PERFORMANCE WEIGHTED BLENDED
POWER SPECTRUM ESTIMATION

by

Jeffrey Tucker
A Thesis
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Master of Science
Electrical Engineering

Committee:

_____	Dr. Kathleen E. Wage, Thesis Director
_____	Dr. Jill Nelson, Committee Member
_____	Dr. Zhi Tian, Committee Member
_____	Dr. Monson H Hayes, Chairman, Department of Electrical and Computer Engineering
_____	Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering

Date: _____ Summer 2020
George Mason University
Fairfax, VA

Performance Weighted Blended Spectral Estimation

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at George Mason University

By

Jeffrey Tucker
Bachelor of Science
George Mason, 2019

Director: Dr. Kathleen E. Wage, Professor
Department of Electrical and Computer Engineering

Summer 2020
George Mason University
Fairfax, VA

Dedication

I dedicate this thesis to Noa, Charlie, Benji, and Isaac.

Acknowledgments

I would like to thank the following people who made this possible; my parents Isaac and Gigi Ho'opi'i, Joe and Jackie Tucker. My advisor Kathleen Wage for her support, and encouragement. John Buck for his advice and developing the beamformer that this algorithm is based on. Lora Van Uffelen for allowing me to use her glider data. All the members of the Ocean Acoustics and Signal Processing group, especially Vaibhav Chavali for the camaraderie and many productive conversation. My sisters Bess and Kukana for always being willing to listen to me complain. Finally I would like to thank Jill Parsons for always believing in me and making the majority of the coffee I consumed while writing this thesis. This research was supported by Office of Naval Research Award N00014-18-1-2669.

Table of Contents

List of Figures

Abstract

PERFORMANCE WEIGHTED BLENDED SPECTRAL ESTIMATION

Jeffrey Tucker

George Mason University, 2020

Thesis Director: Dr. Kathleen E. Wage

A classic approach to power spectrum estimation is to apply a time domain window to a signal and then compute the discrete Fourier transform (DFT). The window provides a trade off between the resolution of the estimator, and the ability to detect a quiet signal when loud signals are also present. There are many windows available, and there is often no single window that provides the best balance between resolution and dynamic range. Analysts can often improve their estimates by combining spectra from multiple windowed DFTs. This thesis proposes a performance weighted blended (PWB) spectrum estimator that automates the work of an analyst by blending an ensemble of estimators. The proposed estimator is an adaptation of Buck and Singer's performance weighted blended beamformer. A sensor array samples a signal in space and a beamformer calculates a spatial frequency spectrum. Since planewave beamforming is analogous to spectral estimation, Buck and Singer's approach can be used to blend windowed DFTs. Thus the same approach can be used to blend windowed DFTs. When an ensemble spectral estimators are constrained to have

unity gain in the look direction, then any difference in their estimates is due to noise or interference. With this in mind, accumulated power output was chosen as the performance metric for the PWB estimator. This estimator is guaranteed to perform as well or better than the best performing estimator in the ensemble as the number of data blocks goes to infinity. The PWB estimator was tested on complex exponential signals with uniformly distributed random phase in complex Gaussian white noise and experimental data. Results show that the PWB estimator is able to exhibit improved resolution in regions of the spectrum where there are loud signals, and improved dynamic range in regions where there are quiet signals. Simulations also show that the PWB estimator is able to outperform a minimum power distortionless response (MPDR) estimator when it is calculated using the sample statistics. Since the estimator as it was originally proposed was not robust enough for use with real data, methods to improve robustness will be presented. The algorithm was evaluated using data from a hydrophone mounted on an underwater glider. The experiments show that the PWB algorithm is able approximate the performance of the best estimator in the ensemble as long as certain restrictions on its parameters are respected.

Chapter 1: Introduction

Spectrum estimation is the problem of calculating the power of a signal at a specific frequency or frequencies of interest. Spectral analysis is very important to many fields such as radar, sonar, geophysics, and music [1]. For example, resonant frequencies in sound reinforcement systems may result in a very loud and unpleasant sound. Audio technicians often use spectrum analyzers to identify these resonant frequencies so that they can be are attenuated using a graphic equalizer.

A well trained audio technician can often identify resonant frequencies and the proper amount of attenuation simply by listening carefully. However, many applications require a more rigorous approach that is usually based on windowed versions of the fast Fourier transform (FFT). The FFT can be viewed as a bank of band pass filters, one for each frequency of interest [2]. Each filter's frequency response has a main lobe that passes the frequency of interest but also allows power from nearby signals to corrupt the estimate, and side lobes which allow power from signals outside the main lobe to corrupt the estimate [3]. These two sources of bias in the estimate are known as resolution, and side lobe leakage respectively. They cannot be completely alleviated, however a window can be applied to the time domain signal in order to change the resolution, and side lobe leakage of each filter in the filterbank [4]. Most useful windows that reduce side lobe leakage also reduce resolution, and windows that improve resolution tend to increase side lobe leakage. Much of the art of practical spectrum estimation reduces to selecting a window that provides a balance between resolution, and side lobe leakage that is appropriate for the spectrum to be estimated.

To get a picture of a signal's frequency content, an analyst will often use multiple windows and synthesise the results from each spectral estimate. High resolution windows are at their best when there is a loud line component at a particular frequency because the

improved resolution does a better job of identifying the exact frequency of the line component. When there is a loud line component at a frequency other than the frequency of interest, a window with less side lobe leakage will do a better job of rejecting the interferer signal. When there are both loud and quiet signals present there will be no single window that provides the best estimate at all frequencies of interest, thus different windows must be selected at each frequency. This thesis presents a performance weighted blended (PWB) estimator that automates this process by weighting each estimator in an ensemble of windowed FFTs based on its performance at each frequency of interest. The resulting spectral estimator performs as well or better than each of the estimators in the ensemble at each frequency of interest [5].

This thesis begins with a review of non-parametric power spectrum estimation. Chapter 3 presents historical context as well as a complete description of the blending algorithm and a detailed convergence proof. Chapter 4 presents simulations of the PWB estimator where it is applied to both stationary, and non-stationary environments. Additionally, the PWB estimator is compared to the minimum power distortionless response (MPDR) estimator [6]. Chapter 5 contains a guide for setting the parameters of the algorithm, and designing the window ensemble. Chapter 6 develops some modifications to the PWB estimator to improve its robustness. The improved algorithm is then demonstrated using hydrophone data which has been spiked with a line component. The performance of the improved PWB algorithm is analyzed in terms of regret, which is the difference between the performance of the PWB algorithm and the best performing algorithm in the ensemble. Chapter 6 also examines the performance of the algorithm in terms of its output signal to noise ratio. It is shown that the PWB algorithm is able to perform as well or better than the best estimator in the ensemble as long as certain conditions on the algorithm parameters are respected. The final chapter summarizes the results, and presents opportunities for future study.

Chapter 2: Background

The goal of spectral analysis is to estimate the distribution of a signal's power in the frequency domain from a finite sequence of measurements. From the analysis we should be able to perform tasks such as identifying resonant frequencies, and determining which frequency bands contain the most power [1]. This thesis focuses on a non-parametric approach based on the Fourier transform. The Fourier transform assumes that any signal can be represented as a sum of complex exponentials. Consider a finite sequence of length N stored in a vector:

$$\mathbf{x} = \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix}. \tag{2.1}$$

Vectors will be represented with variables in bold type. The discrete time Fourier transform (DTFT), $X(e^{j\omega})$ of the signal $\mathbf{x}$ is

$$\text{DTFT}\{\mathbf{x}\} = X(e^{j\omega}) = \sum_{n=0}^{N-1} x[n]e^{-j\omega n}. \tag{2.2}$$

The DTFT of a sequence is a continuous function of ω so it cannot be represented digitally. Throughout the thesis we will use square brackets to distinguish discrete functions from continuous functions. A more practical approach to the spectral analysis is to calculate the DTFT for a discrete set of frequencies of interest. When this is done for N equally spaced

frequencies starting from zero the result is the discrete Fourier transform (DFT):

$$\mathrm{DFT}\{\mathbf{x}\} = X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\omega_k n}, \tag{2.3}$$

where

$$\omega_k = \frac{2\pi f_s k}{N}. \tag{2.4}$$

N will be assumed to be even in this thesis so the index k will take the values:

$$k = \left\{-\frac{N}{2}, -\frac{N}{2}+1, \ldots, 0, \ldots, \frac{N}{2}-1\right\} \tag{2.5}$$

When the signal $\mathbf{x}$ only takes real values as like the data from a hydrophone, then the spectra will be even functions. The complex exponential signals in this research will only be assigned positive frequencies. In both cases the negative half of the spectrum does not provide any additional insight so it will be omitted. Each element of $X(k)$ represents the complex amplitude of the complex exponential portion of the signal at the frequency ω_k [7]. Each index k is referred to as the kth bin rather than explicitly referencing a frequency. These complex amplitudes be squared for each k to estimate the power of the signal at each frequency of interest ω_i

$$\hat{P}(\omega_i) = \left|\sum_{n=0}^{N-1} x[n]e^{-j\omega_i n}\right|^2. \tag{2.6}$$

This chapter will explain the DFT approach by considering it as a bank of band pass filters. The characteristics of the filters in the in the filter bank will be altered by windowing the signal. The advantages of and disadvantages of different types of windows will be discussed. Finally, an optimal filter called the minimum power distortionless response (MPDR) filter will be introduced. The MPDR will be used as a basis of comparison for the proposed performance weighted blended (PWB) algorithm.

2.1 Filter Bank Interpretation

Consider a band pass filter with center frequency ω_c and transfer function

$$H(e^{j\omega}) = e^{j(N-1)(\omega_c-\omega)/2}\frac{\sin(N(\omega_c-\omega)/2)}{\sin((\omega_c-\omega)/2)} \tag{2.7}$$

The impulse response of this filter is

$$h(n) = \begin{cases} e^{j\omega_c n} & n = \{0, 1, \ldots, N-1\} \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

Passing a sequence of N measurements x through this filter yields

$$\begin{aligned} \mathbf{x}_f(N-1) &= \textstyle\sum_{n=0}^{\infty} x[n]h[N-1-n] \\ &= \textstyle\sum_{n=0}^{N-1} x[n]e^{j\omega_c(N-1-n)} \\ &= \textstyle\sum_{n=0}^{N-1} x[n]e^{-j\omega_c n} \end{aligned} \tag{2.9}$$

where $\mathbf{x}_f$ is the filtered output. This is the discrete time Fourier transform (DTFT) of $x(n)$ evaluated at the center frequency of the filter. The DFT forms a bank of these filters by evaluating the filter output for each frequency ω_k. The final step in the estimation is to take the norm squared of the estimates $X(k)$ which yields Eq. 2.6 [8]. Different estimates can be formed using different filters. For example the impulse response

$$h(n) = \begin{cases} \frac{1}{\sqrt{N}}e^{j\omega_c n} & n = \{0, 1, \ldots, N-1\} \\ 0 & \text{otherwise} \end{cases} \tag{2.10}$$

will result in the estimator

$$\hat{P}_{Per}(\omega) = \frac{1}{N}\left|\sum_{n=0}^{N-1} x[n]e^{-j\omega n}\right|^2. \tag{2.11}$$

which is the periodogram [8].

In general the filterbank approach produces estimates in the form

$$\hat{P} = \left|\sum_{n=0}^{N-1} q[n]x[n]e^{-j\omega_c n}\right|^2, \tag{2.12}$$

where $q[n]$ is referred to as a taper. To simplify our notation we will define a replica vector $\mathbf{v}(\omega)$ as:

$$\mathbf{v}(\omega_i) = \begin{bmatrix} \exp j\omega_i[0] \\ \exp j\omega_i[1] \\ \vdots \\ \exp j\omega_i[N-1] \end{bmatrix} \tag{2.13}$$

The dependence on ω will sometimes be suppressed for clarity. The replica vector is the length N sequence that would be produced by a complex exponential at frequency ω_i. The replica vector is useful because it can be used to define a window as $\mathbf{w} = \mathbf{q} \odot \mathbf{v}$ where $\odot$ represents element-wise multiplication. The power estimate at the frequency ω_i can then be written as $\left|\mathbf{w}(\omega_i)^H\mathbf{x}\right|^2$, where the superscript H denotes the Hermitian transpose. Fig. 2.1 contains a block diagram of the estimator using the vector notation.

2.2 Windows

The characteristics of the window used in to implement the filterbank determines the quality of the estimates that the filterbank produces. Figure 2.2 shows the frequency response of the

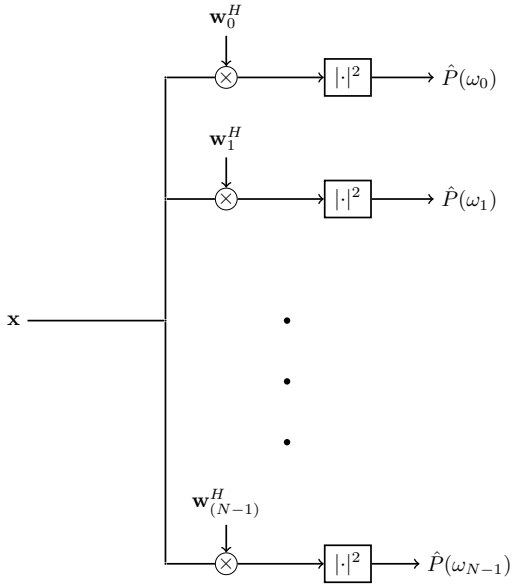

Figure 2.1: A windowed DFT can be interpreted as a bank of band pass filters each implemented by multiplication with the vector $\mathbf{w}$

filter implemented by the uniform window where $q_{uniform}[n] = \frac{1}{N}$, $n = \{0, 1, ..., N-1)$. The most important features of the response are the gain at the center frequency, the resolution, side lobe performance, and coherent gain. This section will look at each of these features and discuss how they effect power estimates generated by the windowed DFT.

The response of the filter in Fig. 2.2 at the center frequency is unity or 0 dB. This is called either the unity gain constraint or the distortionless response constraint [9]. We will require that all windows used in this thesis have this constraint. The reason is that in order to obtain unbiased estimates we have to avoid distorting the signal at the frequency where we are attempting to estimate the power. If a signal $\mathbf{x}$ consists of a single complex exponential at the frequency of interest ω_k then this constraint will insure an unbiased estimate. Further, any bias in the estimate will be the result of interference from either noise, or complex exponential components at frequencies other than the frequency of interest. We can normalize any window to have the distortionless constraint by multiplying the window by $\frac{1}{\sum_{n=0}^{N-1} \tilde{w}(n)}$ where $\tilde{w}(n)$ is the un-normalized window.

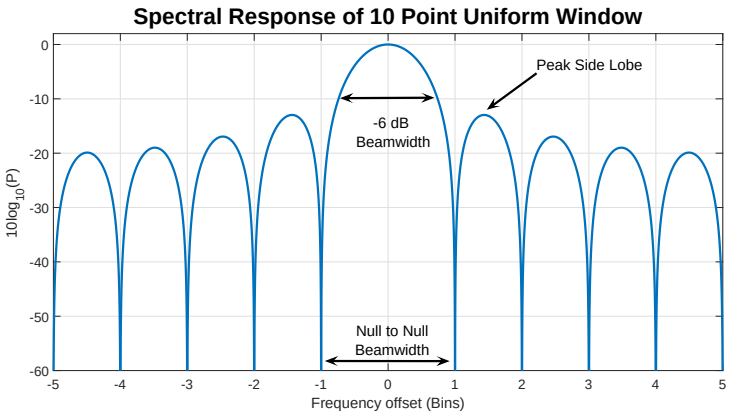

Figure 2.2: Resolution is the ability of a filter to distinguish between signals that are close in frequency space. It is determined by the shape of the filter's main lobe. The side lobes determine the amount of spectral leakage that the filter will admit into the estimate.

The resolution of a filter refers to the width of its main lobe. Another way to describe resolution is as the ability of a filter to tell the difference between two signals that are close in frequency space. If two components of a signal have frequencies that both appear in the main lobe of the filter then they will be difficult to tell apart in the resulting spectrum. The width of the main lobe is often measured from one null to the next called the null to null beamwidth, however it can also be measured at the point either 3 dB, or 6 dB below the maximum gain which are called the -3 dB bandwidth, and the -6 dB bandwidth respectively [4]. Notice that the frequency axis in Fig. 2.2 is in bins. Recall that the center frequencies of each bin $\omega_k = \frac{2\pi f_s k}{N}$ are a function of N the length of the data record. So in addition to the natural resolution of the filter the resolution of the estimator is limited by the length of the data record N. Filters that are longer than the data record say with length $M > N$ can be implemented by zero padding the data, however the zero padding the signal is equivalent to windowing it with a uniform window of length N so we are not able to improve the resolution of the estimates by zero padding [7].

The side lobe performance of a window determines the amount of bias from spectral leakage. The side lobes are usually referenced by both the peak level, and the rate at which they decrease as you move out from the main lobe. The peak side lobe level of the uniform

window in Fig. 2.2 is at -13 dB. Uniform windows have side lobe peaks that decay at -6 dB per octave, however other windows have side lobes that decay at different rates or do not decay at all in the case of Chebychev windows. Most useful window functions represent a trade-off between resolution and side lobe performance. Windows with lower peak side lobe levels tend to have less resolution.

Coherent gain, which is also called processing gain or white noise gain (WNG) is the ratio of the signal to noise ratio (SNR) at the filter output to the SNR at the filter input when there is a single signal at the frequency of interest in white noise. Filters with high coherent gain do a better job of rejecting uncorrelated noise in favor of coherent signals in the spectrum. This is a very important metric because if a filter is incapable of rejecting enough noise it will be unable to estimate the power of the line components in the signal [4]. The WNG of a window can be calculated as $\mathrm{WNG}(\mathbf{w}) = |\mathbf{w}|^{-2}$ [9].

