficients used in the transform inversion. The first section of this plot is shown in more detail in Figure 4.4. For this signal, even on the

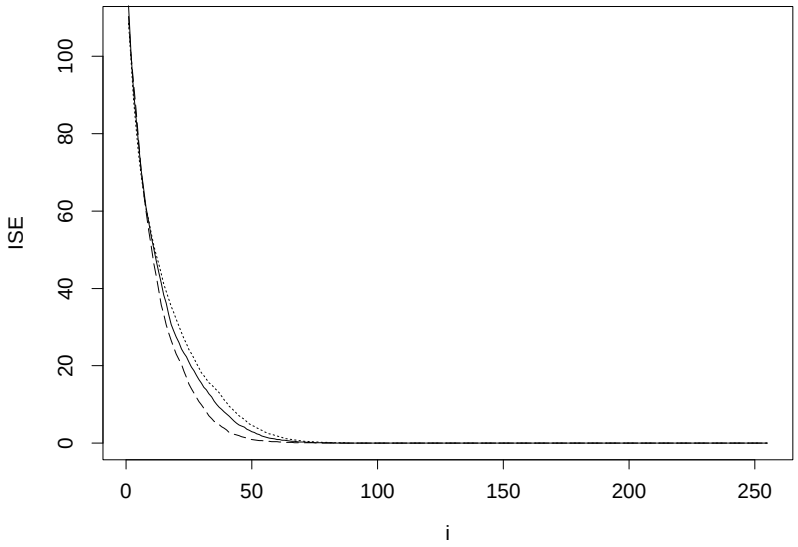

Figure 4.1: The sparsity of our lifting algorithms when decomposing the *Bumps* signal on irregular grids with jitter value $d_3 = 1$: LP1S (dotted); AP1F (solid); AN1 (dashed).

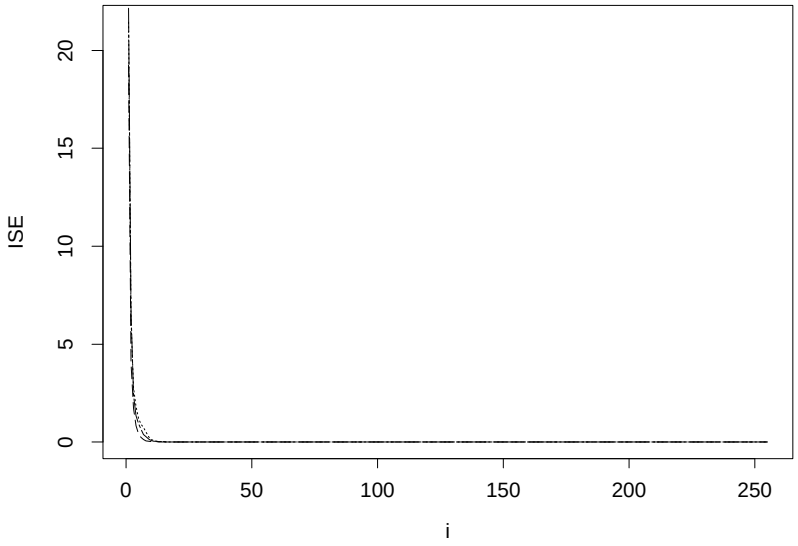

Figure 4.2: The sparsity of our lifting algorithms when decomposing the *Ppoly* signal on irregular grids with jitter value $d_2 = 0.1$: LP1S (dotted); AP1F (solid); AN1 (dashed).

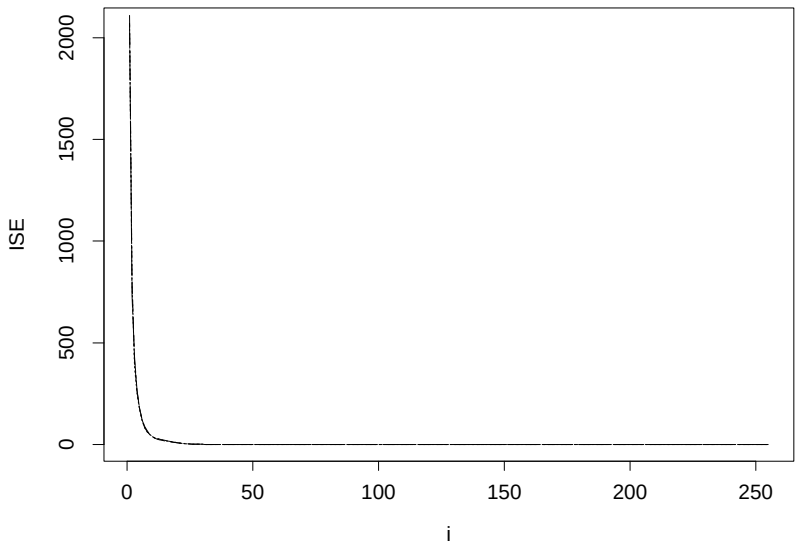

Figure 4.3: The sparsity of AP2N when decomposing the *HeaviSine* signal on irregular grids: d_1 (dotted); d_2 (dot-dashed); d_3 (dashed).

blowup, the sparsity is very similar across grids. However, sometimes a slight difference is observed, such as for *Bumps* (see Figures 4.5 and 4.6).

However, from signal to signal, the sparsity showed different characteristics. For the signals *Blocks* and *Bumps*, the rate of decrease in ISE was slower than with the other three smoother signals. This can be seen by comparing the Figures 4.1 and 4.2.

Comparing the lifting algorithms with the other wavelet transforms (KS and DWT), we found that our methods remained competitive. For the smoother signals, there is not as much difference between the methods, than for the more discontinuous signals. This is shown in Figures 4.7, 4.8 and 4.9. For the *HeaviSine* signal, all methods show a dramatic decrease in ISE error by adding only a few non-zero detail coefficients into the coefficient sequence before the transform inversion. The KS and classical wavelets have identical sparsity, having better compression than our methods for $i \leq 10$. However, this is relatively insignificant when considering the whole range of i.

In Figure 4.9, which gives an example for the *Blocks* signal, the behaviour of

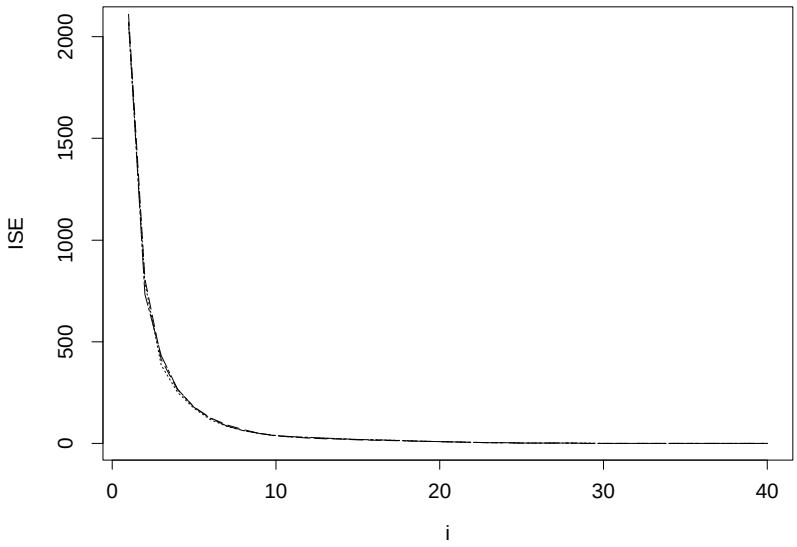

Figure 4.4: A blowup of Figure 4.3, showing sparsity for when few non-zero detail coefficients were included in the wavelet decomposition: d_1 (dotted); d_2 (dot-dashed); d_3 (dashed). Even on this graph, it is difficult to separate the curves for the three grid jitter values.

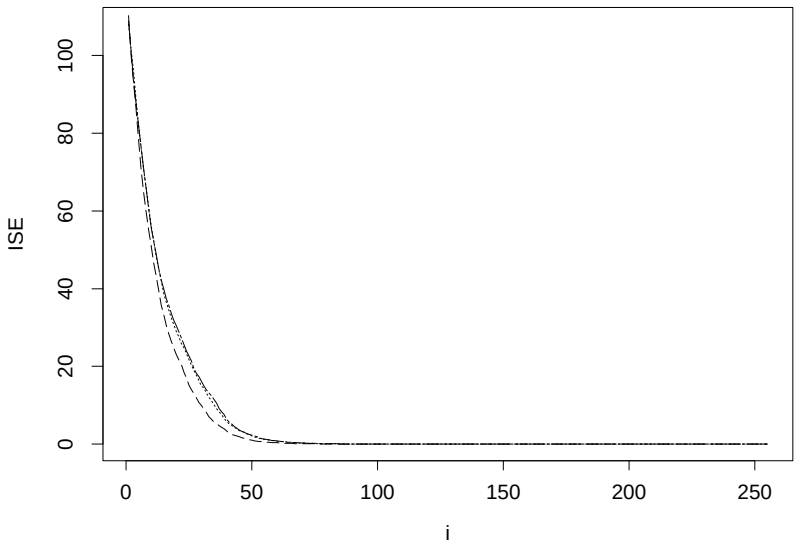

Figure 4.5: The sparsity of AN1 when decomposing the *Bumps* signal on irregular grids: d_1 (dotted); d_2 (dot-dashed); d_3 (dashed).

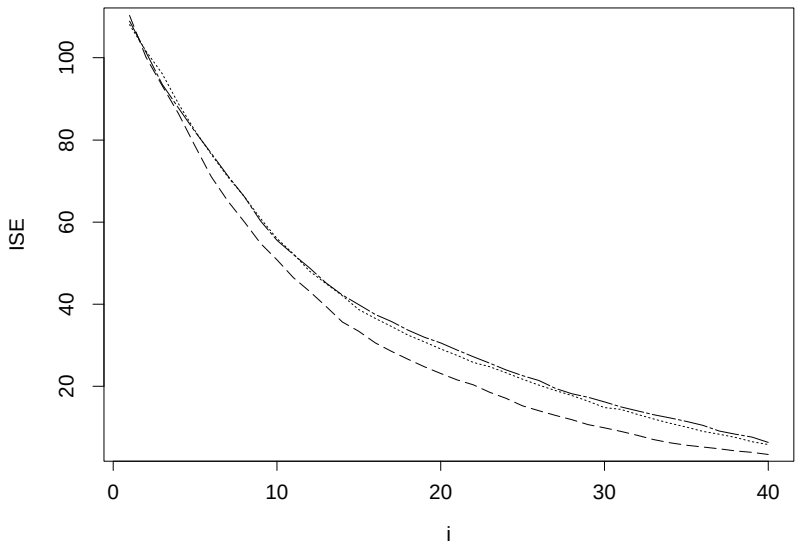

Figure 4.6: A blowup of Figure 4.5, showing sparsity for when few non-zero detail coefficients were included in the wavelet decomposition: d_1 (dotted); d_2 (dot-dashed); d_3 (dashed).

each algorithm is clearer. Our algorithm, AN1, has best sparsity out of all the algorithms, being closely followed by Haar wavelets (it should be stressed that the sparsity curve is for the DWT on a regular grid). Note that unlike the other algorithms, the ISE curve corresponding to the KS algorithm does not decay to zero. Remember here that the last points in a sparsity curve represents the ISE computation when nearly all coefficients are left alone (only a few small detail coefficients are set to zero). So for this region, the curve should tend to zero, since the ISE computations are increasingly closer to a straight decomposition-inversion implementation (where the ISE value would be zero).

As mentioned before, the KS algorithm uses interpolation to form its function estimates. If the signal values on the regular grid (obtained through interpolation) are not close to the actual "true" signal values on the interpolated grid, then the ISE for this algorithm will not be zero at the end of the ISE curves. This phenomenon will occur especially when the algorithm input shows some degree of erratic behaviour, or for example signals with many

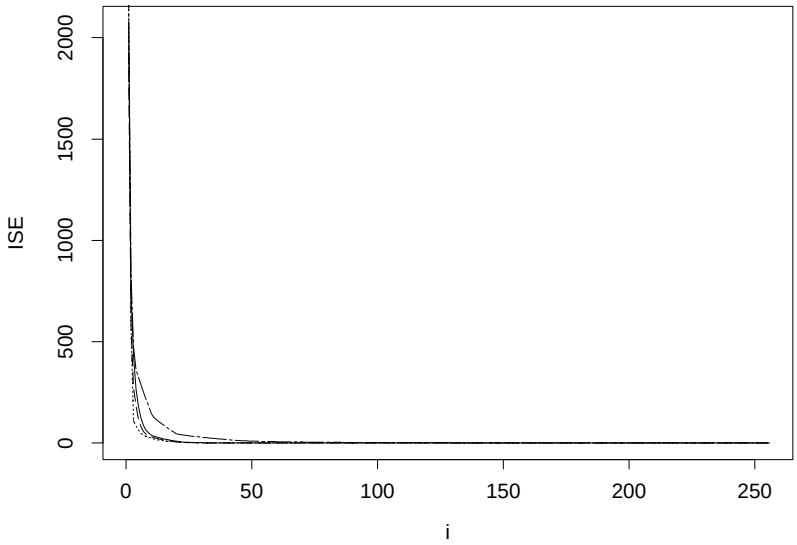

Figure 4.7: The sparsity of different algorithms when decomposing the *Heavi-Sine* signal. For the lifting algorithms and KS, irregular grids with jitter value $d_2 = 0.1$ were used: AP2N (solid); AN1 (dashed); KS with D5 (sparsely dotted). Regular grid curves (DWT with Daubechies Extremal Phase wavelets): best sparsity was using D5 (dotted); worst sparsity was using Haar wavelets (dot-dashed).

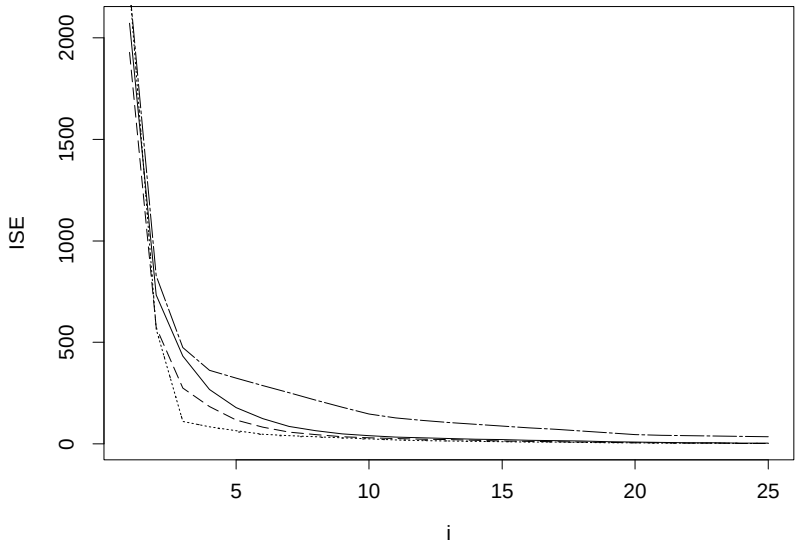

Figure 4.8: A blowup of Figure 4.7, showing sparsity for the different algorithms when few non-zero detail coefficients were included in the wavelet decomposition: AP2N on $d_2 = 0.1$(solid); AN1 on $d_2 = 0.1$ (dashed); KS with D5 on $d_2 = 0.1$ (sparsely dotted); DWT using D5 (dotted); DWT using Haar wavelets (dot-dashed). Note: the DWT using D5 and KS with D5 coincide.

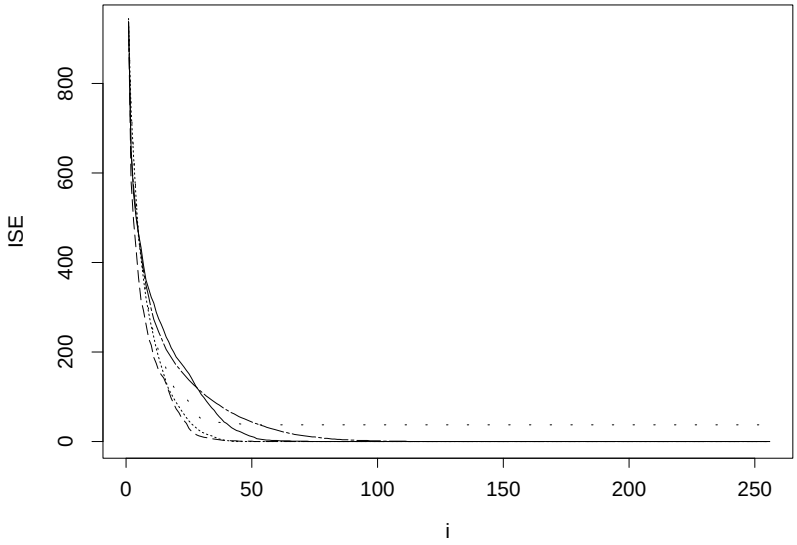

Figure 4.9: The sparsity of different algorithms when decomposing the *Blocks* signal. For the lifting algorithms and KS, irregular grids with jitter value $d_2 = 0.1$ was used: AP2N (solid); AN1 (dashed); KS with Haar wavelets (sparsely dotted). Regular grid curves (DWT with Daubechies Extremal Phase wavelets): best sparsity was using Haar wavelets (dotted); worst sparsity was using D7 (dot-dashed).

discontinuities.

4.3 Investigation into denoising performance of lifting algorithms

In this section, we explore the denoising capability of our lifting transforms compared to other existing nonparametric regression techniques, by means of a simulation study and application of the methods to real data examples.

The test signals used in the denoising simulations have already been described in Section 4.1. We performed the study over the three irregular grid values $d_1 = 0.01$, $d_2 = 0.1$ and $d_3 = 1$, and for three signal-to-noise (SNR) ratios 3,5 and 7. The signal-to-noise ratio is given by $SNR = \sqrt{\text{var}(g)}/\sigma$, where

g is the signal. The SNR gives a measure of how big the noise contribution is compared to the signal. The lower SNR levels have more noise content and so are harder to smooth successfully. In practice, we rescaled the signals to have unit variance, so that the standard deviations of the added noise were 1/3, 1/5 and 1/7 respectively.

For each signal, the function was sampled on an irregular grid (of length $n = 256$) and zero mean Gaussian noise was added according to a chosen SNR to form $\mathbf{f}^k$. The function is then smoothed with a denoiser to obtain an estimate of the true signal. We denote this by $\widehat{\mathbf{g}}^k$. We compared this to the true function sampled on the irregular grid, $\mathbf{g}^k$. This procedure was repeated for $k = 1, \ldots, K = 100$ simulations. We are interested in the average estimation of the function at the sample points. The measure of error we use for this is the *average mean square error* (AMSE), defined by

$$\text{AMSE} = (nK)^{-1} \sum_{k=1}^{K} \sum_{i=1}^{n} (g_i^k - \hat{g}_i^k)^2. \tag{4.1}$$

Our lifting methodology is compared to the following denoising algorithms, all of which have been described in Section 3.1.

Locfit. This is a local polynomial fitting algorithm. *Locfit* [69, 70] has a bandwidth smoothing parameter. To create the *Locfit*'s best estimate, we choose the parameter by cross-validation.

Smoothing spline. The S-plus function `smooth.spline()` was used to perform cubic spline regression on the test signals. The smoothing parameter for this function was also chosen by cross-validation. This method is denoted by SSCV.

Comte-Rozenholc. This algorithm fits a polynomial basis to the data, similar to smoothing splines [30]. The parameters of this method were taken according to the authors' recommendations in [30], with the maximum

number of partition knots $Dmax = n$, and the maximum polynomial degree of $rmax = 74$. The noise variance is estimated by the algorithm. In the simulation results, we give this method the abbreviation CR.

KS method. As with the sparsity simulations, Daubechies Extremal Phase wavelets were taken for the choice of wavelet family for the KS algorithm. With this basis, the specific wavelet (characterized by its vanishing moments) and primary resolution level also need to be chosen. Furthermore, there is the issue of which thresholding technique to be used with the KS method. Nason [77] investigates these choices for the KS method with soft and hard thresholding. In order to answer these decisions in our context, we ran simulations for every wavelet D1,...,D10 and every primary resolution level 0,...,7 for *SureShrink* [40] thresholding. The *SureShrink* thresholding technique is summarized in Section 2.3. We also ran the simulations with *EbayesThresh* thresholding, but since [63] state that this technique is insensitive to the choice of primary resolution level, a maximum level decomposition (primary resolution set to zero) was made. The full results of these simulations are given in [80]. The investigation shows that there is a lot of variability of denoising performance when using *SureShrink*. The KS method has less variation when changing the wavelet and using *EbayesThresh*. However, the AMSE values are higher with this thresholding method on all signals apart from *Bumps*. When reporting the AMSE results, we only give the values from the best wavelet/primary resolution/thresholding combination for each of the five signals – *Blocks*: (Haar,2,*SureShrink*); *Bumps*: (D2,0,*EbayesThresh*); *HeaviSine*: (D4,4,*SureShrink*); *Doppler*: (D4,5,*SureShrink*); and *Ppoly*: (D5,4,*SureShrink*).

Note that similar to the sparsity simulations, estimates for the KS algorithm are actually computed on regular interpolated grids, so the AMSE values are also computed on these grids, not the generated irregular

datasets. Hence when comparing the values, this should be kept in mind.

In all simulations relating to our lifting algorithms, we used *EbayesThresh* [61, 63] empirical Bayesian thresholding, with the "quasi-cauchy" prior and posterior thresholding choice. This technique is modified as in Section 3.5.2. Since our methodology lifts one detail coefficient at each stage, there is a similar choice to the primary resolution level in the KS algorithm for our method – the number of lifting steps to perform. We perform full decompositions for our denoisers (down to two scaling coefficients) since simulations in the next chapter show that this choice is not critical,as long as the stopping time is kept low.

Example 4.1. The test function *Doppler* was sampled on an irregular grid of length $n = 256$ (jitter value $d_2 = 0.1$, and Gaussian noise added with SNR=5 (see Figure 4.10). The noisy function was then denoised with the AP1S version of our lifting algorithm, and also the other smoothers KS, `smooth.spline()`, and *Locfit*. For KS, Daubechies' Extremal Phase wavelet D4 with primary resolution 5 and *SureShrink* thresholding was used, since for *Doppler*, this wavelet-primary resolution combination generally worked best. The graphs in Figure 4.11 show the four estimates of the original *Doppler* signal. It is apparent that for both the `smooth.spline()` and *Locfit* estimates, there is a certain amount of "wiggliness". This is due to the smoothness assumptions used in the techniques. The KS estimate is better, but seems to be sharper, with a number of "glitches". Our estimate successfully smooths the noisy data. However, there is a small unexplained spike at around x=0.6.

We repeated this test for the *Blocks* function, which was sampled using a jitter value of $d_1 = 0.01$. Gaussian noise with signal-to-noise ratio SNR=7 was added. The sampled signal and its noisy version can be seen in Figure 4.12. The Haar wavelet with primary resolution 2 and *SureShrink* thresholding was

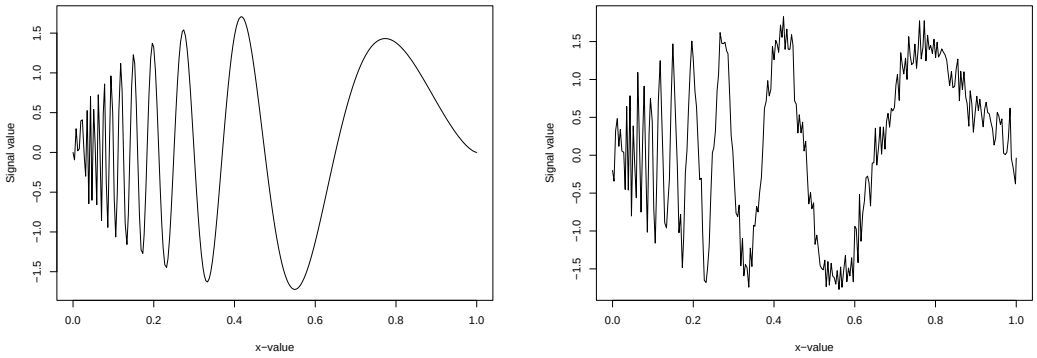

Figure 4.10: An irregularly-sampled *Doppler* signal (jitter value $d_2 = 0.1$) and the same signal with added Gaussian noise, SNR=5.

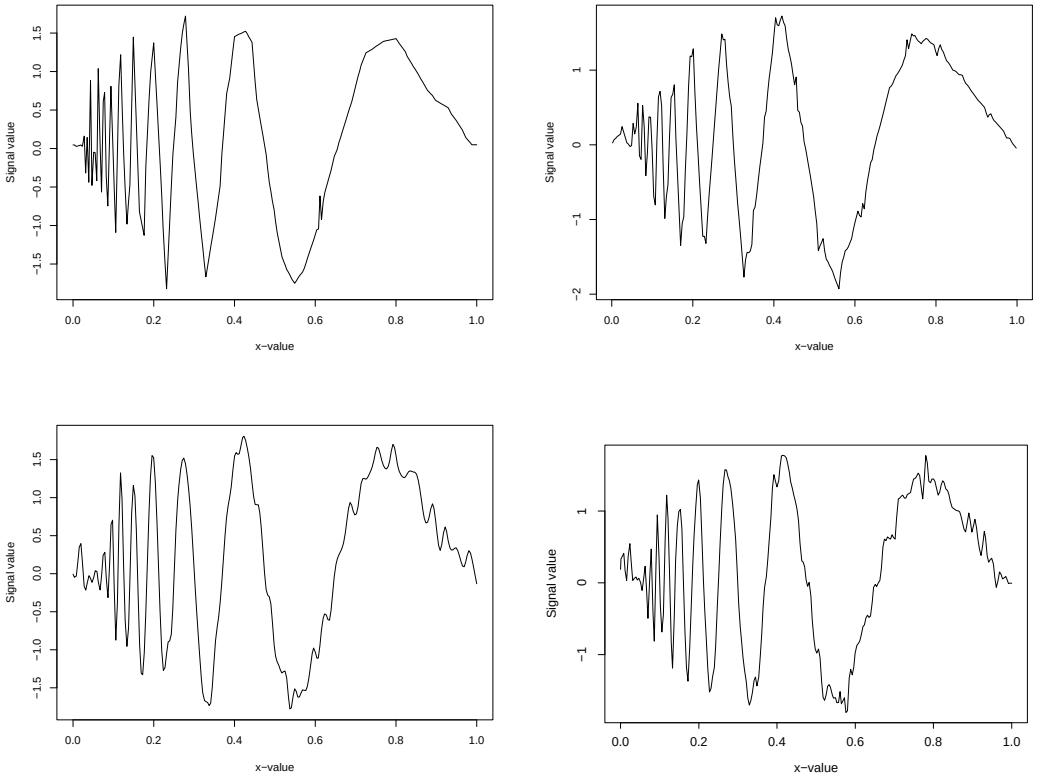

Figure 4.11: Estimates for the *Doppler* signal in Figure 4.10, using different denoising algorithms. Top left: AP1S; top right: KS using D4, resolution level 5 with *SureShrink* thresholding; bottom left: `smooth.spline()`; bottom right: *Locfit*. The plot shows our method AP1S to smooth the noisy data more successfully than the other smoothing procedures.

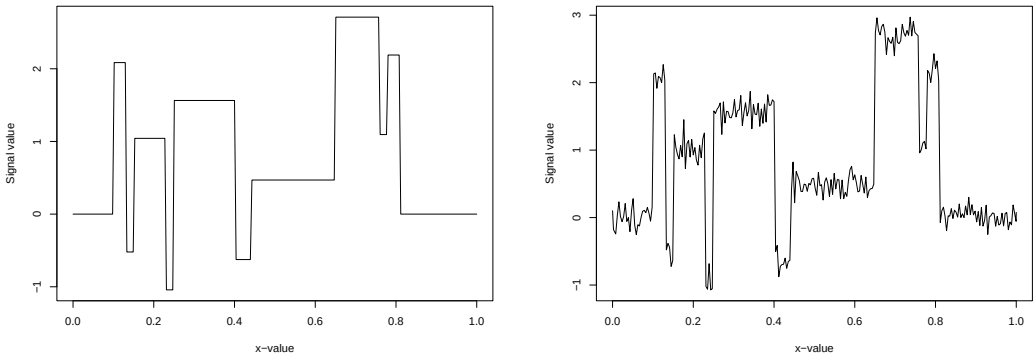

Figure 4.12: An irregularly-sampled *Blocks* signal (jitter value $d_1 = 0.01$) and the same signal with added Gaussian noise, SNR=7.

used for the KS estimate. None of the four resulting estimates are particularly visually pleasant (see Figure 4.13), but this is not surprising, since this function is more difficult to denoise due to the discontinuities across the range of the signal. Again, the two non-wavelet methods do not denoise the signal well. The Kovac-Silverman method and our algorithm (AN1) produce similar estimates, but it could be said that our estimate preserves the lower constant parts of the signal slightly better.

4.3.1 Denoising results

Tables 4.1, 4.2 and 4.3 show the denoising simulation results for the three different noise levels. For each of the five test signals and jitter values d_ℓ, the recorded value is the AMSE $\times 10^3$ of the 100 runs. The tables do not include results from our quadratic and cubic fixed lifting schemes or the AdaptNeigh algorithms with neighbourhoods larger than 2, since these did not work so well.

The AMSE tables show that there is the obvious trend of decreasing AMSE as the SNR level is increased (the noise content is decreased). The grid irregularity does not seem to have a significant effect on denoising performance of the lifting algorithms.

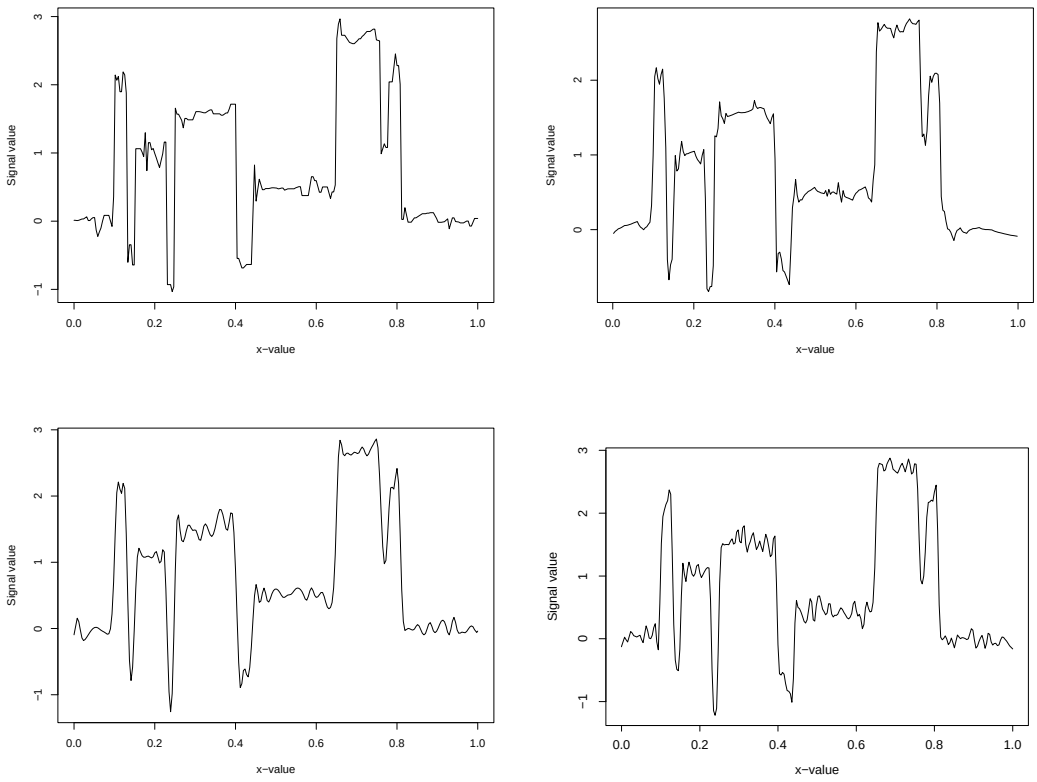

Figure 4.13: Estimates for the *Blocks* signal in Figure 4.12, using different denoising algorithms. Top left: AN1; top right: KS using Haar wavelet, resolution level 2 with *SureShrink* thresholding; bottom left: `smooth.spline()`; bottom right: *Locfit*. The plot shows our method AP1S to smooth the noisy data more successfully than the other smoothing procedures.

Table 4.1: AMSE ($\times 10^3$) simulation results for test signals with SNR=3 with three levels of jitter, d_ℓ, for various denoising methods described in the text.

	Blocks			*Bumps*			*HeaviSine*			*Doppler*			*Ppoly*		
Method	d_1	d_2	d_3	d_1	d_2	d_3	d_1	d_2	d_3	d_1	d_2	d_3	d_1	d_2	d_3
LP1S	72	71	68	81	80	73	20	20	21	54	53	52	16	16	18
LP2N	70	73	67	84	83	73	20	20	22	55	56	51	16	16	17
AP1S	72	68	59	77	77	62	20	20	23	52	50	48	16	17	18
AP2N	69	70	59	78	75	64	21	21	22	53	52	48	15	16	17
AP3N	69	68	68	76	74	73	46	44	41	64	65	61	42	39	36
AN1	55	54	52	66	67	61	36	39	37	61	61	59	38	33	32
Locfit	73	72	64	110	108	101	11	11	11	58	58	54	21	20	19
SSCV	74	74	67	307	315	250	12	11	12	61	60	53	20	20	19
KS	79	78	87	179	181	259	13	12	15	51	52	57	18	17	18
CR	119	119	133	332	313	284	25	25	25	155	155	148	13	13	13

Table 4.2: AMSE ($\times 10^3$) simulation results for test signals with SNR=5 with three levels of jitter, d_ℓ, for various denoising methods described in the text.

	Blocks			*Bumps*			*HeaviSine*			*Doppler*			*Ppoly*		
Method	d_1	d_2	d_3	d_1	d_2	d_3	d_1	d_2	d_3	d_1	d_2	d_3	d_1	d_2	d_3
LP1S	24	25	22	31	28	27	10	10	10	23	23	23	6	6	7
LP2N	23	23	22	30	30	27	10	10	11	23	23	22	6	6	6
AP1S	22	23	20	30	29	23	10	10	10	22	22	21	6	6	7
AP2N	23	23	20	30	29	23	10	10	11	22	21	21	6	6	7
AP3N	27	27	26	30	30	29	18	18	16	26	26	26	16	15	14
AN1	19	20	18	26	26	24	15	16	16	25	24	24	13	13	12
Locfit	35	35	34	40	40	39	7	7	7	25	26	25	12	12	11
SSCV	51	51	46	277	285	227	7	7	7	37	37	30	11	12	11
KS	52	52	59	130	134	213	8	7	8	29	28	33	9	9	10
CR	85	85	101	288	272	247	23	23	23	136	137	133	11	11	12

Table 4.3: AMSE ($\times 10^3$) simulation results for test signals with SNR=7 with three levels of jitter, d_ℓ, for various denoising methods described in the text.

