NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

STOCHASTIC MATCHED FILTERS FOR SIGNAL DETECTION APPLICATIONS

by

Michelle M. Welch

June 2022

Thesis Advisor: Monique P. Fargues
Second Reader: Ric Romero

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2022	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE
STOCHASTIC MATCHED FILTERS FOR SIGNAL DETECTION APPLICATIONS

5. FUNDING NUMBERS

6. AUTHOR(S) Michelle M. Welch

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT (maximum 200 words)

The stochastic matched filter (SMF) is a variation of the matched filter that can detect stochastic signals in noisy environments. Some earlier studies suggest that the SMF can be extended to the detection of frequency time-variant (nonstationary) signals, namely wideband modulated sonar in shallow water. This thesis considers the SMF algorithm first proposed by J.-F. Cavasillas in signal detection and estimation scenarios, and investigates its application to narrowband and chirp signals imbedded in white noise. In medium to high signal-to-noise ratio (SNR) values, results indicate that the SMF is a viable technique for signal detection and estimation, and could be employed in passive, real-time signal detection and estimation scenarios.

14. SUBJECT TERMS
matched filter, stochastic matched filter, non-stationary signals, acoustic signals, detection

15. NUMBER OF PAGES
83

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

STOCHASTIC MATCHED FILTERS FOR SIGNAL DETECTION APPLICATIONS

Michelle M. Welch
Lieutenant Commander, United States Navy
BS, United States Naval Academy, 2009

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 2022

Approved by: Monique P. Fargues
Advisor

Ric Romero
Second Reader

Douglas J. Fouts
Chair, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The stochastic matched filter (SMF) is a variation of the matched filter that can detect stochastic signals in noisy environments. Some earlier studies suggest that the SMF can be extended to the detection of frequency time-variant (nonstationary) signals, namely wideband modulated sonar in shallow water. This thesis considers the SMF algorithm first proposed by J.-F. Cavasillas in signal detection and estimation scenarios, and investigates its application to narrowband and chirp signals imbedded in white noise. In medium to high signal-to-noise ratio (SNR) values, results indicate that the SMF is a viable technique for signal detection and estimation, and could be employed in passive, real-time signal detection and estimation scenarios.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

GEP	generalized eigenvalue problem
KLE	Karhunen-Loève Expansion
KLT	Karhunen-Loève Transform
MSE	mean square error
PSD	power spectral density
SMF	stochastic matched filter
SNR	signal-to-noise ratio
STFT	short-time Fourier transform
WSS	wide sense stationary

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I'd like to express my heartfelt gratitude to Professor Fargues for her patience, guidance, and expertise. Throughout this process, I took inspiration from her work ethic and dedication to her students. She is a true scholar and a delight to work with. A special thanks to my fellow students, Major Nick Balk, USMC, and Captain Sandy DeBock, USMC, for their enthusiastic support. I'd also like to thank Professor Romero and the ECE Department. Finally, I am grateful to CAPT Herring for his exceptional leadership and professionalism.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

The U.S. Navy has mission sets in multiple domains, from outer space to the bottom of the oceans, and even cyberspace. Arguably, the most challenging of these missions is undersea warfare, due to the unpredictable nature of the underwater acoustic environment. The path of an underwater signal depends on myriad factors such as the signal frequency and depth of transmission; the temperature, depth, and salinity of the water columns through which it travels; the composition of the bottom of the ocean; the nature of underwater noise; and much more. These complications make the Navy undersea mission of detecting adversary submarines difficult. When narrowed down to a signal processing problem, there are many techniques that show promise.