2.2.1 Dolph-Chebychev Windows

For this research the Dolph-Chebychev window is of particular interest. The window was developed by antenna engineers and it is popular in both temporal and spatial spectrum estimation. The window is derived by minimizing the main lobe width of the frequency response of the window,while fixing the level of the side lobes. The window is given by

$$W(k) = (-1)^k \frac{N \cos^{-1}\left[\beta \cos\left(\pi \frac{k}{N}\right)\right]}{\cosh\left[N \cosh^{-1}(\beta)\right]}; \quad 0 \leq |k| \leq N-1 \tag{2.14}$$

where

$$\beta = \cosh\left[\tfrac{1}{N} \cosh^{-1}(10^{\alpha})\right]$$

and

$$\cos^{-1}(x) = \begin{cases} \frac{\pi}{2} - \tan^{-1}\left[\frac{x}{\sqrt{1-x^2}}\right]; & |x| \leq 1 \\ ln\left[x + \sqrt{x^2-1}\right]; & |x| \geq 1 \end{cases}$$

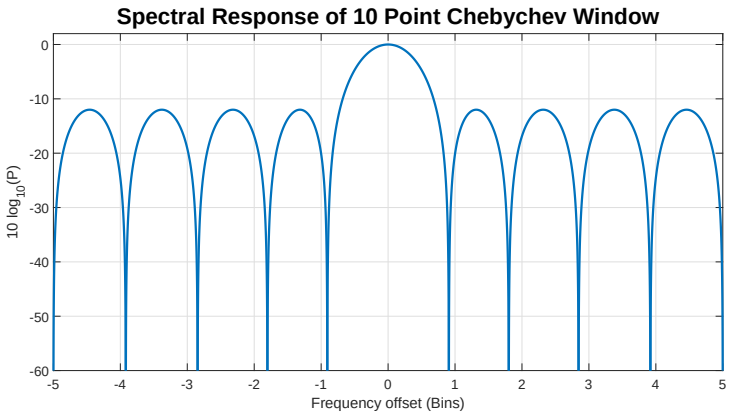

Figure 2.3: The frequency response of a 10 point Chebychev window

The parameter α is the log base 10 of the ratio of mainlobe level which we will assume to be unity to the sidelobe level [9]. Fig. 2.3 contains a plot of a 10 point Chebychev window with a peak side lobe level of -12 dB

The Dolph-Chebychev window has maximum resolution for a particular side lobe level, so it represents a direct trade between resolution, and dynamic range. In fact when the side lobe levels are set very high, the window can achieve better resolution than even a rectangular weighting. This fact will be very important in choosing which windows to include as sub-estimators for a universal estimator.

For practical signals there is generally no single window that can provide an unbiased power estimate. Windows with higher resolution will be preferred for signals that contain complex exponential components that are closely spaced in frequency. Windows with lower side lobes will be preferred for signals that contain both loud and quiet complex exponential components that are far apart in frequency space. The difficulty occurs because the nature of the spectrum to be estimated is generally not known in advance so appropriate windows must be found through trial and error. The next section presents a data adaptive window that seeks to solve this problem by minimizing the power output of the window while maintaining the distortionless response requirement.

2.3 The Minimum Power Distortionless Response Estimator

The minimum power distortionless response (MPDR) estimator was first proposed by Capon in a beamforming context [6]. The estimator is derived by minimizing the power of the estimate subject to the constraint that the output of the estimator has unity gain. Recall that the replica vector $\mathbf{v}$ defined in Eq. 2.13 is that it represents a unit amplitude complex exponential input at a particular frequency of interest ω_i. So we would like to find the weight vector $\mathbf{w}$ that minimizes $|\mathbf{w}^H\mathbf{x}|^2$ with the constraint that $\mathbf{w}^H\mathbf{v} = 1$. This can be done with Lagrange multipliers, and the resulting weight vector is

$$\mathbf{w} = \frac{\mathbf{R}_x^{-1}\mathbf{v}}{\mathbf{v}^H\mathbf{R}_x^{-1}\mathbf{v}}, \tag{2.15}$$

where $\mathbf{R_x}$ is the covariance matrix of $\mathbf{x}$ [10]. The expected value of the power estimate is then $\mathcal{E}\{|\mathbf{w}^H\mathbf{x}|^2\} = \mathbf{w}^H\mathcal{E}\{\mathbf{x}\mathbf{x}^H\}\mathbf{w} = \mathbf{w}^H\mathbf{R}_x\mathbf{w} = \frac{1}{\mathbf{v}^H\mathbf{R}_x^{-1}\mathbf{v}}$ [9]. The key insight here is that when the ensemble covariance matrix is known, then the MPDR estimate is a deterministic function of the covariance matrix. Another important property of the MPDR is that if $\mathbf{x}$ consists of only a single exponential signal in white noise then the MPDR estimator becomes the uniform window [9].

In practice the ensemble covariance matrix is not available, so $\mathbf{R_x}$ must be replaced by the sample covariance matrix estimated from a block of the data which introduces bias into the estimate. We will refer to the MPDR calculated with the sample statistics as the sample MPDR to distinguish it from the ensemble MPDR in Capon's original formulation. The sample covariance matrix can be estimated as

$$\hat{\mathbf{R}}_x = \frac{1}{M}\sum_{m=1}^{M}\mathbf{x}_m\mathbf{x}_m^H, \tag{2.16}$$

where here the data record $\mathbf{x}$ has been segmented into M non-overlapping blocks of length

N. The mth block is stored int he vector $\mathbf{x}_m$. Capon and Goodman showed that when the signal consists of independent complex Gaussian random variables, the sample MPDR underestimates power in the look direction by a factor of $\frac{M-N+1}{M}$. This bias can be corrected by multiplying the estimate by $\frac{M}{M-N+1}$, but the correction should be done with caution as it also increases the variance of the estimate [11].

2.4 Examples

This section presents two examples to illustrate the importance of the window properties in the previous section. The examples use estimators built with Chebychev windows. The main focus is examples that demonstrate the effects of resolution, spectral leakage, and coherent gain. Each example consists of complex exponentials with a uniformly distributed random phase, and zero mean additive Gaussian white noise. The power and location of the complex exponentials in frequency space is shown in Fig. 2.4. The sampling frequency is 1000 Hz. A total of 100,000 samples of the processes were generated, and the time series was split into 1,000 blocks of 100 samples each. The spectral estimates were all calculated using a 10,000 point DFT on each block. The power estimates were then averaged over the 1,000 Monte Carlo trials. Each example is processed using two different Chebychev windows, one with -6 dB peak side lobes, and the other with -30 dB side lobes. The spectral responses of the two windows are shown in Fig. 2.5.

For the first example the noise power was set to unity to demonstrate the effects of resolution and side lobe levels. The spectra generated by each of the two Chebychev windows and an ensemble MPDR is shown in Fig. 2.6. The ensemble MPDR provides a more accurate estimate however it requires advance knowledge of the signal's covariance matrix to implement. The window with -6 dB side lobes has enough resolution that the spectrum it generates clearly displays the signals at 250 Hz, and 258 Hz as separate components. The cost of this improved resolution is poor side lobe performance, thus the spectrum shows the signal at 400 Hz is covered by spectral leakage from the two louder signals. The window

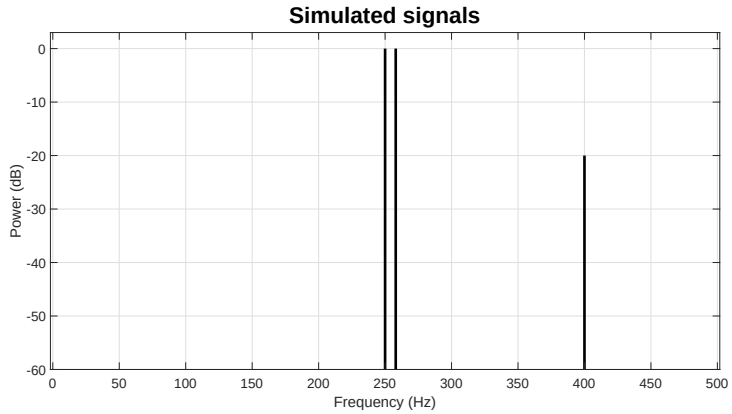

Figure 2.4: The locations and relative powers of the complex exponential signals in the example were chosen to demonstrate the trade off between resolution and side lobe performance for spectrum estimation with fixed windows.

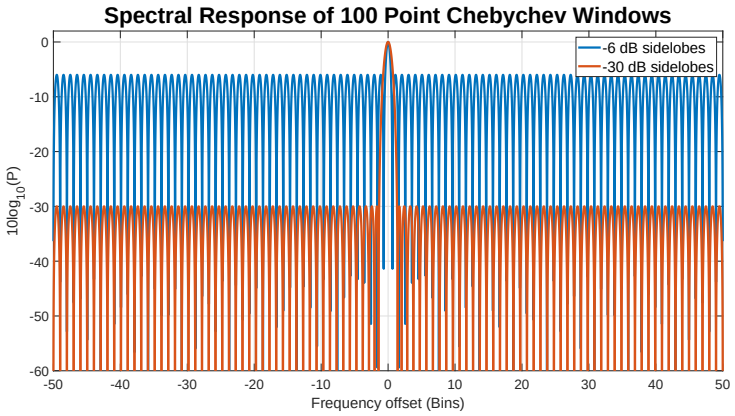

Figure 2.5: Chebychev windows with high side lobe levels have better resolution, and Chebychev windows with low side lobe levels have worse resolution

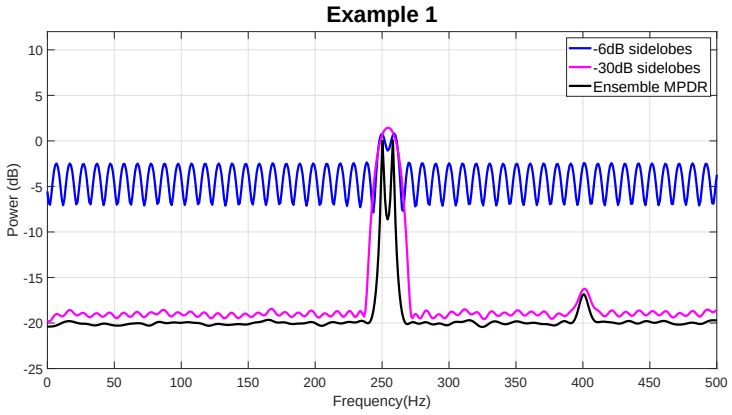

Figure 2.6: The window with -6 dB side lobes can resolve the signals at 250 Hz and 258 Hz, however it cannot detect the signal at 400 HZ. The window with -30 dB side lobes can detect the signal at 400 Hz but it cannot resolve the other two signals. The ensemble MPDR provides a more accurate estimate however it requires advance knowledge of the signal's covariance matrix.

with -30 dB side lobes estimates the power of the 400 Hz signal more accurately, however it lacks the resolution to distinguish between the signals at 250 Hz and 258 Hz. Recall that by definition the Chebychev window provides the maximum resolution for a given peak side lobe level, so this problem is not limited to the case of these two windows it is a fundamental difficulty in spectrum estimation using windowed DFTs.

To demonstrate the effects of coherent gain, the noise power has been increased by in the second example. The spectra are shown in Fig. 2.7. The ensemble MPDR is not able to reject enough noise to reveal the signals in this case. The Chebychev window with -6 dB side lobes has much less coherent gain than the other window, so even though it has superior resolution it is unable to provide a satisfactory spectral estimate because the SNR in the example is too low. The window with -30 dB side lobes performs better however it still unable to resolve the signals at 250 Hz and 258 Hz. The key insight is that the coherent gain of a window determines the minimum SNR at which it can still detect line components in the signal.

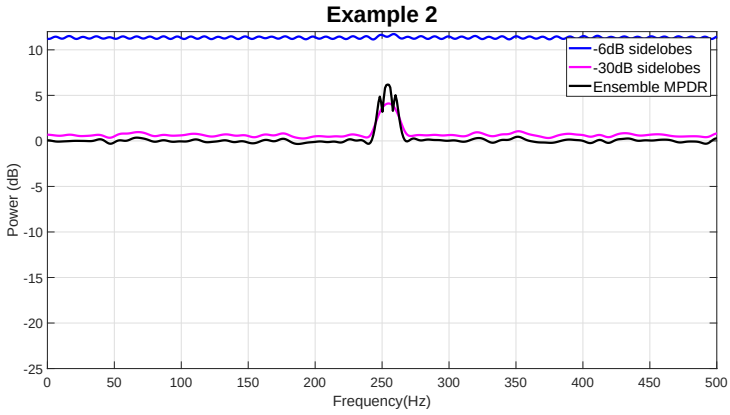

Figure 2.7: The window with -6 dB side lobes doesn't have enough coherent gain to reveal any of the signals . The window with -30 dB side lobes can detect that there is power in the 250 Hz region but it cannot resolve the other two signals.

2.5 The Short Time Fourier Transform

In order for the periodogram to provide a reasonable spectral estimate we require that the process be wide sense stationary, however this is rarely the case with practical signals. To make the process more general we can assume that process is locally stationary that is, the autocorrelation function $r_x(k)$ is only a function of the lag for $|k| \leq M$ for some finite integer M. This assumption justifies processing the time series with a periodogram M samples at a time. The resulting spectrum is a function of both frequency and time and is called the short time Fourier transform (STFT). The STFT is defined as

$$X(n,\omega) = \sum_{m=0}^{M-1} x[n+m]q[m]e^{-j\omega m} \tag{2.17}$$

where $q[n]$ is a taper [1].

The spectrogram is a plot of the output of the STFT. It is useful because it shows how the power spectrum of a non-stationary signal changes over time. The STFT produces a spectrum for each time n in the data record, however in creating a spectrogram we typically do not calculate the STFT for each time n. The STFT in Eq. 2.17 can be

viewed as advancing the analysis window by one time sample for each new periodogram. If the analysis window has length M then instead of advancing by one sample at a time we can advance the window by M samples so that each periodogram in the spectrogram is calculated using a different non-overlapping block of the time series data. Alternatively the blocks can be allowed to overlap, and the amount of overlap is usually specified as a percentage of the length of the block. So if $M = 100$ then 50% overlap would require an advance of 50 samples for each periodogram.

Chapter 3: Algorithm

This chapter discusses the history of the proposed PWB algorithm. It also presents John Buck's proof that the loss of the PWB converges to the best estimator in the ensemble [12]. There is a brief discussion about how to efficiently calculate the PWB estimates. Finally, a renormalization method that mitigates numerical issues with the PWB algorithm is presented.

3.1 History

The PWB algorithm is a special case of universal coding algorithms, an idea that comes from information theory. Kolmogorov gave the first description of universal coding in a paper on the quantitative definition of information [13]. A formal definition is given by Davisson in [14]. Davisson describes universal coding as any method for converting message blocks into code blocks when the encoding is based only on an observed source, and some performance measure is attained in the limit as the block length goes to infinity. As an example, the performance measure could be vanishing MSE in the case of a prediction algorithm, or some other loss function to be minimized. The universal approach can also be used for the problems of estimation, and prediction, as suggested in [15] and [16], by expanding the definition so that the only requirement is that some performance measure is attained in the limit as the number of estimates or predictions goes to infinity. In 1999 Singer and Feder proposed a universal predictor that uses a function for weighting predictors of different model orders [17]. Finally, Buck, and Singer proposed using Singer and Feder's weighting function with Dominant Mode Rejection Beamformers of different model orders [5].

3.1.1 Universal Linear Prediction Using Soft-max

Linear prediction is the problem of predicting future values of a sequence $x[t]$ by a linear transformation of the past values so that $\hat{x}[t] = f(x[0], x[1], \ldots, x[t-1]) \approx x[t]$ where f is a linear function. The weighting function used in the PWB estimator was also used in [17] by Singer and Feder to combine an ensemble of linear predictors. They proposed a universal linear predictor of the form

$$\hat{x}_u[t] = \sum_{k=1}^{M} \mu_k[t]\hat{x}_k[t] \tag{3.1}$$

where

$$\mu_k[t] = \frac{\exp(-\frac{1}{2\nu}l_{t-1}(x, \hat{x}_k))}{\sum_{j=1}^{M}\exp(-\frac{1}{2\nu}l_{t-1}(x, \hat{x}_j))}. \tag{3.2}$$

where ν is a sensitivity parameter that determines the estimator's speed of convergence. The function $l_n(x, \hat{x}_n)$ is a loss function that acts as the performance metric for each predictor in the ensemble. Singer and Feder used the squared error between the estimate and the true value of $x[t]$ is used as the loss function in this case so that:

$$l_n(x, \hat{x}_n) = \sum_{t=1}^{n}(x[t] - \hat{x}_u[t])^2$$

Singer and Feder went on to show that for any bounded sequence $|x[t]| \leq A$, and $\nu \geq 4A^2$

$$\frac{1}{n}l_n(x, \hat{x}_u) \leq \min_k \frac{1}{n}l_n(x, \hat{x}_k) + \frac{2\nu ln(M)}{n}$$

.

This universal predictor can be interpreted as a performance weighted sum of an ensemble

of predictors. The weightings are determined by applying what is often called the soft-max, multivariate-sigmoid, or multivariate-logistics function to the output of the loss function l_n. The PWB algorithm makes use of the same weighting function.

3.1.2 The Universal Dominant Mode Rejection Beamformer

Buck and Singer adapted Singer and Feder's prediction algorithm for use with dominant mode rejection beamformers [5]. A new loss function was the only alteration to Singer, and Feder's prediction algorithm required to use it with dominant mode rejection beamformers of different dominant subspace dimensions instead of the predictors that Singer and Feder used [5]. In the beamforming problem a spatial spectrum is estimated from snapshots which are vectors of complex phasors that represent narrowband planewaves passing through a sensor array. Each snapshot is an $N \times 1$ complex valued vector where N is the number of sensors in the array. Snapshots are the beamforming analog of the blocks used in the STFT for temporal spectrum estimation. The dominant mode rejection (DMR) beamformer was introduced by Abraham and Owsley as an improvement to the MPDR beamformer [18]. To calculate the MPDR weight vector the covariance matrix of the data must be inverted. If the ensemble statistics are known then the matrix inversion does not cause any problems. However, when the covariance matrix is estimated from the data the estimate may not be an invertible matrix. To overcome this difficulty the DMR imposes a structure on the sample covariance matrix. The algorithm replaces the $N-d$ smallest eigenvalues with their average. The remaining d eigenmodes form the dominant subspace of the resulting estimate of the covariance matrix S_{DMR}. The new DMR covariance matrix is then used in place of the ensemble covariance matrix in the MPDR beamformer. The difficulty in application of the DMR beamformer is estimating the appropriate dominant subspace dimension d. Buck and Singer use an ensemble of D dominant mode rejection beamformers each with a different number of dimensions in the noise subspace. The window vector for each beamformer will be called $\mathbf{w}_d$.