	Blocks			*Bumps*			*HeaviSine*			*Doppler*			*Ppoly*		
Method	d_1	d_2	d_3	d_1	d_2	d_3	d_1	d_2	d_3	d_1	d_2	d_3	d_1	d_2	d_3
LP1S	11	11	11	15	15	14	6	6	6	12	12	13	3	3	3
LP2N	11	11	10	16	15	13	6	6	6	13	12	12	3	3	3
AP1S	10	10	10	15	14	12	6	6	6	12	12	11	3	3	4
AP2N	11	10	10	15	14	12	6	6	6	12	12	12	3	3	4
AP3N	14	14	14	16	16	16	10	10	10	14	14	14	8	8	7
AN1	10	10	9	14	14	13	9	9	8	14	14	13	7	7	6
Locfit	20	20	19	21	20	20	5	5	5	13	13	16	9	9	8
SSCV	44	44	39	269	273	220	5	5	5	30	30	23	8	8	8
KS	45	45	52	119	122	195	6	6	5	22	22	25	5	5	6
CR	78	79	92	280	269	230	22	22	22	132	130	128	11	11	11

Table 4.4: Results of the Simulation Study, $n = 100$, SNR=4. AN1 result computed here, all other results as computed by [38]. First row: square root of median MSE value; Second row: interval shows square root of 1st and 3rd quartiles of the MSE results over 500 simulations. All results $\times 10^3$.

	Delouille *et al.*				
AN1	With Update	No update	ANTO/FAN	KS	SUPSMO
588	610	792	819	775	706
[517, 654]	[526, 675]	[661, 989]	[759, 875]	[688, 856]	[629, 807]

Examining the AMSE tables, the adaptive algorithms are shown to be able to denoise the test functions well. In particular, AN1 performed very well, especially on *Bumps* and *Blocks*, where it outperformed the competitors on all noise levels. On the three smoother signals, AP methods with 2 neighbours prove to denoise best. Our method is outperformed on *HeaviSine* with SNR=3 and 5 though with SNR=7, the AP methods produce similar results to the other denoisers. For the *Doppler* signal, the KS algorithm comes close to our denoising performance with SNR=3, but on the other two noise levels, our method outperforms the competitors, with *Locfit* being closest. On *Ppoly*, the KS method is again the algorithm which comes closest to our denoising performance, though our method outperforms all competitors. The CR method does not perform well at all, probably due to oversmoothing in a lot of cases. The algorithm also takes a lot longer to run than the other denoisers.

Extra simulations

A simulation study was also run on a modified *HeaviSine* function, as used in [38]. The function is altered so that the discontinuity jumps are of size 4 instead of 2. The simulation was run for $K = 500$, with the x-values $\{x_i\}_{i\in\{1,\ldots,100\}}$ being distributed $N(0.5, (0.2)^2)$. The performance is compared with the ANTO/FAN method [7] and SUPSMO is the "super smoother" of [50].

Table 4.4 shows the results of the simulation. There is a slight improvement with our method. However, it should be pointed out that *HeaviSine* was the

signal on which our algorithm performed worst.

Our method performs well against the KS method according to Table 4.4, but in Tables 4.1 and 4.2, it does not. This is due to the modification to the *HeaviSine* signal. The bigger jumps in the modified signal cause the wavelet coefficients to be larger (relatively); the signal content can be viewed as being more significant around the discontinuities, and so it is as if the SNR is locally higher. It is in this case that our methods are more competitive (see Table 4.3).

4.3.2 Real data examples

We now show how our lifting methods perform on two real datasets.

4.3.3 Inductance plethysmography data

The original plethysmography dataset introduced by Nason [75] consists of 4096 datapoints. A plethysmograph is an instrument which measures the flow of air whilst breathing of a patient.

For analysis with the different denoising techniques in this study, the dataset was sampled to form a new irregularly-spaced dataset of length $n = 700$. For the KS method, since the algorithm only has scope to handle data defined on [0,1], we first preprocessed the data by (linearly) translating it to the unit interval. After denoising, the estimate was translated back to the original range.

All estimates apart from the Comte-Rozenholc method preserved the general shape of the data, removing the noise successfully, as observed by Nason [75] for wavelet methods on the original regular grid (Figure 4.14). The CR estimate oversmooths the data drastically, not picking up any of the features of the signal, and moreover does not eliminate the noise from the data fully. We believe in our estimate the peaks are slightly better discovered than the other methods (the shapes of the individual peaks represent the data more ac-

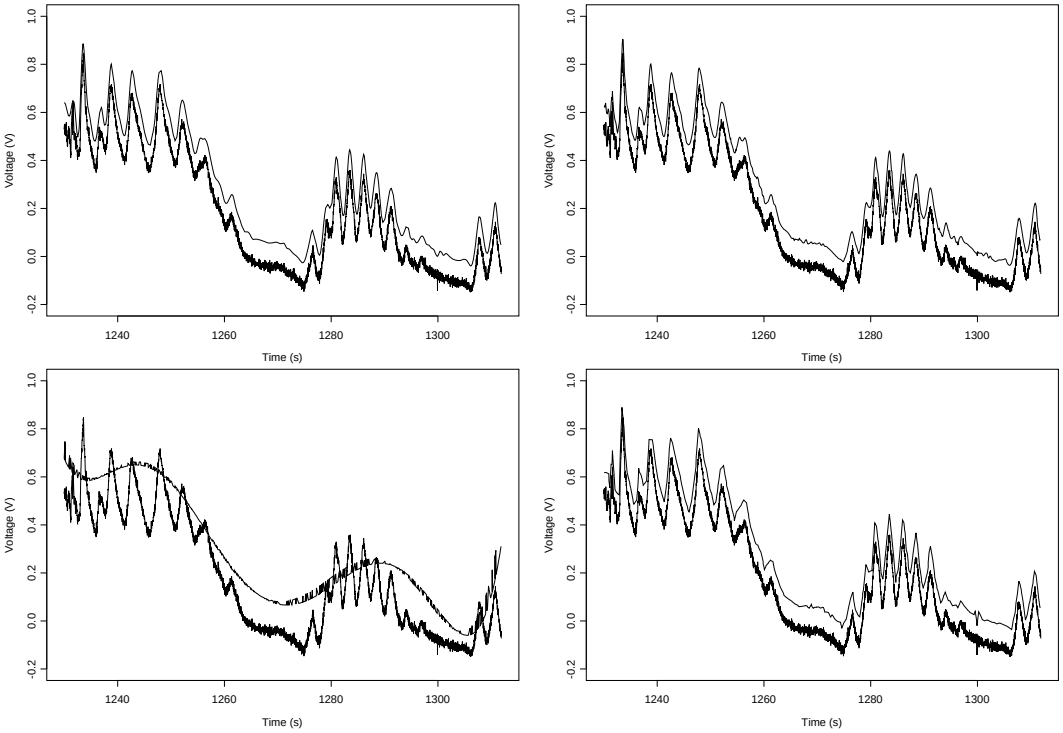

Figure 4.14: Estimators of the inductance plethysmography data (irregularly-sampled dataset of length $n = 700$). Noisy data shown with estimates shifted up by 0.1 to enhance visibility. Top left: smoothing spline with cross-validated smoothing parameter; top right: KS estimate using D6, primary resolution 3, using *SureShrink* thresholding; bottom left: CR estimate; bottom right: adaptive lifting using AP1S and *EbayesThresh* posterior median thresholding.

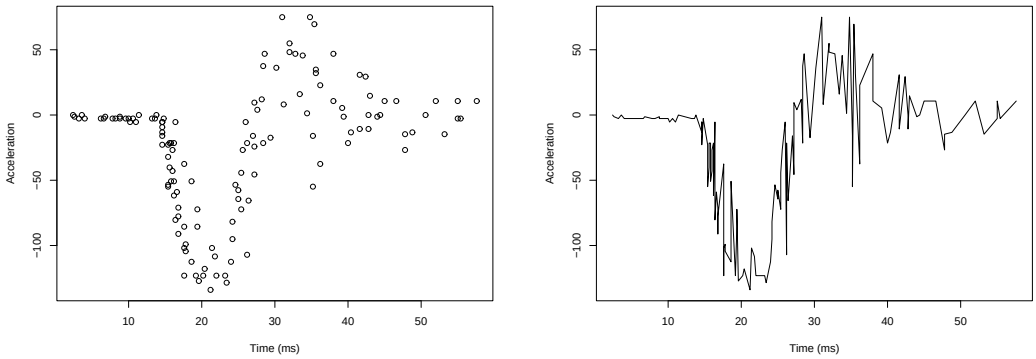

Figure 4.15: The motorcycle crash dataset. The left plot shows the original dataset (113 points); the right plot shows the same dataset joined with a line.

curately). However, our estimate has a short glitch around 1300 ms, which is not present in any of the other estimates. This is unlikely to be a true feature of the signal.

Motorcycle data

Figure 4.15 shows the motorcycle crash dataset, an example of a dataset with multiple observations at each x-value. This dataset was introduced by Silverman [89]. The dataset describes the head acceleration through time from simulated motorcycle crashes, to test crash helmets for efficacy. The dataset consists of 133 acceleration measurements at 94 time points. A linear depiction of the data is also given in Figure 4.15, since from this plot it is easier to see the job the smoothers have to "denoise" the motorcycle data.

This dataset was denoised using the smoothers described in the simulation study.

For the KS method, since the algorithm cannot handle data with multiple observations, for this smoother, we treated the dataset as being of length $n = 94$. The acceleration values for the distinct x-values were taken to be the means of the corresponding multiple observations. Similar to the inductance plethysmography data, for the KS algorithm, the motorcycle data was also translated to the unit interval prior to denoising, and then shifted back after

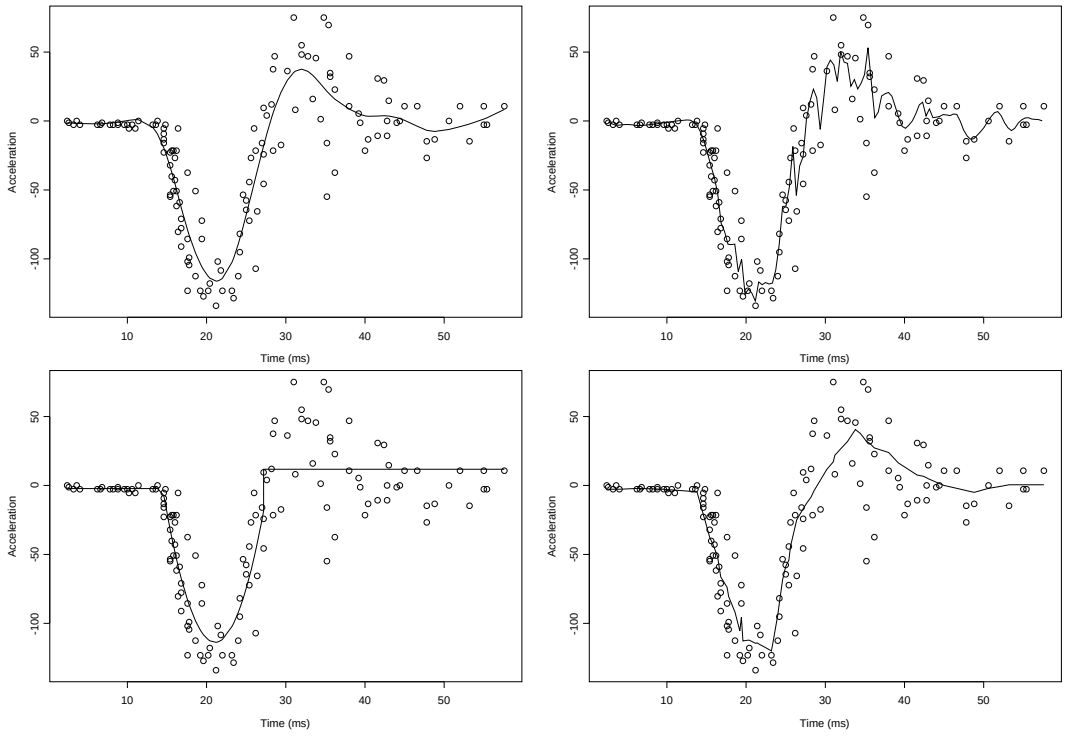

Figure 4.16: Motorcycle crash data and denoised estimates. Small circles=data; solid line=estimate. Top left: smoothing spline with cross-validated smoothing parameter; top right: KS estimate (applied to "averaged" dataset) using D6, primary resolution 3 with *SureShrink* thresholding; bottom left: CR estimate; bottom right: adaptive lifting using AP1S with heteroscedastic variance computation (see text) and *EbayesThresh* posterior median thresholding.

smoothing.

For our estimate, we used AP1S. The changing variance of the signal was estimated using the sliding window heteroscedastic variance procedure described in Section 3.5.2.

Figure 4.16 gives the estimates of the trend of head acceleration during motorcycle crashes for various denoisers. The four estimates are all quite different. Our method is less erratic than the KS method, but we should report that Kovac [66] applies the motorcycle data to an improved procedure to remove outliers, which produces better estimates. The peak of our estimate occurs later than the smoothing spline estimate, but this feature is also present

in some of the estimates for this dataset produced in [66]. The CR smoother does not seem to represent the data very well, producing a very conservative estimate. The overall trend of the data is present in the estimate for the first half of the signal, but it is obvious that the flat part of the estimate does not describe the true signal. The authors propose a second run of the algorithm using a variance estimate based on the first signal estimate (see [30]). For the motorcycle data, this causes the estimate to have a linear downward trend for the second half of the data. Even though this is an improvement over the first estimate, it does not pick up the clear variable nature of the tail of the data.

4.3.4 Further examples

Knight and Nason [64] apply our lifting methods to a proteomics problem. The transforms are used in the application of predicting transmembrane protein segments. Classical wavelet methods assume a regularly-spaced model for protein chain segments. However, the approach taken by the authors of [64] is to incorporate information from the 3D structure of the proteins to form an irregularly-spaced model for the residues based on the known structure of proteins of similar classifications. The paper shows improvements of up to 13% over the classical Daubechies wavelets when using the adaptive lifting algorithms.

4.4 Conclusions

In this chapter we have provided substantial evidence in support of the advantages of using our adaptive lifting algorithms over classical wavelet methods. Our methods are consistently competitive, and show better compression than competitors.

They outperform existing wavelet and non-wavelet methods for irregular design nonparametric regression problems in nearly all cases. The simulations show that the algorithms are particularly suitable for signals which have dis-

continuities. The application of the lifting transforms to the real examples demonstrate that the procedures produce good estimates.

The software (and associated help pages) for the adaptive lifting schemes are available from the website

`http://www.stats.bris.ac.uk/~maman/computerstuff/Adlift/`

or alternatively, it is available from the CRAN R package resource

`http://cran.r-project.org/`

The next chapter discusses the choice of stopping times in the adaptive lifting schemes, to try to give further improvements in denoising.

Chapter 5

Stopping times in adaptive lifting

Introduction

The previous two chapters have introduced adaptive lifting transforms, following the lifting "one coefficient at a time" approach proposed by Jansen *et al.* [59, 60]. These algorithms are suitable for use on irregular data of any length. Alongside these transforms, we have proposed a modified version for use on data with multiple observations for each x-value. The simulations in the last chapter show that the methods are very competitive in terms of both sparsity and denoising performance.

In the last two chapters, we did not discuss the primary resolution (PR) level in the lifting transforms, but assumed data to be decomposed to as many detail coefficients as possible. In this chapter, we will discuss the choice of primary resolution in the adaptive lifting transforms. The primary resolution level will also be referred to as the *stopping time* of the transform. We want to establish the importance of the stopping time for the adaptive lifting schemes through simulation, and based on the findings, we will attempt to devise an

automatic way of choosing the primary resolution level.

The first section of this chapter reviews the previous and existing work on primary resolution level selection. Section 5.2 examines the choice of primary resolution level in the context of the adaptive lifting schemes introduced in Chapter 3. The next section investigates the effect of stopping times in lifting schemes through simulation, and analyzes the results. Afterwards, a method for automatically selecting the stopping time for the adaptive lifting transform is suggested and assessed.

5.1 Previous work on primary resolution level in wavelet transforms

In classical wavelet transforms, such as the DWT, we have seen that a sampled signal vector $\mathbf{f}$ can be decomposed by a wavelet basis through Mallat's pyramidal algorithm as in 2.15

$$\mathbf{d} := \mathrm{DWT}(\mathbf{f}) = (\mathbf{c_{j_0}}, \mathbf{d_{j_0}}, \mathbf{d_{j_0+1}}, \ldots, \mathbf{d_{J-1}}),$$

where the number j_0 is the primary resolution level associated to the wavelet transform.

When addressing the nonparametric model in equation (2.45), the primary resolution level will affect the signal estimate, as well as the thresholding technique and threshold level chosen. A low choice of primary resolution level corresponds to coarse (as well as fine) level wavelet coefficients being thresholded, whereas when the primary resolution is chosen to be high, only the fine coefficients are thresholded. Wavelet estimators are generally forgiving of oversmoothing: too low a value for the primary resolution will have relatively little undesired effect, since the thresholding usually compensates for the low

resolution. Due to this resistance against oversmoothing, wavelet estimators provide a bound for the order of magnitude of the associated ISE. However, wavelet estimators are not forgiving of undersmoothing, so choosing the PR too high will cause noisy estimates. This is the reason why in standard implementations of wavelet transforms, the primary resolution level is taken to be fairly low.

The effect of the primary resolution parameter in wavelet decompositions has already been discussed in the literature for different settings.

Early work on the choice of the truncation parameter (resolution level) for general wavelet transforms is investigated in Hall and Patil [54]. The authors address the choice of truncation parameter and threshold level for different types of wavelet estimators on uniform and non-uniform design regression problems. Both light-tailed (normal or bounded) and heavy-tailed error distributions are considered. Suggestions for the form of the optimal resolution level are made for the different situations set out in the paper.

Hall and Nason [53] discuss the choice of the primary resolution level without the restriction to integer values, which is often assumed to make use of Mallat's pyramidal algorithm for the DWT (see Section 2.1.3). Integer choice of resolution level parallels a certain choice of bandwidth parameter for kernel methods, which is stated to be often unacceptable. The authors also point out that dyadic choice of resolution level in wavelet estimators is sometimes inappropriate, and can lead to under- or oversmoothing. The advantages of non-integer primary resolution level are quantified and a density estimation example given, showing that the extra computational effort involved in choosing non-integer primary resolution levels is insignificant. However, the computational implications of using non-integer primary resolution levels in nonparametric regression is not addressed.

The paper [55] by Hall and Penev introduces a cross-validatory technique

for choosing the primary resolution level for nonlinear wavelet estimators. The algorithm works by obtaining an initial estimate of the true curve, and then dividing the range of the signal into subregions according to where the estimator has similar roughness. Then cross-validation is used to select a primary resolution level chosen for each subregion. The minimum of these resolutions taken as the final chosen primary resolution level, and an estimate is made using this resolution across the whole signal region. Theoretical properties of the estimator are discussed.

The denoising effect of decomposing a signal to different primary resolution levels becomes especially apparent when combined with other smoothing parameters, such as shrinkage rules and thresholding values. For example, in [77], cross-validatory techniques are used to find the optimal primary resolution level, while simultaneously choosing other parameters. A simulation study is provided which shows the cross-validation method selects good values of primary resolution level for certain unknown functions. A fast algorithm to select these denoising parameters is implemented by Nason for the Kovac-Silverman denoising method (see [67]).

Recent work in [79] and [80] has also shown the choice of primary level in the Kovac-Silverman algorithm combined with other thresholding techniques to affect its ability to smooth noisy signals.

In Johnstone and Silverman [62, 63], the empirical Bayesian thresholding technique *EbayesThresh* is introduced. This method has been shown to perform well in many situations and with different wavelet methods. The authors propose that if *EbayesThresh* is used, it is insensitive to the choice of the primary resolution level in wavelet transforms.

Barber and Nason [10, 11] investigate the effect of the primary resolution level as part of their simulation study on thresholding rules for real and complex wavelet transforms. They find that certain thresholding techniques are sensitive to the choice of primary resolution level, whereas wavelet transforms combined with other methods (including *EbayesThresh*) show similar perfor-

mance across resolution levels.

The adaptive lifting algorithms from the previous chapters discussed here are nonlinear since their transform matrices depend on the input signal. Changing the primary resolution could hence have an interesting effect on their denoising capability. Because of this, as well as the factors of prediction lifting step, neighbourhood choice and thresholding methods, there is motivation for determining whether the primary resolution level affects the ability of the transform to smooth a noisy data input.

5.2 The rôle of primary resolution level in single coefficient lifting schemes

The adaptive lifting schemes of this thesis use the "one coefficient at a time" lifting procedure introduced in [59] and [60]. We now recap a few details of this variant of the lifting scheme, in order to place the notion of primary resolution level in our context. For more details on the adaptive lifting schemes, refer to the previous chapters.

Suppose we have a dataset of length n, representing a sampled function. The algorithm is as follows:

1. Associate scaling functions to each datapoint, by the interval construction detailed in Section 3.2 (the intervals correspond to the scaling function integrals);

2. Choose a point to lift by considering the data point corresponding to the interval with minimum length;

3. Perform prediction and update lifting steps on this coefficient to produce its detail coefficient and new scaling function integrals;

4. Repeat this for as many steps as required.

Due to the dyadic nature of classical wavelet transforms, the primary resolution level in this case can only take a limited number of values, namely 0 to $\log_2 n - 1$, where n is the length of the signal. In our lifting transforms, however, due to lifting only one coefficient at a time, we can have a much wider choice of possible values for the stopping time; for single coefficient lifting transforms, the choice of primary resolution level translates into choosing the number of lifting steps to perform (equivalent to deciding how many scaling coefficients to have in the function decomposition). Because of the interval construction used to represent the scaling function integrals, the maximum number of lifting steps in the transform is $n-1$, though in simulations we take the maximum to be $n - 2$.

5.3 Simulation Study: does the stopping time affect denoising performance of adaptive algorithms?

5.3.1 Simulation preliminaries

To determine the effect of the stopping time in the adaptive lifting algorithms, a denoising simulation study was run for the standard Donoho and Johnstone test signals *Doppler*, *Bumps*, *Blocks*, and *HeaviSine* first used in [39], as well as the now well-known *Ppoly* function introduced in [76]. These signals are described fully in Section 4.1.

The same *jittering* method described in Chapter 4 was used to produce 100 irregularly-sampled vectors of length 256 for each signal, which were then corrupted with Gaussian noise having signal-to-noise ratio equal to 3. The simulations were then repeated for SNR=5.

Different variants of the "one coefficient at a time" lifting algorithm were

used together with *Ebayesthresh* empirical Bayesian thresholding to eliminate the noise: the "quasi-cauchy" prior and posterior median thresholding were chosen for the *EbayesThresh* parameters.

For each algorithm, the transform was performed to every resolution level possible. Due to the interval construction described above, we take the maximum number of lifting steps which can be performed to be $n - 2$. Hence, for each primary level in the range 2 to 255, the error between the denoised estimate and the true signal was computed. This resulted in an *integrated squared error* (ISE) curve for each simulation run, showing the variability in denoising performance for different primary resolution levels. This was repeated for each of the 100 simulation runs and for each function.

Examples of the ISE curves obtained from the simulation study can be seen in Figure 5.1.

5.3.2 Simulation results: frequency plots

In what follows, we follow the same abbreviations for the lifting algorithms as explained in earlier sections. Frequency plots were constructed to show how often different numbers of scaling coefficients produced the best denoised estimates: for each of the 100 signals denoised in the MISE simulation study in Section 5.3, the individual ISE curves were examined to find when the lowest error value was obtained, and the results recorded.

When examining the results from the frequency plots, a few observations become apparent. The denoising behaviour of the lifting algorithms is not affected significantly by the level of jitter in the irregular grids on which the test signals were generated. This holds no matter which signal was denoised and which algorithm was implemented. A typical example of this is shown in Figure 5.2.

However, the number of lifting steps needed for optimal denoising seems

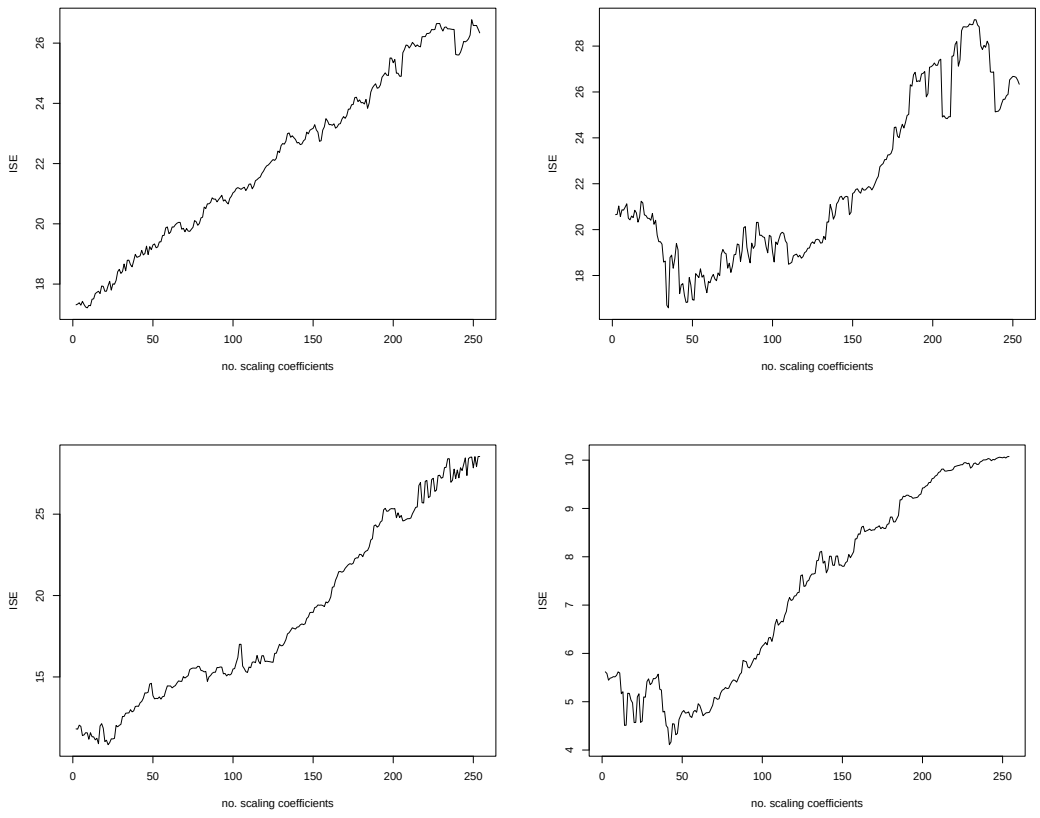

Figure 5.1: True ISE curves corresponding to four different datasets: (a) *Bumps* on grid d_3 with added SNR=3 Gaussian noise, denoised using AN1 (top left); (b) *Bumps* on grid d_3 with added SNR=3 Gaussian noise, denoised using LP2N (top right); (c) *Doppler* on grid d_1 with added SNR=3 Gaussian noise, denoised using AP2N (bottom left); (d) *Ppoly* on grid d_2 with added SNR=5 Gaussian noise, denoised using AN1 (bottom right).

to change dramatically across signals, and also across the denoisers used for the smoothing. The higher order non-adaptive algorithms, namely QP2S and CP2S, have greater spreads in the indices for best denoising, especially for *Blocks* and *Bumps* where the lifting steps required in the 100 runs can range over most of the possible values (see Figure 5.3). There is also sometimes no obvious choice for how many lifting steps should be performed in these algorithms to give the best denoised signal. This reinforces the unpredictability of these transforms, and their erratic behaviour when used in denoising, shown by the simulations in [79].

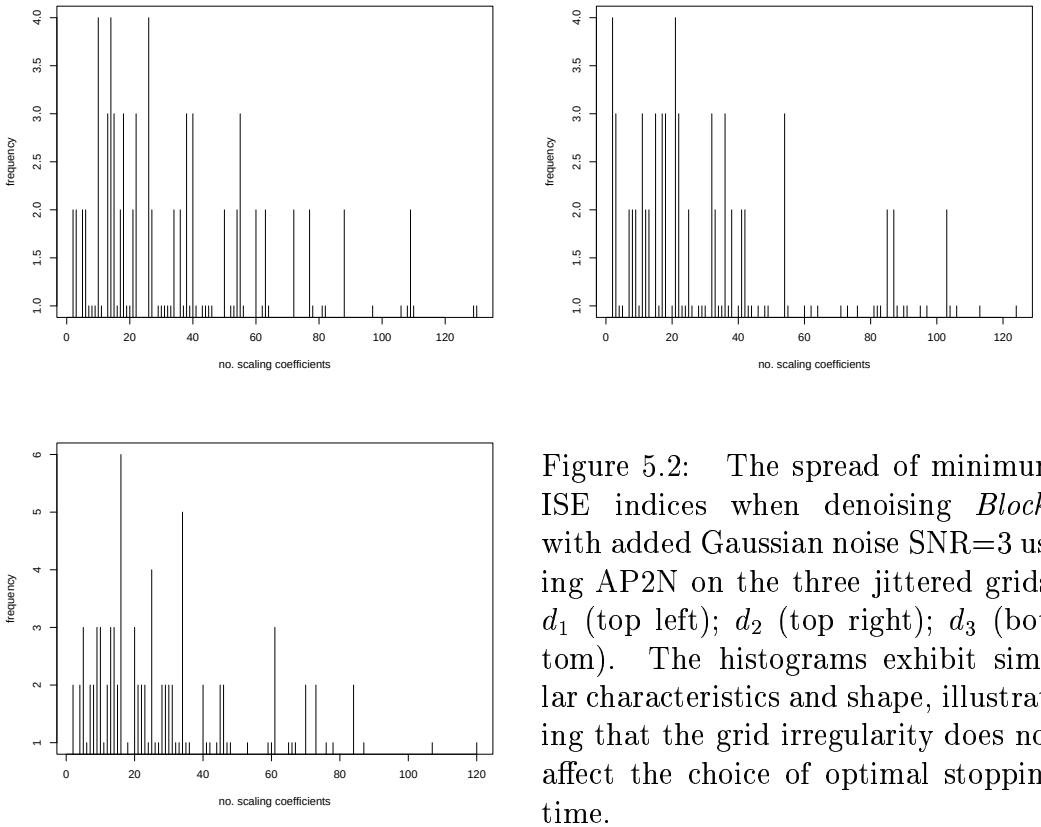

Figure 5.2: The spread of minimum ISE indices when denoising *Blocks* with added Gaussian noise SNR=3 using AP2N on the three jittered grids: d_1 (top left); d_2 (top right); d_3 (bottom). The histograms exhibit similar characteristics and shape, illustrating that the grid irregularity does not affect the choice of optimal stopping time.

The linear prediction scheme was more stable in view of the range of stopping times appearing from the simulations. However, the adaptive schemes showed a more pronounced indication as to the best number of lifting steps, with AN1 selecting the fewest number of stopping times. In general, the signals with discontinuities produced the most erratic behaviour with all denoisers. There was a marked difference compared to the smoother *HeaviSine* and *Ppoly* signals, which seemed to give more similar histograms across the denoising algorithms (Figure 5.4).

5.4 Automatic prediction of stopping times

The simulation study illustrates evidence for the link between the stopping time of the adaptive lifting schemes and their denoising performance.

The time taken to denoise a dataset of length $n = 256$ to every primary

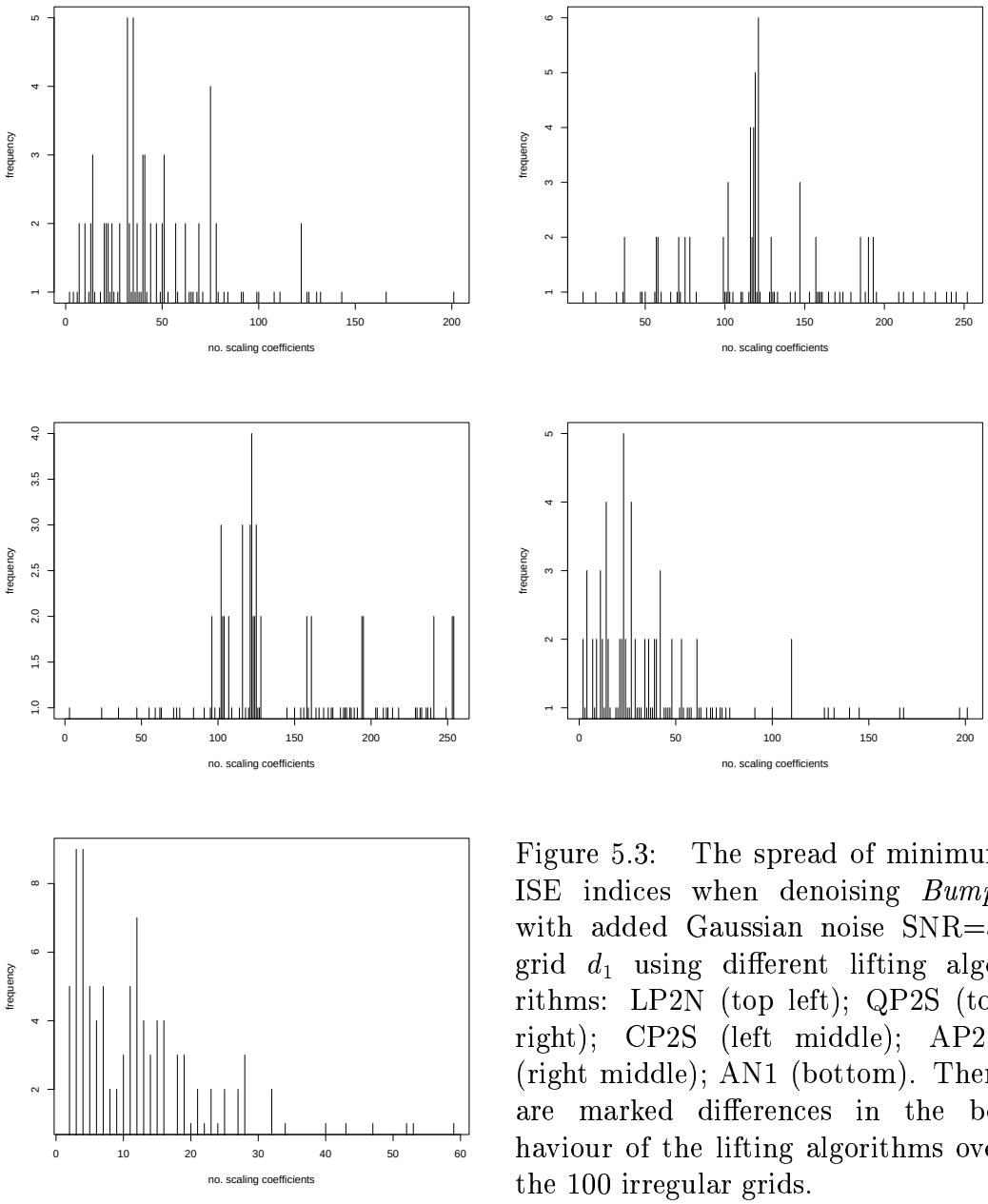

Figure 5.3: The spread of minimum ISE indices when denoising *Bumps* with added Gaussian noise SNR=3, grid d_1 using different lifting algorithms: LP2N (top left); QP2S (top right); CP2S (left middle); AP2N (right middle); AN1 (bottom). There are marked differences in the behaviour of the lifting algorithms over the 100 irregular grids.