Though the matched filter is a classic solution to signal detection, it is not necessarily well suited to the complicated conditions presented by real-world acoustic environments. Adaptations to the matched filter, such as the stochastic matched filter (SMF), introduced by J. F. Cavassilas in 1991 [1], have opened doors into solving more complex signal processing problems. Of note, P. Courmontagne's 2010 paper [2] provided an overview of his years of work using the SMF to detect signals and de-noise imagery, paving the way for new applications. Bonnal, Danès, and Renaud used the SMF to detect and isolate acoustic patterns in speech [3]. Mori and Gounon explored its use in active sonar [4]. When using active sonar, a known signal is transmitted, and its return is received and analyzed. The transmitted signal, however, can be detected by the adversary, making active sonar a less desirable detection tool in many military situations. More recently, however, L. Bouffaut's 2019 dissertation [5] detailed her use of SMF in passively detecting whale calls, an application that could directly contribute to the U.S. Navy ability to passively detect submarines. The purpose of this research is to determine if the SMF warrants further study and application to military problem sets, most notably submarine detection.

Chapter II introduces concepts needed to provide background knowledge for the SMF, to include discussions about the matched filter and eigenanalysis. Next, Chapter III describes the SMF theory and algorithmic implementation. Chapter IV presents an

overview of the processes and associated parameters used to conduct our simulations. Next, Chapter V presents simulation results obtained using fixed frequency tones and various chirp signals. Finally, conclusions and recommendations for future work are presented in Chapter VI.

II. BACKGROUND

The SMF incorporates some important concepts from the world of signal processing, including the classic matched filter and eigenanalysis. This chapter begins with a review of the matched filter, specifically its derivation for deterministic signals, followed by an overview of eigenanalysis and the Karhunen-Loève Expansion (KLE).

A. MATCHED FILTER FOR DETERMINISTIC SIGNALS

The matched filter approach is commonly applied to detect signals embedded in noise, where the filter impulse response is derived by optimizing the signal-to-noise ratio (SNR) quantity [6]. It is a popular technique in radar and sonar detection, specifically active applications, as the transmitted signal characteristics must be known in order to build the filter. Although the technique can be adapted to random signals with known statistics, this section will discuss the traditional application of the matched filter for deterministic signals only.

Let us define $s(n)$ as a deterministic signal of interest with known structure that occurs between samples $n = 0, \dots, N-1$, $w(n)$ as interfering noise, and $z(n) = s(n) + w(n)$ the resulting observed discrete signal. Let us assume that the signal of interest and noise are uncorrelated, and both are zero-mean [7]. Consider a linear time invariant (LTI) filter with impulse response $h(n)$ and output $y_z(n) = y_s(n) + y_w(n)$, where $y_s(n)$ represents the response of the signal of interest and $y_w(n)$ the response of the noise contribution. According to [6], the goal of the matched filter is to maximize the output SNR at the signal endpoint sample n_N such that

$$SNR_{out} = \frac{|y_s(n_N)|^2}{\mathbb{E}\{|y_w(n_N)|^2\}}, \tag{1}$$

with $y_s(n_N) = \boldsymbol{h}^T \tilde{\boldsymbol{s}}$, $y_w(n_N) = \boldsymbol{h}^T \tilde{\boldsymbol{w}}$, where $\boldsymbol{h}$ is defined as the impulse response of the matched filter of length *N*, and $\tilde{\boldsymbol{s}}$ and $\tilde{\boldsymbol{w}}$ are the reversed signal vector and reversed noise vector, respectively. Note that in this context, a reversed vector is considered a flipped

vector. Then the output can be expressed in terms of the filter impulse response vector $\boldsymbol{h}$ and the observation in vector form $\boldsymbol{z}$:

$$y_z(n_N) = \sum_{k=0}^{N-1} h[k]z[n_N - k] = \boldsymbol{h}^T\tilde{\boldsymbol{z}}, \tag{2}$$

where $\tilde{\boldsymbol{z}}$ is the reversed observation vector. The numerator of the expression shown in (1) can be expressed in terms of the impulse response vector and signal vectors as:

$$|y_s(n_N)|^2 = \boldsymbol{h}^H\tilde{\boldsymbol{s}}^*\tilde{\boldsymbol{s}}^T\boldsymbol{h}, \tag{3}$$

and the denominator of (1) can be expressed in terms of the impulse response vector and noise autocorrelation matrix $\boldsymbol{R}_w$:

$$\mathbb{E}\{|y_w(n_N)|^2\} = \boldsymbol{h}^H\boldsymbol{R}_w\boldsymbol{h}. \tag{4}$$