A major change to Singer and Feder's estimator was required to adapt it for beamforming. In the linear prediction case we have access to the ground truth so the square error is a natural loss function, but this is not true in the beamforming problem. In [5] Buck and Singer propose a universal dominant mode rejection (UDMR) beamformer with the loss function

$$l_n(\mathbf{w}, \mathbf{x}) = \sum_{\tau=1}^{n} |\mathbf{w}[\tau]^H \mathbf{x}[\tau]|^2 \tag{3.3}$$

which is simply the accumulated power output of the beamformers.

The motivation behind the loss function is the unity gain constraint placed on the DMR beamformers. If a collection of DMR beamformers are steered towards a signal the signal itself will be passed through unaltered. So any variation in the outputs is due to interference or noise, and can only increase the output power. The new loss function changes the formulation of the estimator slightly so that the UDMR beamformer is

$$w_u[n] = \sum_{d=0}^{D} \mu_d[n] \mathbf{w}_d[n] \tag{3.4}$$

with

$$\mu_d[n] = \frac{exp(-\frac{1}{2\nu} l_{n-1}(\mathbf{w}_d, \mathbf{x}))}{\sum_{q=0}^{D} exp(-\frac{1}{2\nu} l_{n-1}(\mathbf{w}_q, \mathbf{x}))}. \tag{3.5}$$

The beamformer converges so that

$$\frac{1}{n} l_n(\mathbf{w}_u, \mathbf{x}) \leq \min_d \frac{1}{n} l_n(\mathbf{w}_d, \mathbf{x}) + \frac{2\nu ln(M)}{n} \tag{3.6}$$

when

$$\nu \geq \max_{d,n} ||\mathbf{x}[n]||^2 ||\mathbf{w}_d[n]||^2. \tag{3.7}$$

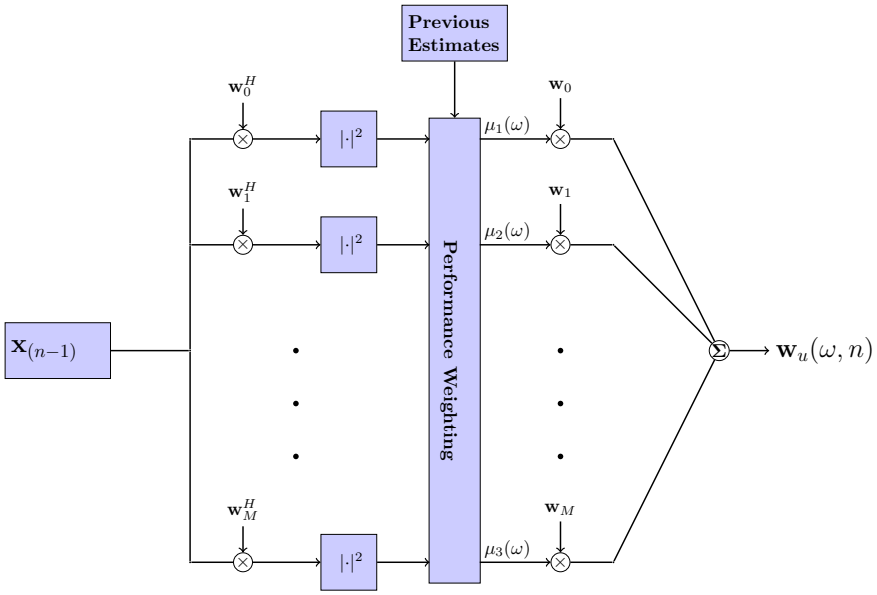

Figure 3.1: A block diagram of the algorithm

Although Buck and Singer's beamformer was designed for use with DMR beamformers, it works with any ensemble of linear estimators with a unity gain constraint. The next section describes the PWB algorithm which has the same basic structure as the UDMR.

3.2 The Performance Weighted Blended Spectral Estimator

For this discussion of the PWB algorithm we will make the assumptions that we are estimating the power of a signal at a single frequency from the nth block of data. We will assume that each block has length N. The algorithm uses each of M windows, which we will call the ensemble to calculate the power of the current data block at a frequency of interest. We will use the notation $\mathbf{w}_m$, suppressing the dependence on ω to refer to the mth filter in the ensemble. The output from each window is combined with the output from all previous data blocks and a loss is calculated. Then weights are found as a function of the loss for each window in the ensemble, and the windows are multiplied by their weights, and summed to form a new window $\mathbf{w_u}$. The new window is then applied to the next block to find the power estimate. Fig. 3.1 contains a block diagram of the algorithm.

We require the windows in the ensemble to be normalized so that the filters pass a signal at the frequency of interest with no amplification or attenuation. As in the beamforming problem, windows with this normalization will provide estimates that can only have a positive bias resulting from noise, or interaction with signals at other frequencies. This property allows us to use accumulated power

$$\mathcal{L}_n(\mathbf{w}, \mathbf{x}) = \sum_{\tau=1}^{n} \left| \mathbf{w}[\tau]^H \mathbf{x}[\tau] \right|^2 \tag{3.8}$$

as the loss function. The weighting function is

$$\mu_m(\omega) = \frac{\exp(-(\frac{1}{2\nu})\mathcal{L}_{n-1}(\mathbf{w_m}, \mathbf{x}, \omega))}{\sum_{i=1}^{M} \exp(-(\frac{1}{2\nu})\mathcal{L}_{n-1}(\mathbf{w_i}, \mathbf{x}, \omega))} \tag{3.9}$$

where the parameter ν will be explained in the convergence proof. The universal window is

$$\mathbf{w}_u[n, \omega] = \sum_{m=1}^{M} \mu_m[n, \omega] \mathbf{w}_m[n]. \tag{3.10}$$

The algorithm is guaranteed to converge such that

$$\frac{1}{n}\mathcal{L}_n(\mathbf{w}_u, \mathbf{x}) \leq \min_m \frac{1}{n}\mathcal{L}_n(\mathbf{w}_m, \mathbf{x}) + \frac{2\nu ln(M)}{n} \tag{3.11}$$

3.3 Convergence Proof

John Buck proved the assertion in Eq. 3.11 [12]. Start by defining a psuedo-probability on the loss function

$$P_{\mathbf{w}}(\mathbf{X}^n) = c \exp\left[\frac{-1}{2\nu}\mathcal{L}_n(\mathbf{w}, \mathbf{x})\right] \tag{3.12}$$

the psuedo-probability for the PWB estimator is then

$$\begin{aligned} P_u(\mathbf{X}^n) &= c \exp\left[\tfrac{-1}{2\nu}\mathcal{L}_n(\mathbf{w}_u, \mathbf{x})\right] \\ &= c \exp\left[\tfrac{-1}{2\nu}\sum_{\tau=1}^{n}\left|\mathbf{w}_u[\tau]^H\mathbf{x}[\tau]\right|^2\right] \\ &= c \exp\left[\tfrac{-1}{2\nu}\sum_{\tau=1}^{n}\left|\sum_{m=1}^{M}\mu_m[\tau]\mathbf{w}_m[\tau]^H\mathbf{x}[\tau]\right|^2\right] \\ &= c\prod_{\tau=1}^{n}\exp\left[\frac{-1}{2\nu}\left|\sum_{m=1}^{M}\mu_m[\tau]\mathbf{w}_m[\tau]^H\mathbf{x}[\tau]\right|^2\right] \end{aligned} \tag{3.13}$$

Defining the function

$$f_\tau(\mathbf{z}) = \exp\left[\frac{-1}{2\nu}\left|\mathbf{z}^H\mathbf{x}[\tau]\right|^2\right] \tag{3.14}$$

and substituting yields the psuedo-probability as a function of a weighted blend of the windows in the ensemble

$$P_u(\mathbf{X}^n) = c\prod_{\tau=1}^{n} f_\tau\left(\sum_{m=1}^{M}\mu_m[\tau]\mathbf{w}_m[\tau]\right) \tag{3.15}$$

The psuedo-probability of the average estimator is then

$$P_{Avg}(\mathbf{X}^n) = \frac{1}{M}\sum_{m=1}^{M} P_m(\mathbf{X}^n) \tag{3.16}$$

where P_m is the psuedo-probability associated with the mth estimator in the ensemble. Using successive conditioning the average pseudo-probability is rewritten as

$$\begin{aligned} P_{Avg}(\mathbf{X}^n) &= P_{Avg}(\mathbf{x}[n]|\mathbf{X}^{n-1}P_{Avg}(\mathbf{x}[n-1]|\mathbf{X}^{n-2}\ldots P_{Avg}(\mathbf{x}[1]) \\ &= \textstyle\prod_{\tau=1}^{n} P_{avg}(\mathbf{x}[\tau]|\mathbf{X}^{\tau-1}) \end{aligned} \tag{3.17}$$

where

$$P_{Avg}(\mathbf{x}[1]|\mathbf{X}^0) = P_{Avg}(\mathbf{x}[1]) \tag{3.18}$$

The next step is to invoke Bayes rule to write

$$\begin{aligned} P_{Avg}(\mathbf{x}[n]|\mathbf{X}^{n-1}) &= \frac{P_{Avg}(\mathbf{x}[n],\mathbf{X}^{n-1})}{P_{Avg}(\mathbf{X}^{n-1})} \\ &= \frac{P_{Avg}(\mathbf{X}^n)}{P_{Avg}(\mathbf{X}^{n-1})} \\ &= \frac{\sum_{m=1}^{M} P_m(\mathbf{X}^n)}{\sum_{k=1}^{M} P_k(\mathbf{X}^{n-1})} \\ &= \frac{\sum_{m=1}^{M} P_m(\mathbf{x}[n]|\mathbf{X}^{n-1})P_m(\mathbf{X}^{n-1})}{\sum_{k=1}^{M} P_k(\mathbf{X}^{n-1})} \\ &= \sum_{m=1}^{M} P_m(\mathbf{x}[n]|\mathbf{X}^{n-1})\frac{P_m(\mathbf{X}^{n-1})}{\sum_{k=1}^{M} P_k(\mathbf{X}^{n-1})} \end{aligned} \tag{3.19}$$

but

$$\frac{P_m(\mathbf{X}^{n-1})}{\sum_{k=1}^{M} P_k(\mathbf{X}^{n-1})} = \frac{\exp(\frac{-1}{2\nu}\mathcal{L}_{n-1}(\mathbf{w}_m,\mathbf{x}))}{\sum_{k=1}^{M}\exp(\frac{-1}{2\nu}\mathcal{L}_{n-1}(\mathbf{w}_k,\mathbf{x}))} = \mu_m[n] \tag{3.20}$$

so

$$P_{avg}(\mathbf{x}[n]|\mathbf{X}^{n-1}) = \sum_{m=1}^{M} P_m(\mathbf{x}[n]|\mathbf{X}^{n-1})\mu_m \tag{3.21}$$

Using Bayes rule again we can write $P_m(\mathbf{x}[n]|\mathbf{X}^{n-1})$ in terms of the function f defined earlier

$$\begin{aligned} P_m(\mathbf{x}[n]|\mathbf{X}^{n-1}) &= \frac{P_m(\mathbf{x}[n],\mathbf{X}^{n-1})}{P_m(\mathbf{X}^{n-1})} &&= \frac{P_m(\mathbf{X}^n)}{P_m(\mathbf{X}^{n-1})} \\ &= \frac{\exp(\frac{-1}{2\nu}\mathcal{L}_n(\mathbf{w}_m,\mathbf{x}))}{\exp(\frac{-1}{2\nu}\mathcal{L}_{n-1}(\mathbf{w}_m,\mathbf{x}))} &&= \frac{\exp(\frac{-1}{2\nu}\sum_{\tau=1}^{n}|\mathbf{w}_m[\tau]^H\mathbf{x}[\tau]|^2)}{\exp(\frac{-1}{2\nu}\sum_{t=1}^{n-1}|\mathbf{w}_m[t]^H\mathbf{x}[t]|^2)} \\ &= \exp(\frac{-1}{2\nu}|\mathbf{w}_m[n]^H\mathbf{x}[n]|^2) &&= f_n(\mathbf{w}_m[n]). \end{aligned} \tag{3.22}$$

Putting it all together

$$P_{avg}(\mathbf{x}[n]|\mathbf{X}^{n-1}) = \sum_{m=1}^{M}\mu_m P_m(\mathbf{x}[n]|\mathbf{X}^{n-1}) = \sum_{m=1}^{M}\mu_m f_n(\mathbf{w}_m[n]) \tag{3.23}$$

and

$$P_{Avg}(\mathbf{X}^n) = \prod_{\tau=1}^{n} P_{avg}(\mathbf{x}[\tau]|\mathbf{X}^{\tau-1}) = \prod_{\tau=1}^{n}\sum_{m=1}^{M}\mu_m f_\tau(\mathbf{w}_m[\tau]) \tag{3.24}$$

Jensen's inequality guarantees that

$$f_\tau(\sum_{m=1}^{M}\mu_m[\tau]\mathbf{w}_m[\tau]) \geq \sum_{m=1}^{M}\mu_m[\tau]f_\tau(\mathbf{w}_m[\tau]) \tag{3.25}$$

The inequality holds as long as f_τ is convex. To make sure this condition is met it is required to place an lower bound on the parameter ν so that

$$|\mathbf{w}_m^H\mathbf{x}[\tau]| \leq \sqrt{\nu} \tag{3.26}$$

or

$$\nu \geq |\mathbf{w}_m^H\mathbf{x}[\tau]|^2 \tag{3.27}$$

Comparing the products terms by term in equations 3.24, and 3.15 shows that

$$P_u(\mathbf{X}^n) \geq P_{Avg}(\mathbf{X}^n) \tag{3.28}$$

The next step is to bound the average estimator in terms of the best estimator.

$$P_{Avg}(\mathbf{X}^n) = \frac{1}{M}\sum_{m=1}^{M} P_m(\mathbf{X}^n) \geq \frac{1}{M}\min_m P_m(\mathbf{X}^n) \tag{3.29}$$

Therefore

$$P_u(\mathbf{X}^n) \geq \frac{1}{M}\max_m P_m(\mathbf{X}^n) \tag{3.30}$$

To complete the proof we take the natural logarithm of both sides, multiply both sides by negative one to flip the inequality, and finally substitute from the definitions of the pseudo-probabilities.

$$\begin{aligned}
&ln(P_u(\mathbf{X}^n)) \geq ln(\tfrac{1}{M}\max_m P_m(\mathbf{X}^n)) \\
&-ln(P_u(\mathbf{X}^n)) \leq -ln(\tfrac{1}{M}) - ln(\max_m P_m(\mathbf{X}^n)) \\
&-ln(c\exp[\tfrac{-1}{2\nu}\mathcal{L}_n(\mathbf{w}_u,\mathbf{x})] \leq -ln(\tfrac{1}{M}) - ln(\max_m c\exp[\tfrac{-1}{2\nu}\mathcal{L}_n(\mathbf{w}_m,\mathbf{x})])) \\
&-ln(c) + \tfrac{1}{2\nu}\mathcal{L}_n(\mathbf{w}_u,\mathbf{x})] \leq -ln(\tfrac{1}{M}) - ln(c) + \tfrac{1}{2\nu}\min_m \mathcal{L}_n(\mathbf{w}_m,\mathbf{x}) \\
&\mathcal{L}_n(\mathbf{w}_u,\mathbf{x}) \leq \min_m \mathcal{L}_n(\mathbf{w}_m,\mathbf{x}) + 2\nu ln(M)
\end{aligned} \tag{3.31}$$

Dividing both sides by n yields eq. 3.11 as claimed.[12]

3.4 Efficient Calculation

It is not necessary to calculate the PWB window in order to get the resulting power estimate. Using vector notation for the filters and suppressing the dependence on the frequency of

interest yields:

$$\hat{P} = \left|\mathbf{w}_u^H \mathbf{x}\right|^2 = \left|(\sum_{m=1}^{M} \mu_m \mathbf{w}_m)^H)\mathbf{x}\right|^2 = \left|\sum_{m=1}^{M} \mu_m (\mathbf{w}_m^H \mathbf{x})\right|^2. \tag{3.32}$$

The insight here is that we can calculate the power estimate for a frequency of interest by calculating the weight coefficients, and applying them directly to the complex output of each filter in the filter-bank formed by the windowed DFT without the need to explicitly calculate the PWB window.

3.5 Re-normalization

The PWB algorithm has a numerical problem built in to the performance weighting. The performance weighting function is vulnerable to saturation when the values of the loss are very large. The exponential terms can get truncated to zero, and if this happens for each of the estimators the denominator of the weighting function becomes zero. A method for stabilizing the softmax function is presented by Goodfellow et al. in [19]. The maximum of the inputs to the function is subtracted from each of the inputs to the soft-max function.

$$\text{softmax}(-\frac{1}{2\nu}\mathcal{L}) = \text{softmax}(-\frac{1}{2\nu}\mathcal{L} - \max_{\text{i}}(-\frac{1}{2\nu}\mathcal{L}_{\text{i}})) \tag{3.33}$$

To show that this does not change the weights define the unnormalized performance weight $\tilde{\mu} = \exp(\frac{-1}{2\nu}\mathcal{L}_m)$ where here $\mathcal{L}_m$ denotes the loss of the mth window in the ensemble. Multiplying each $\tilde{\mu}$ by a constant β gives us

$$\frac{\beta\tilde{\mu}_m}{\sum_{m=1}^{M} \beta\tilde{\mu}_m} = \frac{\tilde{\mu}_m}{\sum_{m=1}^{M} \tilde{\mu}_m} = \mu_m. \tag{3.34}$$

So multiplying the unnormalized weights by a constant does not change the output of the

weighting function. With this in mind we can choose

$$\beta = \exp(\frac{1}{2\nu} \min_m \mathcal{L}_m) \tag{3.35}$$

so that

$$\tilde{\mu}_m = \exp(\frac{-1}{2\nu}(\mathcal{L}_m - \min_k \mathcal{L}_k)). \tag{3.36}$$

This technique limits the growth of the arguments in the exponential terms and hence keeps the numerical issues under control.