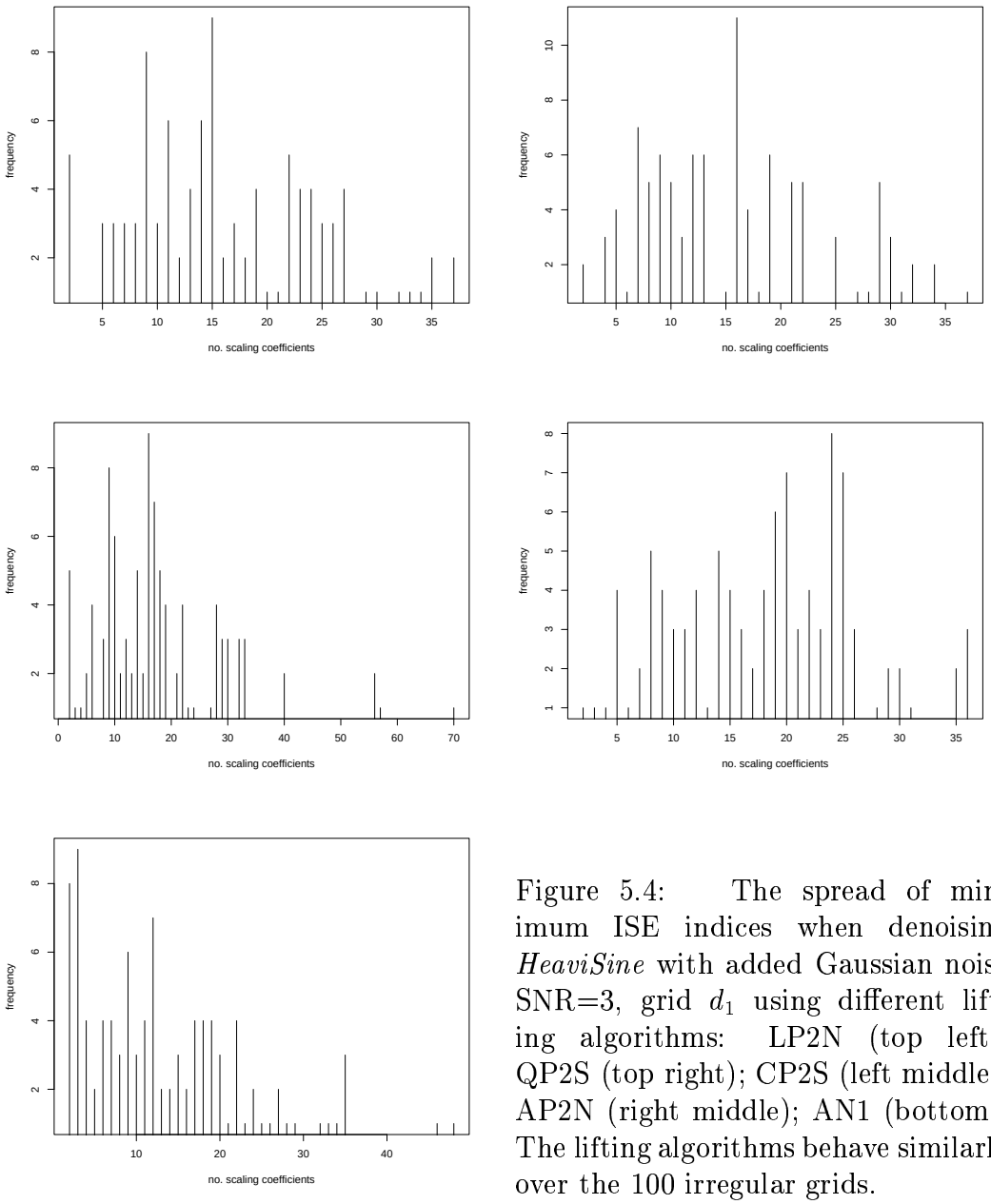

Figure 5.4: The spread of minimum ISE indices when denoising *HeaviSine* with added Gaussian noise SNR=3, grid d_1 using different lifting algorithms: LP2N (top left); QP2S (top right); CP2S (left middle); AP2N (right middle); AN1 (bottom). The lifting algorithms behave similarly over the 100 irregular grids.

resolution level, using AP2N and AN1 is, on average 10 minutes 14 seconds and 25 minutes 20 seconds respectively. This is taken from above simulations, running the lifting algorithms with a 2.8 GHz processor. From this, one could then find the best primary resolution level and hence best estimator for a particular dataset. Though this is not excessively long, for larger datasets this time could increase dramatically. There is an obvious trade-off between finding the best estimate for a signal, and the computation time involved in using the lifting transform to denoise data for every resolution level. Hence there is clear motivation for finding an efficient way to obtain the optimal stopping time, and in turn the best signal estimate. This would be of interest in situations where large datasets are to be denoised, or for example when speed of computation is of importance.

Having established that there is indeed a connection between the number of lifting steps to perform and the denoising capability of the various transforms, the question arises of how to develop an efficient automatic way of choosing the resolution level which would give improved performance in eliminating noise during smoothing procedures.

When denoising a general signal, we are just given the noisy data from which to estimate the true signal. We would like to know the ISE curve for the dataset, so that we could then choose the primary resolution level accordingly. However, since the true function is unknown, this is not possible, so our aim is to somehow estimate the ISE curve for a particular dataset.

5.4.1 ISE curve prediction procedure

To achieve this aim, we now present a method which attempts to recreate the ISE curve for a given dataset. We proceed as follows. Suppose we have an estimate for the true signal (we discuss this matter in more detail below). We could then compute estimates for the underlying signal at certain other resolution levels and use this information to calculate estimates for the ISE

values at those points. From this an estimate for the true signal's ISE curve can be formed. There are a few issues about this general procedure to discuss.

Initial estimate for true signal. Since we have no idea (*a priori*) of what the underlying true signal is, we could pick one resolution level and then use the smoothed version of the data based on this resolution as our estimate of the true signal. One disadvantage of doing this though, is that our initial true signal estimate would be heavily dependent on which resolution we choose, and may look very different to an estimate using a different resolution level. There would be some degree of uncertainty as to which one is a "good" or more suitable estimate, since the actual function is unknown. One way round this issue is to form an initial estimate for the true signal using some "average" of the curves obtained from denoising the dataset from several resolution levels. This estimate will more likely be a balance of the individual "good" and "bad" function estimates.

Signal estimate update. We may then decide to pick more resolution levels to get a better estimate of the signal's ISE curve. The current true function estimate can be updated by the new information gained. This update will give a better overall estimate of the true function. The resulting errors will then be judged according to this new current estimate. In this way we have an "in-place" improvement of the original function estimate.

Alternatively, the ISE values could be computed by comparing all the thresholded estimates with some *global* "average" of all the thresholded estimates.

Choice of resolution points. There is also an issue of how to pick subsequent resolution points to use for the ISE curve prediction. Suppose we have an estimate for the true function. The next resolution level could be chosen by considering a curve of best fit through the ISE values already

computed, picking the number of scaling coefficients to be used in the next decomposition corresponding to the minimum of the predicted ISE curve. Alternatively, we could just use the ISE information we have to pick the next stopping time, choosing it near the present resolution level giving lowest error.

Let us recap: our aim is (after some prechosen number of denoising iterations) to be able to recreate the shape and therefore minimum of the ISE curve of a given dataset. This will enable us to select the number of scaling coefficients to keep in the function decomposition to give the best denoised signal. Our procedure will be:

1. Pick a number of initial stopping times, and create an estimate for the true signal, M_k. Then for $k > 1$:
2. Pick another resolution level, $PR_{i(k)}$;
3. compute thresholded estimate using the primary resolution level $PR_{i(k)}$. Call this f_k;
4. Compute ISE_k between f_k and M_k;
5. Update M_k with f_k: $M_{k+1} := U(M_k, f_k)$;
6. Choose next resolution level i_{k+1} e.g. by fitting curve of best fit through all ISE points already computed.

Then (after say, K iterations), we then predict optimal number of scaling coefficients by either a curve of best fit through the ISE points already obtained, or recalculating the errors from some global function estimate.

5.4.2 ISE curve prediction: simulations

To test this method of simulating ISE curves, different datasets and denoisers were chosen, and the true ISE curves computed (see Figure 5.1). Best stopping times for the four datasets are (a) 9; (b) 35; (c) 22; (d) 42.

The various methods outlined in the prediction procedure above were then implemented to try and reconstruct them. Two "starting values" for the initial function estimate, namely $m = 5, 10$ were used, with either the mean or median of the m denoised functions as the estimate.

To choose subsequent stopping times, either a smoothing spline or the present ISE points were examined. The motivation behind fitting a smoothing spline to the ISE points is that it provides a method of predicting the ISE curve when we only have a relatively small number of points. It must be noted, however, that even though (ideally) we would like an ISE curve to be smooth, in reality most true ISE curves will have some degree of "spikiness". When we used the ISE points already calculated, the resolution level closest to the current optimal stopping time was chosen.

The procedures were stopped after $K_1 = 20$, $K_2 = 30$, and $K_3 = 40$ total iterations.

Updates

When we employed the mean as our initial function estimate, one of the following update choices was selected for the "true" signal estimates:

(**none**) The function estimate M was not updated from the initial estimate, i.e. $M_k = \text{mean}(f_1, \ldots, f_m) \quad \forall k$.

(**m1**) The mean of all previous denoised estimates: $M_{k+1} = \text{mean}(f_1, \ldots, f_k)$.

(**m2**) A weighted mean of the present estimate and the new estimate: $M_{k+1} = \text{mean}(M_k, f_k)$. The idea behind using this update is to weight the updated function estimate towards the estimate just computed.

(**ts**) A mean of the estimates after the initial primary resolution levels had been used to form M: if there are m initial primary resolutions, $M_{k+1} = \text{mean}(f_{m+1}, \ldots, f_k)$ (for $k > m$). Our hope is that since these points are chosen later in the procedure, they are more likely to lie on (or close to)

the true ISE curve, and hence predict a more accurate estimate of the best stopping time.

When the median was chosen to form the initial function estimate, we made a choice from two updates:

(**none**) The function estimate M was not updated from the initial estimate, i.e. $M_k = \text{median}(f_1, \ldots, f_m) \quad \forall k$.

(**m3**) The median of all previous estimates: $M_{k+1} = \text{median}(f_1, \ldots, f_k)$.

Two global function estimates were also used. These were computed from (K) function estimates, resulting from primary resolution levels chosen regularly over the range $2, \ldots, 255$ (all possible primary resolution levels):

(**mean**) A global mean of all estimates: $M = \text{mean}(f_1, \ldots, f_K)$.

(**median**) A global median of all estimates: $M = \text{median}(f_1, \ldots, f_K)$. For consistency, this was only computed when the median was also taken for the initial signal estimate.

The global estimates were then used to produce new ISE curves by comparing them to each individual thresholded function estimate, f_k. The best stopping time was recorded from these alternative ISE curves.

5.4.3 ISE curve prediction: results

The results from the prediction procedure proved to be disappointing. There was no appreciable difference in between the shape of the ISE curves from starting values $m = 5$ and 10. The mean and median updates produced very erratic and spiky ISE estimates, not reliable at all for predicting the best resolution level. The other updates, although more stable, did not represent the shape of the true ISE curves in Figure 5.1.

One of the problems with the updating methods in the ISE curve prediction is that the curve is inherently dependent on which resolutions are used for the initial signal estimate. For example, if initially the resolution levels 10, 20, 30, 40, and 50 were chosen instead of picking them regularly over the range 2 to 255, then the resulting signal estimate and therefore ISE curve could be significantly different.

Computing the global mean and median produced, in general, smoother ISE curves. Increasing the total iterations in the prediction procedures increased the overall error magnitude, though this is not necessarily an indication of an worsening of the update scheme, since the error is only relative to the current function estimate.

Using the smoothing spline to select resolution levels after the initial estimate seemed to increase the "spiky" features of the predicted ISE curves.

The behaviour of the prediction schemes was independent of which of the four datasets were denoised to form the error curves, in that all the datasets produced similar ISE curve estimates.

The poor performance of the update methods in these simulations could be attributed to the fact that they are very sensitive to the stopping times which are chosen from which to form the ISE predictions. The resolutions tend to be clustered around certain parts of the stopping time range, and so some primary levels which would produce globally minimal error might not be used, therefore skewing the ISE information in the procedures.

Tables 5.1 – 5.4 show the predicted optimal stopping times corresponding to the ISE curves in Figure 5.1 for the datasets (a) – (d). Although there are update scheme/initial function estimate combinations which predict stopping times the same or near to the true optimal resolution levels (given in Section 5.4.2), there is no combination which remains consistent over the four datasets.

No curve fitting:

		Starting value $m = 5$			Starting value $m = 10$		
		K_1	K_2	K_3	K_1	K_2	K_3
Mean Updates	none	127	127	127	138	138	138
	m1	127	127	127	138	138	138
	m2	16	21	21	13	13	24
	ts	6	6	6	10	10	10
Median Updates	none	127	127	127	138	138	138
	m3	16	24	127	16	21	24

	Total iterations		
	K_1	K_2	K_3
Global mean	121	130	131
Global median	131	121	128

Smoothing spline fitting:

		Starting value $m = 5$			Starting value $m = 10$		
		K_1	K_2	K_3	K_1	K_2	K_3
Mean Updates	none	129	128	130	128	129	128
	m1	158	139	182	254	254	217
	m2	143	141	144	126	122	118
	ts	137	138	138	127	126	128
Median Updates	none	128	126	128	128	127	127
	m3	2	64	176	149	157	103

	Total iterations		
	K_1	K_2	K_3
Global mean	130	129	129
Global median	130	134	130

Table 5.1: Predicted stopping times for dataset (a) (target=9) for initial function estimate types *mean* and *median*, and starting values $m = 5, 10$. Predicted stopping times are after $K_1 = 20$, $K_2 = 30$ and $K_3 = 40$ total iterations of signal estimate update methods described in the text. Mean update methods: no update (none), mean (m1); weighted mean (m3) and global mean (mean); median update methods: no update (none); median (m3) and global median (median). Top: no ISE curve interpolation to choose resolution points in algorithm; bottom: smoothing spline ISE curve interpolation to choose resolution points.

No curve fitting:

		Starting value $m = 5$			Starting value $m = 10$		
		K_1	K_2	K_3	K_1	K_2	K_3
Mean Updates	none	127	127	127	115	115	138
	m1	16	26	36	115	21	32
	m2	16	26	26	10	10	26
	ts	6	6	6	10	10	10
Median Updates	none	127	127	127	115	138	138
	m3	127	127	127	16	19	26

	Total iterations		
	K_1	K_2	K_3
Global mean	121	130	131
Global median	131	133	130

Smoothing spline fitting:

		Starting value $m = 5$			Starting value $m = 10$		
		K_1	K_2	K_3	K_1	K_2	K_3
Mean Updates	none	129	131	131	123	120	134
	m1	96	2	2	38	2	2
	m2	144	111	109	131	132	133
	ts	119	118	122	130	130	130
Median Updates	none	85	144	16	110	86	87
	m3	129	131	131	123	134	134

	Total iterations		
	K_1	K_2	K_3
Global mean	128	130	129
Global median	126	135	130

Table 5.2: Predicted stopping times for dataset (b) (target=35) for initial function estimate types *mean* and *median*, and starting values $m = 5, 10$. Predicted stopping times are after $K_1 = 20$, $K_2 = 30$ and $K_3 = 40$ total iterations of signal estimate update methods described in the text. Mean update methods: no update (none), mean (m1); weighted mean (m3) and global mean (mean); median update methods: no update (none); median (m3) and global median (median). Top: no ISE curve interpolation to choose resolution points in algorithm; bottom: smoothing spline ISE curve interpolation to choose resolution points.

No curve fitting:

		Starting value $m = 5$			Starting value $m = 10$		
		K_1	K_2	K_3	K_1	K_2	K_3
Mean Updates	none	127	127	127	138	138	138
	m1	16	26	36	138	21	32
	m2	14	23	23	14	14	32
	ts	5	5	5	13	13	13
Median Updates	none	127	127	127	138	138	138
	m3	16	25	32	16	4	30

	Total iterations		
	K_1	K_2	K_3
Global mean	133	136	131
Global median	123	133	134

Smoothing spline fitting:

		Starting value $m = 5$			Starting value $m = 10$		
		K_1	K_2	K_3	K_1	K_2	K_3
Mean Updates	none	129	132	127	139	139	139
	m1	93	2	2	2	2	121
	m2	118	114	107	129	124	122
	ts	122	123	123	129	127	127
Median Updates	none	130	127	131	134	139	135
	m3	88	254	31	135	88	166

	Total iterations		
	K_1	K_2	K_3
Global mean	133	133	133
Global median	132	127	130

Table 5.3: Predicted stopping times for dataset (c) (target=22) for initial function estimate types *mean* and *median*, and starting values $m = 5, 10$. Predicted stopping times are after $K_1 = 20$, $K_2 = 30$ and $K_3 = 40$ total iterations of signal estimate update methods described in the text. Mean update methods: no update (none), mean (m1); weighted mean (m3) and global mean (mean); median update methods: no update (none); median (m3) and global median (median). Top: no ISE curve interpolation to choose resolution points in algorithm; bottom: smoothing spline ISE curve interpolation to choose resolution points.

No curve fitting:

		Starting value $m = 5$			Starting value $m = 10$		
		K_1	K_2	K_3	K_1	K_2	K_3
Mean Updates	none	127	127	127	138	138	138
	m1	127	21	127	138	138	138
	m2	5	5	31	10	10	31
	ts	6	6	6	10	10	10
Median Updates	none	127	127	127	138	138	138
	m3	13	22	127	8	3	22

	Total iterations		
	K_1	K_2	K_3
Global mean	139	136	139
Global median	139	139	134

Smoothing spline fitting:

		Starting value $m = 5$			Starting value $m = 10$		
		K_1	K_2	K_3	K_1	K_2	K_3
Mean Updates	none	129	129	133	135	135	142
	m1	154	87	64	100	201	204
	m2	129	122	117	147	150	156
	ts	139	141	142	2	7	8
Median Updates	none	129	133	127	135	142	139
	m3	95	65	177	175	138	249

	Total iterations		
	K_1	K_2	K_3
Global mean	142	134	143
Global median	142	143	130

Table 5.4: Predicted stopping times for dataset (d) (target=42) for initial function estimate types *mean* and *median*, and starting values $m = 5, 10$. Predicted stopping times are after $K_1 = 20$, $K_2 = 30$ and $K_3 = 40$ total iterations of signal estimate update methods described in the text. Mean update methods: no update (none), mean (m1); weighted mean (m3) and global mean (mean); median update methods: no update (none); median (m3) and global median (median). Top: no ISE curve interpolation to choose resolution points in algorithm; bottom: smoothing spline ISE curve interpolation to choose resolution points.

5.5 MISE curve investigation

To examine the results from the simulation study further, the ISE curves from the 100 simulation runs were averaged for each signal/grid/SNR combination to form *mean integrated square error* (MISE) curves, to see how the lifting transforms behave on average.

5.5.1 Graphical interpretation

The three best denoising methods, namely LP2N, AP2N and AN1, have a relatively smooth change in performance as the number of scaling coefficients increases, and whilst the MISE curves of the quadratic (QP2F) and cubic (CP2F) transforms are more variable, they too have resolution levels which give better smoothing performance than others. The unpredictability of the higher order transforms is illustrated by their sometimes erratic behaviour. There is often a big difference in the shape of the MISE curves as we change the grid jitter, whereas for the better algorithms there is only a slight change in behaviour in certain cases (see Figure 5.5). The third plot shows that the irregular grids with jitter value $d_3 = 1$ are denoised similarly to the irregular grids with jitter values $d_1 = 0.01$ and $d_2 = 0.1$, which nearly coincide. This observation substantiates the findings in Chapter 3 that the higher order prediction schemes and the lifting algorithms employing larger neighbourhood configurations are "ill-conditioned" and unstable.

Figure 5.6 shows the effect of changing the primary resolution used in the signal denoising for different lifting transforms. The graphs reinforce the conclusions from simulations in Chapters 3 and 4 that the quadratic and cubic algorithms are not competitive with the adaptive lifting transforms, since their estimates have relatively high error compared to both AP2N and AN1 at most primary resolution levels. In view of this, and the observations highlighted above, we do not discuss QP2S or CP2S further in this section.

Considering how the resolution level affects the three main denoisers in

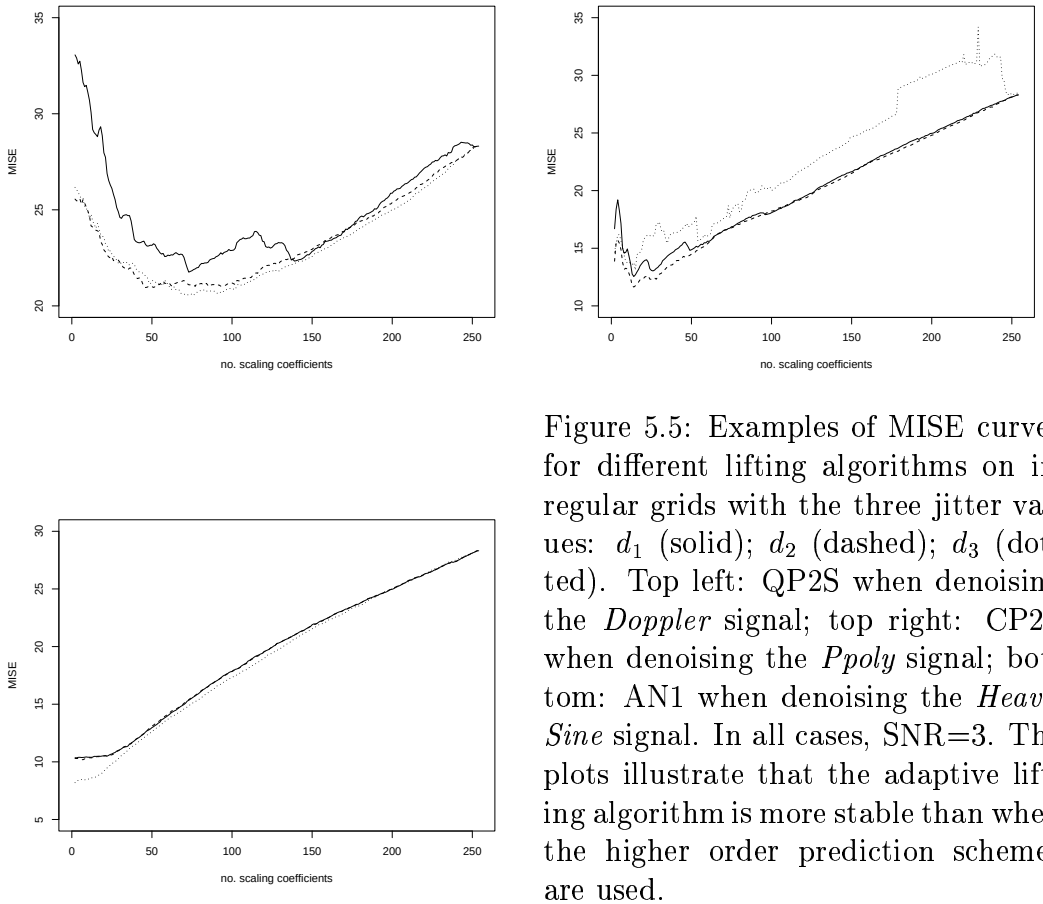

Figure 5.5: Examples of MISE curves for different lifting algorithms on irregular grids with the three jitter values: d_1 (solid); d_2 (dashed); d_3 (dotted). Top left: QP2S when denoising the *Doppler* signal; top right: CP2S when denoising the *Ppoly* signal; bottom: AN1 when denoising the *HeaviSine* signal. In all cases, SNR=3. The plots illustrate that the adaptive lifting algorithm is more stable than when the higher order prediction schemes are used.

more detail, we note from Figure 5.6 that for the *Doppler* signal, the fully adaptive transform AN1 cannot outperform either LP2N or AP2N, with these two transforms being very similar in performance. Comparing this now to Figure 5.7, where the *Bumps* signal is denoised, we notice that AN1 has lower error than the other two algorithms when fewer scaling functions are included in the signal decomposition (*more* lifting steps are performed in the transform). Indeed in general, this behaviour is characteristic of the better denoisers – for the signals with discontinuities, AN1 performs the best with a low stopping time, whereas the other two algorithms denoise better on *Doppler*, *HeaviSine* and *Ppoly*. This property agrees with the simulation study in Chapter 4, where a full decomposition down to two scaling coefficients was performed: AP methods are better for the smoother signals, and AN1 outperforms the

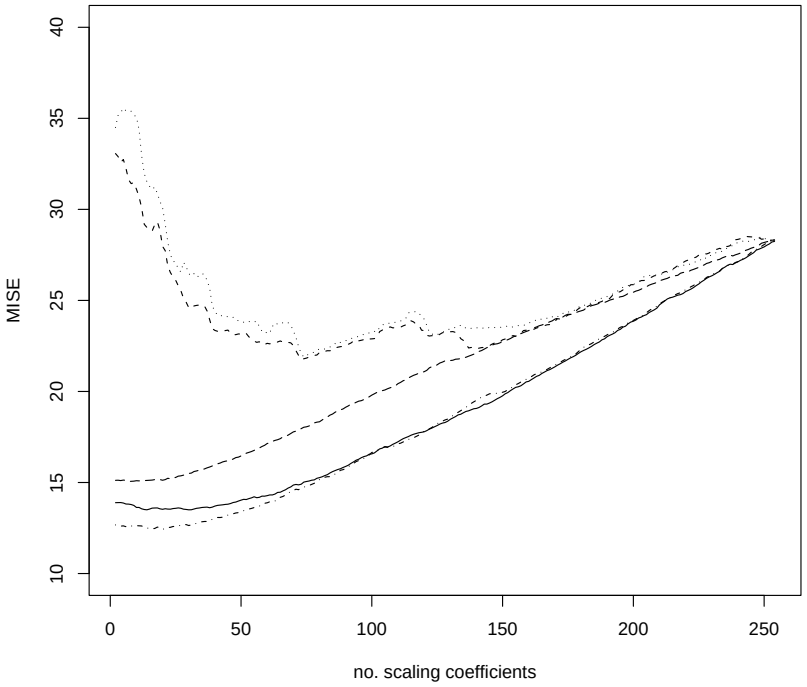

Figure 5.6: MISE curves of the lifting algorithms when smoothing the *Doppler* signal with added SNR=3 Gaussian noise on irregular grids with jitter value d_1: LP2N (solid); QP2S (dashed); CP2S (dotted); AP2N (dot-dashed); AN1 (long-dashed).

other lifting algorithms for *Blocks* and *Bumps*.

5.5.2 Optimal stopping times from MISE curves

Table 5.5 shows the number of scaling coefficients in the signal decompositions giving the best denoising performance (i.e. the number of scaling coefficients corresponding to the minima of the MISE curves). The erratic behaviour of the worse transforms is reflected in the varying number of coefficients across the grid jitters. Focussing on the better denoising methods, the performance when denoising the smoother test functions (*Doppler*, *HeaviSine* and *Ppoly*) is best irrespective of the denoiser at approximately 20 scaling coefficients. With

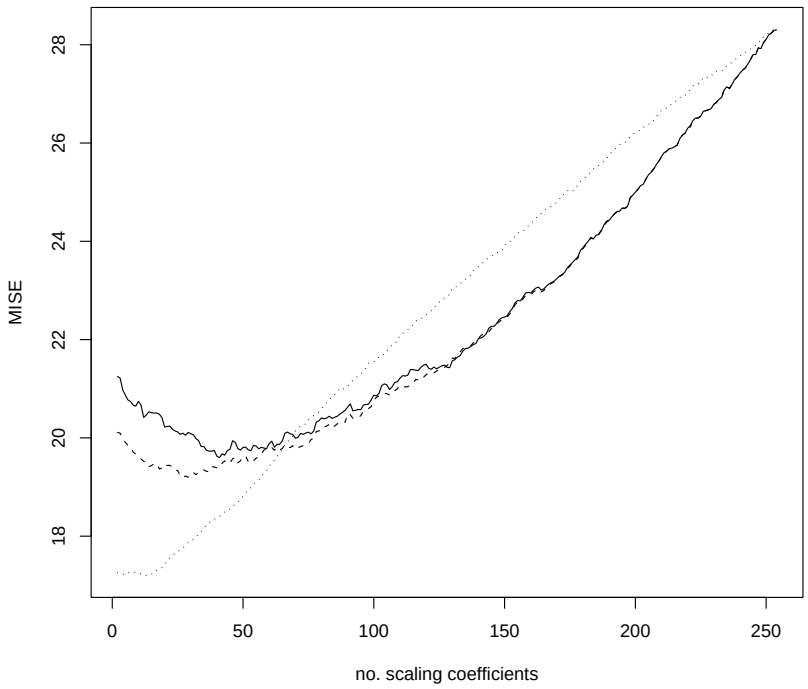

Figure 5.7: MISE curves for the better lifting algorithms when smoothing the *Bumps* signal with added SNR=3 Gaussian noise on irregular grids with jitter value d_1: LP2N (solid); AP2T (dashed); AN1 (dotted).

the signals with more pronounced discontinuities, namely *Bumps* and *Blocks*, the fully adaptive algorithm performs best with a low (less than 10) number of scaling coefficients in the decomposition, whereas the other two algorithms need slightly fewer lifting steps to obtain minimal error.

When SNR=5, the transforms exhibit similar MISE curves as when SNR=3. Corresponding patterns can also be seen in how the algorithms perform on the functions with discontinuities versus the smoother signals. Table 5.6 shows the number of scaling coefficients which give the best denoised signal for the different jitter values and signals for SNR=5. Again, with this signal-to-noise ratio, we draw the conclusion that to produce a good signal estimate, a low stopping time is needed.

	Blocks			*Bumps*			*HeaviSine*			*Doppler*			*Ppoly*		
Method	a	b	c	a	b	c	a	b	c	a	b	c	a	b	c
LP2N	55	41	45	41	41	43	7	7	17	30	20	19	7	7	10
QP2S	121	101	135	121	128	146	17	18	14	73	46	72	13	15	2
CP2S	122	123	250	122	204	247	17	18	251	74	79	97	14	14	15
AP2N	15	17	32	27	39	20	20	21	5	20	25	7	2	2	7
AN1	9	6	3	12	11	2	3	7	5	7	8	3	13	2	2

Table 5.5: The resolution level (number of scaling coefficients in the signal decomposition) giving the best denoised signal, when considering the average ISE curve of 100 denoised estimates, SNR=3.

	Blocks			*Bumps*			*HeaviSine*			*Doppler*			*Ppoly*		
Method	a	b	c	a	b	c	a	b	c	a	b	c	a	b	c
LP2N	26	37	28	34	38	27	23	22	21	27	34	19	2	2	2
AP2N	6	15	20	39	35	17	24	29	21	9	20	27	2	2	2
AN1	14	9	2	8	7	2	3	8	3	13	18	9	2	2	2

Table 5.6: The resolution level (number of scaling coefficients in the signal decomposition) giving the best denoised signal, when considering the average ISE curve of 100 denoised estimates, SNR=5.

5.6 Conclusions and further work

In this chapter, we have shown through simulations that varying the stopping times in the adaptive lifting schemes presented in Chapter 3 does affect the algorithms' denoising capability.

We proposed an automatic method for finding the best stopping time, although we were we were unable to correctly predict the optimal resolution level. Increasing the total iterations and starting values could give better results. It would be informative to use the automatic procedure on every primary level, to see if the ISE curve obtained resembled the true curves shown in Figure 5.1.

Collating the histogram information with the MISE curve simulation results in section 5.3, all algorithms show variation in the best resolution levels

from the 100 denoised signals. The range of this variation was sometimes quite big, whereas we would prefer to have the histograms showing more localized ranges with clear modes for the best stopping times. This would allow us to give definite best stopping times for each of the lifting algorithms. However, one could argue that 100 runs is not enough to give a true representation of this information, and so increasing the number of simulations might be useful.

In addition, the results are still informative. From Chapter 4, it is recommended to use AP2N (or something similar) for the smoother signals and AN1 for *Bumps* and *Blocks*. For these algorithms, both the MISE curves and the histograms are more consistent, and hence the conclusions drawn from them more reliable. Consequently, we suggest using these algorithms with low stopping times, e.g. 10 and 20 scaling coefficients respectively (`nkeep=10,20` in the *adlift* software) for signals similar to these as optimal stopping times, since on average this choice will give good denoising results.

An interesting idea also arises from the work in this chapter. Denoising with many lifting steps is obviously more computationally intensive compared to keeping a higher stopping time. Further work could be to attempt to provide an "in-place" decision-making process in the adaptive lifting algorithms to choose whether or not lifting an extra coefficient would be beneficial when denoising.

Chapter 6

A Haar-Fisz Algorithm for Binomial Probability estimation

Introduction

This chapter investigates the problem of estimating the proportion parameter associated with a binomial process.

Let us introduce the general set up for this problem. Suppose we have n observations x_k from a sequence of binomial random variables X_k, $k \in \{1, \ldots, N\}$, where we assume the variables to be independent: $X_k \sim Bin(n_k, p_k)$. Our aim is to try and estimate the proportions p_k from the observations x_k, where the binomial sizes are often chosen to be equal. We assume $p_k = P(k/N)$ for $k \in \{1, \ldots, N\}$, where P denotes the underlying (unknown) binomial proportion function; we also assume that P has some degree of regularity, for example that P is piecewise constant. In practice, this type of problem is difficult, since the classical 'function plus noise' model usually assumed for nonparametric regression problems may be unsuitable.

A similar situation often arises in applications where it is of interest to discover change points in the binomial probabilities during long periods of fairly homogeneous behaviour. This type of problem is interesting from a theoretical

point of view, and it also has many applications, especially in genetics.

For example, the Eisenberg and Levis simulated data [43] is often analyzed for this type of situation. Ion channels are large proteins controlling some aspects of cell function, which exist in a finite number of states, according to their ion conductance. The Eisenberg and Levis time series describes the current through a single ion channel with two states. The problem can be set up in the framework of a Bernoulli times series, with the model of being (independently) in one state with a certain (unknown) probability. Hodgson and Green [57] test their MCMC methodology on this data to investigate models for ion channel data. This setting is similar to the isochore prediction situation described in Oliver *et al.* [81] and Zhang and Chen [103], where the prediction of change points between DNA segments of homogeneous G+C nucleotide content is desired.