Thus, (1) can be expressed as:

$$SNR_{out} = \frac{\boldsymbol{h}^H\tilde{\boldsymbol{s}}^*\tilde{\boldsymbol{s}}^T\boldsymbol{h}}{\boldsymbol{h}^H\boldsymbol{R}_w\boldsymbol{h}}. \tag{5}$$

In the next sections, we will present applications of the matched filter to signal detection in white and colored noise environments.

1. Deterministic Signal in White Noise

When the noise $w(n)$ is white, $\boldsymbol{R}_w = \sigma_w^2\boldsymbol{I}$, where σ_w^2 is the noise power and $\boldsymbol{I}$ is the identity matrix, resulting in a diagonal noise correlation matrix. Then (5) can be expressed as:

$$SNR_{out} = \frac{\boldsymbol{h}^H\tilde{\boldsymbol{s}}^*\tilde{\boldsymbol{s}}^T\boldsymbol{h}}{\sigma_w^2|\boldsymbol{h}|^2} = \frac{|\boldsymbol{h}^H\tilde{\boldsymbol{s}}^*|^2}{\sigma_w^2|\boldsymbol{h}|^2}. \tag{6}$$

The impulse response $\boldsymbol{h}$ can be computed by maximizing the numerator of (6) and normalizing the denominator to a fixed value equal to 1. By the Cauchy-Schwartz inequality, $\boldsymbol{h} = K\tilde{\boldsymbol{s}}^*$ maximizes the numerator expression of (6) [6]. Substituting this expression of $\boldsymbol{h}$ into the denominator portion of (6) and setting it to equal one, yields $K = \frac{1}{\sigma_w|\boldsymbol{s}|}$, which leads to $\boldsymbol{h} = \frac{1}{\sigma_w|\boldsymbol{s}|}\tilde{\boldsymbol{s}}^*$, where $\tilde{\boldsymbol{s}}^*$ is the reversed conjugate of the signal

expression [6]. Therefore, the impulse response of the matched filter is a scaled version of the reversed expression of the signal of interest.

2. Deterministic Signal in Colored Noise

In the case of colored noise, the noise correlation function, $\boldsymbol{R}_w$ is no longer diagonal but can be expressed using the Cholesky decomposition: $\boldsymbol{R}_w = \boldsymbol{L}_w \boldsymbol{L}_w^H$, and (5) can be rewritten as:

$$SNR_{out} = \frac{|\boldsymbol{h}^H \tilde{\boldsymbol{s}}^*|^2}{\boldsymbol{h}^H \boldsymbol{L}_w \boldsymbol{L}_w^H \boldsymbol{h}}. \tag{7}$$

Again, the numerator is maximized using the Cauchy-Schwartz inequality leading to

$\boldsymbol{L}_w^H \boldsymbol{h} = K \boldsymbol{L}_w^{-1} \tilde{\boldsymbol{s}}^*$, which yields $\boldsymbol{h} = K \boldsymbol{R}_w^{-1} \tilde{\boldsymbol{s}}^*$ [7]. We then normalize the denominator of (7) by setting it equal to one, resulting in $K = \frac{1}{\sqrt{\boldsymbol{s}^H \boldsymbol{R}_w^{-1} \boldsymbol{s}}}$. Substituting K in the expression for the impulse response leads to

$$\boldsymbol{h} = \frac{1}{\sqrt{\boldsymbol{s}^H \boldsymbol{R}_w^{-1} \boldsymbol{s}}} \boldsymbol{R}_w^{-1} \tilde{\boldsymbol{s}}^*. \tag{8}$$

In this case, the optimal filter can be viewed as a whitening filter combined with a matched filter derived in the white noise environment [7].