Chapter 4: Simulations

The performance of the PWB algorithm was verified with four types of simulations. The first simulation is a simple example that demonstrates how the algorithm operates. The next set of simulations demonstrate the performance of the PWB estimator under stationary conditions. The third set of simulations compares the PWB estimator to the ensemble MPDR and the sample MPDR. Finally, a simulation in a non-stationary environment is examined.

4.1 Example

The estimator ensemble for this example is two Chebychev windows with peak side lobe levels of -12 dB and -120 dB. The simulation uses a sampling frequency of 1000 Hz, and the data record is segmented into a total of 311 non-overlapping blocks of 100 samples each. The first block was used only to initialize the blend weights. The power estimates were averaged over 1000 Monte Carlo trials. The first 11 blocks consist only of white noise with a power of -24 dB below unity at a single sample, i.e., $10\log_{10}(\sigma^2) = -24$, where σ^2 is the variance of the noise process. The subsequent blocks consist of the same white noise with two complex exponential signals. The exponential signals were assigned random phases at the beginning of each Monte Carlo trial. The first signal is at 250 Hz and unity gain. The second signal has a power of -24 dB below unity at 400 Hz. The sensitivity parameter of the PWB estimator was set to 1.

Spectrograms using each of the two Chebychev windows are shown in Fig. 4.1. The estimator using a Chebychev window with -12 dB side lobes admits less bias through its main lobe and the estimator using a Chebychev window with -120 dB side lobes admits less bias through its side lobes. The result is that in the spectrogram produced by the window

with -12 dB side lobes the signal component at 400 Hz is not visible. In the spectrogram produced by the estimator with -120 dB side lobes both components are visible however the cost of the reduced spectral leakage is that the signals are not as well localized due to the lack of resolution.

Figure 4.1 also contains a spectrogram generated by the PWB algorithm. By the end of the simulation the PWB estimator is able to combine the estimates from the two ensemble estimates into an estimate that has the best properties of each. The signal at 250 Hz appears with the resolution of the estimate produced by the Chebychev window with -12 dB side lobes. The 400 Hz signal also appears in the estimate however it does not have the same resolution as the signal at 250 Hz. Also notice that in the region of improved resolution around 250 Hz there is more spectral leakage from the 250 Hz signal than there is in other regions. This is fundamentally how the PWB works. For each frequency of interest the PWB selects the estimator or combination of estimators that have resolution and side lobe characteristics that minimize the bias of the PWB estimate.

To understand how the algorithm is combining the two estimates we can look at the blend weights associated with estimator using the Chebychev window with -12 dB side lobes which we will call $\mu_1(m, \omega)$. There are only two estimators in the ensemble so the blend weights for the second window is simply $1 - \mu_1(m, \omega)$. Figure 4.2 contains a color scale plot of $\mu_1(\mu, \omega)$ as well as vertical slices of the color scale plot at 250, 260, 281, and 400 Hz. For the first 10 blocks the blend weights are all at roughly 0.5 because there are no line components in the signal. When the line components are turned on the weights adapt to the new environment. At 250 Hz the weights stay at 0.5 because both estimators pass the 250 Hz signal at unity gain. At 260 Hz the PWB weights the window with -12 dB side lobes more heavily. The 250 Hz signal appears in the main lobe of the window with -120 dB side lobes at 260 Hz so the improved resolution of the window with -12 dB side lobes provides a less biased estimate. At 281 Hz the PWB algorithm prefers the window with -120 dB side lobes because the 250 Hz signal now appears in the side lobes of both windows. Similarly at 400 Hz, the algorithm weights the window with -120 dB side lobes more heavily because

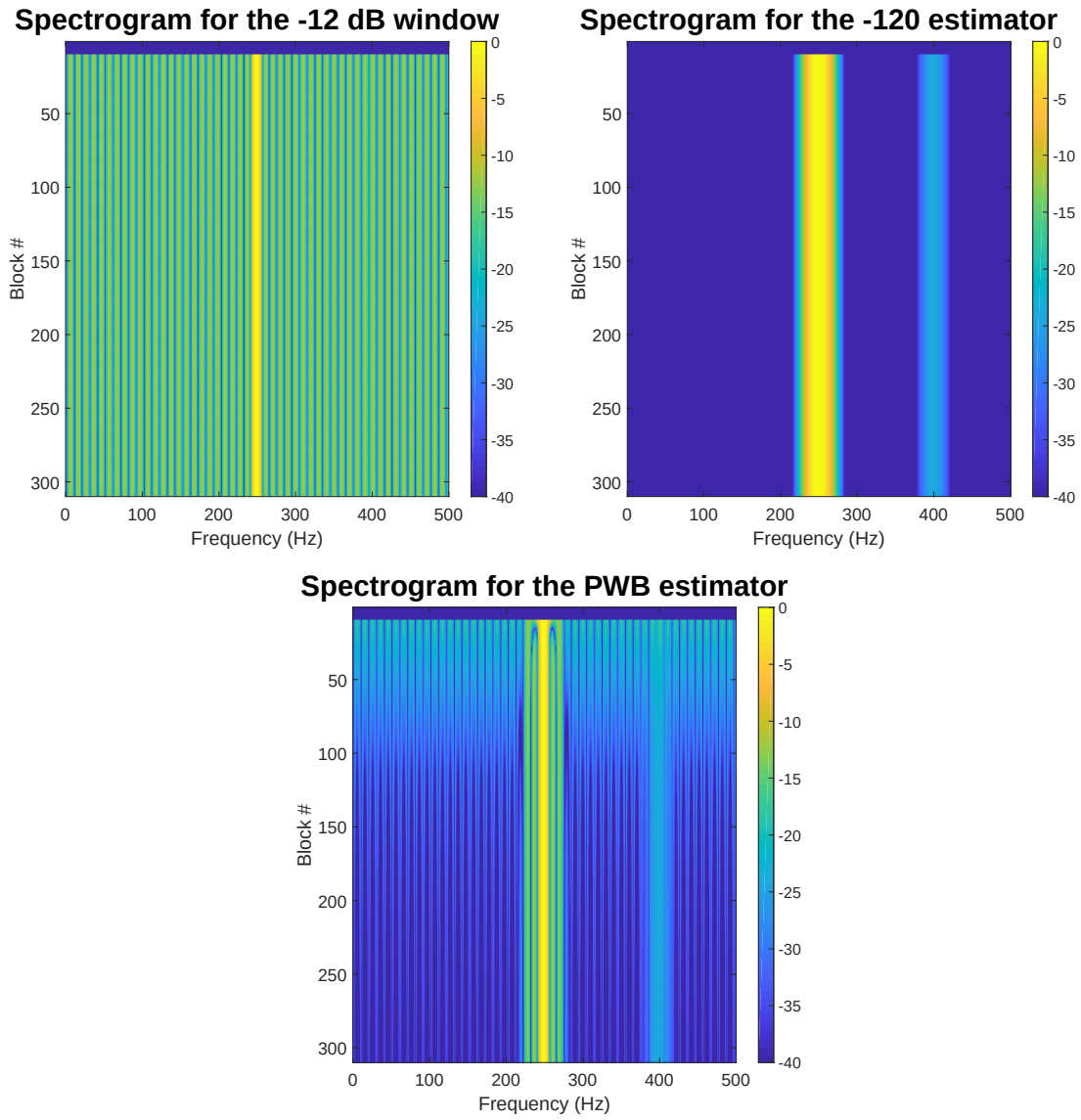

Figure 4.1: The Chebychev window with -12 dB side lobes has better resolution, however the Chebychev window with -120 dB side lobes has less spectral leakage. The PWB estimate has the best properties of each estimator in the ensemble.

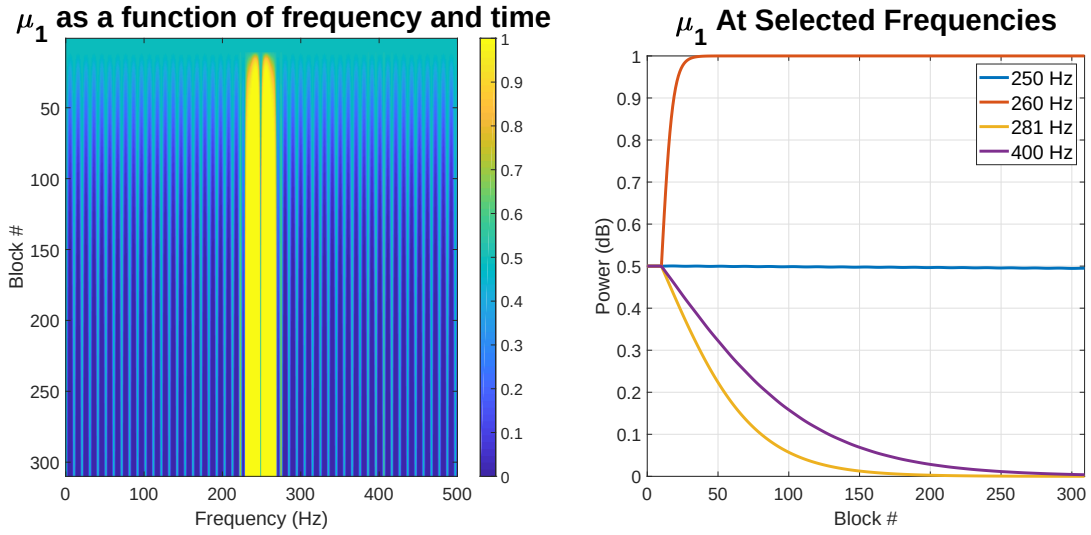

Figure 4.2: The PWB blend weights illustrate how the PWB chooses the appropriate estimator at each frequency of interest.

its improved side lobe performance reduces the bias of the estimate there.

This simulation also reveals the effect of the sensitivity parameter ν. At 260 Hz it takes 41 blocks after the complex exponentials are added for the blend weight to reach its final value of 1. In contrast at 400 Hz the blend weight takes the entire remainder of the simulation 300 blocks to reach its final value of 0. The reason for the difference is the sensitivity parameter ν. When the power of the estimator outputs is close to the value of ν then the algorithm will react more quickly, and conversely if the estimator outputs are much lower than ν the estimator will converge more slowly. In each case the bound established by Eq. 3.11 is met, however the regret term $\frac{2\nu ln(M)}{n}$ isn't related to the signal power so there is more relative error for quiet signals than for loud signals. It is very important to choose a value of ν that is low enough for the regret to be reduced fast enough to reveal the quiet components of the signal, while still respecting the lower bound in Eq. 3.26 required for convergence.

4.2 Stationary Simulations

The first of three simulations in this section consists of three complex exponential signals with random phase in white noise. The location in frequency space, and power of each signal is the same as the example in Section 2.4. The scenario is shown in Fig. 2.4. The signals and windows were chosen so that no single window in the ensemble would be able to return a spectrum where all the line components are clearly visible. The power of the noise is set to -6 dB below unity. There are two unity gain line components at 250 Hz, and the bin center nearest 258 Hz. Finally there is a third line component at the bin center nearest 400 Hz that is -20 dB below unity. The block length was 100 samples, and 51 blocks were generated with the first block used only to initialize the weights. Each estimate was calculated using the 1024 point DFT of the windowed signal. The outputs at the final block were averaged over 1000 Monte Carlo trials to create the final estimate. For these stationary simulations the sensitivity parameter was set to 1, and the sampling frequency was 1000 Hz. The windows in the ensemble are five Chebychev windows with peak side-lobe levels of -6 dB, -12 dB, -24 dB, -48 dB, and -96 dB. The -6 dB, -12 dB, are able to at least marginally resolve the line components at 250 Hz, and 258 Hz, however because of their poor peak side-lobe performance they are unable to detect the line component at 400 Hz. In contrast, the three windows with the lowest peak side-lobe levels are able to detect the line component at 400 Hz but they fail to resolve the signals at 250 Hz, and 258 Hz. The situation is illustrated in Fig. 4.3. The PWB processor is able to combine all five estimates into a single estimate where each line component is visible. Fig. 4.4 shows the spectrum estimate from the PWB processor as well as the estimates from each of the estimators in the ensemble.

Fig. 4.5 shows the same example with the noise power increased to 10 dB above unity. In this case all of the windows fail to resolve the two components at 250 Hz, and 258 Hz and they fail to detect the component at 400 Hz. The problem is that all of the windows lack enough coherent gain to eliminate the noise. The PWB estimator fails as well. The key insight is that when choosing the windows in the estimator ensemble the SNR in the

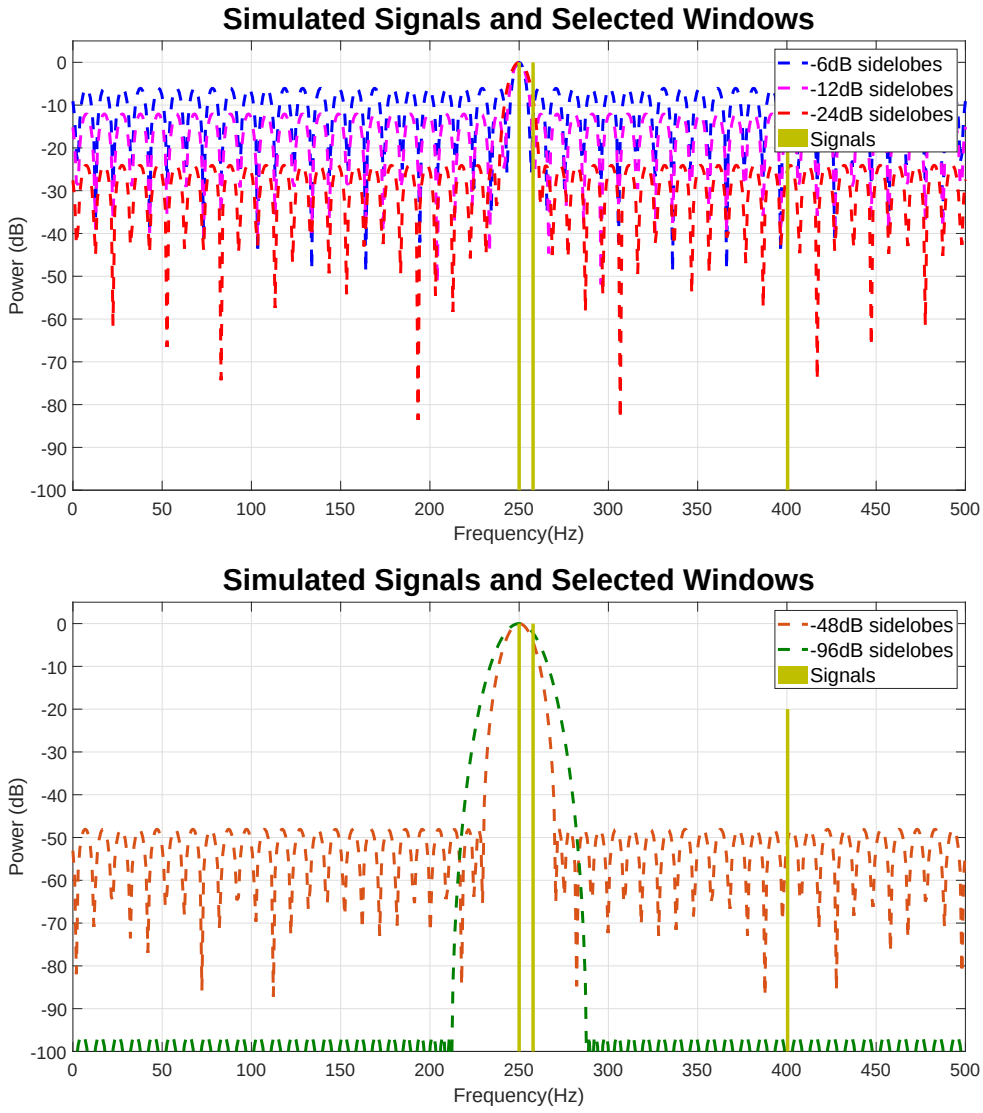

Figure 4.3: To show that no window in the ensemble can reveal all three line components the signal locations and powers are plotted on top of the responses of the windows at 250 Hz.

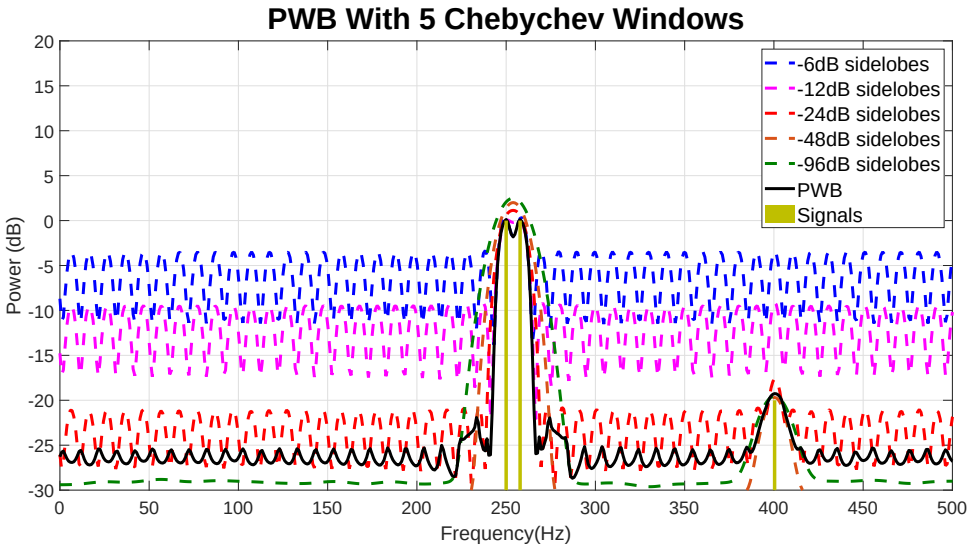

Figure 4.4: The PWB processor returns a spectrum that clearly contains three line components even though no single window in the ensemble can detect all three signals.

environment limits both the maximum resolution, and the maximum dynamic range that the PWB processor can achieve.