This chapter is organized as follows. In the first section, we give an overview on estimation methods for binomial processes.

Section 6.2 discusses the Fisz transform, an operator on two random variables which, in the limit, has a Gaussian distribution for variables with suitable properties [47]. This discussion provides motivation for our Gaussianizing approach to the binomial count estimation problem described in later sections.

Afterwards, the Haar-Fisz algorithm [51] is reviewed in Section 6.3. This procedure uses the Haar wavelet transform and the Gaussianizing and variance stabilizing properties of the Fisz transform to bring data closer to being normally distributed. In view of these properties, a more detailed exposition of previous work with the Haar-Fisz algorithm for intensity estimation is included.

The Fisz transform, although applicable to binomial random variables, does not possess the same variance stabilizing properties in the binomial case as for other random variables, for example, for Poisson variables. Hence in the subsequent section, we propose an alternative Gaussianizing transform to the

original Fisz transform for the particular case of binomially distributed input random variables.

We conclude this chapter by providing numerical evidence for the asymptotic properties of this new transform, and compare its performance with the traditional inverse sine Gaussianizing technique of Anscombe [6]. We also propose a technique for binomial proportion estimation and apply it to real data.

6.1 Previous work on binomial proportion estimation

We now give a brief outline of work in the literature for binomial distribution estimation problems.

6.1.1 Wavelet methods for binomial processes

Antoniadis and Leblanc [8] consider linear wavelet smoothers for the binary regression situation. A generalized linear model is imposed on the regression function, and via usual wavelet projection an estimator of the smooth model function $s(x)$ (see Section 6.1.2). A particular form of empirical wavelet coefficient is proposed to obtain smoother regression estimators than other coefficient estimators. The estimator is then modified to give a suitable estimator of the regression function $P(x)$. The choice of resolution parameter in resulting wavelet expansions is implemented in the binary regression context by generalizing existing selection criteria. The estimators' properties are shown through simulations and examples.

Wavelet shrinkage is used in the methodology by Antoniadis and Sapatinas [9], extending the idea to exponential families of functions with quadratic variance functions, a class of functions which includes the binomial distribution. An estimator of the risk is formed by assuming the function estimate to be

a diagonal linear shrinker and using a cross-validation approach. The function estimate is then constructed using a function which minimizes the risk estimate. A simulation study is carried out for the binomial case of an exponential family, and the estimates produced are shown to have good properties compared to traditional techniques.

Sardy *et al.* [85] propose a generalization of the *WaveShrink* wavelet smoother [39] to include a range of non-Gaussian distributions such as the binomial and Bernoulli distributions. The procedure uses interior point numerical techniques to find the (unique) solution to a penalized log-likelihood problem based on the l^1-norm of the wavelet coefficients in a wavelet estimator representation, and generalizes the notion of the universal threshold for distributions with log-likelihood function possessing certain properties.

Kolaczyk and Nowak [65] present a multiscale generalized linear model for the estimation of functions in a general one-dimensional nonparametric regression setting. Piecewise polynomials defined on recursive partitionings of the unit interval are used to construct estimators of the regression function, optimizing a penalized likelihood criterion to choose a piecewise polynomial fit. The method is applied to real-life examples, including estimation of packet loss rates in computer network information transmission. This can be modeled as a Bernoulli (or with modification, binomial) process.

6.1.2 Other techniques for binomial processes

Nonparametric regression techniques for proportions usually assume that the underlying proportion function has a certain degree of smoothness. For example, generalized linear models assume that the proportion function $P(x)$ follows the relation

$$g(P(x)) = s(x),$$

where g is a monotone smooth function called the *link function*, and $s(x)$

is a smooth function which is estimated by methods suitable for smooth (continuous) regression functions. The choice of the link function is obviously an important issue for this model, since it affects the fit of an estimator to the data. For a more involved discussion of generalized linear models, see for example [56, 73]. For different assumptions and estimation techniques for $s(x)$, and also link function choice, see [45, 46]. Antoniadis and Leblanc [8], described in Section 6.1.2, also uses a generalized linear model construction for their wavelet regression technique.

Altman and MacGibbon [5] use cross-validation for the bandwidth selection in kernel estimators for binary regression. The asymptotic risk of the kernel estimators is shown to have good convergence properties under certain smoothness conditions on the regression function.

Another approach to the binomial problem also includes Gaussianization of the original observed data, which involves a transformation of the observations so that the (transformed) data can be assumed to be normally distributed. There are many suitable estimation techniques in the literature which can then be used on this data.

Anscombe [6] suggests transformations to bring data from well-known distributions closer to normality. For the binomial distribution, the following transformation is proposed. Suppose $\{x_i\}$ are realizations from i.i.d. binomial random variables $X_i \sim Bin(n, p)$. Then the transformed data given by

$$\mathcal{A}x_i = \sin^{-1}\sqrt{\left(\frac{x_i + c}{n + 2c}\right)} \tag{6.1}$$

for the original data $\{x_i\}$ will be distributed more normally. Anscombe states that the value $c = \frac{3}{8}$ is optimal for μ and $n - \mu$ large (where μ is the

mean of the binomial distribution). Donoho [42] uses Anscombe's similar result for Poisson data, applying it to low light photon counts.

Freeman and Tukey [49] discuss a similar transformation for binomial data which takes the form of an averaged inverse sine function

$$\sin^{-1}\sqrt{\left(\frac{x_i}{n+1}\right)} + \sin^{-1}\sqrt{\left(\frac{x_i+1}{n+1}\right)}.$$

This is said to have good variance stabilization for almost all cases when the binomial mean is at least one.

An alternative Gaussianization transformation is given in Fisz [47]. This function acts on pairs of random variables, and takes the form of a ratio of powers of their sum. The properties of Fisz's result are discussed fully in the next section.

Fryźlewicz and Nason [51] combine the properties of the Fisz transform and the Haar wavelet transform and propose a preprocessing algorithm for Poisson count data to stabilize the variance of the data and to convert it so that any Gaussian denoiser can be used. This procedure uses a simple modification to the wavelet coefficients produced from the Haar wavelet transform, so can be performed without adversely affecting computational time. The Haar-Fisz algorithm is also applied to chi-squared data in [52] to denoise the wavelet periodogram with similar benefits. The Haar-Fisz transform is described in more detail in Section 6.3.

Other recent related work concerns providing reliable confidence intervals for binomial proportions. The classical so-called Wald interval, used for confidence intervals for a binomial success probability in standard statistical texts, is now known to have erratic interval coverage for many situations, especially near the boundaries zero and one. Agresti and Coull [4] in particular show that the true coverage probability for the 95% confidence interval to be variable for small binomial sizes. Brown *et al.* [16, 17] consider alternative intervals

for binomial proportions motivated by different mathematical reasoning, and present recommendations on the use of the proposed intervals, based on their merits in different binomial size situations.

Cox and Snell [32] study many different aspects of binary (Bernoulli) data. Examples of where binary data arises in applications are explored, and statistical models for these data are discussed. The authors also analyze departures from the classical Bernoulli problem, including abandoning the constant success probability, the introduction of dependence between responses, as well as more complex situations.

6.2 Fisz Gaussianization

As mentioned above, Fisz [47] proved a theorem which asserts the asymptotic normality of a special ratio of random variables under certain conditions. This theorem was used to develop a successful transform with good Gaussianization and variance stabilization properties for Poisson and chi-squared variables [51, 52]. Since its implementation with the simple Haar wavelet transform is both effective and not computationally intensive, we hope to achieve similar success for binomial random variables.

We now give some notation and recall a couple of definitions which are used in the Fisz theorem and which we will use later.

Let $\xi_1(\lambda_1)$ and $\xi_2(\lambda_2)$ be two independent non-negative random variables based on distributions with respective parameters λ_r. We denote, for $r = 1, 2$,

$$m_r = \mathbb{E}(\xi_r), \quad \sigma_r^2 = \mathrm{var}(\xi_r), \quad \psi = \sqrt{\sigma_1^2 + \sigma_2^2}, \quad \text{and} \quad \beta_r = \frac{\sigma_r}{\psi}. \tag{6.2}$$

Definition 6.1. *A random variable* $\xi(\lambda)$ **converges in probability to a**

number c *if, for any* $\varepsilon > 0$,

$$\lim_{\lambda\to\infty} \mathbb{P}(|\xi(\lambda) - c| > \varepsilon) = 0.$$

Definition 6.2. *A random variable* $\xi(\lambda)$ *is* **asymptotically normal** $\boldsymbol{N(u(\lambda), v(\lambda))}$ *if there exist functions* $u(\lambda)$ *and* $v(\lambda) > 0$ *such that* $\forall\, x \in \mathbb{R}$,

$$\lim_{\lambda\to\infty} \mathbb{P}\left(\frac{\xi(\lambda) - u(\lambda)}{v(\lambda)} < x\right) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-x^2/2}\, dx = \Phi(x).$$

As motivation for our approach to the binomial problem, we now state and discuss the Fisz theorem and its application to binomial random variables.

Theorem 6.3. *(Fisz, 1955). If*

(a) the variable $\xi(\lambda)/m(\lambda)$ *converges in probability to the number 1,*

(b) the variable $\xi(\lambda)$ *is asymptotically normal* $N(m(\lambda), \sigma(\lambda))$,

(c) the variables ξ_1 *and* ξ_2 *are independent and*

$$\lim_{\substack{\lambda_1\to\infty \\ \lambda_2\to\infty}} \frac{m_1}{m_2} = 1, \tag{6.3}$$

then the variable

$$\zeta^{\alpha}(\lambda_1, \lambda_2) = \frac{\xi_2 - \xi_1}{(\xi_2 + \xi_1)^{\alpha}}, \tag{6.4}$$

where α *is an arbitrary positive number, is asymptotically normal*

$$N\left(\frac{m_2 - m_1}{(m_1 + m_2)^{\alpha}}, \frac{\psi}{(m_1 + m_2)^{\alpha}}\right), \quad \textit{when } \lambda_1 \to \infty, \lambda_2 \to \infty. \tag{6.5}$$

(We use the convention here that if ξ_1 and ξ_2 are both zero, then ζ^{α} takes the value zero as well).

We now explicitly define the *Fisz transform* of two random variables.

Definition 6.4. *The* **Fisz transform with exponent α** *of two non-negative random variables X_1 and X_2 is*

$$\zeta^{\alpha}(X_1, X_2) = \frac{X_2 - X_1}{(X_1 + X_2)^{\alpha}},$$

with the convention that $0/0 = 0$.

Example 6.5. As an example, let us apply the Fisz theorem to two binomial random variables, as in [47]. Suppose $X_1 \sim Bin(n, p_1)$ and $X_2 \sim Bin(n, p_2)$ are independent random variables. The random variables are clearly non-negative, and due to the Law of Large Numbers and the Central Limit Theorem respectively, assumptions (a) and (b) in Theorem 6.3 are satisfied. Assuming that condition (c) holds with $m_r = np_r$ for $r = 1, 2$, we conclude that $\zeta^{\alpha}(X_1, X_2)$ is asymptotically normal

$$N\left(\frac{n^{1-\alpha}(p_2 - p_1)}{(p_1 + p_2)^{\alpha}}, \frac{n^{1/2-\alpha}\sqrt{p_1(1-p_1) + p_2(1-p_2)}}{(p_1 + p_2)^{\alpha}}\right). \tag{6.6}$$

Fisz notes that the hypothesis of equal binomial probabilities p_r can be tested for binomial random variables, since in this case ($p_1 = p_2 = p$), the asymptotic Normal distribution reduces to

$$N\left(\frac{n_2 - n_1}{(n_1 + n_2)^{\alpha}} p^{1-\alpha}, \frac{(1-p)^{1/2}}{(n_1 + n_2)^{\alpha - 1/2}} p^{1/2-\alpha}\right). \tag{6.7}$$

For the random variables X_1 and X_2 in Example 6.5 (i.e. of equal size), this asymptotic distribution simplifies further to

$$N\left(0, \frac{(1-p)^{1/2}}{(2n)^{\alpha - 1/2}} p^{1/2-\alpha}\right).$$

In this last case, the assumption (c) of Theorem 6.3 is automatically satisfied, since the ratio of the binomial means is exactly one.

Note that in all of these cases of binomial random variables, the variance function of the asymptotic normal distribution depends on p, and so the variance is *not* stabilized by the usual Fisz transform.

6.3 The Haar-Fisz transform

I will now digress briefly to provide motivation for the mathematical structure to tackle the binomial proportion estimation problem. A special case of the Fisz theorem is used in [51], and a preprocessing procedure is proposed for use with Poisson intensity estimation.

Applying the Fisz transform to Poisson random variables $\xi_r(\lambda_r) \sim Poi(\lambda_r)$, $r = 1, 2$, the distribution in (6.5) becomes

$$N\left(\frac{\lambda_2 - \lambda_1}{(\lambda_1+\lambda_2)^\alpha}, \frac{\sqrt{\lambda_1+\lambda_2}}{(\lambda_1+\lambda_2)^\alpha}\right), \tag{6.8}$$

when $\lambda_1 \to \infty, \lambda_2 \to \infty$.

The Haar-Fisz transform is motivated by the fact that for Poisson random variables, a choice of $\alpha = 1/2$ leads to an asymptotic Normal distribution with unit variance. Fryźlewicz and Nason [52] also take advantage of the variance stabilization properties of the Fisz transform for Chi-square random variables (Fisz transform with exponent $\alpha = 1$).

Let us explore the transform in a bit more detail. Suppose a positive data vector, $\mathbf{v}$, of length $N = 2^J$ has been observed. The algorithm proposed is as follows:

1. Perform the Haar discrete wavelet transform on the data, to transform $\boldsymbol{v} = \boldsymbol{c_J}$ into $(\boldsymbol{c_0}, \boldsymbol{d_0}, \boldsymbol{d_1}, \ldots, \boldsymbol{d_{J-1}})$, where as usual, $\boldsymbol{c_0}$ denotes the smooth

component and the $\boldsymbol{d_j}$ represent the detail components in the transform. However, as each level is computed, perform the modification

$$f_{j,k} = \begin{cases} 0 & \text{if } c_{j,k} = 0, \\ d_{j,k}/\sqrt{c_{j,k}} & \text{otherwise} \end{cases} \tag{6.9}$$

2. Perform the inverse Haar DWT on the vector $(\boldsymbol{c_0}, \boldsymbol{f_0}, \boldsymbol{f_1}, \ldots, \boldsymbol{f_{J-1}})$. Call the result $\mathbf{u}$.

These two steps are known as the *Haar-Fisz transform* of $\mathbf{v}$. We denote the transform as an operator by $\mathbf{u}:=\mathcal{F}\mathbf{v}$. Note that these steps can be easily inverted.

The motivation as to why the modification is applied to the detail coefficients needs a few words of explanation. The Haar DWT is performed on the input data, $\mathbf{v}$ by iterating the steps

$$c_{j,k} = (c_{j+1,2k} + c_{j+1,2k+1})/2$$

$$d_{j,k} = (c_{j+1,2k} - c_{j+1,2k+1})/2,$$

for $j = J-1, \ldots, 0$.

The inverse DWT can be expressed in the two equations

$$c_{j+1,2k} = c_{j,k} + d_{j,k}$$

$$c_{j+1,2k+1} = c_{j,k} - d_{j,k}.$$

(Note that the forward and inverse steps described above translate into using $\frac{1}{2}(1,1)$ and $\frac{1}{2}(1,-1)$, which differs from the filters in Example 2.6, which make the Haar basis orthonormal).

Using the last two equations, the original data vector, $\mathbf{v}=\boldsymbol{c_J}$, can be expressed in terms of a linear combination of the smooth component and the detail components, and then in turn each of the detail and smooth components can be again expressed in terms of the original data vector. As an example, suppose $N = 4$. Then using the inversion equations,

$$v_0 = c_{2,0} = c_{1,0} + d_{1,0} = c_{0,0} + d_{0,0} + d_{1,0}$$

$$v_1 = c_{2,1} = c_{1,0} - d_{1,0} = c_{0,0} + d_{0,0} - d_{1,0}$$

$$v_2 = c_{2,2} = c_{1,1} + d_{1,1} = c_{0,0} - d_{0,0} + d_{1,1}$$

$$v_3 = c_{2,3} = c_{1,1} - d_{1,1} = c_{0,0} - d_{0,0} - d_{1,1}.$$

Repeating the forward equations, we can express the scaling and detail coefficients in the transform in terms of the original data:

$$c_{0,0} = \frac{1}{2}(c_{1,0} + c_{1,1}) = \frac{1}{4}(c_{2,0} + c_{2,1} + (c_{2,2} + c_{2,3})) = \frac{1}{4}(v_0 + v_1 + v_2 + v_3)$$

$$d_{0,0} = \frac{1}{2}(c_{1,0} - c_{1,1}) = \frac{1}{4}(c_{2,0} + c_{2,1} - (c_{2,2} + c_{2,3})) = \frac{1}{4}(v_0 + v_1 - v_2 - v_3)$$

and

$$d_{1,0} = \frac{1}{2}(c_{2,0} - c_{2,1}) = \frac{1}{2}(v_0 - v_1)$$

$$d_{1,1} = \frac{1}{2}(c_{2,2} - c_{2,3}) = \frac{1}{2}(v_2 - v_3)$$

$$c_{1,0} = \frac{1}{2}(c_{2,0} + c_{2,1}) = \frac{1}{2}(v_0 + v_1)$$

$$c_{1,1} = \frac{1}{2}(c_{2,2} + c_{2,3}) = \frac{1}{2}(v_2 + v_3).$$

However, if we incorporate the coefficient modification described above in

step 1 by combining the last two sets of equations, we get

$$u_0 = \frac{1}{4}\sum_{i=0}^{3} v_i + \frac{\frac{1}{4}(v_0 + v_1 - v_2 - v_3)}{(\frac{1}{4}\sum_{i=0}^{3} v_i)^{\frac{1}{2}}} + \frac{\frac{1}{2}(v_0 - v_1)}{(\frac{1}{2}(v_0 + v_1))^{\frac{1}{2}}} \tag{6.10}$$

$$u_1 = \frac{1}{4}\sum_{i=0}^{3} v_i + \frac{\frac{1}{4}(v_0 + v_1 - v_2 - v_3)}{(\frac{1}{4}\sum_{i=0}^{3} v_i)^{\frac{1}{2}}} - \frac{\frac{1}{2}(v_0 - v_1)}{(\frac{1}{2}(v_0 + v_1))^{\frac{1}{2}}} \tag{6.11}$$

$$u_2 = \frac{1}{4}\sum_{i=0}^{3} v_i - \frac{\frac{1}{4}(v_0 + v_1 - v_2 - v_3)}{(\frac{1}{4}\sum_{i=0}^{3} v_i)^{\frac{1}{2}}} + \frac{\frac{1}{2}(v_2 - v_3)}{(\frac{1}{2}(v_2 + v_3))^{\frac{1}{2}}} \tag{6.12}$$

$$u_3 = \frac{1}{4}\sum_{i=0}^{3} v_i - \frac{\frac{1}{4}(v_0 + v_1 - v_2 - v_3)}{(\frac{1}{4}\sum_{i=0}^{3} v_i)^{\frac{1}{2}}} - \frac{\frac{1}{2}(v_2 - v_3)}{(\frac{1}{2}(v_2 + v_3))^{\frac{1}{2}}}. \tag{6.13}$$

Each of the transformed points are thus expressed as linear combinations of ratios of the form of the Fisz theorem (with $\alpha = 1/2$). Hence if the observations v_i come from independent Poisson random variables, then all the terms in (6.10) – (6.13) will be approximately normal, provided that the Poisson means satisfy the conditions at the beginning of the theorem. Fryźlewicz and Nason [51] provide a general formula for the terms in the Haar-Fisz transform.

The choice of $\alpha = 1/2$ in equation (6.8) demonstrates the variance stabilizing property of the Haar-Fisz transform: it causes the asymptotic normal distribution in (6.5) to have unit variance (for Poisson random variables). Fryźlewicz and Nason [51] also prove that if the Poisson observations are from i.i.d. random variables, then, the above observation can be used to prove the following result

Proposition 6.6. *If* $\mathbf{v}$ *is a sequence of observations (of length n) of i.i.d Poisson random variables with mean* λ. *Let* $\mathbf{u}=\mathcal{F}\mathbf{v}$ *be the Haar-Fisz transform of* $\mathbf{v}$. *Then* $\forall k \in \{1, \ldots, n\}$

$$u_k - \lambda = \nu + Y_k,$$

where $\nu \to 0$ *as* $\lambda/k \to 0$ *and* $Y_k \to N(0,1)$ *as* $(\lambda, k) \to (\infty, \infty)$.

In other words, the vector $\mathbf{u}$ is just the Poisson intensity with additional Gaussian noise. This result motivates Fryźlewicz and Nason [51] to propose a method for Poisson intensity estimation as follows:

1. Perform the Haar-Fisz transform on a vector of Poisson observations, $\mathbf{v}$, to bring the data closer to normality.

2. Use any wavelet denoiser suitable for Gaussian noise.

3. Invert the Haar-Fisz transform to obtain the estimate of the Poisson intensity.

Now let us look at the general form for coefficients in the decomposition stage of the Haar-Fisz transform. As an example, let us take a vector of length $2^J = n = 8$ representing observations from independent random variables X_k.

$$\mathbf{X}: \qquad X_1 \quad X_2 \quad X_3 \quad X_4 \quad X_5 \quad X_6 \quad X_7 \quad X_8$$

$$\mathbf{f_2}: \qquad \frac{\frac{1}{2}(X_1-X_2)}{(\frac{1}{2}(X_1+X_2))^\alpha} \quad \frac{\frac{1}{2}(X_3-X_4)}{(\frac{1}{2}(X_3+X_4))^\alpha} \quad \frac{\frac{1}{2}(X_5-X_6)}{(\frac{1}{2}(X_5+X_6))^\alpha} \quad \frac{\frac{1}{2}(X_7-X_8)}{(\frac{1}{2}(X_7+X_8))^\alpha} \tag{6.14}$$

$$\mathbf{f_1}: \qquad \frac{\frac{1}{4}((X_1+X_2)-(X_3+X_4))}{(\frac{1}{4}(X_1+X_2+X_3+X_4))^\alpha} \quad \frac{\frac{1}{4}((X_5+X_6)-(X_7+X_8))}{(\frac{1}{4}(X_5+X_6+X_7+X_8))^\alpha} \tag{6.15}$$

$$\mathbf{f_0}: \qquad \frac{\frac{1}{8}\left(\sum_{i=1}^4 X_i - \sum_{i=5}^8 X_i\right)}{(\frac{1}{8}\sum_{i=1}^8 X_i)^\alpha} \tag{6.16}$$

The numerators in the level decompositions (6.14) – (6.16) are just the Haar transform detail coefficients, and the sums in the denominators are the corresponding (scaled) smooth coefficients. The detail coefficients at level j

of the transform can be described as $d_{j,k} = 2^{(J-j)(1-\alpha)}\zeta^{\alpha}(Y_1, Y_2)$, where Y_1 and Y_2 are both sums of 2^{J-j-1} original random variables X_k. These are the components which are used to express the data $\mathbf{X}$ as linear combinations of Fisz-transformed random variables.

Let us now return to the case when the X_k are binomial (or simply Bernoulli) random variables. The random variables Y_1 and Y_2 will also be binomial random variables, provided that the binomial probabilities are equal.

We would like to have variance stability on each decomposition level j. Unfortunately, the variance stabilizing properties of the Fisz transform for Poisson variables (with exponent $\alpha = 1/2$) and Chi-square variables (with exponent $\alpha = 1$) cannot be achieved for binomial random variables - there is no choice of α in (6.7) which produces an asymptotic variance constant in p.

However, for $X_r \sim Bin(n, p)$, through examination of the distribution of $\zeta^{\alpha}(X_1, X_2)$ for different values of α, some interesting mathematical parallels can be observed. Since these distributional features somewhat deviate from the aim of this chapter, the (direct) numerical computation of the mass functions are explored in Chapter 7. The left hand side of Figures 6.1 – 6.4 give graphical representation of the probability mass function of $\zeta^1(X_1, X_2)$ for different binomial sizes and proportions. The sequence of graphs show that the shape of the distribution does indeed resemble a normal distribution for larger n. More remarks are made about the distribution of ζ^1 in Example 6.12.

6.4 An alternative Fisz transform for binomial random variables

In this section we introduce a new Fisz-like transform for binomial random variables. The idea is based on the original Fisz theorem [47], but since the

exponent in the denominator of $\zeta^{\alpha}(\lambda_1, \lambda_2)$ cannot be chosen to achieve asymptotic variance stabilization, we must approach the problem from a different direction.

We essentially divide the Haar difference $X_2 - X_1$ by its standard error, $\sqrt{\text{var}(X_1) + \text{var}(X_2)}$. We use the observations from X_1 and X_2 as estimates for the individual binomial means $n_r p$ $(r = 1, 2)$ in the expression for the standard error, so that the common proportion p estimated by $\frac{X_1+X_2}{n_1+n_2}$. Note that this is the Haar sum divided by the joint binomial size. This is similar to the Poisson Haar-Fisz case; since the mean and variance of Poisson variables are equal, the denominator of the transform is simply the square root of the Haar sum.

We first state and prove our alternative theorem to Theorem 6.3.

Theorem 6.7. *(Nunes, Nason) Let $X_r \sim Bin(n_r, p_r)$, for $r = 1, 2$ with $p_r \in (0, 1)$ (fixed). Let m_r and ψ be as in Theorem 6.3. If the random variables X_1 and X_2 are independent and*

$$\lim_{\substack{\lambda_1 \to \infty \\ \lambda_2 \to \infty}} \frac{m_1}{m_2} = 1, \tag{6.17}$$

Then the random variable defined by

$$\zeta_B(n_1, n_2) = \frac{X_2 - X_1}{\left(\frac{X_1+X_2}{n_1+n_2}(n_1 + n_2 - (X_1 + X_2))\right)^{1/2}}$$

is asymptotically normal $N(\mu_B, \sigma_B)$ when $n_1 \to \infty$, $n_2 \to \infty$, where

$$m_B = \frac{m_2 - m_1}{\left(\frac{m_1+m_2}{n_1+n_2}(n_1 + n_2 - (m_1 + m_2))\right)^{1/2}}$$

and

$$\sigma_B = \frac{\psi}{\left(\frac{m_1+m_2}{n_1+n_2}(n_1 + n_2 - (m_1 + m_2))\right)^{1/2}}. \tag{6.18}$$

In the definition of ζ_B, we assume that the random variable takes the value zero when both X_1 and X_2 are zero.

Proof. The proof of this theorem follows the ideas used for the proof of Theorem 6.3 in [47]. We begin by presenting some introductory lemmas.

Lemma 6.8. *If ξ_1 and ξ_2 are independent and $\xi_r(\lambda_r)/m_r(\lambda_r)$ converges in probability to 1, then*

$$\lim_{\substack{\lambda_1\to\infty \\ \lambda_2\to\infty}} \mathbb{P}\left(\left|\frac{\xi_1+\xi_2}{m_1+m_2}-1\right|>\varepsilon\right)=0, \tag{6.19}$$

where ε is an arbitrary positive number.

Proof of Lemma 6.8. This proof is taken directly from Fisz (see [47]). The assumption of the lemma says that for λ_1 and λ_2 large enough, the inequalities

$$m_r(1-\varepsilon)<\xi_r<m_r(1+\varepsilon) \qquad r=1,2$$

happens with probability greater than $1-\delta$, where $\delta>0$ is an any positive small number. Since the two random variables are independent, the probability that both inequalities will occur is at least $(1-\delta)^2$. The probability of the inequality

$$(m_1+m_2)(1-\varepsilon)<\xi_1+\xi_2<(m_1+m_2)(1+\varepsilon)$$

is greater than the probability of the occurrence of both the inequalities above; since δ can be arbitrarily small, the lemma is proved.

□

As we noted before, X_r/m_r converges in probability to 1 for $p_r\in(0,1)$ and $r=1,2$. Thus, taking ξ_r to be the binomial random variables X_r, it follows

that $R(n_1, n_2) = \frac{X_1+X_2}{m_1+m_2}$ also converges to 1 in probability when $n_1 \to \infty$ and $n_2 \to \infty$.

Lemma 6.9. *For the random variable* $R_1(n_1, n_2) = \frac{n_1+n_2-(X_1+X_2)}{n_1+n_2-(m_1+m_2)}$,

$$\lim_{\substack{n_1\to\infty \\ n_2\to\infty}} \mathbb{P}\left(|R_1(n_1, n_2) - 1| > \varepsilon\right) = 0, \tag{6.20}$$

where ε *is an arbitrary positive number.*

Proof of Lemma 6.9. For $\varepsilon_1 > 0$, Lemma 6.8 implies that for sufficiently large values of n_1 and n_2, the inequality

$$-\varepsilon_1 < 1 - R(n_1, n_2) < \varepsilon_1 \tag{6.21}$$

occurs with probability greater than $1 - \delta$, for $\delta > 0$ an arbitrarily small positive number. Then using the definition of R and R_1,

$$\begin{aligned}
& -\varepsilon_1 < && 1 - R(n_1, n_2) && < \varepsilon_1 \\
\Leftrightarrow\ & \frac{-(m_1+m_2)\varepsilon_1}{n_1+n_2-(m_1+m_2)} < && \tfrac{(m_1+m_2)-(X_1+X_2)}{n_1+n_2-(m_1+m_2)} && < \frac{(m_1+m_2)\varepsilon_1}{n_1+n_2-(m_1+m_2)} \\
\Leftrightarrow\ & \frac{-(m_1+m_2)\varepsilon_1}{n_1+n_2-(m_1+m_2)} < && R_1(n_1, n_2) - 1 && < \frac{(m_1+m_2)\varepsilon_1}{n_1+n_2-(m_1+m_2)}.
\end{aligned}$$

Now let $\varepsilon = \frac{p\ \varepsilon_1}{1-p}$, where $p = \max\{p_1, p_2\}$. Since

$$0 \le \frac{(m_1+m_2)}{n_1+n_2-(m_1+m_2)} \le \frac{m_1+m_2}{(n_1+n_2)(1-p)} \le \frac{(n_1+n_2)p}{(n_1+n_2)(1-p)} = \varepsilon/\varepsilon_1,$$

for the values of n_1 and n_2 such that the inequality (6.21) holds, we have

$$\begin{aligned}
\mathbb{P}\left(|R_1(n_1, n_2) - 1| < \varepsilon\right) &= \mathbb{P}\left(|R_1(n_1, n_2) - 1| < \frac{p\ \varepsilon_1}{1-p}\right) \\
&\ge \mathbb{P}(|1 - R(n_1, n_2)| < \varepsilon_1) \ge 1 - \delta,
\end{aligned}$$

where the numbers ε, ε_1 and δ are arbitrarily small. Hence

$$\lim_{\substack{n_1\to\infty \\ n_2\to\infty}} \mathbb{P}\left(|R_1(n_1,n_2)-1|>\varepsilon\right)=0, \tag{6.22}$$

i.e. $R_1(n_1,n_2)$ converges in probability to 1.

□

Lemma 6.10. *(Fisz [47]) If $\xi_r(\lambda_r)$ is asymptotically normal $N(m_r(\lambda_r),\sigma_r(\lambda_r))$, then the random variable $\xi_2-\xi_1$ is asymptotically normal $N(m_2-m_1,\psi)$ when $\lambda_1\to\infty$, $\lambda_2\to\infty$.*

For the proof of this lemma, see Fisz [47].

We now state a theorem by Cramér [33]* which we will also use in the proof of our theorem.

Theorem 6.11. *(Cramér, [33]) Let $\xi_1,\xi_2,\ldots$ be a sequence of random variables with distribution functions $F_1,F_2,\ldots$. Suppose that $F_n(x)$ tends to a distribution function $F(x)$ as $n\to\infty$.*

Let $\eta_1,\eta_2,\ldots$ be another sequence of random variables and suppose that η_n converges in probability to a constant c. Put

$$X_n=\xi_n+\eta_n,\ Y_n=\xi_n\eta_n,\ Z_n=\frac{\xi_n}{\eta_n}.$$

Then the distribution function of X_n tends to $F(x-c)$. Further, if $c>0$, the distribution function of Y_n tends to $F\left(\frac{x}{c}\right)$, while the distribution function of Z_n tends to $F(cx)$.

We can now prove our theorem, Theorem 6.7. Let

$$A=\frac{X_1+X_2}{n_1+n_2},\qquad B=(n_1+n_2-(X_1+X_2)),$$

*Theorem 6.11 and its proof can be found in Cramér [33], Section 20.6, p.254.

$$C = \frac{m_1 + m_2}{n_1 + n_2}, \qquad D = (n_1 + n_2 - (m_1 + m_2)),$$

and

$$y = (AB)/(CD) = \left(\frac{X_1 + X_2}{m_1 + m_2}\right)\left(\frac{n_1 + n_2 - (X_1 + X_2)}{n_1 + n_2 - (m_1 + m_2)}\right).$$

Then

$$\begin{aligned}
\tau(n_1, n_2) &= \frac{\zeta_B - m_B}{\sigma_B} \\
&= \frac{\frac{X_2 - X_1}{(AB)^{1/2}} - \frac{m_2 - m_1}{(CD)^{1/2}}}{\frac{\psi}{(CD)^{1/2}}} \\
&= \frac{(CD)^{1/2}(X_2 - X_1) - (m_2 - m_1)(AB)^{1/2}}{(ABCD)^{1/2}} \times \frac{(CD)^{1/2}}{\psi} \\
&= \frac{(CD)^{1/2}(X_2 - X_1 - (m_2 - m_1)) + (CD)^{1/2}(m_2 - m_1) - (m_2 - m_1)(AB)^{1/2}}{(AB)^{1/2}\psi} \\
&= \frac{(CD)^{1/2}(X_2 - X_1 - (m_2 - m_1)) + (m_2 - m_1)((CD)^{1/2} - (AB)^{1/2})}{(AB)^{1/2}\psi} \\
&= \frac{y^{-1/2}(X_2 - X_1 - (m_2 - m_1))}{\psi} + \frac{(m_2 - m_1)(y^{-1/2} - 1)}{\psi} \\
&= \frac{\eta + \frac{m_2 - m_1}{\psi}(1 - y^{1/2})}{y^{1/2}}, \qquad (6.23)
\end{aligned}$$

where $\eta(n_1, n_2) = \frac{X_2 - X_1 - (m_1 + m_2)}{\psi}$ is the random variable defined in the proof of Lemma 6.10 for our specific binomial case (see Fisz [47]). Note that $y(n_1, n_2) = \left(\frac{X_1 + X_2}{m_1 + m_2}\right)\left(\frac{n_1 + n_2 - (X_1 + X_2)}{n_1 + n_2 - (m_1 + m_2)}\right) = R(n_1, n_2)R_1(n_1, n_2)$, where R and R_1 are as defined in Lemma 6.8 and Lemma 6.9.