B. EIGENANALYSIS

The concepts of eigenvectors and eigenvalues form the foundation of the SMF, which is the extension of the matched filter when applied to random signals [8]. In this section we briefly review these concepts before introducing the SMF approach in the next section.

1. The Generalized Eigenvalue Problem

Let $\boldsymbol{u}(n)$ be a wide sense stationary discrete-time stochastic process in the form of a $N \times 1$ vector. Then the autocorrelation of $\boldsymbol{u}(n)$ is a $N \times N$ Hermitian matrix $\boldsymbol{R}$. An

eigenvector $\boldsymbol{q}$ is defined as a vector of dimension $N \times 1$, which satisfies the following relationship,

$$\boldsymbol{R}\boldsymbol{q} = \lambda \boldsymbol{q}, \tag{9}$$

where the constant λ is the eigenvalue associated with a specific eigenvector.

Eigenvalues and eigenvectors may be computed by rewriting (9) in the form:

$$(\boldsymbol{R} - \lambda \boldsymbol{I})\boldsymbol{q} = \boldsymbol{0}, \tag{10}$$

where $\boldsymbol{I}$ is the $N \times N$ identity matrix and $\boldsymbol{0}$ is the $N \times 1$ null vector. For $\boldsymbol{q}$ to be a non-zero solution of (10), $(\boldsymbol{R} - \lambda \boldsymbol{I})$ must be singular (i.e., have a determinant equal to zero); therefore:

$$\det(\boldsymbol{R} - \lambda \boldsymbol{I}) = 0. \tag{11}$$

Expanding (11) yields a polynomial of degree N, which means (11) (also known as the characteristic equation of $\boldsymbol{R}$) has N roots λ, and there are N vectors $\boldsymbol{q}$ that satisfy (10). Let λ_i for $i = 1, \dots, N$ be the roots, or eigenvalues, of the characteristic equation (11), and $\boldsymbol{q}_i$ for $i = 1, \dots, N$ be the corresponding eigenvectors such that

$$\boldsymbol{R}\boldsymbol{q}_i = \lambda_i \boldsymbol{q}_i. \tag{12}$$

Eigenvectors derived from a correlation are linearly independent and can be normalized to a length of one; therefore, they are considered an orthonormal set [8] such that:

$$\boldsymbol{q}_i^H \boldsymbol{q}_j = \begin{cases} 1, i = j \\ 0, i \neq j \end{cases}. \tag{13}$$

2. Karhunen-Loève Expansion

One important property of eigenvalues and eigenvectors is the Karhunen-Loève Expansion (KLE) [8]. Let $\boldsymbol{u}(n)$ be a $N \times 1$ vector that represents data derived from a zero-mean wide sense stationary process with $N \times N$ autocorrelation matrix $\boldsymbol{R}$. Let $\boldsymbol{q}_i$, $i = 1, \dots, N$ be the eigenvectors of $\boldsymbol{R}$, sorted in descending order by their corresponding

eigenvalues $\lambda_i, i = 1, \dots, N$. Then we can expand $\boldsymbol{u}(n)$ into a linear combination of the eigenvectors:

$$\boldsymbol{u}(n) = \sum_{i=1}^{N} c_i(n)\, \boldsymbol{q}_i, \tag{14}$$

where $c_i(n)$ are the coefficients of the expansion defined by the Karhunen-Loève Transform (KLT) [6]:

$$c_i(n) = \boldsymbol{q}_i^H \boldsymbol{u}(n), i = 1, \dots, N. \tag{15}$$

The coefficients are zero-mean uncorrelated random variables, therefore [6]:

$$\mathbb{E}\{c_i(n)\} = 0, \tag{16}$$

and

$$\mathbb{E}\{c_i(n) c_j^*(n)\} = \begin{cases} \lambda_i, i = j \\ 0, i \neq j \end{cases}. \tag{17}$$