Fig. 4.6 shows the original example again however this time the power of the component at 258 Hz has been lowered to -20 dB below unity. The two signals at 250 Hz, and 258 Hz can no longer be resolved because the high resolution windows do not allow for enough dynamic range to see both components. This will be a problem for any window ensemble because there is a general trade off between resolution and side-lobe performance in window design. In order for the high resolution windows in the ensemble to resolve two signals, the dynamic range between them must be low. This implies that the PWB processor will always have difficulty resolving components that are close together in frequency space, and have dramatically different powers.

The final stationary example is shown in Fig. 4.7. Here the original simulation has been modified by increasing the power of the signal at 400 Hz to 20 dB above unity. The result is that the two signals at 250 Hz and 258 Hz are no longer resolved. The problem is that the high resolution windows have too much spectral leakage from the loud component to

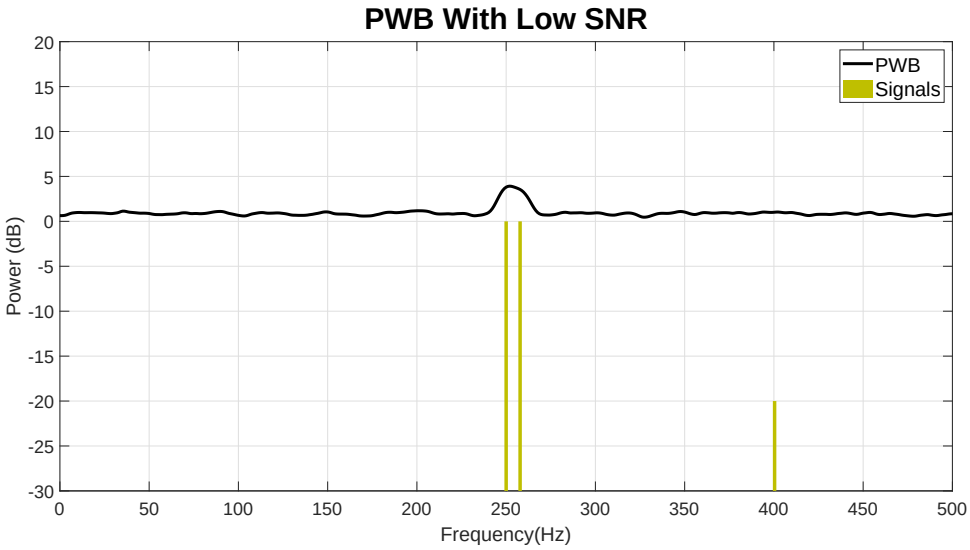

Figure 4.5: When the noise power is increased all of the windows in the ensemble fail and so does the PWB estimator.

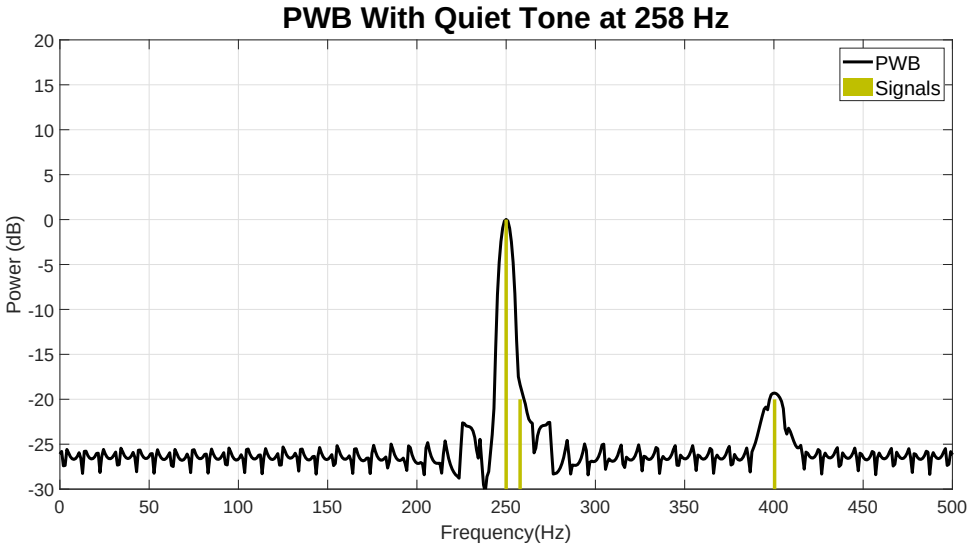

Figure 4.6: When the power of two line components that are close in frequency is not similar then the ability to resolve them is lost.

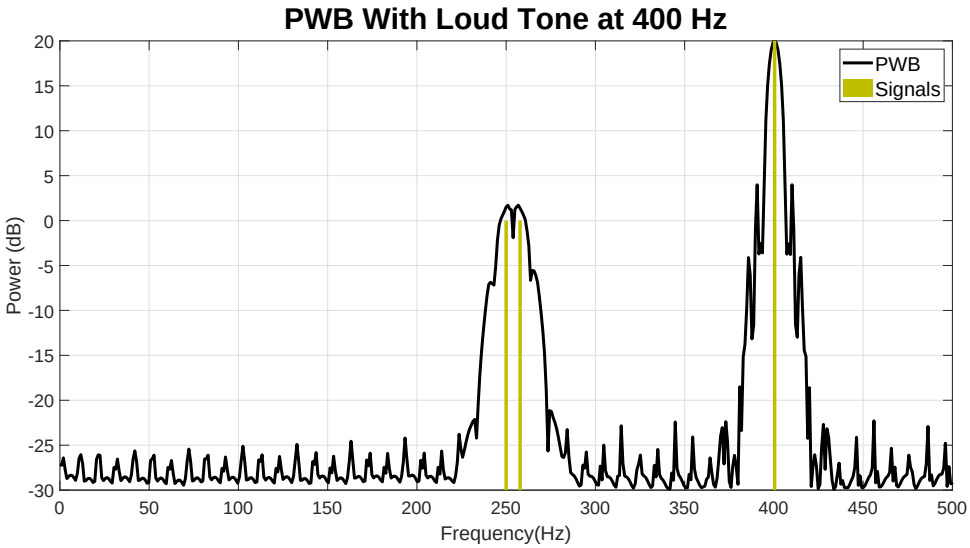

Figure 4.7: A loud line component can destroy the improved resolution of the PWB processor.

be able to identify the two components at 250 Hz, and 258 Hz at all so the only windows that identify those components are those without the ability to resolve them.

The performance of the PWB estimator is limited by the performance of the estimators in the ensemble. It is synthesizing the results of the estimators in the ensemble so if they all fail the PWB estimator will fail as well. When the environment is stationary this means that the PWB estimator will never outperform the best estimator in the ensemble.

4.3 Comparison to MPDR

The simulations in this section compare the PWB to both the ensemble MPDR and the sample MPDR with the bias correction. There are three sets of simulations. The first examines the PWB's ability to estimate a single signal in noise. The second simulation contains a signal with in noise and an interferer with varying power levels. The final simulation contains a signal with noise and the interferer has a fixed power and its location in frequency space is varied. The PWB ensemble consists of 5 windows, 4 Chebychev windows with peak side lobe levels of -6 dB, -12 dB, -24 dB, and -48 dB and a uniform window. The

simulated data is segmented into 51 blocks of 50 samples each. The simulations use a 512 point FFT and the signal has unity power and its frequency places it at the center of bin 129. In each case 1000 Monte Carlo trials were simulated for each value of the independent variable. The mean and variance of the power estimates at the final block and the bin that contains the signal from the PWB and sample MPDR are compared.

The results of the first simulation are shown in Fig. 4.8. The independent variable in this case is the noise power and it was varied from an SNR of -50 dB to 20 dB. The power outputs at the bin containing the signal show that the ensemble MPDR, the sample MPDR and the PWB perform similarly in terms of the estimate they produce. The variance plots show that the variance of the PWB estimator is lower than the sample MPDR estimate. Recall that when only a signal and noise are present the ensemble MPDR reduces to the uniform window, therefore the optimal estimator in this case is the uniform window. The PWB ensemble contains a uniform window so the PWB estimates are equivalent in this case to the ensemble MPDR but with the addition of the PWB regret term. The sample MPDR is limited by its ability to accurately estimate the sample covariance matrix. Given enough blocks the sample MPDR will eventually converge to the ensemble MPDR estimate. The regret term of the PWB also vanishes with increased numbers of data blocks. This simulation indicates that the PWB is able to converge to the ensemble MPDR more quickly than the sample MPDR.

For the second simulation the SNR is fixed at 0 dB and an interferer signal has been added at bin 50. The power of the interferer signal was varied from -10 dB below the signal power to 40 dB above the signal power. The power and variance plots are shown in Fig. 4.9. This time the power estimates from each of the estimators in the PWB ensemble have been included to illustrate how the PWB algorithm fails. As the power of the interferer signal is increased, the estimators in the PWB ensemble fail one at a time. The PWB estimator performs similarly to the ensemble MPDR until the interferer power is increased to the point where none of the estimators in the PWB ensemble can reject the interference. Before this point, the PWB estimate has less variance than the sample MPDR. It is also

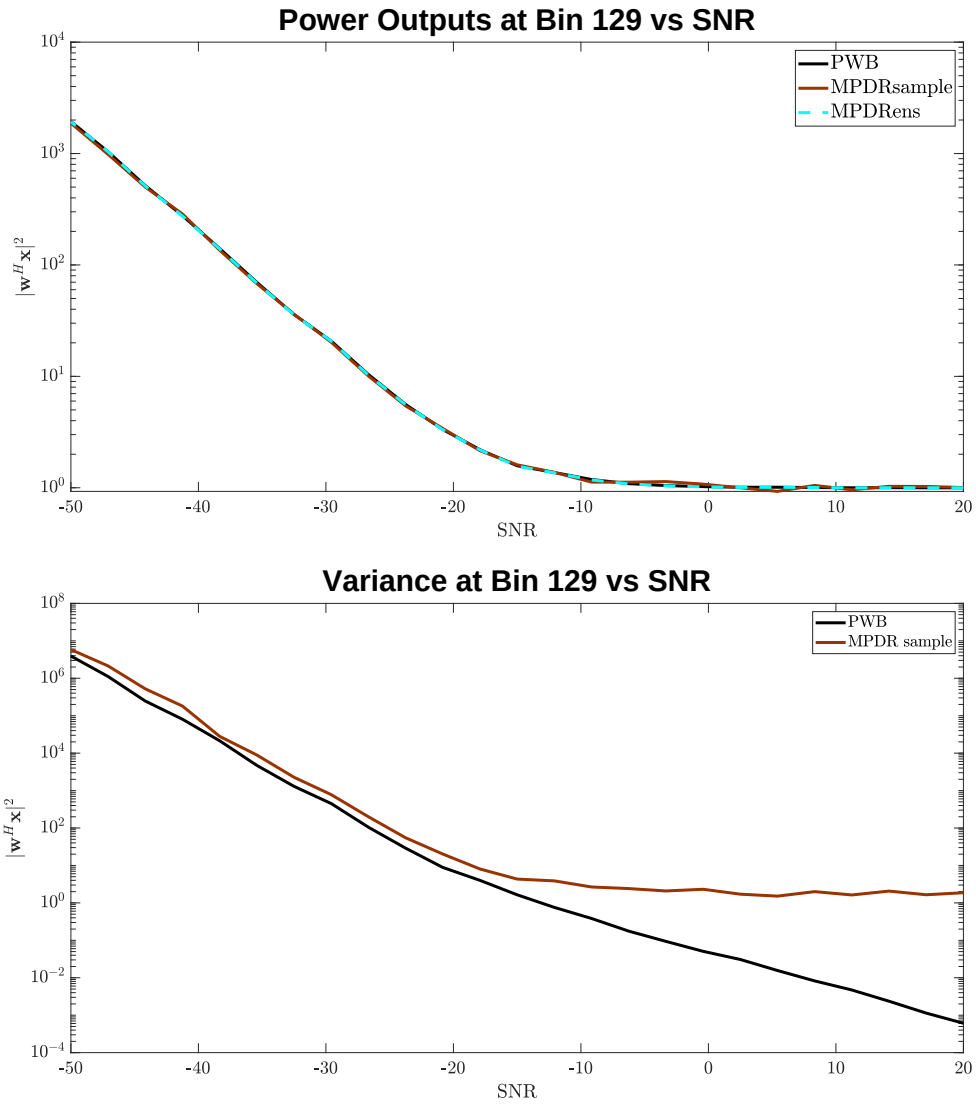

Figure 4.8: As the SNR is increased the PWB estimate is similar to the ensemble MPDR estimate and it has lower variance than the sample MPDR estimate

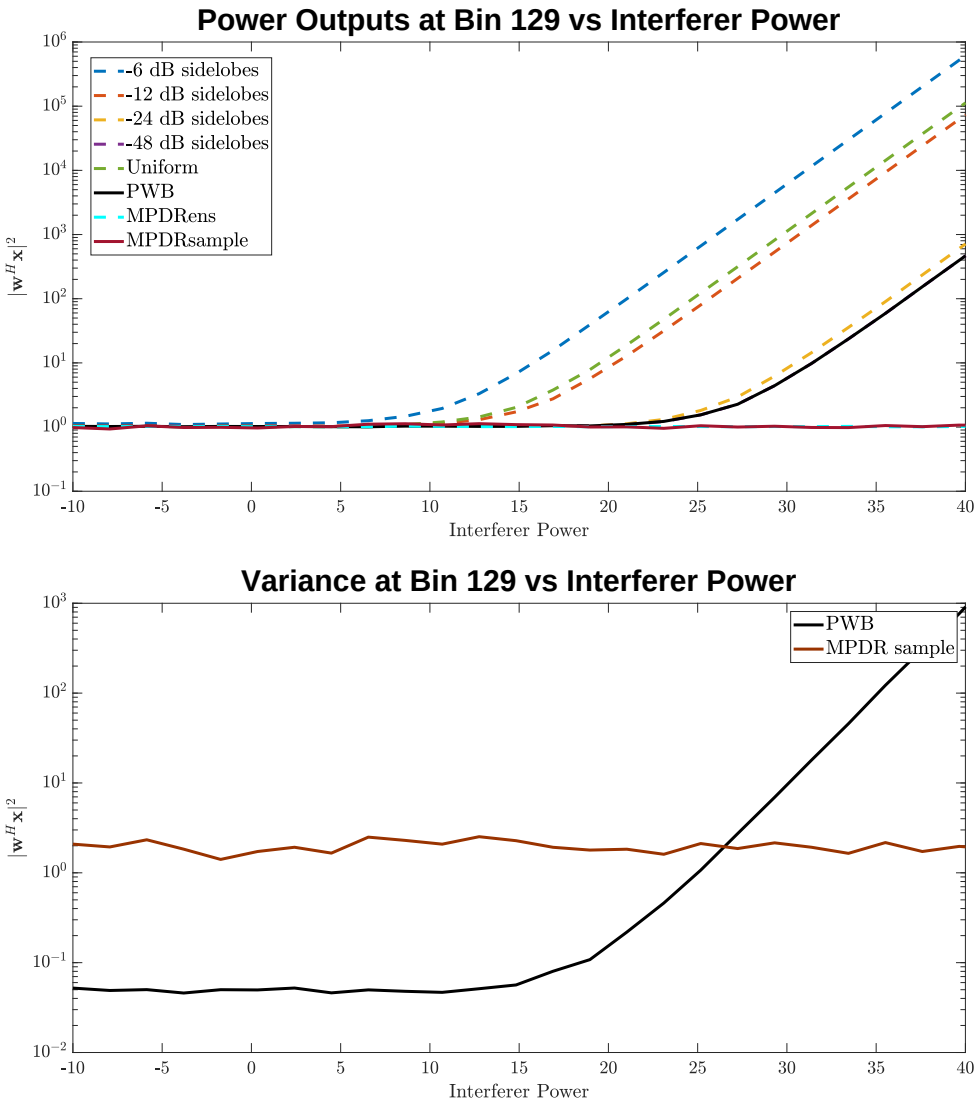

Figure 4.9: As the power of an interferer is increased the windows in the PWB ensemble fail one by one. The PWB estimate is similar to the ensemble MPDR estimate and has lower variance than the sample MPDR estimate until all of the windows in the estimator ensemble have failed

important to notice that if more interferer rejection is required, another window with lower side lobes could be added to the estimator ensemble.

The final simulation in this section fixes the SNR at 0 dB, and fixes the power of the interferer signal at 20 dB. The location of the interferer is then varied from bin 50 to bin 129 where the signal resides. This simulation highlights the trade that the PWB makes between side lobe performance and resolution. When the interferer is far from the signal the PWB weights estimators with lower side lobes more heavily and is able to approximate the performance of the ensemble MPDR. However when the interferer gets close to the signal bin the PWB weights the estimators with high resolution more heavily. The result is that the bias and variance of the estimates is increased because of the sacrifice in side lobe performance.

Overall when the ensemble is chosen appropriately the PWB algorithm is able to provide a better approximation of the ensemble MPDR's performance than the sample MPDR. This is especially true when there are few blocks of data available for the calculation of the sample covariance matrix. The PWB estimator only requires one block of data to initialize its blend weights. In contrast the sample MPDR requires at least $N + 1$ samples to guarantee an invertible estimate of the covariance matrix, and even then the variance of the estimates is greater than those produces by the PWB algorithm. This is an important feature of the PWB estimator because in practical situations there is not always enough data available for the sample MPDR to return a reliable estimate. In the next set of simulations the performance of the PWB in a non-stationary simulation is examined. In the non-stationary case the sample MPDR would never be able to satisfactorily estimate a sample covariance matrix however, the PWB is still able to return an estimate.