Note also that $\tau(n_1, n_2)$ is the random variable $\zeta_B(n_1, n_2)$ standardized by the asymptotic normal mean and standard deviation from equation (6.18). To prove the theorem, we need to show that τ is asymptotically normal $N(0, 1)$.

Due to Lemmas 6.8 and 6.9, the random variables $R(n_1, n_2)$ and $R_1(n_1, n_2)$ both converge in probability to 1. It follows from a proposition due to Slutsky[†],

[†]This proposition can be found in Cramér [33], Section 20.6, p.255.

a corollary to Theorem 6.11, that their product $y(n_1, n_2)$ also converges in probability to 1.

Using the same proposition again, this in turn implies that the function $y^{1/2}(n_1, n_2)$ converges in probability to $1^{1/2} = 1$, since this is a rational function in $y(n_1, n_2)$.

Since X_1 and X_2 are asymptotically normal $N(m_r, \sigma_r)$, then Lemma 6.10 applies here; thus $\eta(n_1, n_2)$ is asymptotically normal $N(0, 1)$, i.e. its distribution function converges to $\Phi(x)$.

Let us now consider the other expression in the numerator of τ. Note that when regarded as a function of y, we can write

$$(1 - y^{1/2}) = (1/2 + \theta)(1 - y),$$

where $\theta(y) = \frac{1-\sqrt{y}}{2(1+\sqrt{y})}$. Note that $\theta \to 0$ as $y \to 1$, which means that

$$\exists\ \delta_0 > 0 \quad \text{such that } |y - 1| < \delta_0 \Rightarrow |\theta| < \varepsilon_0, \tag{6.24}$$

for any positive number ε_0. Since y converges in probability to 1, for sufficiently large n_1, n_2 we also have

$$\mathbb{P}(|y - 1| < \delta_0) > 1 - \gamma, \tag{6.25}$$

for the $\delta_0 > 0$ as in equation 6.24 and for any arbitrarily small number γ. Combining equations 6.24 and 6.25, we obtain

$$\mathbb{P}(|\theta| < \varepsilon_0) \geq \mathbb{P}(|y - 1| < \delta_0) > 1 - \gamma,$$

for the values of n_1 and n_2 valid for the relation in equation 6.25; since γ can be arbitrarily small, θ converges in probability to 0.

Let us now write

$$z = \frac{m_2 - m_1}{\psi}(1 - y^{1/2}) = \frac{m_2 - m_1}{\psi}(1/2 + \theta)(1 - y).$$

Now $y = \left(\frac{X_1+X_2}{m_1+m_2}\right)\frac{B}{D}$, using the quantities B and D defined earlier. Thus

$$\begin{aligned} 1 - y &= 1 - \left(\frac{X_1 + X_2}{m_1 + m_2}\right)\frac{B}{D} \\ &= \frac{(m_1 + m_2)D - (X_1 + X_2)B}{(m_1 + m_2)D}. \end{aligned}$$

So

$$\begin{aligned} z &= -\frac{m_2 - m_1}{m_1 + m_2}(1/2 + \theta)\frac{(X_1 + X_2)B - (m_1 + m_2)D}{\psi D} \\ &= -\frac{m_2 - m_1}{m_1 + m_2}(1/2 + \theta)\varrho(n_1, n_2), \end{aligned}$$

where $\varrho(n_1, n_2) = \frac{(X_1+X_2)B-(m_1+m_2)D}{\psi D}$.

Note that ϱ can be expressed as $\varrho(n_1, n_2) = \frac{(X_1+X_2)R_1-(m_1+m_2)}{\psi}$.

A slight modification to the proof of Lemma 6.10 shows that $(X_1 + X_2)$ is asymptotically normal $N(m_1 + m_2, \psi)$. Due to Theorem 6.11, the random variable $(X_1 + X_2)R_1$ is also asymptotically normal $N(m_1 + m_2, \psi)$, since from Lemma 6.9, R_1 converges in probability to 1. It follows that ϱ is asymptotically normal $N(0, 1)$.

We want to show that z converges to zero in probability, in order to use Theorem 6.11 again to complete the proof of our theorem.

Let ε, $\delta > 0$ be arbitrary given numbers. Then

$$\mathbb{P}(|z| > \varepsilon) = \mathbb{P}\left(|z| > \varepsilon \middle| |\theta| > \delta\right) \mathbb{P}(|\theta| > \delta) + \mathbb{P}\left(|z| > \varepsilon \middle| |\theta| < \delta\right) \mathbb{P}(|\theta| < \delta). \tag{6.26}$$

Now

$$\mathbb{P}\left(|z| > \varepsilon \middle| |\theta| > \delta\right) \mathbb{P}(|\theta| > \delta) \to 0, \tag{6.27}$$

since θ converges in probability to zero, and thus it remains to show that the second summand in the expression (6.26) converges to zero. From the definition of z, we have

$$\begin{aligned} \mathbb{P}\left(|z| > \varepsilon \middle| |\theta| < \delta\right) &\leq \mathbb{P}\left(\left|\frac{m_2 - m_1}{m_1 + m_2}\right| (1/2 + |\theta|)|\varrho(n_1, n_2)| > \varepsilon \middle| |\theta| < \delta\right) \\ &\leq \mathbb{P}\left(\left|\frac{m_2 - m_1}{m_1 + m_2}\right| (1/2 + \delta)|\varrho(n_1, n_2)| > \varepsilon \middle| |\theta| < \delta\right). \end{aligned}$$

Let the event in the last expression above be denoted by E. Then conditional on $|\theta| < \delta$, the probability of E^c can be expressed as

$$\begin{aligned} & \mathbb{P}\left(\left|\frac{m_2 - m_1}{m_1 + m_2}\right| (1/2 + \delta)|\varrho(n_1, n_2)| < \varepsilon\right) \\ = \; & \mathbb{P}\left(\left|\frac{m_2 - m_1}{m_1 + m_2}\right| \varrho(n_1, n_2) < \frac{\varepsilon}{1/2 + \delta}\right) - \mathbb{P}\left(\left|\frac{m_2 - m_1}{m_1 + m_2}\right| \varrho(n_1, n_2) < \frac{-\varepsilon}{1/2 + \delta}\right) \\ \to \; & 1 - 0 = 1, \end{aligned}$$

since $\left|\frac{m_2 - m_1}{m_1 + m_2}\right| \leq \left|\frac{m_2 - m_1}{2\min\{m_1, m_2\}}\right| \to 0$ (due to the assumption (6.17) of the theorem) and the fact that $\varrho(n_1, n_2)$ is asymptotically normal $N(0, 1)$.

This implies that

$$\mathbb{P}\left(\left|\frac{m_2 - m_1}{m_1 + m_2}\right| (1/2 + \delta)|\varrho(n_1, n_2)| > \varepsilon \middle| |\theta| < \delta\right) \to 0. \tag{6.28}$$

The two relations (6.27) and (6.28) together imply that z converges in probability to 0.

Recall that we have

$$\tau(n_1, n_2) = \frac{\eta(n_1, n_2) + z(n_1, n_2)}{y(n_1, n_2)}.$$

Using the Cramér result, we see that the distribution of $(\eta + z)(n_1, n_2)$ tends to $\Phi(x - 0) = \Phi(x)$, since $\eta(n_1, n_2)$ is asymptotically normal $N(0, 1)$ and z converges in probability to 0.

Using the result again, the distribution of $\tau(n_1, n_2) = \left(\frac{\eta+z}{y}\right)(n_1, n_2)$ tends to $\Phi(1 \cdot x) = \Phi(x)$, since the distribution of $(\eta + z)(n_1, n_2)$ tends to $\Phi(x)$ and y converges in probability to 1. This completes the proof of the theorem.

□

Example 6.12. Let us now give an example of Theorem 6.7. Suppose $X_r \sim Bin(n_r, p)$ for $r = 1, 2$, i.e. the binomial random variables have equal trial probabilities. Then according to our theorem, the random variable $\zeta_B(X_1, X_2)$ will be asymptotically normal

$$N\left(\frac{(n_2 - n_1)p^{1/2}}{((n_1 + n_2)(1 - p))^{1/2}}, 1\right), \tag{6.29}$$

when $n_1 \to \infty$, $n_2 \to \infty$. In other words, using the transform $\zeta_B(X_1, X_2)$ will stabilize the variance of the asymptotic distribution.

Note also, that if in addition, we impose the constraint that the binomial sample sizes are the same, i.e. $n_1 = n_2$, the asymptotic distribution will be N(0,1), which is what we want.

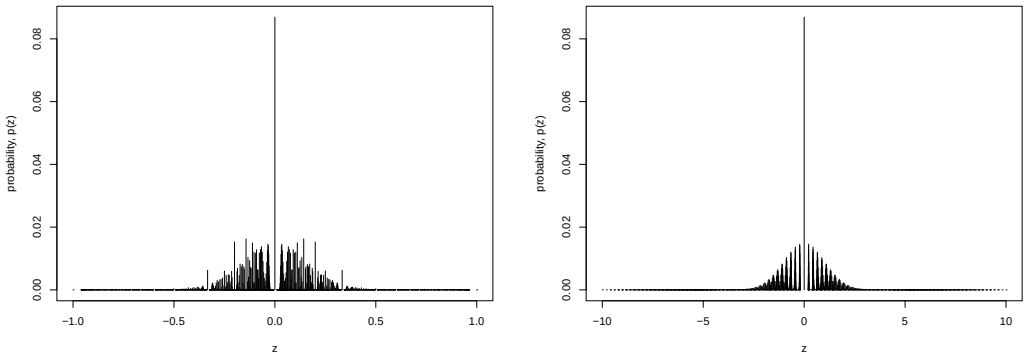

Figure 6.1: Distribution of $\zeta^1(X_1, X_2)$ (left) and $\zeta_B(X_1, X_2)$ (right) with $p_1 = p_2 = 0.3$ for $n_r = 50$.

Comparing the distributions of the Fisz transform for binomial random variables to this new Fisz-like transform is illustrative. Figure 6.1 shows the "exact" distribution of ζ^1 against ζ_B for fixed $n_r = 50$ and fixed $p_r = 0.3$, computed using the binomial probabilities associated to the variables X_1 and X_2. The graphs show a characteristic spike at zero. This represents where the random variables X_1 and X_2 are equal, i.e. where ζ^1 and ζ_B take the value zero. This is discussed further in Chapter 7.

Figures 6.2–6.4 demonstrate the convergence of the distribution of the random variables ζ^1 and ζ_B to the normal distribution, for trial probabilities $p_r = 0.3, 0.5, 0.7$ (respectively) and for increasing binomial sizes $n_r = 10, 50, 200$. The spike at $z = 0$ has been omitted for graphical clarity.

Note that for ζ_B, the shape of the distributions are much more similar to the normal distribution than ζ^1, even for smaller n. For some of the plots, the probabilities for a specific value of ζ^1 or ζ_B can be quite large compared to its neighbouring values, though this is less marked for ζ_B. This phenomenon is due to the values having different probability contributions (for a fixed n) from repetitions in the mass function table (see for example Table 7.1 in Chapter 7).

The properties of the new Fisz-like transform are discussed in the next section.

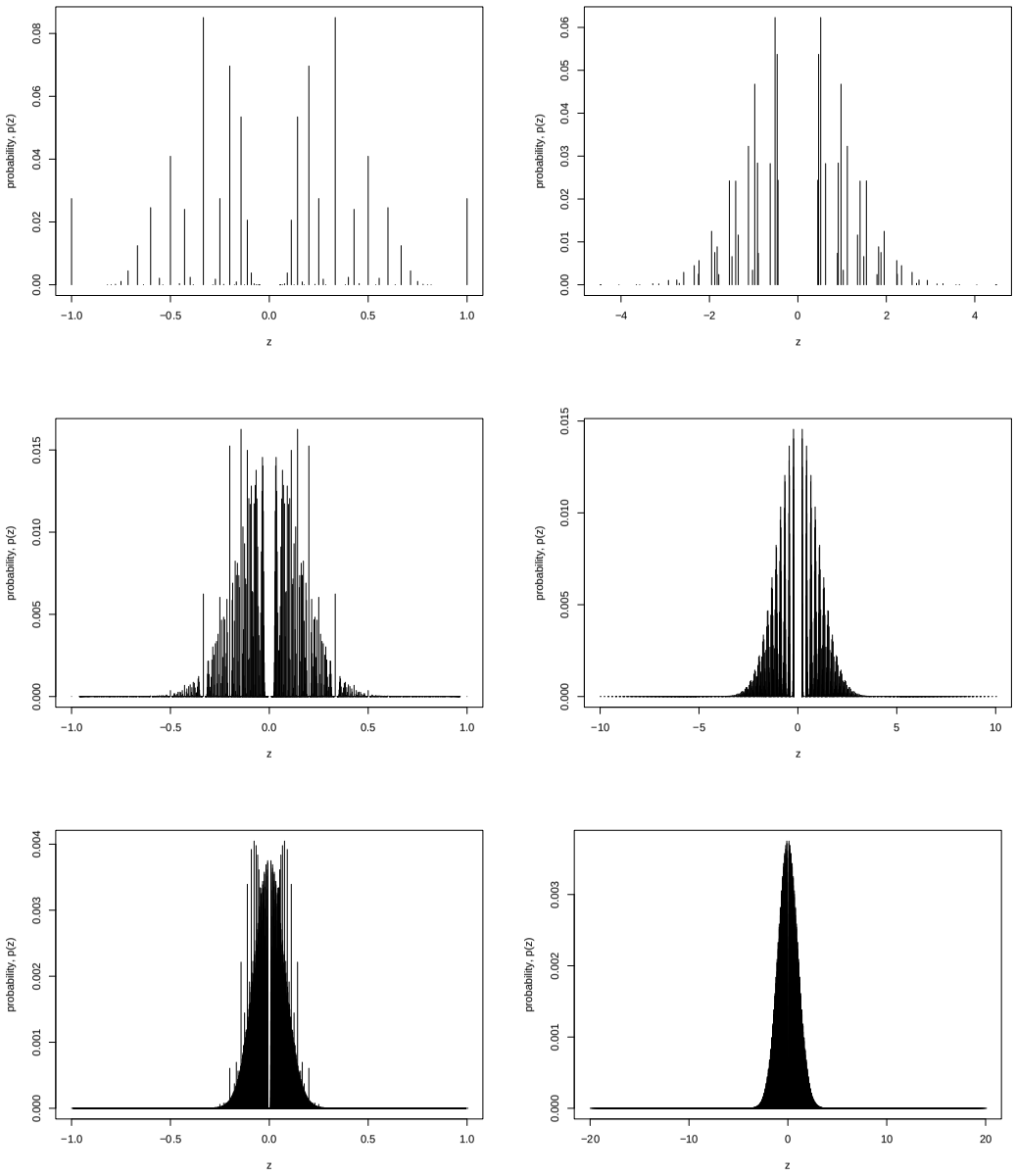

Figure 6.2: Distribution of $\zeta^1(X_1, X_2) \setminus \{0\}$ (left) and $\zeta_B(X_1, X_2) \setminus \{0\}$ (right) with $p_1 = p_2 = 0.3$ and different binomial sizes. Top: $n_r = 10$; middle: $n_r = 50$; and bottom: $n_r = 200$.

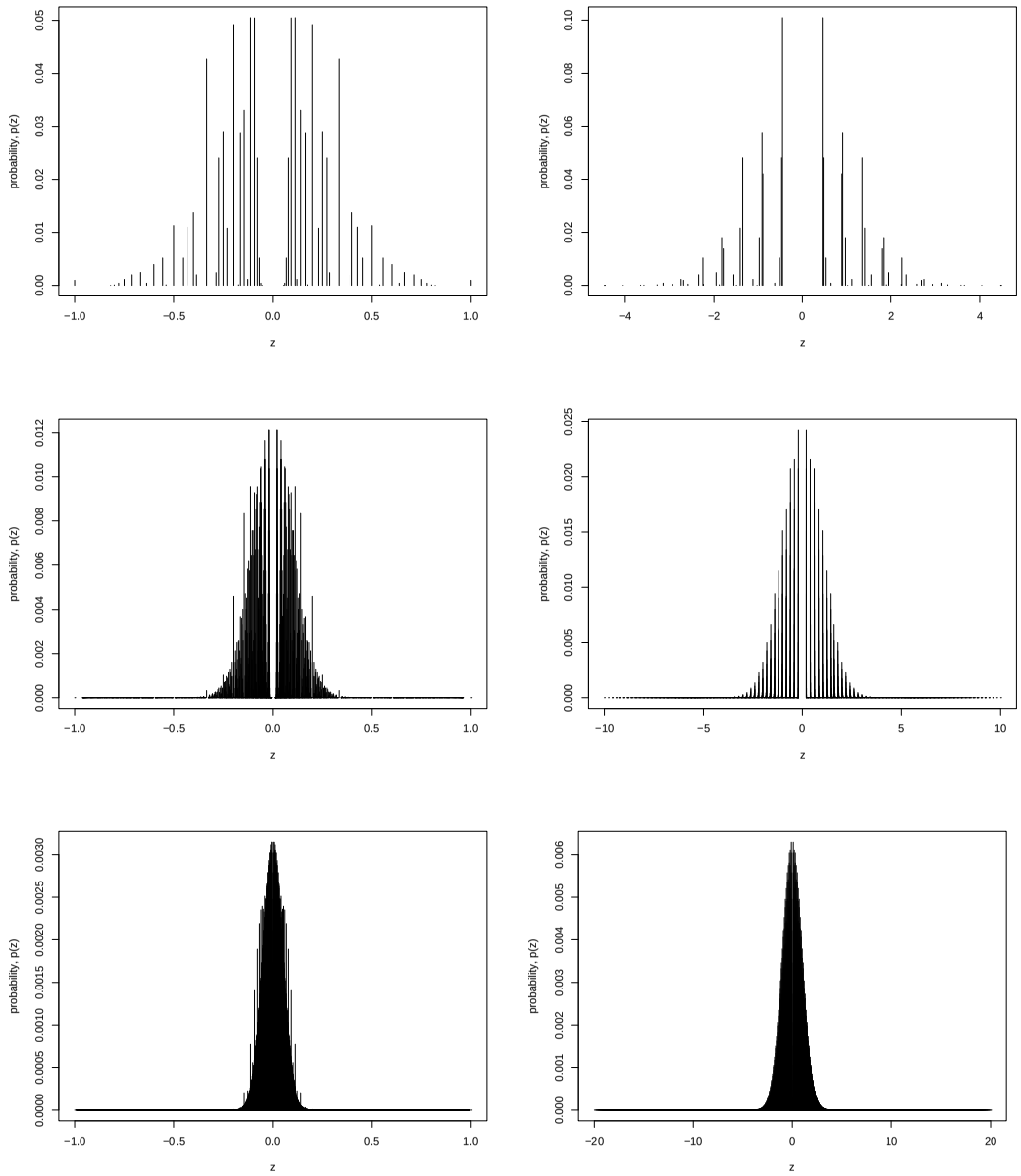

Figure 6.3: Distribution of $\zeta^1(X_1, X_2) \setminus \{0\}$ (left) and $\zeta_B(X_1, X_2) \setminus \{0\}$ (right) with $p_1 = p_2 = 0.5$ and different binomial sizes. Top: $n_r = 10$; middle: $n_r = 50$; and bottom: $n_r = 200$.

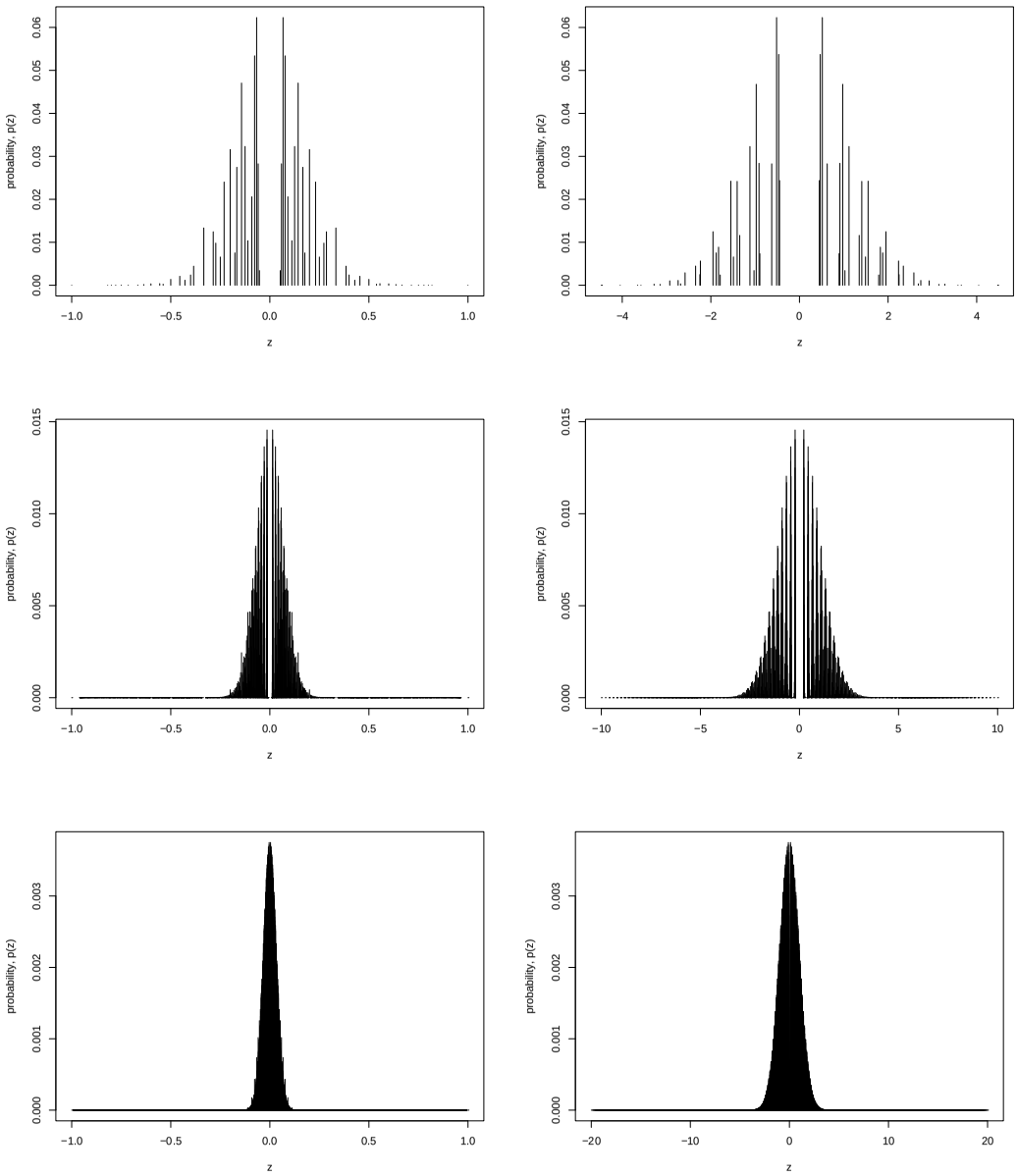

Figure 6.4: Distribution of $\zeta^1(X_1, X_2) \setminus \{0\}$ (left) and $\zeta_B(X_1, X_2) \setminus \{0\}$ (right) with $p_1 = p_2 = 0.7$ and different binomial sizes. Top: $n_r = 10$; middle: $n_r = 50$; and bottom: $n_r = 200$.

6.5 Gaussianization and Variance stabilization properties of the alternative Fisz transform

Note: this section should be read together with Appendix A; for clarity of presentation of the ideas, we limit the number of figures in this section, giving additional plots in the appendix.

In this section we demonstrate, through simulations, how well the transform ζ_B can bring binomial data closer to normality, whilst stabilizing the variance of the data. We might also like to know how fast the mean of ζ_B converges to the asymptotic normal mean. Even though the asymptotic normal distribution in equation (6.18) only holds when the means of the two binomial random variables are close (and large) through the condition (6.17), it is interesting to study these properties when this is not the case.

In some of the simulations below, we compare properties of our transform with that of the angular transformation (6.1) outlined in Section 6.1. We follow a similar approach to these simulations as Fryźlewicz and Nason [51]. However, since the size of the binomial means depends on the trial success probability, p, as well as the binomial size, n, the effect of both of these parameters feature in our simulations.

Let $X_r \sim B(n, p_r)$ for $r = 1, 2$. For each experiment, we sampled 10^5 values of X_r for binomial sizes $n = 1, 2, 4, 32, 128$ and for each probability lattice point (p_1, p_2), where p_r ranged from 0 to 1 in steps of 0.05. The binomial samples were then used to compute 10^5 values of the random variable $\zeta_B(X_1, X_2)$, denoted $z_n(p_1, p_2)$. For the comparisons with Anscombe's inverse sine transformation, the values of the binomial variable corresponding to the larger of the

two probabilities p_r was used. Since Anscombe's transformation works better for larger means, doing this would be favourable to Anscombe.

6.5.1 Mean simulations

To investigate the convergence of the samples of $\zeta_B(X_1, X_2)$ to the asymptotic mean in equation (6.18) (which we will denote by $\zeta_n(m_1, m_2)$), we computed their difference $|\bar{z}_n(p_1, p_2) - m_B|$, where the mean $\bar{z}$ is taken over the 10^5 samples.

Figure 6.5 shows the surface plots across the lattice of binomial probabilities (p_1,p_2) for increasing binomial size, n. The surfaces show that for larger n, the difference approaches zero across the whole lattice, which only a slight difference at the lattice boundary. Figure A.1 in Appendix A shows an equivalent figure with the surfaces converted to contour plots.

6.5.2 Variance simulations

Similar to the mean simulations, the sample variance was computed over the 10^5 samples of ζ_B arising from the samples of X_1 and X_2 for each point (p_1, p_2). Figure A.2 in Appendix A gives a 3D representation of the sample variance for each of the binomial sizes $n = 1, 2, 4, 32, 128$, renormalized so that the asymptotic distribution will have unit variance.

The plots show a "flattening" of the surface peaks as the binomial size increases, with the variance of the peak approaching one. In fact, this feature happens most near the line $p_1 = p_2$. This reflects the observation in Example 6.12 that equal binomial probabilities will result in an asymptotic distribution with unit variance. This characteristic is seen more clearly in Figure 6.6, which show contour plots corresponding to the surfaces of Figure A.2.

To further examine when the two binomial proportions are equal, Figure A.3 and Figure 6.7 display this case graphically for ζ_B and Anscombe's trans-

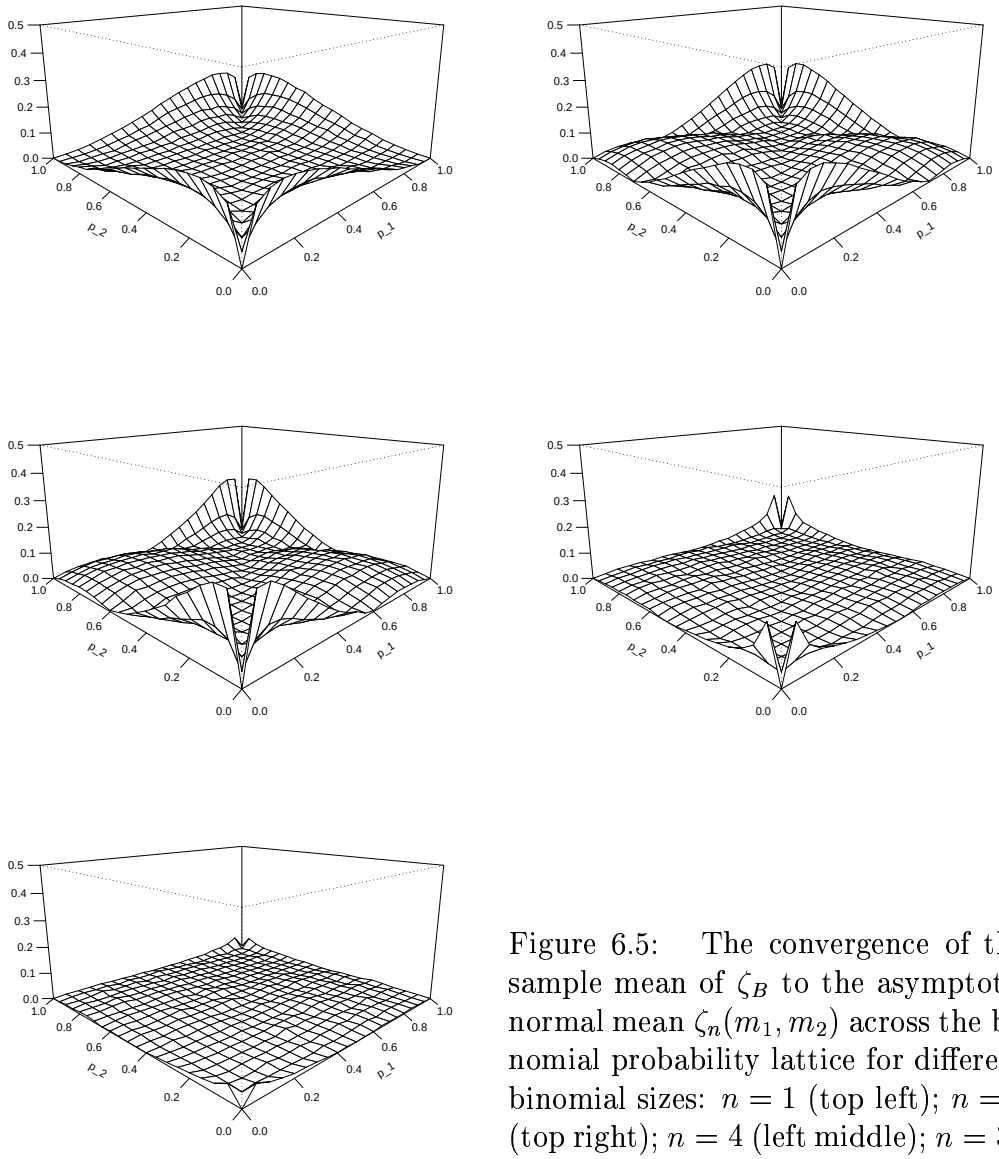

Figure 6.5: The convergence of the sample mean of ζ_B to the asymptotic normal mean $\zeta_n(m_1, m_2)$ across the binomial probability lattice for different binomial sizes: $n = 1$ (top left); $n = 2$ (top right); $n = 4$ (left middle); $n = 32$ (right middle); $n = 128$ (bottom).

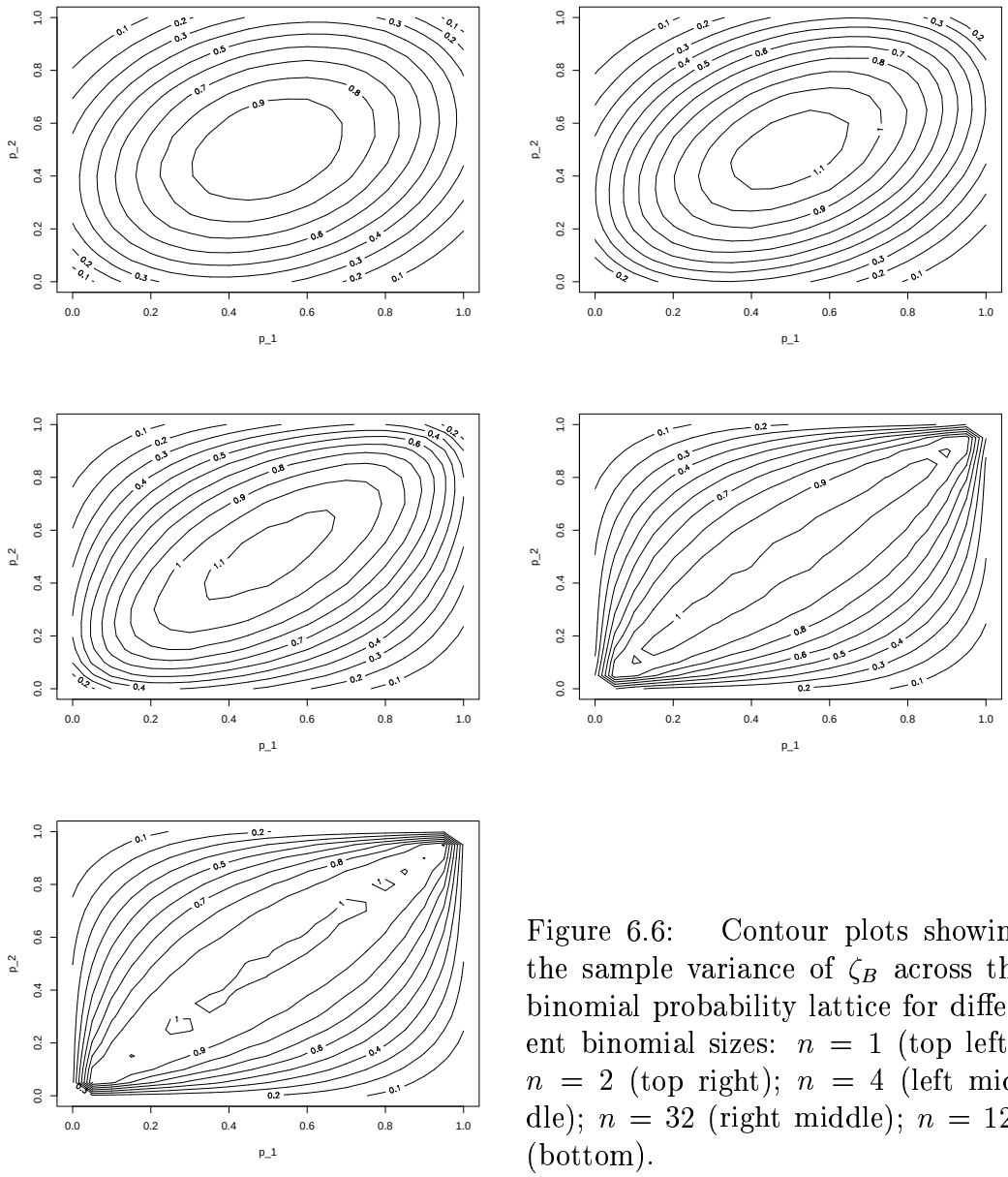

Figure 6.6: Contour plots showing the sample variance of ζ_B across the binomial probability lattice for different binomial sizes: $n = 1$ (top left); $n = 2$ (top right); $n = 4$ (left middle); $n = 32$ (right middle); $n = 128$ (bottom).

formation, $\mathcal{A}$, for increasing n.

Figure A.3 compares the sample variances for $p_1 = p_2 \in (0, 1)$. It is evident that for small n, both transforms have samples variances far from one, though our transform does better. In the middle of the interval, Anscombe's transform has a low sample variance, whereas our transform, ζ_B tends to have a sample variance above one.

To compare the two transforms on this interval more closely, Figure 6.7 plots the squared residual of the variance from one against the (equal) binomial proportion.

From this plot, it is more obvious that for small binomial sizes, our transform has variance closer to one for low and high proportions, and comparable to Anscombe's transformation for the middle half interval (0.25,0.75). For larger n, both transforms do well at stabilizing the variance at one.