Then eigenvectors $\boldsymbol{q}_i$, $i = 1, \dots, N$ form the orthonormal coordinate system of a N-dimensional space in which $\boldsymbol{u}(n)$ is represented as a set of projections $c_i(n)$ onto the axes $\boldsymbol{q}_i$, $i = 1, \dots, N$. According to Haykin [8], from (14) it follows:

$$\sum_{i=1}^{N} |c_i(n)|^2 = \|\boldsymbol{u}(n)\|^2, \tag{18}$$

and from (15) and (17) we have:

$$\mathbb{E}\{|c_i(n)|^2\} = \lambda_i, i = 1, \dots, N. \tag{19}$$

Thus, most of the energy of $\boldsymbol{u}(n)$ is in the direction of the eigenvector (also the basis vector or axis) that corresponds to the largest eigenvalue, $\boldsymbol{q}_1$ and λ_1, respectively [6], as illustrated in Figure 1.

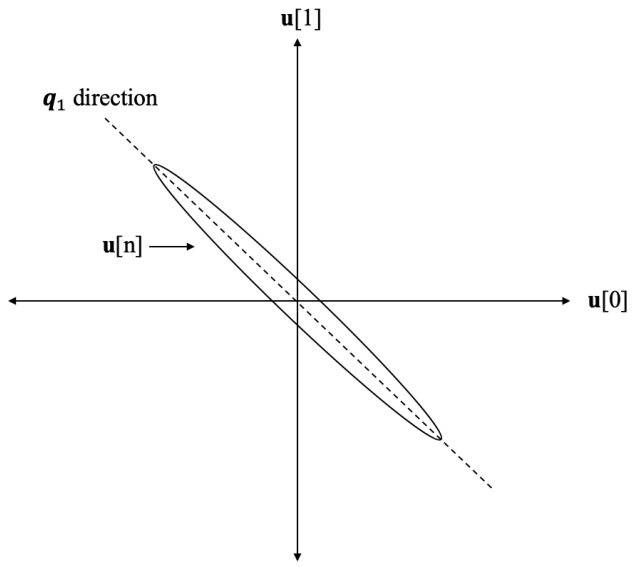

Figure 1. Data projected onto eigenvector. Adapted from [6].

However, there may be a lesser, but still significant amount of energy in the direction of some of the subsequent eigenvectors, which we would want to include in the approximation. Therefore, we would use only data projected onto the eigenvectors associated with largest eigenvalues. Thus, the expansion can be reduced in dimension, meaning it can be truncated to fewer than N orthonormal basis vectors with minimal loss of information [6].

3. Dimension Reduction

We can use the KLE to create a subspace of dimension M, where $M < N$ and the random process $\boldsymbol{u}(n)$ is approximated by

$$\tilde{\boldsymbol{u}}(n) = \sum_{i=1}^{M} c_i(n)\boldsymbol{q}_i\,, \tag{20}$$

where $\boldsymbol{q}_i, 1, \dots, M$ are the basis vectors associated with the M largest eigenvalues $\lambda_i, i = 1, \dots, M$. We choose the value M by minimizing the mean square error (MSE) between the data vector $\boldsymbol{u}(n)$ and the truncated approximation $\tilde{\boldsymbol{u}}(n)$. First, we define the approximation error $\boldsymbol{e}(n)$ as

$$\boldsymbol{e}(n) = \boldsymbol{u}(n) - \widetilde{\boldsymbol{u}}(n), \tag{21}$$

which can be rewritten as

$$\boldsymbol{e}(n) = \sum_{i=M+1}^{N} c_i(n)\boldsymbol{q}_i\,. \tag{22}$$

Then the MSE is defined as the average energy in the approximation error [6]:

$$\begin{aligned}
\mathcal{E} &= \mathbb{E}\{\|\boldsymbol{e}(n)^2\|\} = \mathbb{E}\{\boldsymbol{e}^H(n)\boldsymbol{e}(n)\} \\
&= \mathbb{E}\left\{\sum_{i=M+1}^{N}\sum_{j=M+1}^{N} \boldsymbol{c}_i^*(n)\boldsymbol{c}_j\,(n)\boldsymbol{q}_i^H\boldsymbol{q}_j\right\} \\
&= \sum_{i=M+1}^{N}\sum_{j=M+1}^{N} \mathbb{E}\{\boldsymbol{c}_i^*(n)\boldsymbol{c}_j(n)\}\,\boldsymbol{q}_i^H\boldsymbol{q}_j \\
&= \sum_{i=M+1}^{N} \lambda_i\,,
\end{aligned} \tag{23}$$

where $\lambda_{M+1}, \dots, \lambda_N \ll \lambda_1, \dots, \lambda_M$ [8]. Therefore, leaving out the smallest $N - M$ eigenvalues of $\boldsymbol{R}$ ensures that the approximation $\widetilde{\boldsymbol{u}}(n)$ is an efficient and optimal representation of $\boldsymbol{u}(n)$, which allows us to keep only the top M eigenvalues and corresponding eigenvectors that contain useful information.

When the data sequence is composed of a random signal in additive white noise, its truncated KLE allows us to retain signal information while eliminating much of the noise [6] as illustrated by Figure 2. The expansion effectively changes how we reference the space by using the eigenvectors of the noisy signal vector covariance as the basis to optimally describe the signal and noise, without knowing the initial shape of the signal [9]. Truncating the expansion reduces the dimension and allows us to work within a smaller, more efficient subspace.

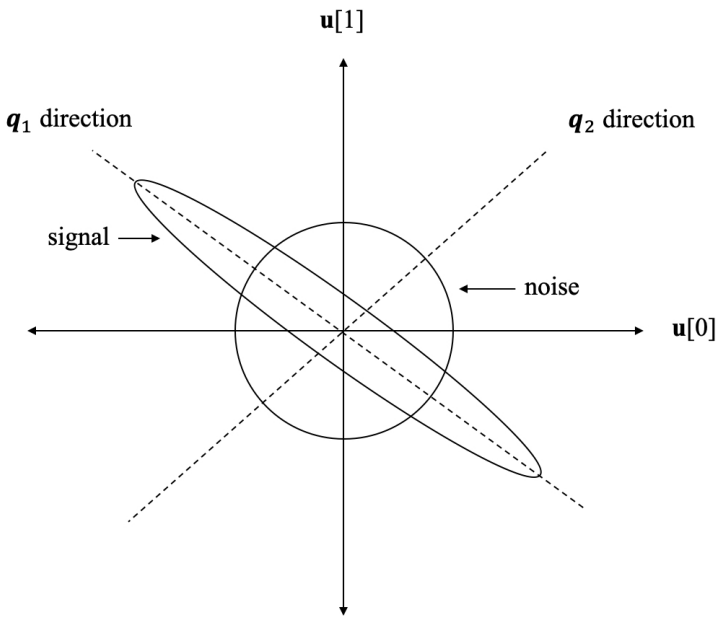

Figure 2. Signal and noise projected onto eigenvectors. Adapted from [6].

C. SUMMARY

In this chapter, we discussed the theory behind the matched filter for deterministic signals and briefly derived the resulting filter impulse response expressions in white and colored noise environments. We then reviewed the concept of eigenanalysis as it relates to the Karhunen-Loève Expansion and dimension reduction. In the next chapter, we introduce the stochastic matched filter, an adaptation of the matched filter designed to handle random signals.