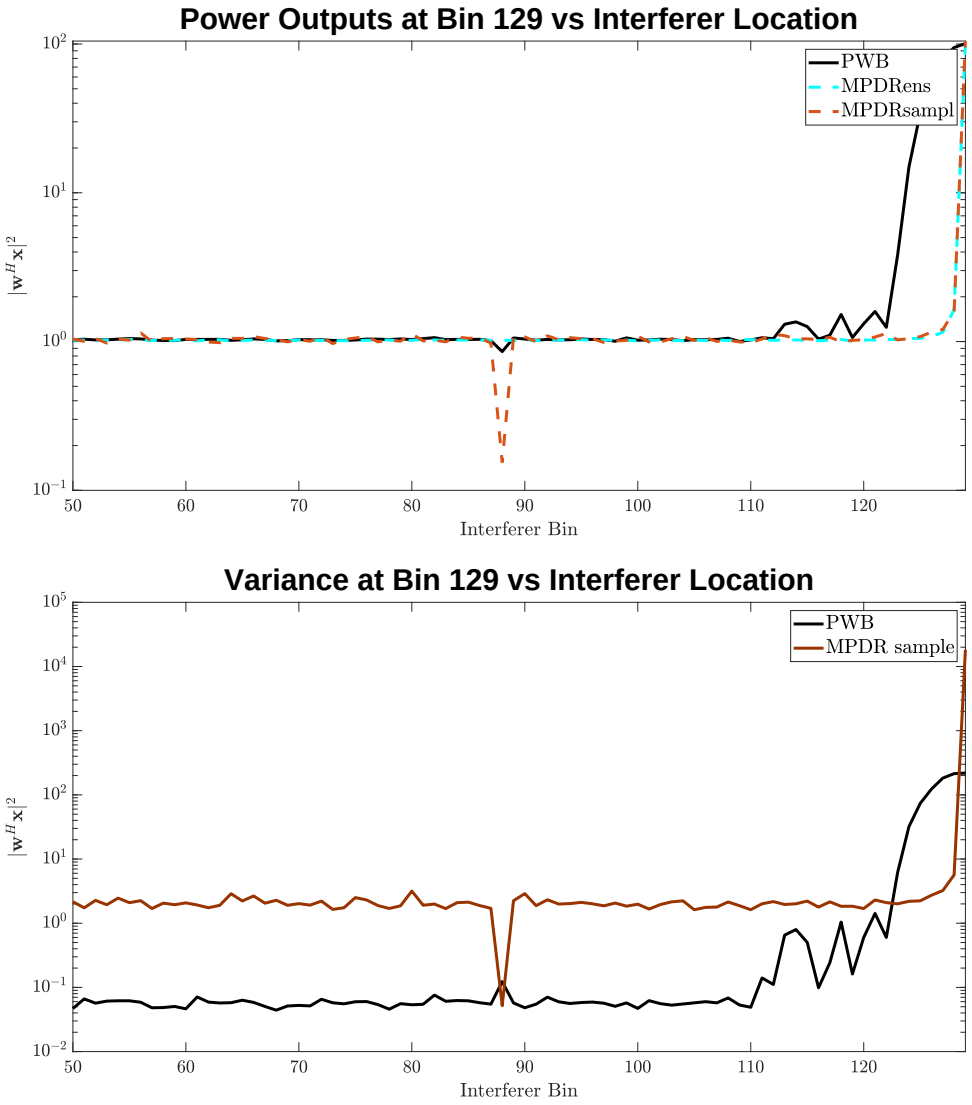

Figure 4.10: The improved side lobe performance of the PWB algorithm is traded for improved resolution as an interferer signal is moved towards the frequency of interest. Until this occurs the PWB estimate has lower variance than the sample MPDR estimate.

4.4 Non-Stationary Trials

The final simulation explores the performance of the PWB algorithm in a non-stationary environment. The simulation includes two stationary line components, a line component with varying power, and noise with unity power. The stationary line components are at 250 Hz with a power 10 dB above unity, and 265 Hz with a power 15 dB above unity. The varying line component is at 400 Hz with power that varies from 0 dB to 50 dB. The simulation was run for a total of 1001 blocks with the first block used to initialize the PWB weights. The window ensemble consisted of only two Chebychev windows. The windows have peak side-lobe levels of -15 dB, and -100 dB. The top left plot in Fig 4.11 shows a spectrogram generated by the window with -15 dB peak side-lobes. At first the window is able to resolve the components at 250 Hz, and 265 Hz, however as the interferer gets loud both signals are lost to side-lobe leakage. On the other hand Fig. 4.11 also shows the spectrogram produced by the -100 dB window. In this case the window is never able to resolve the components at 250 Hz, and 265 Hz, however it has low enough side-lobe leakage that the two unresolved signals are not covered by the loud interferer. Finally, the bottom plot Fig. 4.11 shows the spectrogram produced by the PWB processor. The PWB estimator is able to resolve the two stationary signals at first but it loses the resolution as the interferer at 400 Hz gets loud. Even then the PWB estimator is still able to use the resolution of the -15 dB side-lobe Chebychev window in the local region around where the interferer is located. So the PWB processor is effectively combining estimates from the two sub-estimators to generate the best possible final output. This is further confirmed by Fig. 4.12 which shows the accumulated loss of the PWB, and each sub-estimator throughout the trial at the 250 Hz bin. The loss of the PWB processor closely tracks the loss of the best performing sub-estimator throughout the simulation.

In summary, the simulations have shown that when the window ensemble is appropriate the PWB estimator is able to display high resolution in regions of the spectrum where there are closely spaced line components, and improved side lobe performance in regions where there are quiet components subject to interference by loud components in other

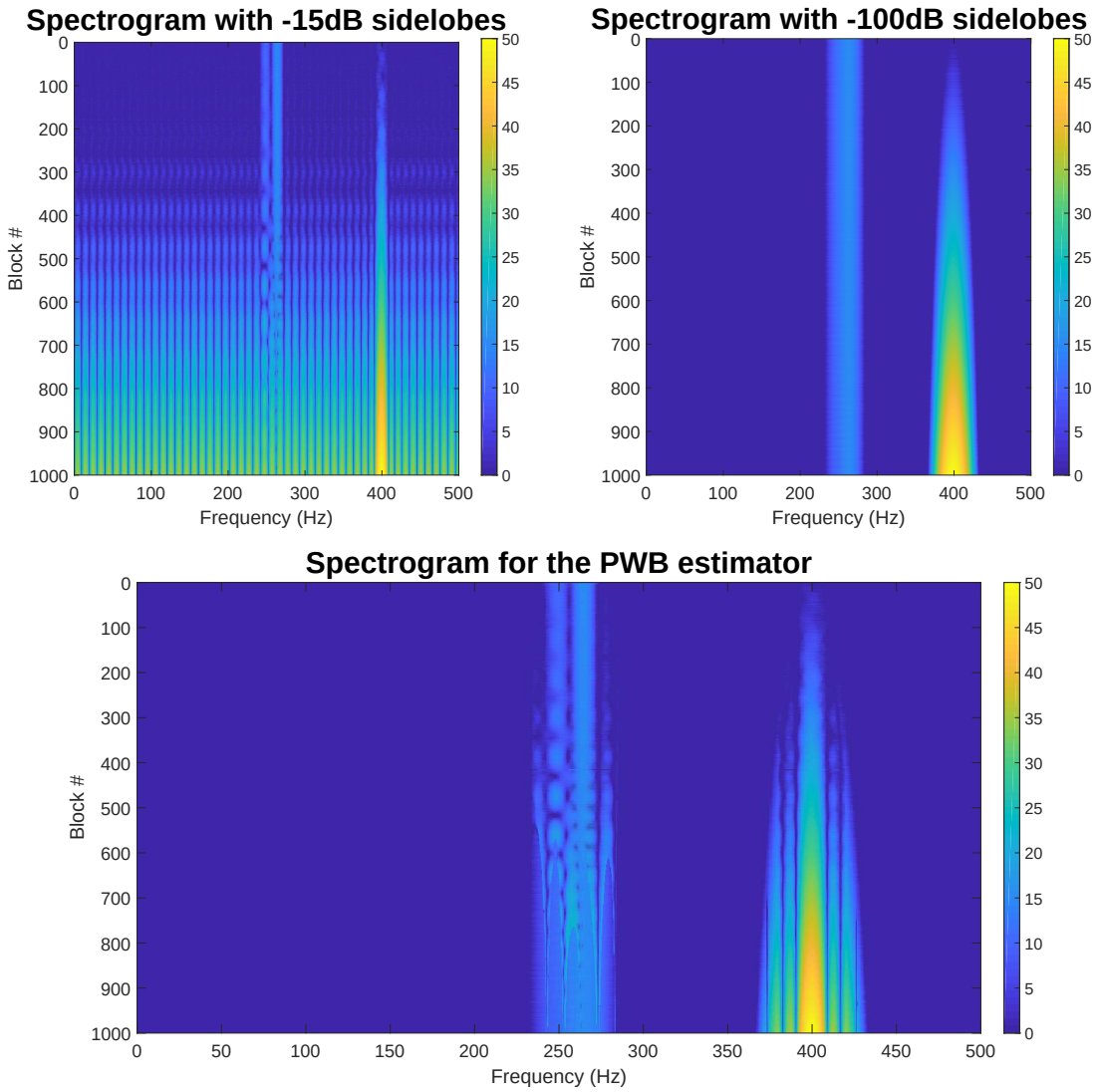

Figure 4.11: The spectrogram produced by a single Chebychev window with -15 dB peak side-lobe cannot reveal the quiet signals as the tone at 400 H gets loud. The spectrogram produced by a single Chebychev window with -100 dB peak side-lobe cannot resolve the two signals at 250 Hz and 258 Hz. The spectrogram produced by the PWB estimator reveals the two quiet signals and though it can no longer resolve them when the 400 Hz signal gets loud it still reveals that there is content in that region.

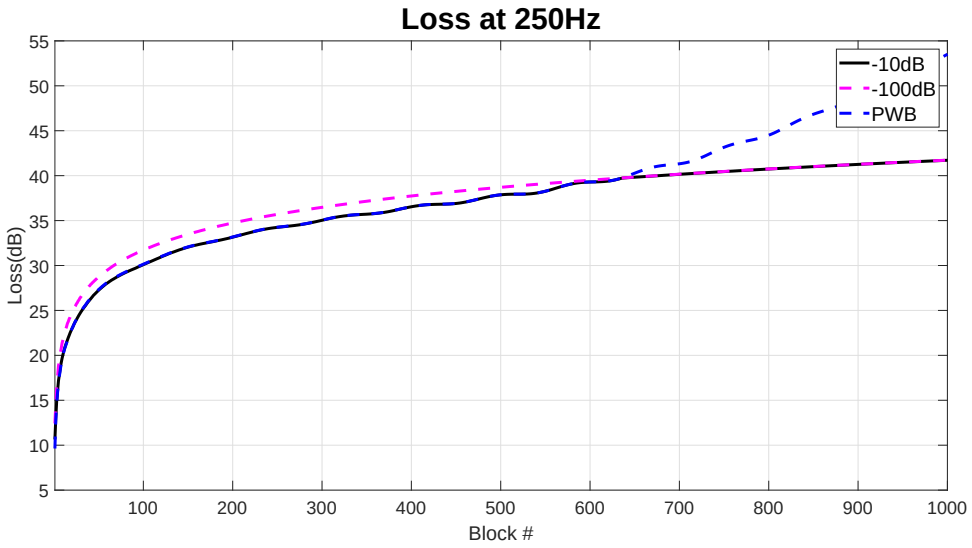

Figure 4.12: The loss of the PWB estimator is similar to the loss of the best performing window at each instance

regions of the spectrum. The simulations also show that the PWB algorithm is able to provide a better approximation of the ensemble MPDR than the sample MPDR, especially when there are not enough blocks of data to get a good estimate of the covariance matrix. Finally, simulations in a non-stationary environment show that the PWB is able to adapt to a changing environment. In general the performance of the PWB algorithm is limited by the performance of the windows in the ensemble. The next chapter proposes a procedure for choosing an appropriate window ensemble.

Chapter 5: Design Guide

In order to design an effective PWB estimator there are several choices to be made. The first is how to segment the data record. This is a well understood problem, and represents a trade between increased resolution gained by using longer blocks of data, and reduced variance of the estimate by using shorter blocks that allow for more time averaging [1]. The second choice is the window ensemble. The PWB spectrum estimator is simply a linear combination of an ensemble of sub-estimators. Once the block length has been selected the performance is completely determined by the windows in the ensemble, and the choice of the sensitivity parameter. It is very important when designing the estimator ensemble that at least one window in the ensemble will give an acceptable estimate at all times. It is also important to notice that the guarantee placed on the average loss of the PWB estimator depends on the number of windows in the ensemble, so there is a cost associated with the size of the window ensemble. In order to design a good estimator, we must determine how many windows to include in the ensemble, as well as which windows to include. To get the lowest possible average loss, the sensitivity parameter should be set as close as possible to the constraint placed on it by the convexity requirement. It should be noted that it is entirely possible to use different window ensembles, and different values of ν for each frequency of interest in the estimate, however for simplicity and clarity that scenario is not considered in this work.

Chebychev windows are a natural choice for the windows in the ensemble because their defining characteristic is maximum resolution for a given peak side-lobe level. Figure 5.1 contains a plot of the peak side-lobe level versus the null to null beamwidth for Chebychev windows with several other canonical windows included for reference. The trade off between side-lobe level and resolution is a very important consideration because it determines a window's ability to resolve line components that are close together, and the window's ability

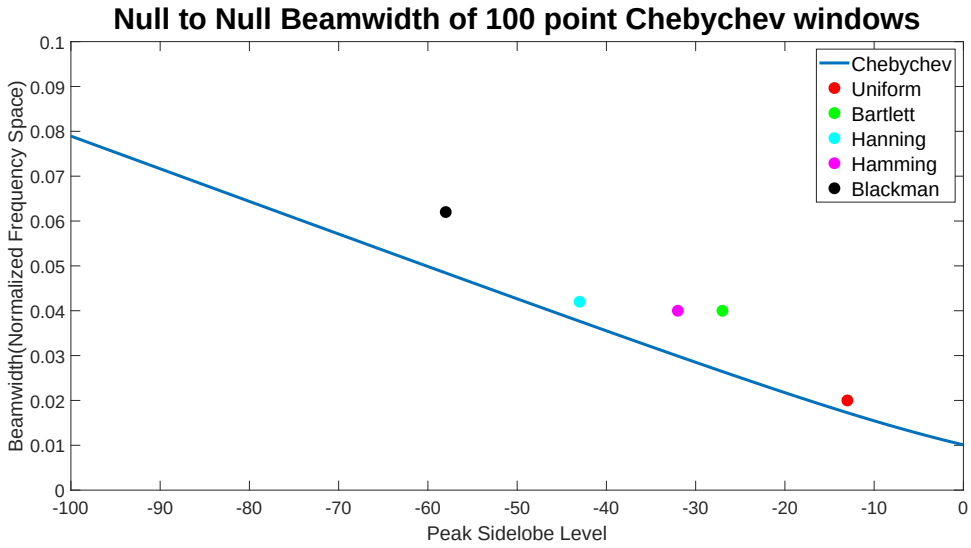

Figure 5.1: The Chebychev window offers the best possible resolution for a given side-lobe level

to reject interference from loud line components. The other important consideration is coherent gain which reflects a window's ability to reject white noise. Fig. 5.2 contains a plot of the coherent gain of Chebychev windows and other canonical windows vs. peak side-lobe level. The coherent gain of the uniform window significantly outperforms the Chebychev window and its side lobes decay unlike the Chebychev window. Therefore, it is useful to include the uniform window in an ensemble that otherwise consists only of Chebychev windows.

Having determined the quality of the windows in the ensemble i.e., Chebychev windows, and a uniform window, the next step is to determine how many Chebychev windows to include, and how to set their peak side-lobe levels. The first consideration is the amount of noise in the environment. The coherent gain property of a window determines how much noise it can reject from the environment so regardless of resolution and side lobe level if a window cannot reject enough noise to reveal the signals it will fail. It is recommended to use an estimate of the SNR that the algorithm will encounter to place upper and lower bounds on the peak side lobe levels of the Chebychev windows in the ensemble. Fig. 5.2 can

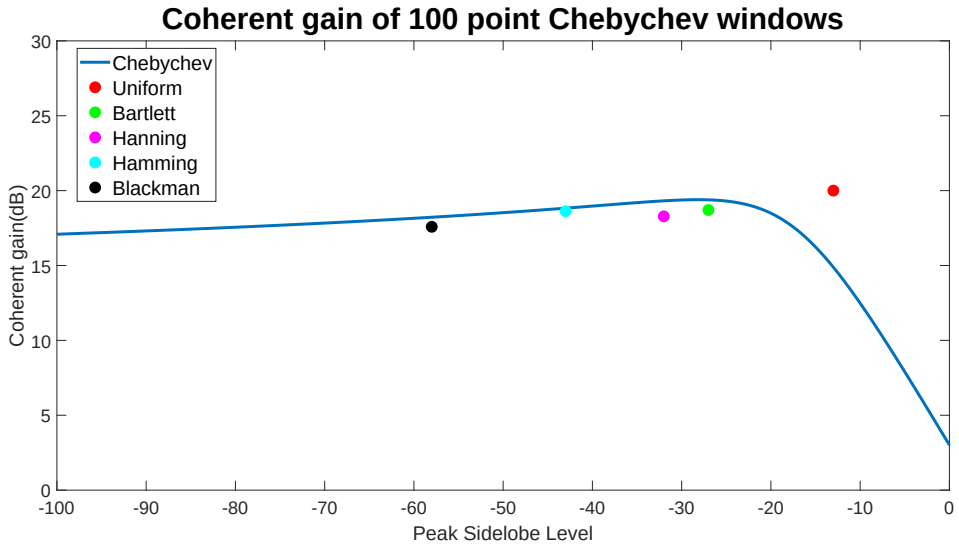

Figure 5.2: The Chebychev window offers good coherent gain or equivalently white noise gain for a given side-lobe level but it is outperformed by a uniform window.

be used to find appropriate endpoints for a given SNR. The endpoints can then be further refined by considering the maximum power of interfering line components if available and the maximum resolution required. Fig. 5.1 can be used to determine maximum and minimum peak side-lobe levels that give the required resolution and side lobe performance.

Having selected the two endpoint windows the final step is to choose the number of windows with intermediate peak side lobe levels that will be in the ensemble, and where to set their peak side lobe levels. The upper bound on the average loss of the PWB estimator increases monotonically with the number of windows in the ensemble. Therefore, the smaller the ensemble, the less average loss the algorithm will have. The downside of a small window ensemble is there are fewer estimates for the PWB algorithm to chose from so it is more likely that all windows will fail to give a good estimate. In simulations, and preliminary work with real data it was found that evenly spacing the side lobe levels of Chebychev windows within the range determined by the SNR, and side-lobe requirements provided acceptable results. However, the performance of a particular ensemble is very much dependent on the characteristics of the environment.