6.5.3 Gaussianization simulations

For judging the relative Gaussianizing properties of the transform ζ_B, we computed the Kolmogorov-Smirnov statistics for both ζ_B and for Anscombe's transformation over the binomial proportion lattice. Lower Kolmogorov-Smirnov statistics are representative of samples which are more Gaussian. Figures A.4 and 6.8 show perspective and contour plots (respectively) of the difference in Kolmogorov-Smirnov statistics between Anscombe's transform and ζ_B. A positive difference in these plots corresponds to our transform being more Gaussian.

Though some of the behaviour is slightly erratic as n gets large, the overall trend is that for any binomial size, the difference in Kolmogorov-Smirnov statistics is positive, irrespective of the binomial proportions p_1 and p_2. This demonstrates that our transform has better Gaussianization properties than Anscombe.

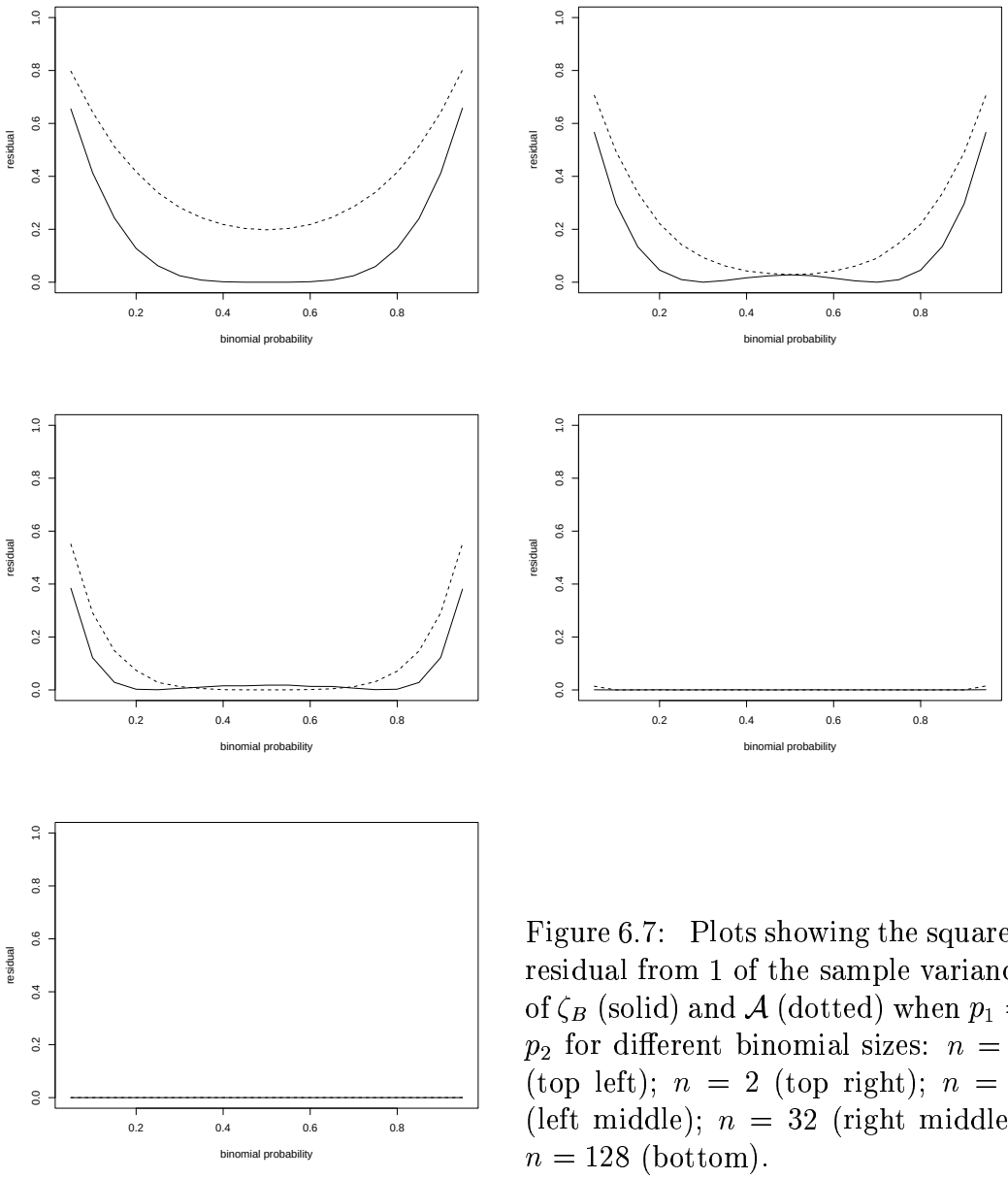

Figure 6.7: Plots showing the squared residual from 1 of the sample variance of ζ_B (solid) and $\mathcal{A}$ (dotted) when $p_1 = p_2$ for different binomial sizes: $n = 1$ (top left); $n = 2$ (top right); $n = 4$ (left middle); $n = 32$ (right middle); $n = 128$ (bottom).

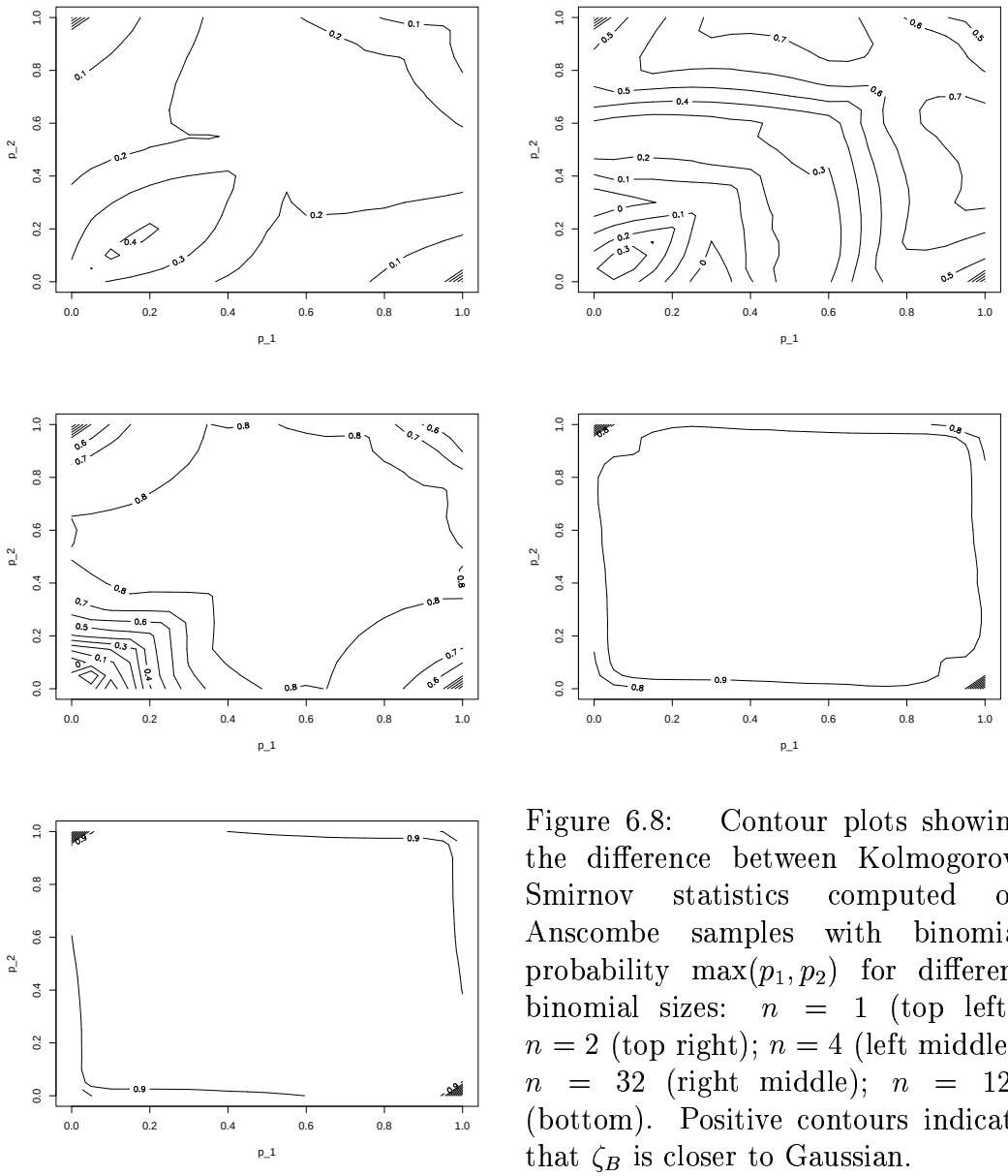

Figure 6.8: Contour plots showing the difference between Kolmogorov-Smirnov statistics computed on Anscombe samples with binomial probability $\max(p_1, p_2)$ for different binomial sizes: $n = 1$ (top left); $n = 2$ (top right); $n = 4$ (left middle); $n = 32$ (right middle); $n = 128$ (bottom). Positive contours indicate that ζ_B is closer to Gaussian.

6.6 Alternative Haar-Fisz transform for binomial random variables

In this section we introduce an algorithm similar to the Haar-Fisz transform described in Section 6.3, based on the asymptotic result from the preceding section.

Let us describe the alternative Haar-Fisz transform. Suppose we have an input vector $\mathbf{v}=(v_0, v_1, \ldots, v_{N-1})$ of length $N = 2^J$, with $0 \leq v_i \leq n$, for some integer n.

1. Perform the Haar DWT on $\mathbf{v}$ to obtain the vector $(\boldsymbol{c_0},\boldsymbol{d_0},\boldsymbol{d_1},\ldots,\boldsymbol{d_{J-1}})$. As each level is produced, modify the coefficients as follows:

$$f_{j,k} = \begin{cases} 0 & \text{if } c_{j,k} = 0 \text{ or } c_{j,k} = n, \\ d_{j,k}/\sqrt{c_{j,k}(n - c_{j,k})/2n} & \text{otherwise} \end{cases} \tag{6.30}$$

2. Perform the inverse Haar DWT on the vector $(\boldsymbol{c_0},\boldsymbol{f_0},\boldsymbol{f_1},\ldots,\boldsymbol{f_{J-1}})$. Call the result $\mathbf{u}$.

We denote this transform by $\mathbf{u}:=\mathcal{F}_B\mathbf{v}$. As with the usual Haar-Fisz transform, $\mathcal{F}_B$ can be inverted by "undoing" the steps 1 and 2.

Let us examine the effect of the modification in step 1 of the above procedure. Consider the coefficients v_0 and v_1. The modified detail coefficient $f_{J-1,0}$ is produced by

$$\begin{aligned} f_{J-1,0} &= \frac{\frac{1}{2}(v_1 - v_0)}{\left(\frac{1}{2}(v_0 + v_1)\left(n - \frac{v_0+v_1}{2}\right)/2n\right)^{1/2}} \\ &= \frac{(v_1 - v_0)}{\left((v_0 + v_1)(2n - (v_0 + v_1))/2n\right)^{1/2}}. \end{aligned}$$

Similarly, for the next coarsest level coefficient, we have

$$\begin{aligned} f_{J-2,0} &= \frac{\frac{1}{2}(c_{J-1,1} - c_{J-1,0})}{\left(\frac{1}{2}\left(c_{J-1,0} + c_{J-1,1}\right)\left(n - \frac{c_{J-1,0}+c_{J-1,1}}{2}\right)/2n\right)^{1/2}} \\ &= \frac{\frac{1}{4}\left((v_0+v_1)-(v_3+v_4)\right)}{\left(\frac{v_0+v_1+v_2+v_3}{4}\left(n - \frac{v_0+v_1+v_2+v_3}{4}\right)/2n\right)^{1/2}} \\ &= \frac{\left((v_0+v_1)-(v_3+v_4)\right)}{\left((v_0+v_1+v_2+v_3)\left(4n-(v_0+v_1+v_2+v_3)\right)/2n\right)^{1/2}}. \end{aligned}$$

This computation is similar for every coefficient within a level, and for each DWT decomposition level. If the data vector $\mathbf{v}$ is representative of observations from i.i.d. binomial random variables $X_k \sim (n, p)$, then the detail coefficients can be expressed as $d_{j,k} = 2^{(J-j-1)/2}\zeta_B(Y_1, Y_2)$, where Y_1 and Y_2 are both sums of 2^{J-j-1} of the random variables X_k, and thus are binomially distributed as well. This is the analogue of the implications from equations (6.14) – (6.16) for the usual Haar-Fisz case.

Since the application of the inverse Haar transform is identical for $\mathcal{F}_B\mathbf{v}$ as for $\mathcal{F}\mathbf{v}$, after performing the transform $\mathcal{F}_B\mathbf{v}$, the original data can be expressed as a linear combination of quantities of the form $\zeta_B(Y_1, Y_2)$ for some binomial random variables Y_1, Y_2, analogous to equations (6.10) – (6.13).

6.7 Gaussianization and Variance stabilization properties of the alternative Haar-Fisz transform

The following investigation compares the Gaussianizing and variance-stabilizing properties of the transform $\mathcal{F}_B$, introduced in Section 6.6 with Anscombe's transformation (6.1), and the identity transformation.

For these simulations, we have chosen a binomial proportion vector, $\boldsymbol{p}$

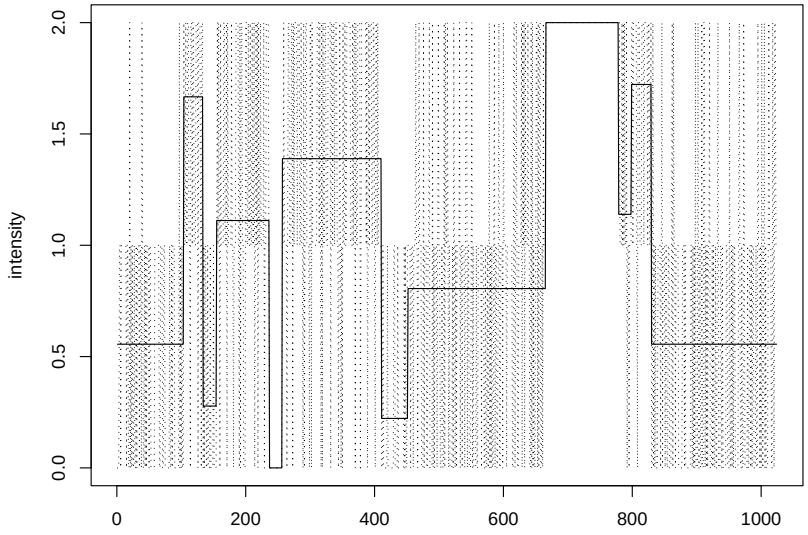

Figure 6.9: Mean intensity vector $\boldsymbol{\lambda}$ based on $\boldsymbol{p}$ and binomial size $n = 2$, together with an example sample path (dotted).

of length $N = 1024$ sampled from a (normalized and stretched) version of the Donoho and Johnstone *Blocks* test signal [39]. For each binomial size $n = 1, 2, 4, 32, 128$, we will denote by $\boldsymbol{\lambda} := n\boldsymbol{p}$ the mean intensity vector corresponding to n. It should be noted that although the mean vector depends on the binomial size, n, this is not included in the notation explicitly, since it will be obvious from the context which value of n we will use. A sample path generated from binomial random variables with the mean vector $\boldsymbol{\lambda}$ will be denoted by $\mathbf{v}$. Figure 6.9 shows the (mean) intensity vector for $n = 2$, overlaid with a sample path generated from it. As expected, the sample path takes the value 1 more often when $\boldsymbol{p}$ is near 1, and hits zero more frequently when $\boldsymbol{p}$ is near zero.

6.7.1 Gaussianizing simulations

We compared the Gaussianizing properties of the different transforms by considering the Q-Q plots of $\mathbf{v}-\boldsymbol{\lambda}$ (identity transform), $\mathcal{A}\mathbf{v}-\mathcal{A}\boldsymbol{\lambda}$ (Anscombe)

and $\mathcal{F}_B\mathbf{v}-\mathcal{F}_B\boldsymbol{\lambda}$ (alternative Haar-Fisz), averaged over 100 sample paths, $\mathbf{v}$. These paths were created from the mean vector $\boldsymbol{\lambda}$ for the binomial sizes $n = 1, 2, 4, 32, 128$.

Figures 6.10 – 6.12 show this comparison for the binomial sizes $n = 2, 4$ and 128.

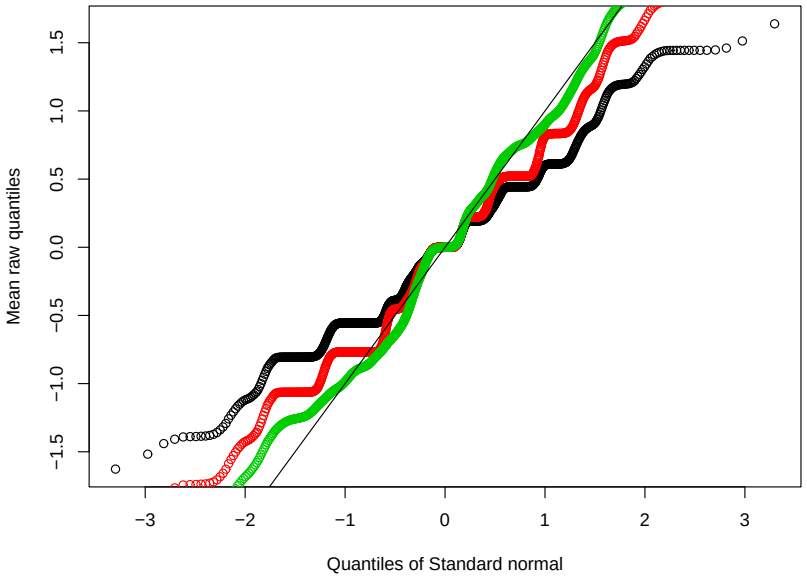

Figure 6.10: Q-Q plot comparison for three different transforms, averaged over 100 paths sampled from binomial variables with size $n = 2$ and proportion vector $\boldsymbol{p}$: $\mathbf{v}-\boldsymbol{\lambda}$ (black); $\mathcal{A}\mathbf{v}-\mathcal{A}\boldsymbol{\lambda}$ (red); $\mathcal{F}_B\mathbf{v}-\mathcal{F}_B\boldsymbol{\lambda}$ (green). Solid line has slope 1, indicating unit variance.

For the lowest binomial sizes, namely $n = 1, 2$, the raw data (marked in black) is quite "stepped". This is expected since the data are discrete. The Anscombe transformed data still exhibits this characteristic, whilst our transform, $\mathcal{F}_B$, has lost most of this stepped character; the data lies closer to a straight line, showing that the data is more Gaussian. Moreover, the data is closer to the solid line, which indicates a variance of one (it has a slope of 1).

As n increases, the Q-Q lines become similar, although it can be said that our transform displays slightly better Gaussianization (and also variance-stabilization), since the quantile points do not deviate from the (solid) straight

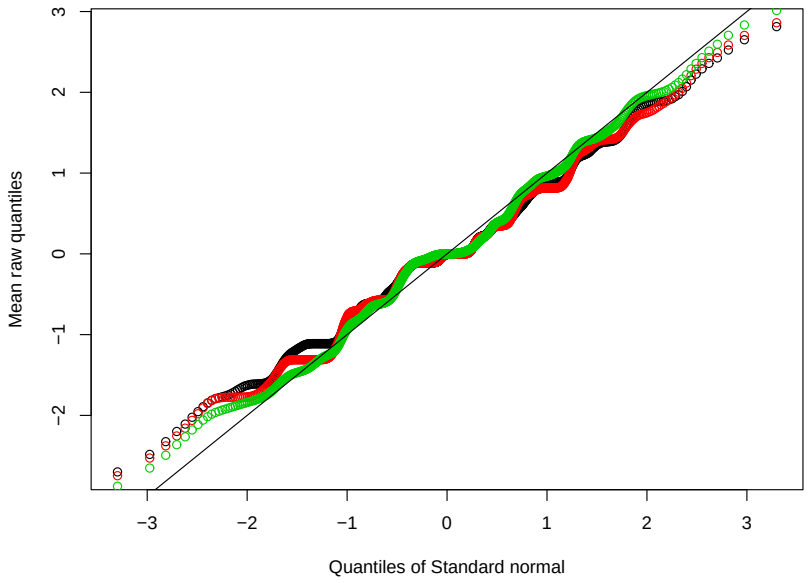

Figure 6.11: Q-Q plot comparison for three different transforms, averaged over 100 paths sampled from binomial variables with size $n = 4$ and proportion vector $\boldsymbol{p}$: $\mathbf{v}-\boldsymbol{\lambda}$ (black); $\mathcal{A}\mathbf{v}-\mathcal{A}\boldsymbol{\lambda}$ (red); $\mathcal{F}_B\mathbf{v}-\mathcal{F}_B\boldsymbol{\lambda}$ (green). Solid line has slope 1, indicating unit variance.

line as much as the other transforms, especially at the tails.

For large n, both the Anscombe transform and our transform do very well at bringing the data to normality: both lines are straight (and coincide). Furthermore, the variance is very close to one. However, this is mostly expected due to the high value of n, since at this large binomial size, the Central Limit Theorem comes into effect.

6.7.2 Variance simulations

To assess how well the transformations $\mathcal{A}$ and $\mathcal{F}_B$ force the data to have variance nearer to one, we plotted the squared residual $|\mathcal{A}\mathbf{v}-\mathcal{A}\boldsymbol{\lambda}|^2$ and $|\mathcal{F}_B\mathbf{v}-\mathcal{F}_B\boldsymbol{\lambda}|^2$ for Anscombe's transformation and our transform (respectively). The residuals were averaged over 1000 sample paths, which were generated from the mean intensity vector $\boldsymbol{\lambda}$ for binomial sizes $n = 1, 2, 4, 32, 128$.

The squared residuals for the two transforms are given in Figures 6.13 –

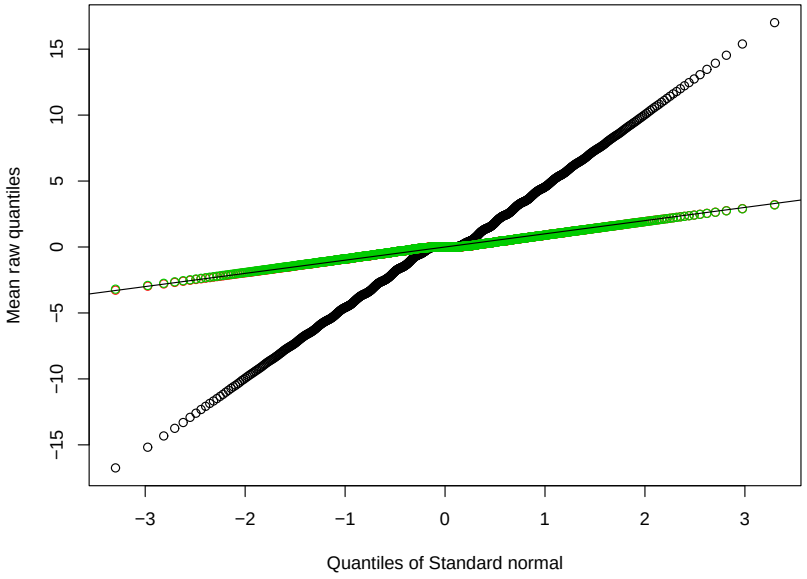

Figure 6.12: Q-Q plot comparison for three different transforms, averaged over 100 paths sampled from binomial variables with size $n = 128$ and proportion vector $\boldsymbol{p}$: $\mathbf{v}-\boldsymbol{\lambda}$ (black); $\mathcal{A}\mathbf{v}-\mathcal{A}\boldsymbol{\lambda}$ (red); $\mathcal{F}_B\mathbf{v}-\mathcal{F}_B\boldsymbol{\lambda}$ (green). Solid line has slope 1, indicating unit variance. Here, both red and green lines coincide.

6.15 for $n = 2, 4$ and 128.

When the binomial size is small, the simulations show that our transform does much better than the competitor, $\mathcal{A}$, at stabilizing the sample path variances. For example, for $n = 2$, the Anscombe transform has the squared residual in the range 0.6 to 0.8, whereas for our transform, the residual is near 1 for most of the sample path range. Further, our transform does relatively well compared to Anscombe when the binomial proportion is small, that is in the three non-zero 'troughs'. However, there is a degree of erratic behaviour near the discontinuities in the proportion vector.

Moderate binomial sizes have the Anscombe transformation beginning to achieve similar stabilization as our transform; when $n = 128$, both transforms do very well at variance stabilization, though Anscombe can be considered to do slightly better in performance in this case, due to the occasional downward

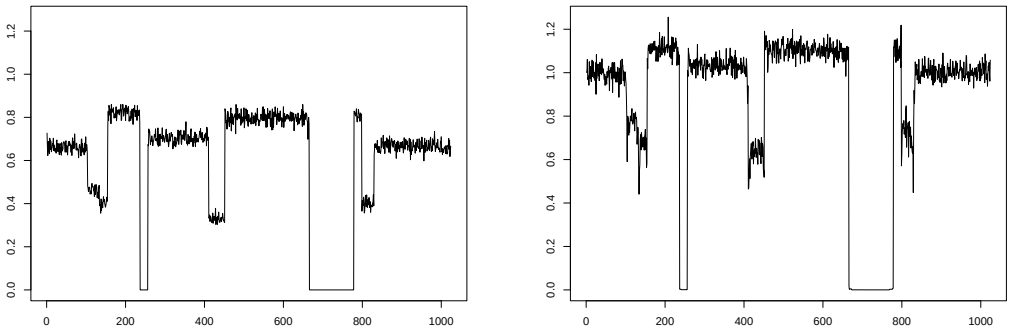

Figure 6.13: Squared residuals for different Gaussianizing transforms, averaged over 1000 sample paths from binomial variables with size $n = 2$ and proportion vector $\boldsymbol{p}$: $|\mathcal{A}\mathbf{v}-\mathcal{A}\boldsymbol{\lambda}|^2$ (left); $|\mathcal{F}_B\mathbf{v}-\mathcal{F}_B\boldsymbol{\lambda}|^2$ (right).

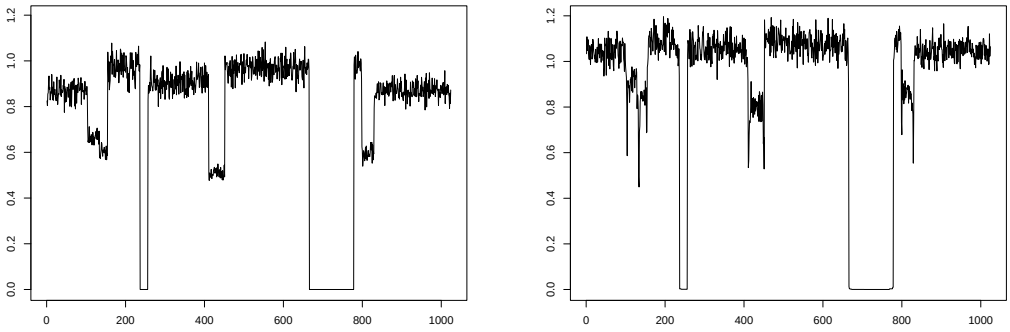

Figure 6.14: Squared residuals for different Gaussianizing transforms, averaged over 1000 sample paths from binomial variables with size $n = 4$ and proportion vector $\boldsymbol{p}$: $|\mathcal{A}\mathbf{v}-\mathcal{A}\boldsymbol{\lambda}|^2$ (left); $|\mathcal{F}_B\mathbf{v}-\mathcal{F}_B\boldsymbol{\lambda}|^2$ (right).

spikes (see Figure 6.15).

6.8 Binomial proportion estimation

Motivated by these observations about the properties of the transform $\mathcal{F}_B$, we now propose an algorithm for binomial probability estimation, similar to that in Section 6.3.

Suppose $\mathbf{v}=(v_0, \ldots, v_{n-1})$ is a vector of observations of length $n = 2^J$ from

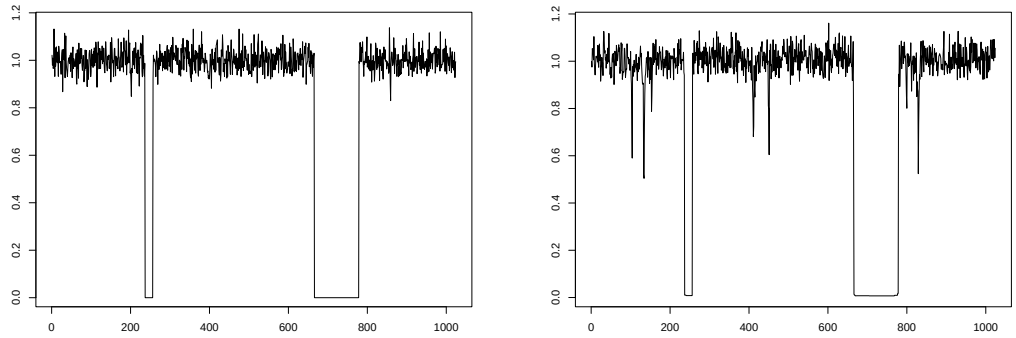

Figure 6.15: Squared residuals for different Gaussianizing transforms, averaged over 1000 sample paths from binomial variables with size $n = 128$ and proportion vector $\boldsymbol{p}$: $|\mathcal{A}\mathbf{v} - \mathcal{A}\boldsymbol{\lambda}|^2$ (left); $|\mathcal{F}_B\mathbf{v} - \mathcal{F}_B\boldsymbol{\lambda}|^2$ (right).

a binomial process with size N (with N large) and unknown probability vector p.

1. Perform the transform $\mathcal{F}_B$ on $\mathbf{v}$ to produce $\mathbf{u}=\mathcal{F}_B\mathbf{v}$. The vector $\mathbf{u}$ should be approximately normally distributed.

2. Use any wavelet denoiser suitable for Gaussian noise.

3. Invert $\mathcal{F}_B$ to obtain the estimate of the binomial probability vector.

6.8.1 Application: DNA Isochore detection

There has been substantial work in the field of bioinformatics in recent years, and the quest to improve existing methods and computational techniques is also of great importance. In particular, DNA sequencing and gene expression methods are a couple of the hot topics in this area.

One of the problems in these areas is the modelling and prediction of isochore clusters in DNA sequence data. This information is useful to know for a range of biological applications.

In this section we hope to use the Gaussianizing and variance stabilizing properties of the random variable $\zeta_B(X_1, X_2)$ for a bioinformatics application.

Biological background to the isochore problem

Before expressing the problem in a mathematical context, we now outline the problem in a biological setting.

DNA sequences are strings (polymers) of nucleotides, which store genetic information. Nucleotides are chemical compounds which play important rôles, for example in cellular behaviour and enzyme regulation.

Each nucleotide is characterized by its nitrogen base, represented by a letter: A (adenine); C (cytosine); G (guanine); and T (thymine). These four nucleotide bases come from two compound groups, namely *purines* (adenine and thymine) and *pyrimidines* (cytosine and guanine), differing in structure. The nucleotides from a specific compound group are referred to as *base pairs*. For a more detailed discussion of the structure of DNA, see any introductory text on genomics, for example [18, 34, 31].

A DNA isochore is a long DNA segment which are (fairly) homogeneous in G+C content [81]. G+C content can be seen as the ratio between the number of pyrimidine nucleotides to the total number of nucleotides in a DNA segment.

A school of thought in bioinformatics accepts an isochore model for DNA, which asserts that genomes (chromosome DNA sequences) are mosaics of long DNA segments with different G+C content in adjacent segments; under this model, the G+C content mosaics differ for different organisms, especially between warm- and cold-blooded vertebrates [13], and so these features of DNA G+C content could be used, for example, in organism classification applications.

Although the isochore features of certain vertebrates have already been investigated, an effective prediction method is of obvious interest.

IsoFinder: an existing approach to the isochore problem

In [81] and [103], a procedure of sequential hypothesis testing is implemented to attempt to model the distribution of G+C cluster sizes of a DNA sequence.

The procedure works as follows. The G+C content of the sequence is counted, and a t-statistic is used to assess the significance of the difference in mean G+C values on either side of a sliding pointer moving along the DNA sequence. After heterogeneity is filtered out, the information is used to split the original sequence into two distinct regions of differing G+C mean value. This is method is then repeated on successive blocks until the original sequence is divided into a number of regions with significantly different mean G+C levels. These obtained clusters are predictions of isochores of the original DNA sequence. We call this method the *IsoFinder* procedure.

The similarities between this method and the multiscale structure of a wavelet decomposition motivates us to investigate whether there is a way of predicting isochores more efficiently using wavelet methods.

Haar-Fisz approach to the isochore problem

Our aim is to use the Haar-Fisz algorithm, applying it to the isochore situation. To set up our procedure, we shall consider a DNA sequence. Since we are interested in the sections of the strand containing G+C content, we can view the DNA section as a binary sequence with a corresponding sequence of indicator values at each nucleotide site, showing whether or not a particular nucleotide comes from the pyrimidine (G or C) base pair:

Example DNA sequence:

ATGCGCTACGTGCATGCAGTACCATGGACGGTACGGTGACGT

Converted sequence:

00111100110110011010011001101110011101011O

For an unseen strand, if we assume each molecule along the sequence is from one of the two nucleotide base pairs independently, we can assign (independent) Bernoulli random variables on the nucleotide sites as follows. Suppose we have a DNA sequence of length $n = 2^J$. Let X_k indicate the type of nucleotide k. Then $X_k \sim$Bernoulli(p_k), and so

$$\begin{aligned} \mathbb{P}(\text{nucleotide } k \text{ has G+C content}) &= \mathbb{P}(X_k = 1) = p_k \\ \mathbb{P}(\text{nucleotide } k \text{ has A+T content}) &= \mathbb{P}(X_k = 0) = 1 - p_k = q_k. \end{aligned}$$

Examples

To test the GC proportion estimation procedure, two chromosome strands were acquired from the Wellcome Trust Sanger Institute Human Genome Sequencing Group[‡], namely the chromosome 6 MHC strand (examined in Oliver *et al.* [81]) and chromosome 20 of the human genome. To make it feasible to process this data with our method, the sequence strands were cropped to $2^{21} = 2097152$ bases, and then converted into binary sequences indicating GC content as outlined above.

In the denoising step of the algorithm in Section 6.8, we used the Haar DWT with *Sureshrink* thresholding (with primary resolution level 3). However, we modified the smoothing procedure. Recall that in the IsoFinder procedure, there is an in-place heterogeneity filtering. This is usually applied to filter out isochores of less than 3 kilobases from the resulting isochore maps, so that these map estimates resemble mammalian genomes [81]. To mimic this filtering, in

[‡]All sequences produced by the Sanger Institute are available online from the website http://www.sanger.ac.uk/HGP/ .

the denoising step of the procedure, we set the finest 11 detail coefficient levels to zero (after thresholding) before inverting the discrete wavelet transform. This has the effect of ensuring that isochore regions of less than $2^{11} = 2048$ bases do not feature in our estimates of GC content produced after inversion of the DWT.