III. STOCHASTIC MATCHED FILTER THEORY

The matched filter described in the previous subsection is designed for deterministic signals, which are signals with known characteristics. In communication applications, this specification implies that there is a single path of propagation from transmitter to receiver, or in the case of multiple paths or receivers, the impulse responses of each are known. However, in a realistic environment, the transmission medium introduces randomness that alters the parameters of the received signal in unpredictable ways [4]. Bouffaut notes in [5] that the matched filter particularly struggles when the signal experiences frequency fluctuations and transient noise, both characteristics of the underwater acoustic environment. Additionally, the matched filter is more easily applied to white noise than to colored noise because white noise is completely uncorrelated. As noise becomes more "colored," it becomes more correlated between samples, the noise autocorrelation function becomes less diagonal, and the impulse response less similar to the signal of interest, raising the optimum SNR at which the signal can be detected [7].

In his 1991 paper [1], J-F. Cavassilas introduced a technique called the stochastic matched filter, an extension of the matched filter that could detect stochastic signals imbedded in additive noise using the known statistical properties of the signal. The SMF can be adapted to passive collection, in which the received signal has unknown and unpredictable parameters due to the effects of the transmission medium. However, when the signal is WSS, second-order statistics can be estimated [5] and used to model the system impulse response [4]. Then the received signal, which is considered stochastic, can be detected, and its transmission time and duration determined. This chapter will introduce the theory behind the SMF, beginning with an introduction of matched filters applied to random signals imbedded in noise. Next, we will discuss the concept of random signal expansion and its use in the derivation of the SMF impulse response for real-time signal detection applications.

A. STOCHASTIC MATCHED FILTER IMPULSE RESPONSE DERIVATION

This section expands the application of the matched filter to the detection of stochastic signals. Although the impulse response of the matched filter is traditionally derived from a deterministic transmitted signal, the technique can be adapted to random signals with known second-order statistics. As with the matched filter scenario described in the previous section, the goal of the SMF is to maximize the signal-to-noise ratio (SNR). According to Borloz and Xerri [10], the goal is to create a multi-dimensional subspace that will optimize signal detection when the data is projected onto it. This subspace is defined by eigenvectors that are associated with a certain number of the largest eigenvalues of the signal autocovariance matrix, allowing for signal detection in nonstationary and colored noise environments [11].

Let $\boldsymbol{z}$ be a received observation of length N, sampled in the discrete time domain such that

$$\boldsymbol{z} = \boldsymbol{s} + \boldsymbol{\eta}, \tag{24}$$

where $\boldsymbol{s}$ is a zero mean random signal vector with known covariance matrix $\boldsymbol{R_s}$, and $\boldsymbol{\eta}$ is a zero mean random noise vector with known covariance matrix $\boldsymbol{R_\eta}$. Let $\sigma_s^2 = \mathbb{E}\{\boldsymbol{s}^2\}$ be the variance of the signal and $\boldsymbol{s}_0$ be the reduced signal such that $\mathbb{E}\{\boldsymbol{s}_0^2\} = 1$. Similarly, let $\sigma_\eta^2 = \mathbb{E}\{\boldsymbol{\eta}^2\}$ be the variance of the noise and $\boldsymbol{\eta}_0$ be the reduced signal such that $\mathbb{E}\{\boldsymbol{\eta}_0^2\} = 1$. Then we can express the observation in terms of signal and noise variances and reduced signal and noise functions as follows:

$$\boldsymbol{z}(k) = \boldsymbol{s}(k) + \boldsymbol{\eta}(k) = \sigma_{\boldsymbol{s}}\boldsymbol{s}_0(k) + \sigma_{\boldsymbol{\eta}}\boldsymbol{\eta}_0(k), \tag{25}$$

where $\boldsymbol{s}_0(k)$ and $\boldsymbol{\eta}_0(k)$ are zero-mean, independent, and second-order stationary processes [2]. Additionally, the covariance matrix of the signal can be expressed in terms of the reduced signal covariance matrix $\boldsymbol{\Gamma}_{\boldsymbol{s}_0}$, and the signal variance:

$$\boldsymbol{R_s} = \sigma_s^2\boldsymbol{\Gamma}_{\boldsymbol{s}_0}. \tag{26}$$