Chapter 6: Experimental Results

This chapter examines the use of the PWB algorithm on data from a hydrophone on an underwater glider. The first section presents some alterations to the algorithm that improve robustness. In the second section the PWB algorithm is tested on the actual hydrophone data. The experiments show that with the improvements the PWB algorithm can outperform each of the fixed window estimators in its ensemble.

6.1 Improving robustness

A problem was encountered using the PWB processor for power spectrum estimation with hydrophone data from an underwater glider. The audio from the glider was intermittently corrupted by mechanical noise from the glider's machinery, which was much louder than the signals being searched for in the data. This poses a problem for the PWB processor because when the sensitivity parameter is tuned for the quiet signals, the restrictions on the parameter ν are grossly violated. The loss is accumulated from the beginning of the data-record so the processor retains memory of the loud transients and chooses the wrong processor even after the transients have disappeared. Figure 6.1 shows a simulated example of the problem. In the example there is a complicated environment with two closely spaced unity gain signals at 250 Hz and 257 Hz, two quiet signals with gain at -20 dB below unity and frequencies of 225 Hz, and 400 Hz. There is a chirp signal at -10 dB below unity gain that moves linearly from 100 Hz to 400 Hz and there is white noise with a power of -20 dB below unity gain. The problem signal is a pulse that occurs at 100 Hz and 100 dB above unity gain but only for blocks 500 through 525. Also note that the pulse looks a broadband signal in the spectrogram, though it is a complex exponential pulse at 100 Hz, because its power leaks into all of the other bins. The sensitivity parameter is set to unity which is

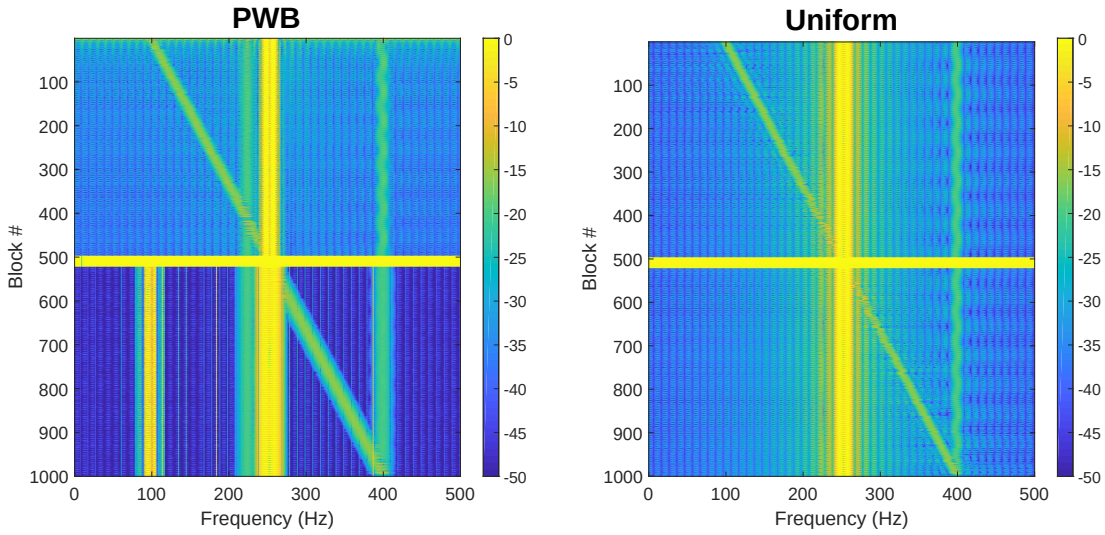

Figure 6.1: A very loud pulse at 100 Hz and 100 dB from blocks 500 to 525 causes the processor to malfunction. The output of a uniform window processor is shown for reference.

approximately correct for the unity gain signals, however it is too low for the loud pulse. The violation of the conditions for ν causes ghost signals in the PWB spectrogram. If we had set the sensitivity parameter to 10^{10} which would be appropriate for the transient, then the parts of the spectrum before and after the loud transient would just be a simple average of the windows in the ensemble.

There are two proposed solutions to solving the problem of the loud transients. The first is altering the loss function so that it accumulates over a sliding window rather than for all time. The second solution is limiting the maximum value that the loss can take. Simulations have shown that the most robust solution is to use both of the proposed methods simultaneously.

The insight behind using a sliding window to accumulate loss is two-fold. The original reason for exploring this idea was to solve numerical problems in calculating the performance weights. The accumulated loss increases monotonically so without renormalization techniques it will eventually grow too large for our processors to calculate the performance weights. Limiting the loss accumulation to a sliding window of a fixed length also limits the

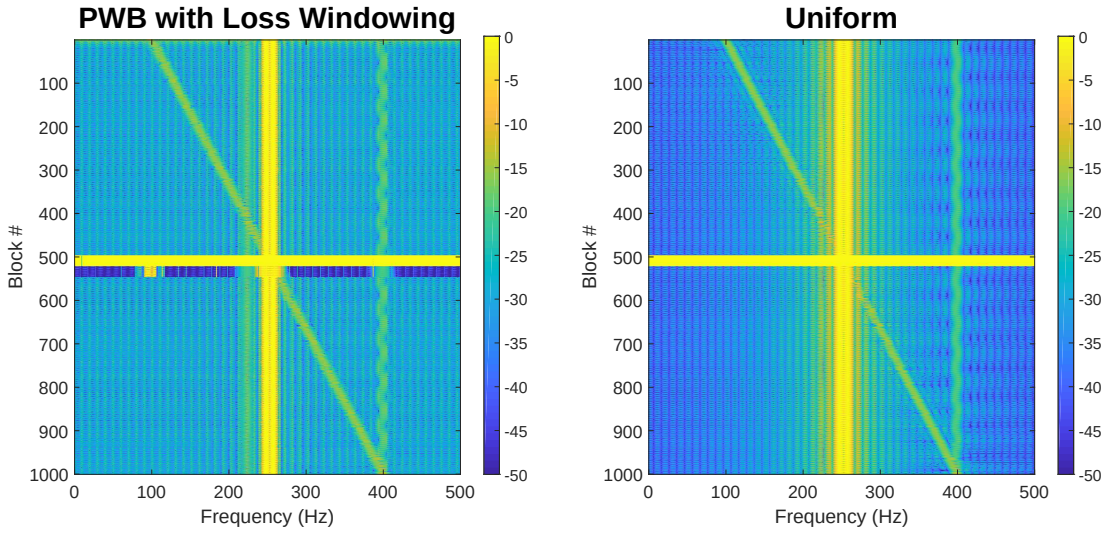

Figure 6.2: Accumulating the loss over a sliding window confines the disruption caused by the transient to a short section of the spectrogram. The output of a uniform window processor is shown for reference.

maximum accumulated loss, which solves the numerical issue. For the purposes of solving the transient problem the sliding window allows the PWB processor to "forget" that the offending transient occurred. The ghost signals still appear, however they only last for the duration of the window. Using the sliding window contains the impact of the loud transients to a smaller region of the spectrogram output. Fig 6.2 shows the same example as Fig. 6.1 however this time it is processed with the loss accumulated only over 25 blocks.

Limiting the accumulation of the loss has a cost. The convergence of the average loss of the PWB processor is a function of the number of blocks used to average over as shown in Eq. 3.11. When we choose the length of the sliding window, we are also fixing the value of the regret term $\frac{2\nu ln(M)}{n}$. Setting the window length too short increases the regret in the PWB output, but setting it too long increases the effects of transients in the signal.

An alternative approach is to limit the maximum value of the loss for each block so that

the new loss function is

$$\mathcal{L}_n(\mathbf{w}, \mathbf{x}) = \sum_{\tau=1}^{n} \min(\gamma, |\mathbf{w}[\tau]^H \mathbf{x}[\tau]|^2), \tag{6.1}$$

where γ represents the maximum possible contribution of a single block to the total loss.

The new definition of loss is used to calculate the performance weights. The sub-estimator outputs are not limited so the resulting spectrum does not have a reduced dynamic range. Choosing the threshold so that $\gamma = \nu$ ensures that the condition required for convexity is never violated. Another way to think about the threshold parameter is as an upper limit on the universality of the PWB estimator. When a narrowband signal with a power above the threshold is present over the entire loss calculation then the output at that frequency is a simple average of the window ensemble outputs and the PWB is no longer universal. The benefit is that the processor is no longer able to overreact to loud transients that violate the restriction on ν. It is tempting to use this method to further reduce the value of ν below the value required by the original proof, however the average of the window ensemble is generally not as good an estimator as the PWB estimator, so if the threshold and corresponding sensitivity parameter are set too low then the output simply becomes an average of the ensemble windows, and we lose the benefit of the performance weighting. Fig. 6.3 shows the same example as earlier however this time the loss is limited but accumulated over the entire simulation. The result of the limiting process is that the effect of the loud transient is mitigated but without the short region with ghost signals seen in the sliding window solution.

A practical PWB spectrum estimator for deployment in an underwater glider will likely need to include both the loss thresholding technique, as well as a sliding window. The loss thresholding technique prevents the estimator from being overwhelmed by loud on-board machinery that doesn't run all the time. The sliding window acts as a second line of defense against loud transients that are below the chosen threshold. If the threshold has been set too high the sliding widow allows the processor to forget a loud signal, and restores the

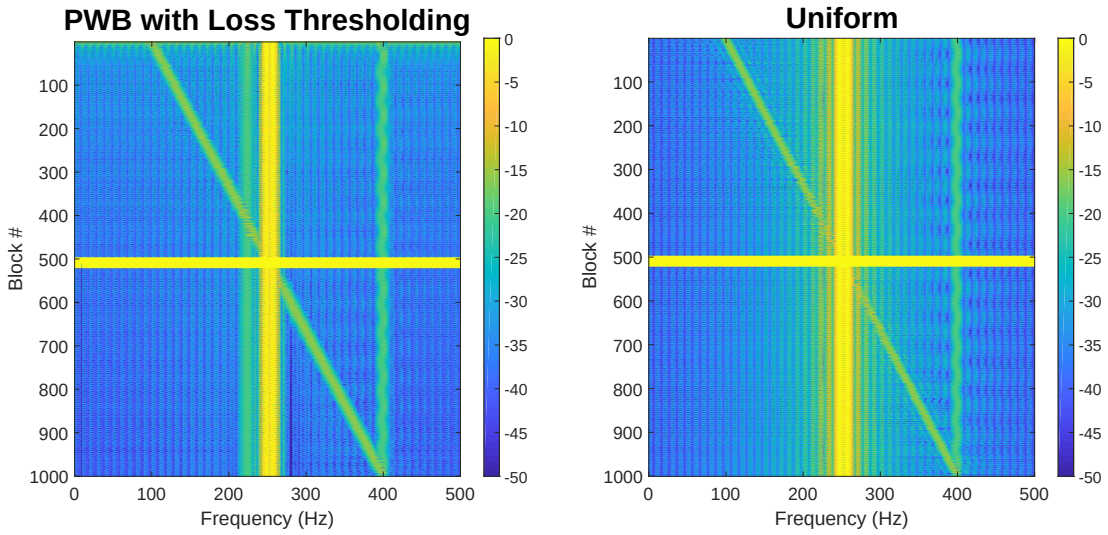

Figure 6.3: Limiting the maximum loss for each block mitigates the effects of a loud transient without simply containing the problem within a sliding window. The output of a uniform window processor is shown for reference.

system to normal operation. Fig. 6.4 shows the output of the same example with a sliding window of 100 blocks, and loss threshold of 1. The result is that we get a lower regret compared to the 25 block sliding window used in the previous example, and we get the transient mitigation from the loss thresholding.

6.2 Experimental results

The PWB algorithm was tested using hydrophone data from a deep water underwater glider experiment near the Scotian Slope off the coast of Nova Scotia in the northern Atlantic. A 25 second segment of data was selected. The data contains a 23 second linear frequency modulated (LFM) signal that began at 20 Hz and increased in frequency to 250 Hz. The power of the LFM signal is roughly -75 dB below unity. The data also contains a section of loud noise from the the machinery of the glider. The original data was sampled at 128 kHz, and it was decimated to a sampling frequency of 4 kHz to reduce the computations required for the analysis. The DC components of the signal were also removed by subtracting the

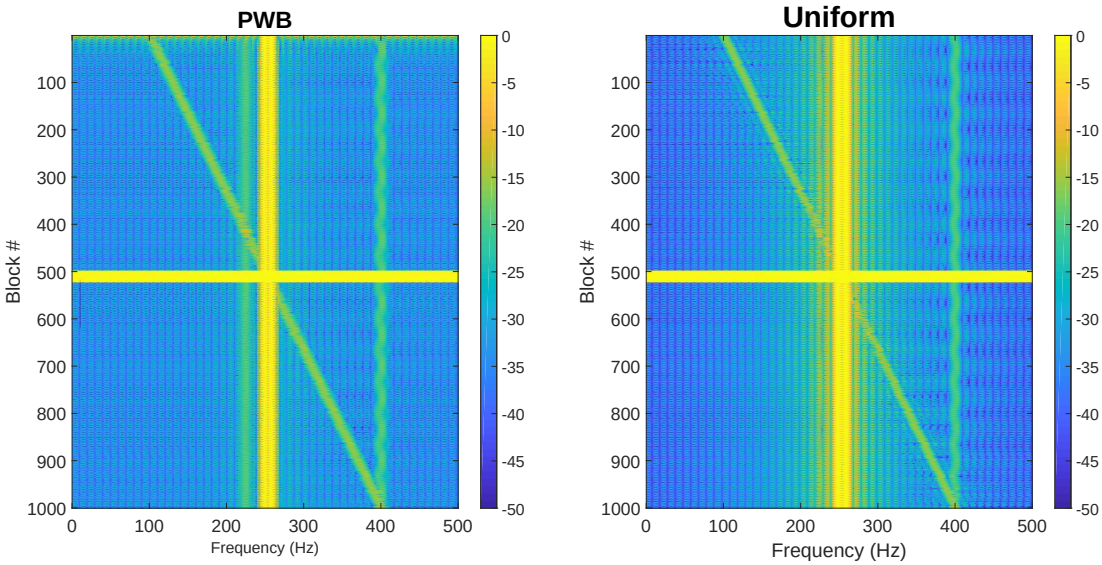

Figure 6.4: Applying both loss limiting, and a sliding window gets the benefits of both methods. The output of a uniform window processor is shown for reference

sample mean of the entire data record [20]. Fig. 6.5 has spectrograms of the data created using a block length of 1000 samples with 75% overlap, which results in 397 total data blocks. The blocks were processed using an 8192 point DFT. For the PWB algorithm, the sensitivity parameter was set to 2×10^{-7}, and the threshold was set to 3×10^{-7}. The PWB window ensemble contains a uniform window as well as Chebychev windows with peak side-lobe levels of -25 dB, -50 dB, -100 dB, and -125 dB. All of these parameters were tuned through experimentation with the data.

The data segment was injected with a 150 Hz complex exponential signal. Fig. 6.6 shows spectrograms created with the PWB algorithm and with a fixed uniform window when the exponential signal is at -35 dB below unity. Visual inspection of the resulting spectrograms show that the PWB algorithm is able to reveal the LFM signal even when the uniform window fails.

When the injected signal is loud it serves as an interferer for the LFM signal. Fig. 6.7 shows two slices of the PWB spectrogram. In the first slice the glider machinery is not running and both the LFM and interferer signals are visible. In the second slice the glider

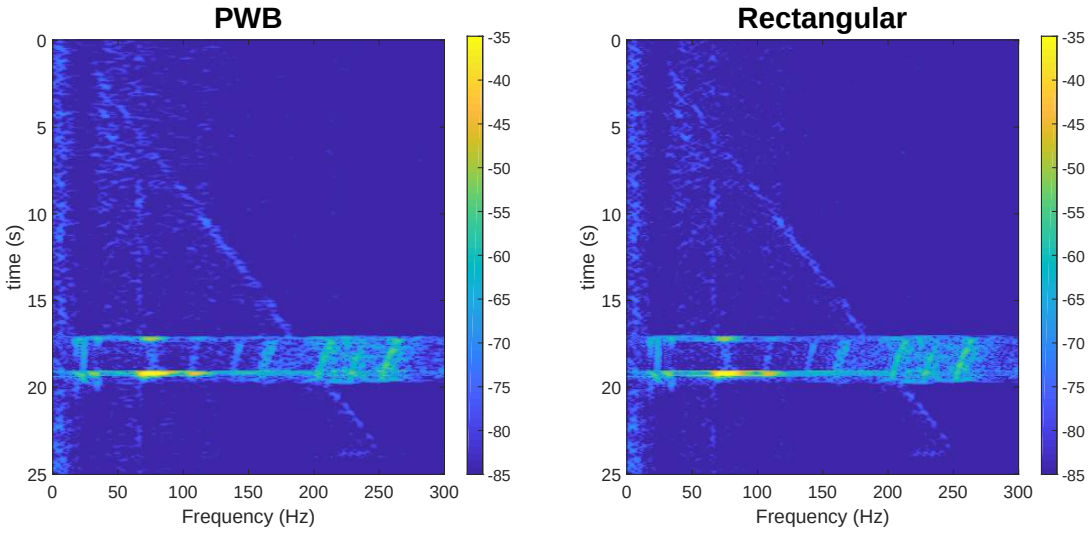

Figure 6.5: The data segment used for the experiment contains an LFM sweep and is corrupted by noise from the glider.

machinery is running which creates a couple loud tonal components, and broadband noise. In this case the interferer signal is still clearly visible and the LFM signal is marginally visible. Slices from the spectrogram produced by a uniform window, and a Chebychev window with -125 dB peak side-lobes are included for reference. It should be noted that in the first slice the PWB algorithm closely tracks the Chebychev window because of the need to suppress side-lobe leakage. In the second slice both estimators fail to definitively reveal the LFM signal therefore the PWB algorithm is also unable to definitively reveal that signal.

When the injected signal is roughly the same power as the LFM signal the PWB algorithm tends to prefer the uniform window. This is because the priority becomes noise reduction rather than reduction of side-lobe leakage. The uniform weighting has the best coherent gain [4] so it does a better job of reducing the noise and is preferred by the algorithm. The two plots for this example are shown in Fig. 6.8.