To assess our isochore map estimates, the *IsoFinder* method was applied to the original (cropped) nucleotide sequences, using the online *IsoFinder* implementation. Figures 6.16 and 6.18 were also created using this web interface[§].

Figures 6.16 and 6.17 show the isochore maps of the MHC nucleotide sequence for the two estimation procedures, whereas Figures 6.18 and 6.19 give the corresponding estimates for the chromosome 20 of the human genome. Whilst the estimates produced using our method are more "spiky" and show shorter isochore regions, it could be said that the estimates for both procedures represent the same underlying proportion function. It should be noted here that our estimates use *SureShrink* thresholding, with no consideration for the effect of the primary resolution level. More complex thresholding procedures, for example, *EbayesThresh* (described in Chapters 3 and 4) could produce estimates more similar to the *Isofinder* procedure, though this increased complexity would lead to increased computation time, due to the length of the datasets being analyzed.

6.9 Conclusions and Further work

This chapter has investigated the classical regression problem of binomial proportion intensity estimation. Since the Fisz transform has no variance stabilizing properties for binomial random variables, we proposed a new transform, ζ_B, that has.

[§]This can be found at http://bioinfo2.ugr.es/IsoF/isofinder.html.

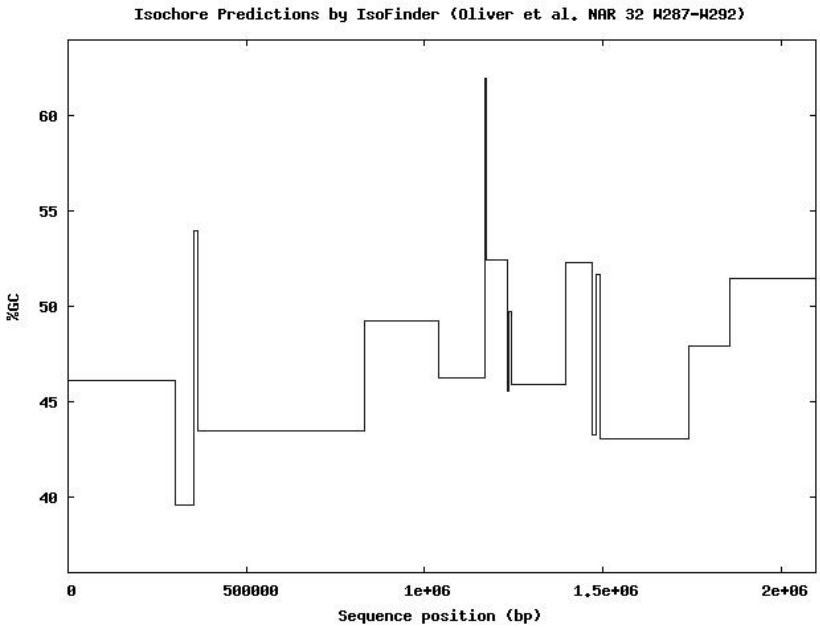

Figure 6.16: Isochore map of the chromosome 6 MHC nucleotide sequence as estimated by the *Isofinder* procedure (with 3 kilobase filtering).

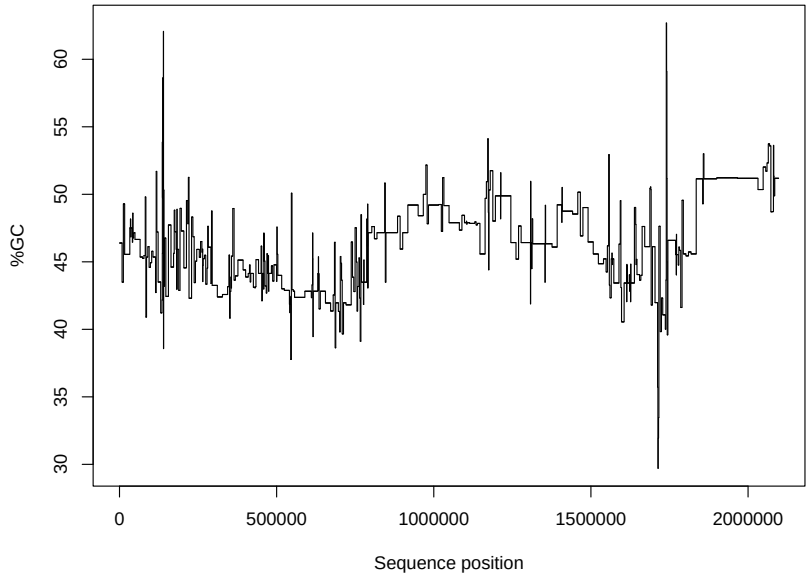

Figure 6.17: Isochore map of the chromosome 6 MHC nucleotide sequence as estimated by our Haar-Fisz Gaussianizing procedure (with 11 finest detail coefficient levels set to zero).

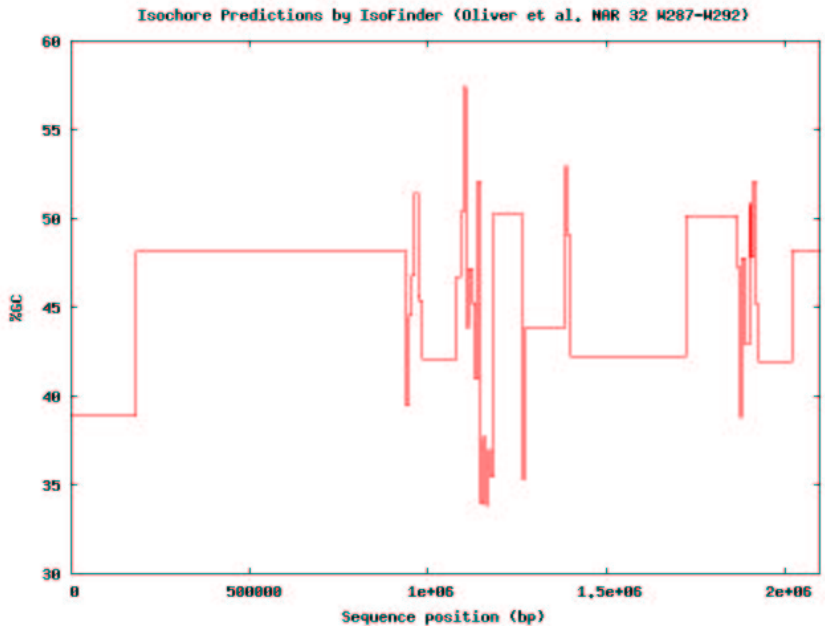

Figure 6.18: Isochore map of chromosome 20 of the human genome as estimated by the *Isofinder* procedure (with 3 kilobase filtering).

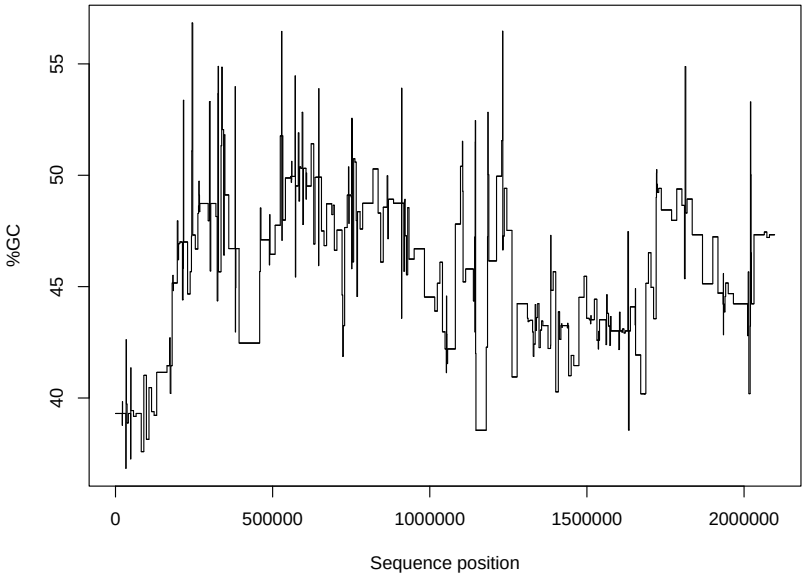

Figure 6.19: Isochore map of chromosome 20 of the human genome as estimated by our Haar-Fisz Gaussianizing procedure (with 11 finest detail coefficient levels set to zero).

An asymptotic result was established about this transform similar to that of Fisz, and simulations for different binomial sizes and probabilities were performed to show how well it Gaussianizes and stabilizes the variance compared to the Anscombe transformation. The results indicate that our transform does very well for smaller binomial sizes, n, and/or for extreme binomial proportions. As n is increased, the two transforms are comparable.

Section 6.6 introduced a new Haar-Fisz transform using our Gaussianizing transform. This was compared to the Anscombe transform also, and it was found to again outperform the traditional transformation for smaller binomial sizes and/or binomial proportions nearer the boundaries of the interval (0,1). This improvement for small n and extreme proportions is important, since in practice, large binomial sizes and "nice" success probabilities could be unrealistic. Both methods perform well when n is large.

The evidence of good properties from the simulations lead us to suggest an algorithm for binomial proportion estimation. We used a real data example to test the performance of the procedure. The example showed the algorithm to perform well, and using more sophisticated thresholding procedures or Gaussian denoisers may lead to predicted isochore maps with more distinct regions.

Chapter 7

Distributional features of Fisz-transformed Binomial Random Variables

7.1 Introduction

This chapter describes results about the distribution of the random variable

$$\zeta^{\alpha}(Y, X) = \frac{X - Y}{(X + Y)^{\alpha}},$$

where we take the random variables X, Y to be i.i.d. $Bin(n, p)$. The function ζ^{α} is discussed in more detail in Chapter 6. We modify the notation above slightly to include the binomial size parameter. Let $\zeta_m^{\alpha}(Y, X) = \frac{X-Y}{(X+Y)^{\alpha}}$ where $X, Y \sim Bin(m, p)$.

Recall from Example 6.12 that due to the Fisz theorem, the asymptotic distribution of $\zeta_m^{\alpha}(Y, X)$ is

$$N\left(0, \frac{(1-p)^{1/2}}{(2m)^{\alpha-1/2}} p^{1/2-\alpha}\right).$$

When considering the Fisz transform of the above random variables, for the lower m values, the asymptotics do not necessarily hold; in these cases, it would be useful to know the actual distribution of the coefficients.

We now give a definition which we use in the distributional results for $\zeta_m^\alpha(Y,X)$.

Definition 7.1. *The generalized hypergeometric function with numerator parameters $a_1,\ldots,a_p$ and denominator parameters $b_1,\ldots,b_q$ is defined by*

$$ {}_rF_s\left(a_1,\ldots,a_r;b_1,\ldots,b_s;\theta\right)=\sum_{k=0}^{\infty}\frac{(a_1)_k(a_2)_k\cdots(a_r)_k}{(b_1)_k(b_2)_k\cdots(b_s)_k}\frac{\theta^k}{k!}, \tag{7.1}$$

where $(a)_k$ is the Pochhammer symbol

$$(a)_k=a(a+1)\cdots(a+k-1)\qquad \text{for } k\geq 0.$$

Note that the generalized hypergeometric *functions* should not be confused with the well-known hypergeometric *distribution*. I do not discuss them here, but for a more detailed description of (generalized) hypergeometric functions, see for example, [68],[44], and [71].

7.2 The distribution of $\zeta_m^1(Y,X)$

For $\alpha=1$, the coefficients take the form of a straight ratio of Binomial random variables $\frac{X-Y}{X+Y}$. With a little work, it can be shown that the distribution has a nice form.

We denote by S_m^1 the set of unique values of ζ_m^1. Through inspection of the

		X								
		0	1	2	3	4	5	6	7	8
	0	0	1	1	1	1	1	1	1	1
	1	-1	0	$\frac{1}{3}$	$\frac{1}{2}$	$\frac{3}{5}$	$\frac{2}{3}$	$\frac{5}{7}$	$\frac{3}{4}$	$\frac{7}{9}$
	2	-1	$-\frac{1}{3}$	0	$\frac{1}{5}$	$\frac{1}{3}$	$\frac{3}{7}$	$\frac{1}{2}$	$\frac{5}{9}$	$\frac{3}{5}$
	3	-1	$-\frac{1}{2}$	$-\frac{1}{5}$	0	$\frac{1}{7}$	$\frac{1}{4}$	$\frac{1}{3}$	$\frac{2}{5}$	$\frac{5}{11}$
Y	4	-1	$-\frac{3}{5}$	$-\frac{1}{3}$	$-\frac{1}{7}$	0	$\frac{1}{9}$	$\frac{1}{5}$	$\frac{3}{11}$	$\frac{1}{3}$
	5	-1	$-\frac{2}{3}$	$-\frac{3}{7}$	$-\frac{1}{4}$	$-\frac{1}{9}$	0	$\frac{1}{11}$	$\frac{1}{6}$	$\frac{3}{13}$
	6	-1	$-\frac{5}{7}$	$-\frac{1}{2}$	$-\frac{1}{3}$	$-\frac{1}{5}$	$-\frac{1}{11}$	0	$\frac{1}{13}$	$\frac{1}{7}$
	7	-1	$-\frac{3}{4}$	$-\frac{5}{9}$	$-\frac{2}{5}$	$-\frac{3}{11}$	$-\frac{1}{6}$	$-\frac{1}{13}$	0	$\frac{1}{15}$
	8	-1	$-\frac{7}{9}$	$-\frac{3}{5}$	$-\frac{5}{11}$	$-\frac{1}{3}$	$-\frac{3}{13}$	$-\frac{1}{7}$	$-\frac{1}{15}$	0

Table 7.1: Possible values of the random variable $\zeta^1(Y,X)$ for $m=8$. The table gives the values of $\zeta_8^1(Y,X)$ produced when X takes the integers on the column headings and Y takes the integers on the row headings.

ratios for different values of $X, Y \in \{1,\ldots,m\}$, we see that

$$S_m^1 = \pm\left\{ \frac{x-y}{x+y} \;\middle|\; x > y \in \{1,\ldots,m\} \text{ and are coprime} \right\} \cup \{0, \pm 1\}.$$

It is interesting to notice that the size of S_m^1 is $2p_m - 1$, where p_m is the mth prime number.

For example, the set of unique values attained by the random variable ζ_8^1 is $S_8^1 = \pm\{\frac{1}{15}, \frac{1}{13}, \frac{1}{11}, \frac{1}{9}, \frac{1}{7}, \frac{1}{6}, \frac{1}{5}, \frac{3}{13}, \frac{1}{4}, \frac{3}{11}, \frac{1}{3}, \frac{2}{5}, \frac{3}{7}, \frac{5}{11}, \frac{1}{2}, \frac{5}{9}, \frac{3}{5}, \frac{2}{3}, \frac{5}{7}, \frac{3}{4}, \frac{7}{9}, 1\} \cup \{0\}$. Table 7.1 shows the values of X and Y which produce these elements of S_8^1. The number of elements in this set is 45.

For simplicity, we introduce the further notation $S_m^1 = \pm S_m^{1*} \cup \{0, \pm 1\}$.

Let us now consider a value of $\zeta_m^1 = z \in S_m^{1*} \cup \{1\}$. We note that if $\frac{x_1-y_1}{x_1+y_1} = \frac{x_2-y_2}{x_2+y_2}$, then this implies that $x_2 = kx_1$ and $y_2 = ky_1$. The random variable ζ_m^1 thus takes the value z with probability

$$\mathbb{P}(\zeta_m^1 = z) = \sum_{k=1}^{\lfloor m/x \rfloor} \mathbb{P}((X,Y) = (kx, ky)),$$

where (x,y) is the coprime pair corresponding to z. Then, due to the independence of X and Y, we have

$$\begin{aligned}
\mathbb{P}(\zeta_m^1 = z) &= \sum_{k=1}^{\lfloor m/x \rfloor} \mathbb{P}(X = kx)\,\mathbb{P}(Y = ky) \\
&= q^{2m} \sum_{k=1}^{m} \binom{m}{kx}\binom{m}{ky}\left(\frac{p}{q}\right)^{k(x+y)} \\
&= q^{2m} \sum_{k=1}^{m} \binom{m}{kx}\binom{m}{ky}\beta^{k(x+y)} \quad \text{where } \beta = \left(\tfrac{p}{q}\right). \qquad (7.2)
\end{aligned}$$

Here, $q = 1 - p$, where p is the binomial success probability. In the penultimate equation, we use the fact that $\binom{m}{kx} := 0$ for $kx > m$ (where $k \geq \lfloor m/x \rfloor$).

Similarly, for $\zeta_m^1 = 0 \in S_m^1$, since the ratio is zero when X and Y take equal values, we get

$$\mathbb{P}(\zeta_m^1 = 0) = \sum_{k=0}^{m} \mathbb{P}(X = k)^2 = q^{2m} \sum_{k=0}^{m} \binom{m}{k}^2 \beta^{2k}. \qquad (7.3)$$

Equations (7.2) and (7.3) are particular cases of an expression for the probability mass function of ζ_m^α for general α.

The generating pair in S_m^1 corresponding to $z = 1$ is $(x,y) = (1,0)$. Appli-

cation of formula (7.2) gives

$$\mathbb{P}(\zeta_m^1 = 1) = q^{2m} \sum_{k=1}^{m} \binom{m}{k} \binom{m}{0} \left(\frac{p}{q}\right)^k = q^m \sum_{k=1}^{m} q^m \binom{m}{k} \left(\frac{p}{q}\right)^k = q^m(1 - q^m),$$

noting that the summand is just the p.m.f. of a binomial random variable.

We now state the distributional result for ζ_m^1.

Proposition 7.2. *Let ζ_m^1, S_m^1 and β be defined as above. For any $z \in S_m^1$,*

$$\mathbb{P}(\zeta_m^1 = z) = \begin{cases} q^m(1 - q^m) & \text{if } z = \pm 1 \\ q^{2m} \left({}_{x+y}F_{x+y-1}([\boldsymbol{\gamma}, \boldsymbol{\delta}]; [\boldsymbol{\eta}, \boldsymbol{\rho}, 1]; (-\beta)^{x+y}) - 1\right) & \text{if } z = \pm\frac{x+y}{x-y} \\ & \textit{(with (x,y) coprime)} \\ q^{2m}\, {}_2F_1([-m, -m]; 1; \beta^2) & \text{if } z = 0 \end{cases}$$

where ${}_rF_s$ denotes the generalized hypergeometric function, and $\boldsymbol{\gamma}$, $\boldsymbol{\delta}$, $\boldsymbol{\eta}$ and $\boldsymbol{\rho}$ are the sequences

$$\boldsymbol{\gamma} = \left(\tfrac{-m+i-1}{x}\right)_{i=1}^{x}, \quad \boldsymbol{\delta} = \left(\tfrac{-m+j-1}{y}\right)_{j=1}^{y}, \quad \boldsymbol{\eta} = \left(\tfrac{t}{x}\right)_{t=1}^{x-1}, \quad \textit{and} \quad \boldsymbol{\rho} = \left(\tfrac{l}{y}\right)_{l=1}^{y-1}.$$

Proof. The $z = \pm 1$ case is shown in the discussion above. It suffices to show the result for $z \in S_m^{1*} \cup \{0\}$, since the negative values of ζ_m^1 have the same probability as their positive counterparts.

Firstly, note that the generalized hypergeometric functions in the proposition converge: Luke [71] states that for ${}_rF_s(\mathbf{a}; \mathbf{b}; \theta)$ with $r = s+1$, the functions converge for $|\theta| < 1$ and converge absolutely for $|\theta| = 1$ if

$$\sum_{h=1}^{r} a_h - \sum_{h=1}^{s} b_h < 0.$$

In our case, this quantity is $-1-2m<0$.

Using Definition 7.1, for (x,y) with $x>y$, the generalized hypergeometric function in Proposition 7.2 is

$$\sum_{k=0}^{\infty}\frac{(\gamma_1)_k(\gamma_2)_k\cdots(\gamma_x)_k(\delta_1)_k(\delta_2)_k\cdots(\delta_y)_k}{(\eta_1)_k(\eta_2)_k\cdots(\eta_{x-1})_k(\rho_1)_k(\rho_2)_k\cdots(\rho_{y-1})_k(1)_k}\frac{[(-\beta)^{x+y}]^k}{k!}.$$

From the definition of the Pochhammer symbol and by substituting the sequences elements of $\boldsymbol{\gamma}$, $\boldsymbol{\delta}$, $\boldsymbol{\eta}$ and $\boldsymbol{\rho}$, this is equal to

$$\begin{aligned}
&\sum_{k=0}^{\infty}\frac{\prod_{i=1}^{x}\left(\prod_{n=0}^{k-1}(\gamma_i+n)\right)\prod_{j=1}^{y}\left(\prod_{n=0}^{k-1}(\delta_j+n)\right)}{\prod_{t=1}^{x-1}\left(\prod_{n=0}^{k-1}(\eta_t+n)\right)\prod_{l=1}^{y-1}\left(\prod_{n=0}^{k-1}(\rho_l+n)\right)}\frac{[(-\beta)^{x+y}]^k}{k!^2}\\
=&\sum_{k=0}^{\infty}\left[\prod_{t=1}^{x-1}\prod_{n=0}^{k-1}\frac{\gamma_t+n}{\eta_t+n}\right]\left[\prod_{l=1}^{y-1}\prod_{n=0}^{k-1}\frac{\delta_l+n}{\rho_l+n}\right]\left[\prod_{n=0}^{k-1}(\gamma_x+n)\right]\left[\prod_{n=0}^{k-1}(\delta_y+n)\right]\frac{[(-\beta)^{x+y}]^k}{k!^2}\\
=&\sum_{k=0}^{\infty}\left[\prod_{t=1}^{x-1}\prod_{n=0}^{k-1}\frac{m-nx-t+1}{t+nx}\right]\left[\prod_{l=1}^{y-1}\prod_{n=0}^{k-1}\frac{m-ny-l+1}{l+ny}\right]v\,w\,\frac{[\beta^{x+y}]^k}{k!^2},
\end{aligned}$$

where $v=\prod_{n=0}^{k-1}\frac{m-(n+1)x+1}{x}$ and $w=\prod_{n=0}^{k-1}\frac{m-(n+1)y+1}{y}$.

We can now reduce the products over t and l:

$$\begin{aligned}
&\prod_{t=1}^{x-1}\prod_{n=0}^{k-1}\frac{m-nx-t+1}{t+nx}\prod_{l=1}^{y-1}\prod_{n=0}^{k-1}\frac{m-ny-l+1}{l+ny}\\
&=\prod_{n=0}^{k-1}\frac{(m-nx)!(nx)!}{(m-(n+1)x+1)!((n+1)x-1)!}\prod_{n=0}^{k-1}\frac{(m-ny)!(ny)!}{(m-(n+1)y+1)!((n+1)y-1)!}.
\end{aligned}$$

Then by cancelling numerator and denominator factors in successive product terms over n, we obtain

$$\sum_{k=0}^{\infty}\left[\prod_{t=1}^{x-1}\prod_{n=0}^{k-1}\frac{m-nx-t+1}{t+nx}\right]\left[\prod_{l=1}^{y-1}\prod_{n=0}^{k-1}\frac{m-ny-l+1}{l+ny}\right]v\,w\,\frac{[\beta^{x+y}]^k}{k!^2}$$
$$=\sum_{k=0}^{\infty}\binom{m}{kx-1}\left[\prod_{n=1}^{k-1}\frac{nx}{n-mx+1}\right]\binom{m}{ky-1}\left[\prod_{n=1}^{k-1}\frac{ny}{n-my+1}\right]v\,w\,\frac{[\beta^{x+y}]^k}{k!^2}.$$

Reintroducing the expressions for v and w and again cancelling denominator and numerator terms, this expression is equal to

$$\sum_{k=0}^{\infty}\binom{m}{kx-1}\binom{m}{ky-1}(k+1)!^2\frac{m-kx+1}{x}\frac{m-ky+1}{y}\frac{[\beta^{x+y}]^k}{k!^2}.$$

Simplifying this further, we finally obtain

$${}_{x+y}F_{x+y-1}([\boldsymbol{\gamma},\boldsymbol{\delta}];[\boldsymbol{\eta},\boldsymbol{\rho},1];(-\beta)^{x+y})=\sum_{k=0}^{\infty}\binom{m}{kx}\binom{m}{ky}\beta^{k(x+y)}.$$

The upper limit of this sum can be taken to be $k=m$, since all values of the sum are zero for $k>m$. Also, noting that the $k=0$ term in the sum is 1, and multiplying the sum by a factor of q^{2m},

$$\begin{aligned}q^{2m}\left({}_{x+y}F_{x+y-1}([\boldsymbol{\gamma},\boldsymbol{\delta}];[\boldsymbol{\eta},\boldsymbol{\rho},1];(-\beta)^{x+y})-1\right) &= q^{2m}\sum_{k=1}^{m}\binom{m}{kx}\binom{m}{ky}\beta^{k(x+y)}\\ &= \mathbb{P}(\zeta_m^1=z),\end{aligned}$$

from equation 7.2. Thus the proposition is proved for $z=\frac{x-y}{x+y}\in S_m^{1*}$, and therefore for any $z\in\pm S_m^{1*}$, by symmetry of x and y.

It remains to prove the proposition for $\zeta_m^1=0$. Taking the last part of the equation in Proposition 7.2,

$$\begin{aligned} q^{2m}{}_2F_1([-m,-m];1;\beta^2) &:= \sum_{k=0}^{\infty} \frac{(-m)_k(-m)_k}{(1)_k}\frac{\beta^{2k}}{k!} \\ &= q^{2m}\sum_{k=0}^{\infty} \frac{[(-m)(1-m)(2-m)\cdots(k-1-m)]^2}{k!}\frac{\beta^{2k}}{k!} \\ &= q^{2m}\sum_{k=0}^{\infty} \frac{[(m)(m-1)(m-2)\cdots(m-k+1)]^2}{k!}\frac{\beta^{2k}}{k!}. \end{aligned}$$

This equation then simplifies to

$$\begin{aligned} & q^{2m}\sum_{k=0}^{m}\binom{m}{k}^2\left(\frac{q}{p}\right)^{2k} \quad \text{since } (-m)_k = 0 \text{ for } k > m \\ = & \sum_{k=0}^{m}\mathbb{P}(W=k)^2 \quad \text{for } W \sim Bin(m,p) \\ = & \mathbb{P}(\zeta_m^1 = 0) \quad \text{from equation (7.3).} \end{aligned}$$

□

Note the similarities between the form of the generalized hypergeometric function for general z and for $z = 0$ in Proposition 7.2. We noticed before that every unique value in $S_m^1 \setminus \{0\}$ had probability contributions from positive integer multiples of a coprime pair (x,y), hence we sum over $k \geq 1$ in (7.2). However, for $z = 0$, we need contributions from all values of (X,Y) where X and Y are equal, which includes zero itself. This is the reason why the sum in (7.3) is over $k \geq 0$. So if we were to view the $z = 0$ case as being generated by the (x,y) pair (1,1) for $k \geq 0$, using the hypergeometric form for general z (ignoring the coprime assumption), we would have

$$\mathbb{P}(\zeta_m^1 = 0) = q^{2m}({}_2F_1([-m,-m];1;\beta^2) - 1) = q^{2m}{}_2F_1([-m,-m];1;\beta^2) - q^{2m},$$

which can be thought of as the contribution from all the positive integer

multiples of the pair (1,1). Hence for the actual probability $\mathbb{P}(\zeta_m^1 = 0)$, we just need to add on the zero multiple contribution, which is q^{2m}. This hence justifies the generalized hypergeometric function equation in the proposition for $z = 0$. Also note that this functional form of $\mathbb{P}(\zeta_m^1 = 0)$ will be the same for *any* value of the exponent α in ζ_m^α, since the random variable will be zero when $X = Y$, irrespective of the denominator.

7.3 The distribution of $\zeta_m^0(Y, X)$

Even though the original Fisz theorem in Section 6.2 is only valid for positive exponents of the denominator, in this section we explore the distribution of ζ_m^0 to demonstrate that the generalized hypergeometric form can be obtained for other values of α. For $\alpha = 0$, $\zeta_m^0 = X - Y$ is the difference of two Binomial distributions. This random variable takes values in $S_m^0 = \{-m, \ldots, 0, \ldots, m\}$. Since the distribution is symmetric, let us consider $l \in \{0, \ldots, m\}$. It can be easily shown that

$$\mathbb{P}(\zeta_m^0 = l) = q^{2m} \sum_{k=0}^{m-l} \binom{m}{k} \binom{m}{k+l} \left(\frac{p}{q}\right)^{2k+l}. \tag{7.4}$$

A similar generalized hypergeometric function form can be proved for ζ_m^0.

Proposition 7.3. *Let ζ_m^0, S_m^0 and β be defined as above. For any $l \in S_m^0$,*

$$\mathbb{P}(\zeta_m^0 = l) = \begin{cases} q^{2m}{}_2F_1([-m,-m];1;\beta^2) & \text{if } l = 0 \\ \beta^l q^{2m}\left(\binom{m}{l+1}\beta^2{}_3F_2([1,1-m,1+l-m];[2,l+2];\beta^2) + \binom{m}{l}\right) & \\ & \text{if } l \in \{1,\ldots,m-2\} \\ mq^{2m}\beta^{m-1}(1+\beta^2) & \text{if } l = m-1 \\ q^m(1-q^m) & \text{if } l = m \end{cases}$$

where ${}_pF_q$ denotes the generalized hypergeometric function.

Proof. From equation (7.4), the proposition is proved for $\zeta_m^0 = m, m-1$. When $l = 0$, equation (7.4) takes the same form as (7.3). Hence we can follow the $\zeta_m^1 = 0$ part of the proof of Proposition 7.2 for the $\zeta_m^0 = 0$ case of this proposition. It then remains to prove the proposition for $\zeta_m^0 = l \in \{1, \ldots, m-2\}$: again, the negative values of ζ_m^0 have the same probability as when the variable takes positive values.

Note that the hypergeometric function in Proposition 7.3 converges in the same way as the functions in Proposition 7.2, since the quantity $\sum_{h=1}^{r} a_h - \sum_{h=1}^{s} b_h = -1 - 2m < 0$.

We start with simplifying the generalized hypergoemetric function term ${}_3F_2([1, 1-m, 1+l-m]; [2, l+2]; \beta^2)$:

$$\begin{aligned} & \sum_{k=0}^{\infty} \frac{(1)_k (1-m)_k (1+l-m)_k}{(2)_k (l+2)_k} \frac{\beta^{2k}}{k!} \\ = & \sum_{k=0}^{m-(l+1)} \frac{k!(-1)^k (m-1)!(-1)^k (m-l-1)!(l+1)!}{(k+1)!(m-k-1)!(m-k-l-1)!(k+l+1)!} \frac{\beta^{2k}}{k!} \end{aligned}$$

from the definition of the Pochhammer symbol. Then we have

$$\begin{aligned} & m\beta^2 \binom{m}{l+1} {}_3F_2([1, 1-m, 1+l-m]; [2, l+2]; \beta^2) + \binom{m}{l} \\ = & \beta^2 \sum_{k=0}^{m-(l+1)} \frac{m!^2}{(k+1)!(m-k-1)!(m-k-l-1)!(k+l+1)!} \beta^{2k} + \binom{m}{l} \\ = & \beta^2 \sum_{k=0}^{m-(l+1)} \binom{m}{k+1} \binom{m}{k+l+1} \beta^{2k} + \binom{m}{l} \\ = & \sum_{i=0}^{m-l} \binom{m}{i} \binom{m}{i+l} \beta^{2i}, \end{aligned}$$

from which the result follows, by considering equation (7.4).

□

The functional form of the probability mass functions of the random variable ζ_m^α for $\alpha = 0, 1$ is an illustration of how the probabilities of the ratios could be computed more efficiently using successful algorithms for generalized hypergeometric functions. One could no doubt find other similar expressions for random variables with different values of α.

Chapter 8

Conclusions and Further work

This chapter gives a summary of the work in this thesis. Each of the main ideas in the Chapters 3–7 is taken in turn, and both the advantages and disadvantages of the new methods introduced are outlined. We give some directions in which the work in the thesis could take for further research.

8.1 Adaptive lifting

The discrete wavelet transform (DWT), described in Chapter 2, can handle regularly-spaced observations. The dyadic nature of classical wavelet transforms means that the number of observations in a dataset is limited to a power of two. Any departure from these conditions involves major modifications of the wavelet transforms or preprocessing of the data, which can lead to unwanted artifacts.

We have introduced a lifting scheme based on "one coefficient at a time lifting" work by Jansen *et al.* [59, 60] for use in the nonparametric regression situation. The algorithm is suitable for data of any length, including irregularly-sampled observations, addressing limitations of some classical wavelet transforms.

We introduced two sources of adaptiveness into the algorithm: the ability to

choose minimum detail coefficients over linear regression orders; and the added freedom to consider different neighbourhood configurations as well. Since the procedure is fully adaptive, it is able to reflect the local properties of the data to recreate an underlying signal without any choices to be made by the user. The algorithm is also of low computational order.

We have also proposed a modified procedure for use on data with multiple observations at each x-point. This aspect is often not accounted for in nonparametric regression techniques.

Our simulations compared the adaptive wavelet transform to other (wavelet and non-wavelet) regression techniques, namely the Kovac-Silverman method [67], *Locfit* [70, 69], Comte-Rozenholc [30] and the Splus `smooth.spline` method. Simulations show that the adaptive lifting transforms perform well in comparison to the other estimators, both in terms of sparsity and denoising. This is especially true for the fully adaptive transform, AN1, when applied to test signals with discontinuous behaviour. Applying the procedure to real datasets, has shown good results.

A disadvantage of our algorithm is that it can exhibit some stability problems for larger regression neighbourhoods. This could be reduced by using ideas proposed by [91, 98].

8.2 Primary resolution levels in adaptive lifting

Previous work on classical wavelet transforms has shown that the choice of primary resolution level when used in conjunction with thresholding methods, such as *SureShrink* [40], affects how well the transforms are able to smooth noisy data. Hence an obvious question to ask is whether the same notion of primary resolution is important for the adaptive lifting transform proposed in Chapter 3.

Motivated by this, we investigated the effect of the primary resolution level in the adaptive lifting scheme proposed in Chapter 3. We concluded that it does affect the denoising performance of our algorithm; the simulations performed show that there is quite a wide variation in best stopping times for all the denoising test signals, grid irregularities and noise ratios.

A method of automatically predicting the resolution level was proposed, based on attempting to predict a signal's resolution level ISE curve. However, this prediction algorithm was unsuccessful, producing erratic results.

Based on the simulation results into optimal stopping times, recommendations for stopping times can be made according to which type of signal were to be denoised; for an unknown signal structure, it is recommended to keep the denoising resolution level low, corresponding to performing many lifting steps.

Further work in this area could include trying to have an efficient lifting algorithm which has the feature of an in-built stopping time decision process. This could be set up using sequential hypothesis testing to investigate the "significance" of detail coefficients. A possibility is to keep track of the detail coefficients as they are produced, assessing each to be "significant" according to some criterion (for example, if they exceed a certain threshold). As the algorithm progresses, we would get an idea of which coefficients have been judged as signal or noise. If a lot of details have been judged significant, then we could use this information to decide to stop the transform, especially if the last few details have been judged as noise.

As it stands, our transform imposes an order on the data by lifting the coefficient with the smallest corresponding scaling function integral. An alternative to the idea above is to consider "signal clustering". When a point comes to be removed, the issue of whether it will have the same significance (meaning whether it was classed as noise or signal content) as its lifted neighbours is an interesting question to consider - the transform could somehow make use of

this information to choose which coefficients to lift/whether to stop the transform according to whether its location is near lifted points which have already been assessed to be signal or noise.

8.3 Binomial proportion estimation

There are many techniques in the literature which tackle the classical nonparametric regression situation of observations assumed to be a signal with added Gaussian noise. However, nonparametric methods specifically for binomial data problems, for example, binomial proportion estimation, are not so widespread.

Motivated by previous Gaussianizing methods in the literature using the Fisz transform [47], we looked at using the same technique for binomial data. However, the equivalent Fisz transform for this type of data does not possess the same desired variance stabilizing properties, though it does lead to interesting distributional results.

We proposed a new Fisz-like transform for binomial random variables, and proved that it does have a certain asymptotic normal distribution.

Simulations provided evidence of the transform's Gaussianization and variance-stabilizing properties, with our transform converging to normal faster than Anscombe's inverse sine Gaussianizing transform [6]. It also stabilizes as well as Anscombe for high binomial sizes, and better than Anscombe for low and high binomial proportions.

Based on this new transform, we have proposed an alternative to the Haar-Fisz transform, replacing the square root division by our Gaussianizing transform. Investigating the properties of this transform, we found that it too is able to bring data closer to normality as well as Anscombe's transformation for large binomial sizes, and better than Anscombe for small n and extreme values of the binomial proportion.

We then presented a method for binomial proportion estimation, using the alternative transform to the Haar-Fisz transform [51] as a preprocessing algorithm. This technique was then applied to an example to demonstrate its capability.

8.4 Fisz-transformed random variables

The initial work in Chapter 6 on the Fisz transform lead to interesting results when applying this transform to binomial random variables.

In particular, the exact distribution of the variable $\zeta_m^1(X_1, X_2)$ was examined for different binomial sizes m, so that it could be compared to the proposed transform of ζ_B in Chapter 6. A functional form for this exact distribution was discovered, based on generalized hypergeometric functions. This form for Fisz-transformed random variables could be used to bring improved computation times for large binomial sizes.

This functional form was also extended to give a similar result for the random variable $\zeta_m^0(X_1, X_2)$, where X_1 and X_2 are binomial random variables.

Appendix A

Properties of ζ_B and $\mathcal{F}_B$: simulation plots

Introduction

This appendix gives extra graphical representation of the findings in Chapter 6; the plots are related to the investigation into the Gaussianization and variance-stabilizing properties of ζ_B corresponding to Section 6.5 in the main text.

- Contour plot for convergence of the sample mean to the asymptotic mean (Figure A.1)
- Perspective plot for the (normalized) sample variance (Figure A.2)
- Comparison between ζ_B and Anscombe's transformation of sample variance for equal binomial proportions (Figure A.3)
- Perspective plot for difference in Kolmogorov-Smirnov statistics between Anscombe and ζ_B (Figure A.4).

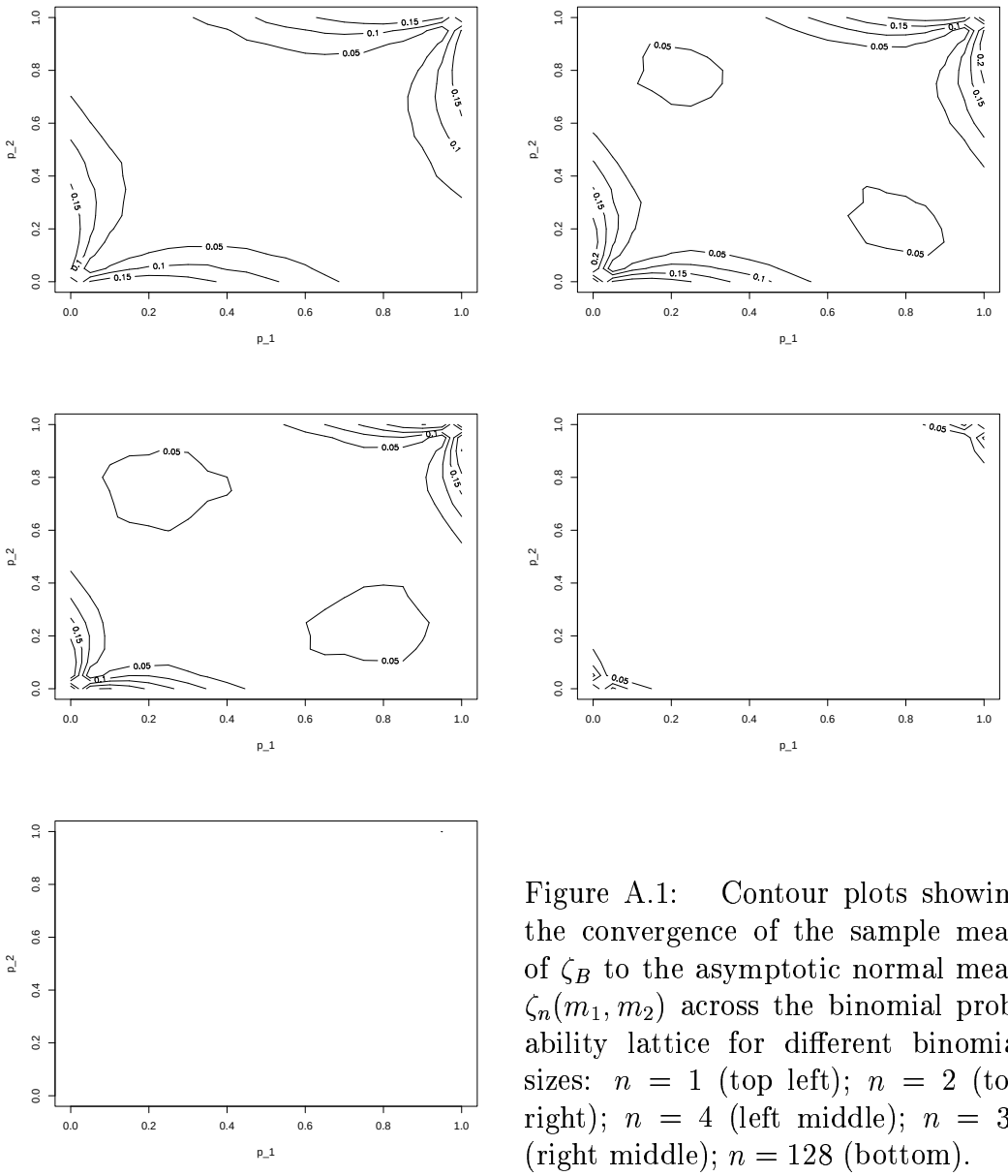

Figure A.1: Contour plots showing the convergence of the sample mean of ζ_B to the asymptotic normal mean $\zeta_n(m_1, m_2)$ across the binomial probability lattice for different binomial sizes: $n = 1$ (top left); $n = 2$ (top right); $n = 4$ (left middle); $n = 32$ (right middle); $n = 128$ (bottom).

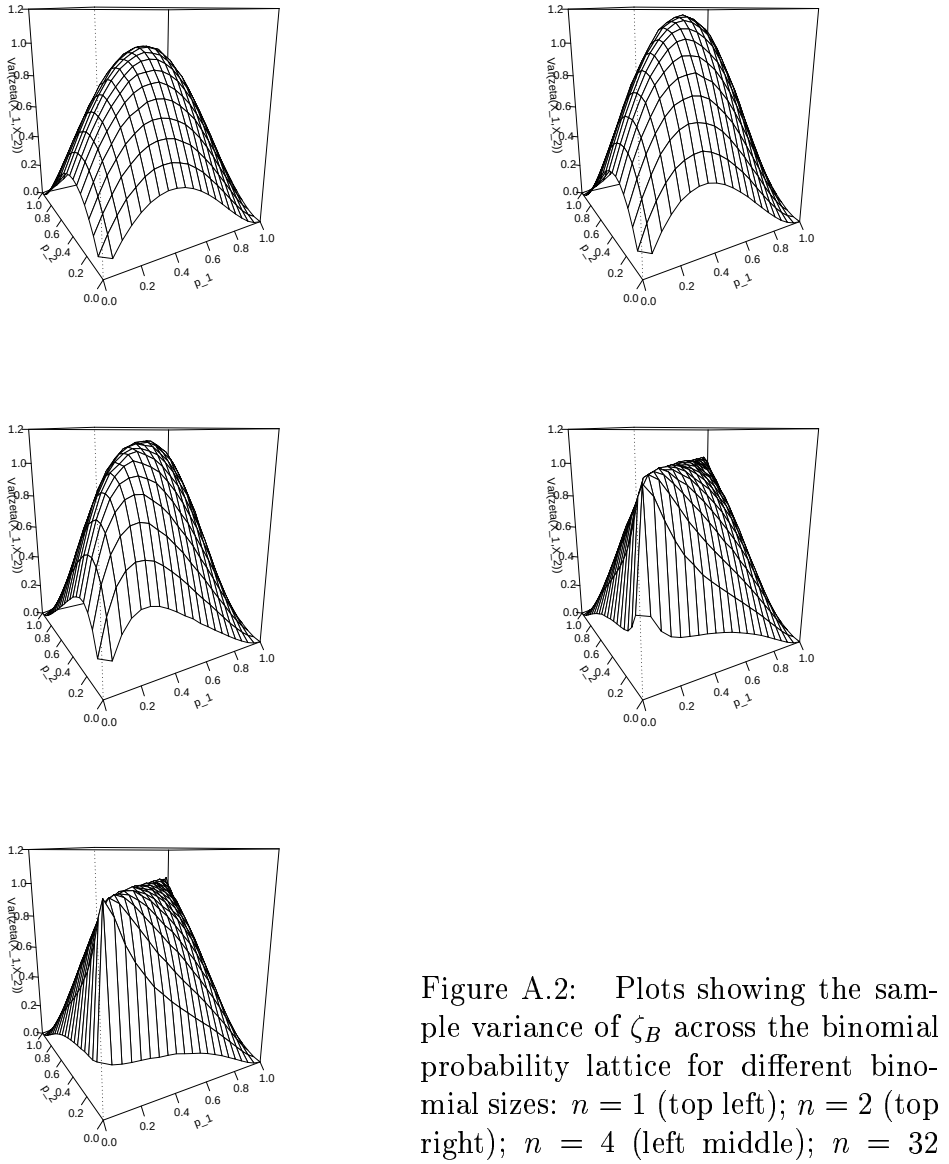

Figure A.2: Plots showing the sample variance of ζ_B across the binomial probability lattice for different binomial sizes: $n = 1$ (top left); $n = 2$ (top right); $n = 4$ (left middle); $n = 32$ (right middle); $n = 128$ (bottom).

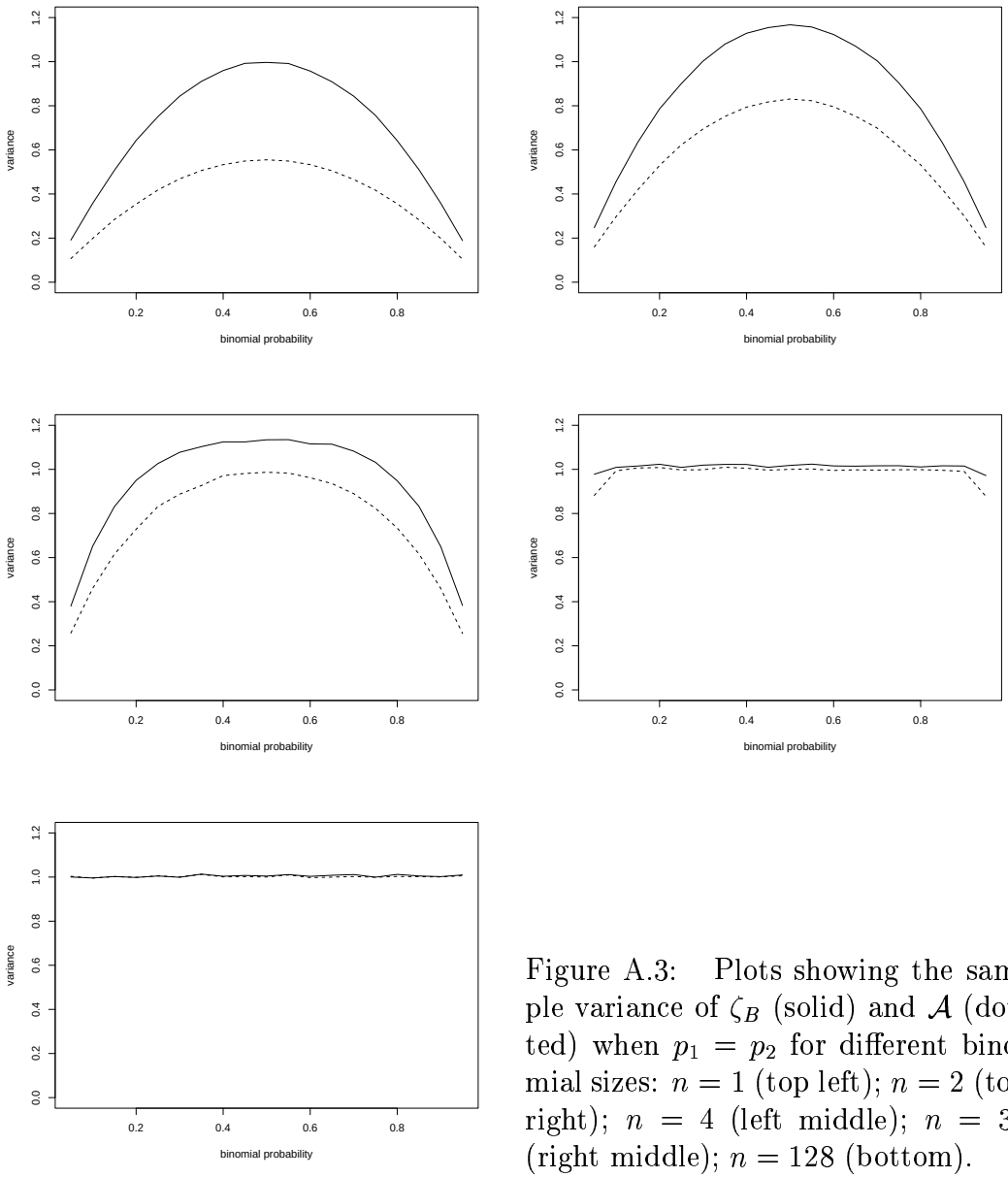

Figure A.3: Plots showing the sample variance of ζ_B (solid) and $\mathcal{A}$ (dotted) when $p_1 = p_2$ for different binomial sizes: $n = 1$ (top left); $n = 2$ (top right); $n = 4$ (left middle); $n = 32$ (right middle); $n = 128$ (bottom).

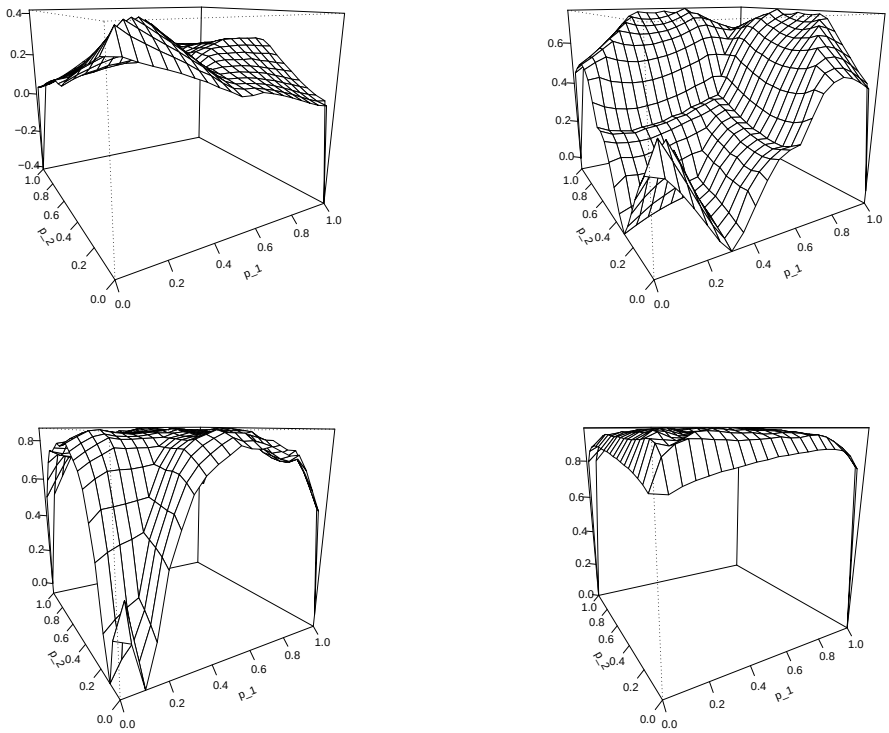

Figure A.4: Plots showing the difference between Kolmogorov-Smirnov statistics computed on Anscombe samples with binomial probability $\max(p_1, p_2)$ for different binomial sizes: $n = 1$ (top left); $n = 2$ (top right); $n = 4$ (left middle); $n = 32$ (right middle); $n = 128$ (bottom). Positive difference indicates that ζ_B is closer to Gaussian.

Bibliography

[1] F. Abramovich, T.C. Bailey, and T. Sapatinas. Wavelet analysis and its statistical applications. *J. Roy. Statist. Soc. D*, 49:1–29, 2000.

[2] F. Abramovich and Y. Benjamini. Adaptive thresholding of wavelet coefficients. *Comp. Stat. and Data Anal.*, 22:351–361, 1996.

[3] F. Abramovich, T. Sapatinas, and B.W. Silverman. Wavelet thresholding via a bayesian approach. *J. Roy. Stat. Soc. B*, 60(4):725–749, 1998.

[4] A. Agresti and B.A. Coull. Approximate is better than "exact" for interval estimation if binomial proportions. *Amer. Stat.*, 52:119–126, 1998.

[5] N.S. Altman and B. MacGibbon. Consistent bandwidth selection for kernel binary regression. *J. Stat. Planning and Inf.*, 70(1):121–137, July 1998.

[6] F.J. Anscombe. The transformation of poisson, binomial and negative binomial data. *Biometrika*, 35(3/4):246–254, 1948.

[7] A. Antoniadis and J. Fan. Regularization of wavelet approximations. *J. Am. Statist. Ass.*, 96:939–967, 2001.

[8] A. Antoniadis and F. LeBlanc. Nonparametric wavelet regression for binary response. *Statistics*, 34:183–213, 2000.

[9] A. Antoniadis and T. Sapatinas. Wavelet shrinkage for natural exponential families with quadratic variance functions. *Biometrika*, 88(3):805–820, 2001.

[10] S. Barber and G.P. Nason. Simulations comparing thresholding methods using real and complex wavelets. Technical Report 03:07, Statistics Group, Department of Mathematics, University of Bristol, UK, 2003.

[11] S. Barber and G.P. Nason. Real nonparametric regression using complex wavelets. *J. Roy. Statist. Soc. B*, 66:927–939, 2004.

[12] Y. Benjamini and Y. Hochberg. Controlling tthe false discovery rate: a practical approach to multiple testing. *J. Roy. Stat. Soc. B*, 57:289–300, 1995.

[13] G. Bernardi. Isochores and the evolutionary genomics of vertebrates. *Gene*, 241:3–17, 2000.

[14] N.V. Boulgouris, D. Tzovaras, and M.G. Strintzis. Lossless image compression based on optimal prediction, adaptive lifting and conditional arithmetic coding. *IEEE Trans. Image Proc.*, 10:1–14, 2001.

[15] A. W. Bowman and A. Azzalini. *Applied Smoothing Techniques for Data Analysis*. Oxford University Press, 1997.

[16] L.D. Brown, T.T. Cai, and A. DasGupta. Interval estimation for a binomial proportion. *Statistical Science*, 16(2):101–133, May 2001.

[17] L.D. Brown, T.T. Cai, and A. DasGupta. Confidence intervals for a binomial proportion and asymptotic expansions. *Ann. Stat.*, 30(1):160–201, 2002.

[18] T.A. Brown. *Genomes*. Oxford: BIOS Scientific, 2nd edition, 2002.

[19] T. T. Cai and L. D. Brown. Wavelet shrinkage for nonequispaced samples. *Ann. Stat.*, 26:1783–1799, 1998.

[20] T. T. Cai and L. D. Brown. Wavelet estimation for samples with random uniform design. *Stat. Prob. Lett.*, 42:313–321, 1999.

[21] H. Chipman, E. Kolaczyk, and R. McCulloch. Adaptive bayesian wavelet shrinkage. *J. Am. Stat. Ass.*, 92:1413–1421, 1997.

[22] R.L. Claypoole, R.G. Baraniuk, and R.D. Nowak. Adaptive Wavelet Transforms via Lifting. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1513–1516, Seattle, WA, May 1998.

[23] R.L. Claypoole, G.M. Davis, W. Sweldens, and R.G. Baraniuk. Nonlinear wavelet transforms for image coding via lifting. *IEEE Trans. Im. Proc.*, 12:1449–1459, 2003.

[24] M. Clyde and E. George. Flexible empirical bayes estimation for wavelets. *J. Roy. Stat. Soc. B*, 62(4):681–2000, 2000.

[25] M. Clyde, G. Parmigiani, and B. Vidakovic. Multiple shrinkage and subset selection in wavelets. *Biometrika*, 85(2):391–401, 1998.

[26] A. Cohen, I. Daubechies, and J. C. Feauveau. Biorthogonal bases of compactly supported wavelets. *Comm. Pure Appl. Math.*, 45(2):485–560, 1992.

[27] A. Cohen, I. Daubechies, and P. Vial. Wavelets on the interval and fast wavelet transforms. *Appl. Comput. Harmon. Anal.*, 1(1):54–81, 1993.

[28] R. R. Coifman and D.L. Donoho. Translation-invariant de-noising. Technical report, Statistics Department, Stanford University, 1995.

[29] R. R. Coifman and M. Wickerhauser. Entropy-based algorithms for best basis selection. *IEEE Transactions on Information Theory*, 38(2):713–718, 1992.

[30] F. Comte and Y. Rozenholc. A new algorithm for fixed design regression and denoising. *Ann. Inst. Stat. Math.*, 56:449–473, 2004.

[31] G.M. Cooper and R.E. Hausman. *The Cell: a molecular approach.* ASM Press, 3rd edition, 2004.

[32] D.R. Cox and E.J. Snell. *Analysis of Binary Data.* Number 32 in Monographs on Statistics and Applied Probability. London: Chapman and Hall, 2nd edition, 1989.

[33] H. Cramér. *Mathematical methods of statistics.* Princeton University Press, 1946.

[34] J.W. Dale and M. von Schantz. *From Genes to Genomes: concepts and applications of DNA technology.* Wiley, 2002.

[35] I. Daubechies. *Ten Lectures On Wavelets.* Philadelphia:SIAM, 1992.

[36] I. Daubechies and W. Sweldens. Factoring wavelet transforms into lifting steps. *J. Fourier Anal. Appl.*, 4(3):245–267, 1998.

[37] V. Delouille, J. Franke, and R. von Sachs. Nonparametric stochastic regression with design-adapted wavelets. *Sankhya A*, 63(3):328–366, 2001.

[38] V. Delouille, J. Simoens, and R. von Sachs. Smooth design-adapted wavelets for nonparametric stochastic regression. *J. Am. Statist. Soc.*, 99:643–658, 2004.

[39] D. L. Donoho and I. M. Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3):425–455, 1994.

[40] D. L. Donoho and I. M. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *J. Am. Statist. Soc*, 90:1200–1224, 1995.

[41] D. L. Donoho, I. M. Johnstone, G. Kerkyacharian, and D. Picard. Wavelet shrinkage: asymptopia? (with discussion). *J. Roy. Statist. Soc. B*, 57:301–337, 1995.

[42] D.L. Donoho. Nonlinear wavelet methods for recovery of signals, densities and spectra from indirect and noisy data. In I. Daubechies, editor, *Proceedings of Symposia in Applied Mathematics: Different Perspectives on Wavelets*, volume 47, pages 173–205. American Mathematical Society, 1993.

[43] R. Eisenberg. Biological signals that need detection: Currents through single membrane channels. In J. Norman and F. Sheppard, editors, *Proceedings of the 16th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 32a–33a, 1994.

[44] H. Exton. *Multiple Hypergeometric Functions and Applications.* Ellis Horwood, 1976.

[45] J. Fan and I. Gijbels. Data-driven bandwidth selection in local polynomial fitting: variable bandwidth and spatial adaptation. *J. Roy. Stat. Soc. B*, 57:371–394, 1995.

[46] J. Fan, N.E. Heckman, and M.P. Wand. Local polynomial kernel regression for generalized linear models and quasi-likelihood functions. *J. Amer. Stat. Ass.*, 90:141–150, 1995.

[47] M. Fisz. The limiting distribution of a function of two independent random variables and its statistical application. *Colloquium Mathematicum*, 3:138–146, 1955.

[48] G. B. Folland. *Fourier Analysis and its Applications.* Pacific Grove: Brooks/Cole, 1992.

[49] M. F. Freeman and J.W. Tukey. Transformations related to the angular and the square root. *Ann. Math. Stat.*, 21(4):607–611, 1950.

[50] J. H. Friedman. A variable span scatterplot smoother. Technical Report 5, Laboratory for Computational Statistics, Stanford University, Stanford, CA, USA, 1984.

[51] P. Fryźlewicz and G. P. Nason. A Haar-Fisz algorithm for poisson intensity estimation. *J. Comp. Graph. Stat.*, 13:621–638, 2004.

[52] P. Fryźlewicz and G. P. Nason. Smoothing the wavelet periodogram using the Haar-Fisz transform. Technical Report 04:06, Statistics Group, Department of Mathematics, University of Bristol, UK, 2004.

[53] P. Hall and G.P. Nason. On choosing a non-integer resolution level when using wavelet methods. *Stat. Prob. Lett.*, 34(1):5–11, 1997.

[54] P. Hall and P. Patil. On the choice of smoothing parameter, threshold and truncation in nonparametric regression by nonlinear wavelet methods. *J. Roy. Statist. Soc. B*, 58:361–377, 1996.

[55] P. Hall and S. Penev. Cross-validation for choosing resolution level for nonlinear wavelet curve estimators. *Bernoulli*, 7(2):317–341, 2001.

[56] T. Hastie and R. Tibshirani. Generalized additive models. *Stat. Sci.*, 1:297–318, 1990.

[57] M.E.A Hodgson and P.J. Green. Investigating Markov model discrimination for ion channels. Technical Report 99:16, Statistics Group, Department of Mathematics, University of Bristol, UK, 1999.

[58] S. Igari. *Real Analysis with an Introduction to Wavelet theory*, volume 177 of *Translations of Mathematical Monographs*. American Mathematical Society, 1998.

[59] M. Jansen, G.P. Nason, and B.W. Silverman. Scattered data smoothing by empirical bayesian shrinkage of second generation wavelet coefficients. In M. Unser and A. Aldroubi, editors, *Wavelet Applications in Signal and Image Processing IX*, volume 4478, pages 87–97. SPIE, 2001.

[60] M. Jansen, G.P. Nason, and B.W. Silverman. Multidimensional nonparametric regression using lifting. Technical Report 04:17, Statistics Group, Department of Mathematics, University of Bristol, UK, 2004.

[61] I. M. Johnstone and B. W. Silverman. Needles and hay in haystacks: Empirical bayes estimates of possibly sparse sequences. *Ann. Stat.*, 32:1594–1649, 2004.

[62] I.M. Johnstone and B.W. Silverman. Ebayesthresh: R and s-plus software for empirical bayes thresholding. Paper accompanying Software package (available from CRAN archive), 2004.

[63] I.M. Johnstone and B.W. Silverman. Empirical bayes selection of wavelet thresholds. *Ann. Stat.*, 33, 2005. (to appear).

[64] M. I. Knight and G. P. Nason. Improving prediction of hydrophobic segments along a transmembrane protein sequence using adaptive multiscale lifting. Technical Report 04:19, Statistics Group, Department of Mathematics, University of Bristol, UK, 2004. Accepted for publication in *SIAM Multisc. Model. & Sim.*

[65] E.D. Kolaczyk and R.D. Nowak. Multiscale generalised linear models for nonparametric function estimation. *Biometrika*, 92(1):119–133, 2005.

[66] A. Kovac. *Wavelet Thresholding for Unequally Time-Spaced Data.* PhD thesis, Statistics Group, Department of Mathematics, University of Bristol, UK, 1998.

[67] A. Kovac and B.W. Silverman. Extending the scope of wavelet regression methods by coefficient-dependent thresholding. *J. Am. Statist. Ass.*, 95:172–183, 2000.

[68] N.N. Lebedev. *Special Functions and their Applications.* Prentice-Hall, 1965.

[69] C. Loader. Locfit: an introduction. *Stat. Comput. Graph. News.*, 8:11–17, 1997.

[70] C. Loader. *Local Regression and Likelihood*. Springer: New York, 1999.

[71] Y. L. Luke. *The Special Functions and their Approximations*, volume 53 of *Mathematics in Science and Engineering*. Academic Press, 1969.

[72] S. G. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Trans. Pattn. Anal. Mach. Intell.*, 11:674–693, 1989.

[73] P. McCullagh and J.A. Nelder. *Generalized Linear Models*. London: Chapman and Hall, 1989.

[74] P. Müller and B. Vidakovic. *Bayesian Inference in wavelet based models*. Number 141 in Lecture Notes in Statistics. Springer-Verlag: New York, 1999.

[75] G. P. Nason. Wavelet shrinkage using cross-validation. *J. Roy. Statist. Soc. B.*, 58:463–479, 1996.

[76] G. P. Nason and B. W. Silverman. The discrete wavelet transform in S. *J. Comp. Graph. Statist.*, 3:163–191, 1994.

[77] G.P. Nason. Choice of wavelet smoothness, primary resolution and threshold in wavelet shrinkage. *Stat. Comput.*, 12:219–227, 2002.

[78] G.P. Nason and B.W. Silverman. The stationary wavelet transform and some applications. In A. Antoniadis and G. Oppeheim, editors, *Wavelets and Statistics*, volume 103 of *Lecture Notes in Statistics*, pages 281–300. New York: Springer-Verlag, 1995.

[79] M. A. Nunes, M. I. Knight, and G. P. Nason. Adaptive lifting for non-parametric regression. Technical Report 04:20, Statistics Group, De-

partment of Mathematics, University of Bristol, UK, 2004. Accepted for publication in *Statistics and Computing*.

[80] M. A. Nunes, M. I. Knight, and G. P. Nason. Adaptive lifting for non-parametric regression: simulation results. Technical report, Statistics Group, Department of Mathematics, University of Bristol, UK, 2004. (accompanying Technical Report 04:20).

[81] J.L. Oliver, P. Carpena, M. Hackenberg, and P. Bernaola-Galvan. Isofinder: computational prediction of isochores in genome sequences. *Nucleic Acids Research*, 32, 2004.

[82] M. Pensky and B. Vidakovic. On non-equally spaced wavelet regression. *Ann. Inst. Statist. Math.*, 53:681–690, 2001.

[83] D. B. Percival and A. T. Walden. *Wavelet methods for time series analysis*. Cambridge University Press: Cambridge, 2000.

[84] G. Piella and H. J. A. M. Heijmans. Adaptive lifting schemes with perfect reconstruction. *IEEE Trans. Sig. Proc.*, 50:1620–1630, 2002.

[85] S. Sardy, A. Antoniadis, and P. Tseng. Automatic smoothing with wavelets for a wide class of distributions. *J. Comp. Graph. Stat.*, 13(2):399–421, 2004.

[86] S. Sardy, D. B. Percival, A. G. Bruce, H.-Y. Gao, and W. Stuetzle. Wavelet shrinkage for unequally spaced data. *Stat. Comput.*, 9:65–75, 1999.

[87] P. Schröder and W. Sweldens. Spherical wavelets: Efficiently representing functions on the sphere. *Computer Graphics Proceedings (SIGGRAPH 95)*, pages 161–172, 1995.

[88] P. Schröder and W. Sweldens. Spherical wavelets: Texture processing.

In P. Hanrahan and W. Purgathofer, editors, *Rendering Techniques '95*. Springer Verlag, Wien, New York, August 1995.

[89] B.W. Silverman. Some aspects of the spline smoothing approach to non-parametric curve fitting. *J. Roy. Statist. Soc. B.*, 47:1–52, 1985.

[90] B.W. Silverman. Wavelets in statistics: Beyond the standard assumptions. *Roy. Soc. Lond. Philos. Trans. Ser. A Math. Phys. Eng. Sci.*, 357:2459–2473, 1999.

[91] J. Simoens and S. Vandewalle. A stabilized contruction of wavelets on irregular meshes on the interval. *SIAM J. Scientific Computing*, 24(4):1356–1378, 2003.

[92] Jeffrey S. Simonoff. *Smoothing Methods in Statistics*. Springer Series in Statistics. New York: Springer, 1996.

[93] M. Stone. Cross-validatory choice and assessment of statistical predictions. *J. Roy. Statist. Soc. B*, 36:111–147, 1974.

[94] G. Strang and T. Nguyen. *Wavelets and Filter Banks*. Wellesley-Cambridge Press, 1996.

[95] W. Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Appl. Comput. Harmon. Anal.*, 3(2):186–200, 1996.

[96] W. Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM J. Math. Anal.*, 29(2):511–546, 1998.

[97] W. Trappe and KJR. Liu. Denoising via adaptive lifting scheme. In *Proceedings of SPIE Wavelet Applications in Signal and Image Processing VIII*, volume 4119, pages 302–312, 2000.

[98] E. Vanraes, M. Jansen, and A. Bultheel. Stabilised wavelet transforms for non-equispaced data smoothing. *Signal Processing*, 82(12):1979–1990, December 2002.

[99] B. Vidakovic. *Practical Nonparametric and Semiparametric Bayesian Statistics*, chapter Wavelet-based nonparametric Bayes methods, pages 133–155. Number 133 in Lecture Notes in Statistics. Springer-Verlag, 1998.

[100] B. Vidakovic. *Statistical modelling by wavelets.* Wiley: New York, 1999.

[101] P. L. Walker. *The Theory of Fourier Series and Integrals.* Wiley: Chichester, 1986.

[102] M.P. Wand and M.C. Jones. *Kernel Smoothing.* Number 60 in Monographs on Statistics and Applied Probability. Chapman and Hall, 1995.

[103] L. Zhang and J. Chen. Scaling behaviors of cg clusters in coding and noncoding dna sequences. *Chaos, Solitons and Fractals*, 24:115–123, 2004.