The previous examples have qualitatively shown the performance of the algorithm. The performance is quantified in two ways. The first is to observe the regret of the PWB

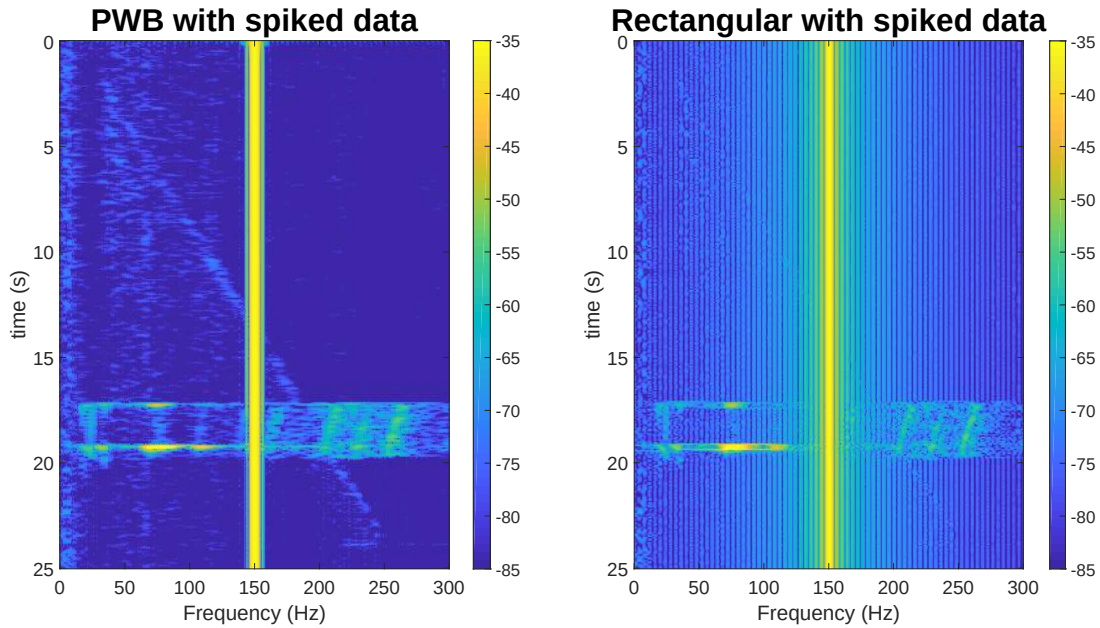

Figure 6.6: The PWB algorithm reveals the LFM signal when the uniform window fails due to a loud interferer.

estimator and compare it to the analytic bound. The second is the output SNR of the PWB estimator will be compared to the output SNR of each of the estimators in the ensemble.

Fig. 6.9 shows the regret at the final data block; of the PWB estimator as a function of the power of the injected signal. The lower plot in Fig. 6.9 shows a detail of the portion of Fig. 6.9 in the black rectangle. In both cases the red vertical line shows the point at which the injected signal power alone is enough to violate the condition on the sensitivity parameter. The regret is shown for 150 Hz where the injected signal is located, and also at 50 Hz where there is no signal. The plots show that as long as the restriction on ν is respected the algorithm performs as advertised in terms of regret. Even when the conditions on ν are violated the regret only increases at the frequency of interest where the condition is violated. That is why the 50 Hz curve remains below the analytic limit as the power of the injected signal is increased.

The final evaluation of the PWB algorithm is to examine the output SNR in comparison

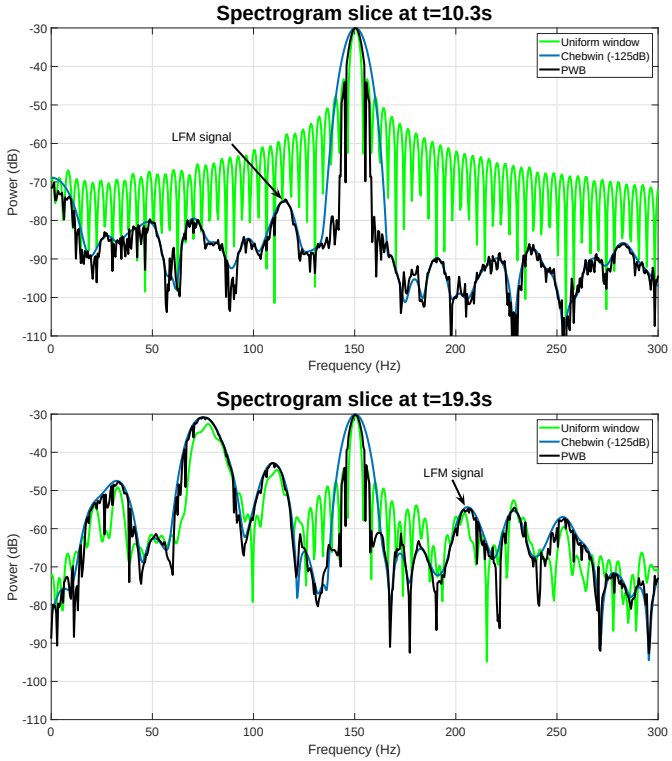

Figure 6.7: When a -30 dB interferer signal is injected the PWB algorithm is able to reveal the LFM signal.

to the fixed window estimators in the ensemble. Fig. 6.10 shows the PWB estimator and each of the estimators in the ensemble. The output SNR was calculated as

$$\text{SNR}_{\text{out}} = 10 * log10 \left(\frac{\hat{X}_s - \hat{X}_n}{\hat{X}_n} \right) \tag{6.2}$$

where $\hat{X}_s$ is the estimate at the last block with the injected signal, and $\hat{X}_n$ is the estimate at the last block of the unaltered signal. The resulting plot shows that even after the conditions on the ν are violated the PWB estimator is still able to relatively closely match the performance of the best estimator in the ensemble in terms of output SNR.

In summary the PWB estimator is able to outperform each of the fixed window estimators in its ensemble however, it requires protection from loud transient signals in the

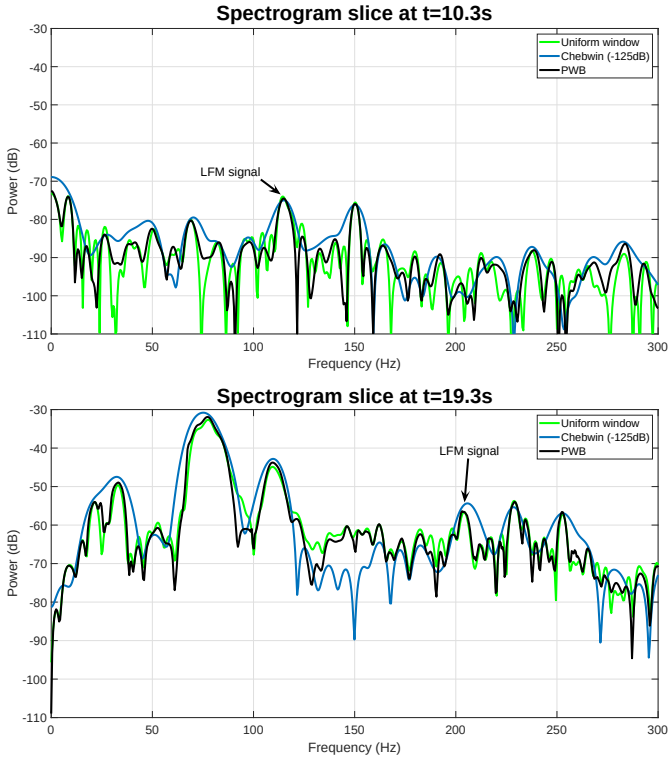

Figure 6.8: When a -75 dB interferer signal is injected the PWB algorithm is able to reveal the LFM signal.

environment. The protection can be provided by applying a sliding window, and thresholding the loss function. When the convexity conditions on the sensitivity parameter are not violated the PWB estimator has a regret that is less than or equal to $\frac{2\nu ln(M)}{n}$. The output SNR of the PWB estimator is also similar to the output SNR of the best performing fixed window estimator in the ensemble. The PWB is a very promising estimator for situations where there is no single fixed window estimator that can reliably return an acceptable estimate.

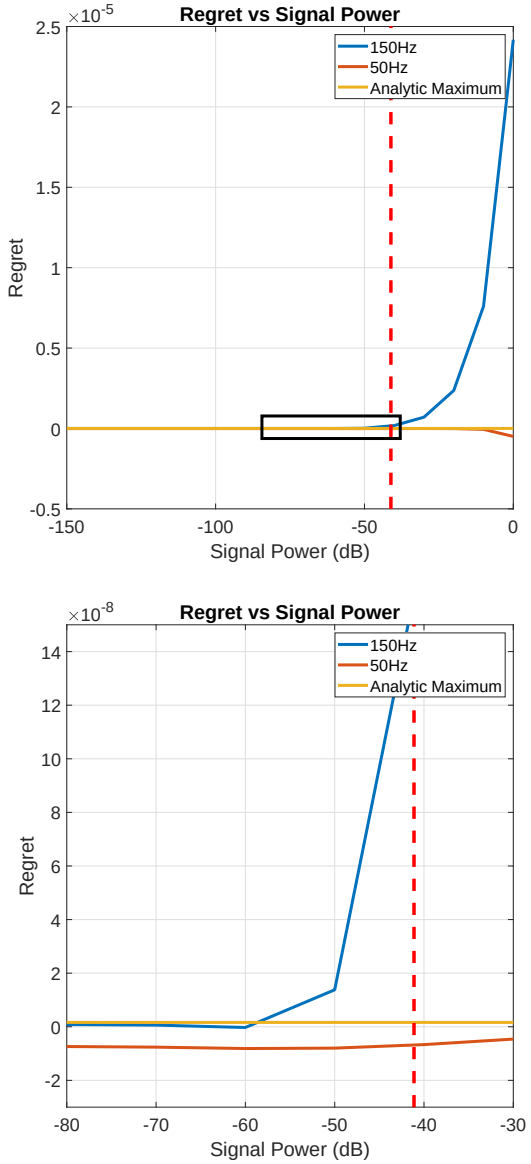

Figure 6.9: The PWB algorithm meets the performance guarantees until the restrictions on the sensitivity parameter are violated and it performs better than the analytic limit in terms of regret especially when there is no signal at the frequency of interest.

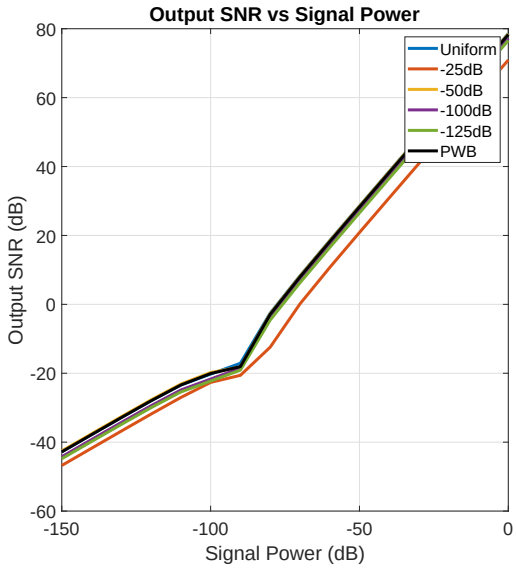

Figure 6.10: The PWB estimator is close to the best performing estimator in terms of output SNR

Chapter 7: Conclusion

This thesis developed the PWB estimator and showed analytically that it performs as well or better than the best fixed window estimator in the ensemble. We also presented a method for numerically stabilizing the algorithm by renormalization. Simulations have shown that the PWB estimator automatically selects the most accurate estimator or combination of estimators at each frequency of interest. If the windows in estimator ensemble are selected as described in the design guide, then they provide the PWB estimator the ability to display high resolution in regions of the spectrum where interferers are not a problem, and lower side-lobe leakage in regions where rejection of an interferer component is required. The simulations also show that the PWB algorithm is able to outperform the sample MPDR. When applied to hydrophone data from an underwater glider experiment the PWB algorithm required some modification to protect the algorithm from very loud transients that violate the conditions on its sensitivity parameter. The issue of loud transients is mitigated by thresholding the loss, and by applying a sliding window to the loss calculation. With these modification the experiments show that the PWB algorithm performs as promised by the convergence proof in terms of regret as long as the restriction on the sensitivity parameter is respected. It was shown that in terms of output SNR the PWB algorithm still performs competitively to the estimators in its ensemble when the restriction is violated. These properties make the PWB estimator especially promising for applications where a spectrum must be calculated without the help of an analyst.

There are still opportunities to improve the algorithm. It may be useful to look at the PWB algorithm in a stochastic setting. Currently we do not have closed form expressions for the distribution, or statistics of the estimator. However, some preliminary work on this problem suggest that it may provide an upper bound on ν instead of the lower bound provided by the deterministic proof presented in this thesis. Another interesting avenue of

study is to borrow from the field of machine learning. The PWB algorithm may be viewed as an unsupervised learning algorithm because it "learns" weights that restrict the loss of the algorithm. In a machine learning context, the function that generates the weights is known as the soft-max function. It would be interesting to replace the soft-max function with some of the other common activation functions in machine learning such as inverse tangent, or hyperbolic tangent.

In this thesis the value of the sensitivity parameter was kept constant across all frequencies of interest, however this is not required by the algorithm. In a situation where the spectrum changes drastically in different regions of the spectrum it might be useful to assign different values of ν in different regions. For example in the hydrophone data the general trend is that there is more noise at the lower frequencies of the spectrum, so at the lower frequencies a relatively high value of ν is required. This means that at higher frequencies the algorithm is less sensitive than it could be. To solve this problem a lower value of ν could be used at the higher frequencies. Finally it would be interesting to explore choosing the window ensemble adaptively. The blend weights provide a description of how much each window is used by the estimator. Windows that consistently have low weights could be altered to make them more useful to the estimator. For example if the ensemble consists of Chebychev windows the peak side-lobe levels of the windows in the ensemble could be adjusted based on the blend weights in an attempt to maximize the usefulness of the ensemble windows.

In conclusion, the PWB estimator is an exciting new algorithm for spectral estimation. While there are many opportunities to improve the estimator it could be used in its current form to improve spectrum estimates especially when a spectrum must be calculated without the help of an analyst, which commonly occurs in autonomous vehicles such as underwater gliders or drones.

Bibliography

Bibliography

[1] J. S. Lim and A. V. Oppenheim, *Advanced Topics in Signal Processing.* Prentice Hall, 1988.

[2] M. Smith and T. Barnwell, "A new filter bank theory for time-frequency representation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 3, pp. 314–327, 1987.

[3] S. M. Kay and S. L. Marple, "Spectrum analysis a modern perspective," *Proceedings of the IEEE*, vol. 69, no. 11, pp. 1380–1419, 1981.

[4] F. J. Harris, "On the use of windows for harmonic analysis with the discrete Fourier transform," *Proceedings of the IEEE*, vol. 66, no. 1, pp. 51–83, Jan 1978.

[5] J. R. Buck and A. Singer, "A performance-weighted blended Dominant Mode Rejection beamformer," 07 2018, pp. 124–128.

[6] J. Capon, "High-resolution frequency-wavenumber spectrum analysis," *Proceedings of the IEEE*, vol. 57, no. 8, pp. 1408–1418, 1969.

[7] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing.* Pearson, 2017.

[8] P. Stoica and R. L. Moses, *Spectral Analysis of Signals.* Pearson/Prentice Hall, 2005.

[9] H. L. Van Trees, *Optimum Array Processing.* New York: Wiley-Interscience, 2002.

[10] C. W. Therrien, *Discrete Random Signals and Statistical Signal Processing.* Prentice-Hall, 1992.

[11] J. Capon and N. Goodman, "Probability distributions for estimators of the frequency-wavenumber spectrum," *Proceedings of the IEEE*, vol. 58, no. 10, p. 17851786, 1970.

[12] J. R. Buck, Private Communication, 2019.

[13] A. N. Kolmogorov, "Three approaches to the quantitative definition of information," *International Journal of Computer Mathematics*, vol. 2, no. 1-4, pp. 157–168, 1968. [Online]. Available: https://doi.org/10.1080/00207166808803030

[14] L. Davisson, "Universal noiseless coding," *IEEE Transactions on Information Theory*, vol. 19, no. 6, pp. 783–795, November 1973.

[15] J. Rissanen, "Universal coding, information, prediction, and estimation," *IEEE Transactions on Information Theory*, vol. 30, no. 4, pp. 629–636, July 1984.

[16] B. Y. Ryabko, "Prediction of random sequences and universal coding," *Probl. Peredachi Inf.*, vol. 24:2, pp. 3–14, 1988.

[17] A. C. Singer and M. Feder, "Universal linear prediction by model order weighting," *IEEE Transactions on Signal Processing*, vol. 47, no. 10, pp. 2685–2699, Oct 1999.

[18] D. Abraham and N. Owsley, "Beamforming with dominant mode rejection," *Conference Proceedings on Engineering in the Ocean Environment*, Sep 1990.

[19] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* The MIT Press, 2017.

[20] G. Heinzel, A. Rüdiger, and R. Schilling, "Spectrum and spectral density estimation by the discrete fourier transform(DFT), including a comprehensive list of window functions and some new flat-top windows." Max-Planck-Institut für Gravitationsphysik (Albert-Einstein-Institut), Tech. Rep., february 2002.

Curriculum Vitae

Jeff Tucker began his career in the music industry. After taking up the French Horn in middle school he earned a Bachelor of the Arts in music from Florida State University. In addition to performing Jeff developed an interest in recording and music production, and after completing a music production program at Hennepin Technical College in 2006 he found a position as a recording engineer at Art of Music Studios. Two years later he took over ownership and continued as the owner operator until 2016. Jeff began to study electronics because of the need to maintain and repair electronic equipment in the recording studio. In 2014 he began taking engineering courses at Northern Virginia Community College. Those courses led to enrollment at George Mason University in Fall 2015 where he finished a Bachelor's degree in electrical engineering magna cum laude. Jeff is currently enrolled in the Master's program as well as the PhD program at George Mason. He is still an active musician and currently plays French Horn with The McLean Symphony.