Similarly, the noise covariance matrix can be expressed:

$$\boldsymbol{R_\eta} = \sigma_\eta^2\boldsymbol{\Gamma}_{\boldsymbol{\eta}_0}, \tag{27}$$

where $\boldsymbol{\Gamma}_{\boldsymbol{\eta}_0}$ is the covariance matrix of the reduced noise [5]. If we consider a linear filter with impulse response $\boldsymbol{h}$, the filter output SNR can be expressed as:

$$oSNR = \frac{\boldsymbol{h}^T \boldsymbol{R}_{\boldsymbol{s}} \boldsymbol{h}}{\boldsymbol{h}^T \boldsymbol{R}_{\boldsymbol{\eta}} \boldsymbol{h}}, \tag{28}$$

which is a Rayleigh quotient [10]. This Rayleigh quotient is maximized when $\boldsymbol{h}$ is the eigenvector corresponding to the greatest eigenvalue of $\boldsymbol{R}_{\boldsymbol{\eta}}{}^{-1}\boldsymbol{R}_{\boldsymbol{s}}$, in which case, $oSNR$ is actually equal to the greatest eigenvalue [7]. From (26) and (27), we note that $\boldsymbol{\Gamma}_{\boldsymbol{s}_0} = \frac{\boldsymbol{R}_{\boldsymbol{s}}}{\sigma_{\boldsymbol{s}}^2}$ and $\boldsymbol{\Gamma}_{\boldsymbol{\eta}_0} = \frac{\boldsymbol{R}_{\boldsymbol{\eta}}}{\sigma_{\boldsymbol{\eta}}^2}$, which allows us to rewrite (28) as:

$$oSNR = \frac{\sigma_{\boldsymbol{s}}^2}{\sigma_{\boldsymbol{\eta}}^2} \frac{\boldsymbol{h}^T \boldsymbol{\Gamma}_{\boldsymbol{s}_0} \boldsymbol{h}}{\boldsymbol{h}^T \boldsymbol{\Gamma}_{\boldsymbol{\eta}_0} \boldsymbol{h}} = \rho \frac{\boldsymbol{h}^T \boldsymbol{\Gamma}_{\boldsymbol{s}_0} \boldsymbol{h}}{\boldsymbol{h}^T \boldsymbol{\Gamma}_{\boldsymbol{\eta}_0} \boldsymbol{h}}, \tag{29}$$

where $\rho = \frac{\sigma_{\boldsymbol{s}}^2}{\sigma_{\boldsymbol{\eta}}^2}$ is the input SNR [4]. According to Courmontagne [2], the Rayleigh quotient $\frac{\boldsymbol{h}^T \boldsymbol{\Gamma}_{\boldsymbol{s}_0} \boldsymbol{h}}{\boldsymbol{h}^T \boldsymbol{\Gamma}_{\boldsymbol{\eta}_0} \boldsymbol{h}}$ can be maximized by solving the following generalized eigenvalue problem (GEP):

$$\boldsymbol{\Gamma}_{\boldsymbol{s}_0} \boldsymbol{\Phi}_i = \lambda_i \boldsymbol{\Gamma}_{\boldsymbol{\eta}_0} \boldsymbol{\Phi}_i, \qquad i \in \{1, \dots, N\} \tag{30}$$

where $\boldsymbol{\Phi}_i$ is the ith eigenvector associated with λ_i, the ith eigenvalue of $\boldsymbol{\Gamma}_{\boldsymbol{\eta}_0}{}^{-1}\boldsymbol{\Gamma}_{\boldsymbol{s}_0}$. The eigenvector $\boldsymbol{\Phi}_1$, corresponding to the largest eigenvalue λ_1, maximizes the output SNR value defined in (29); however, Mori and Gounon [4] note that the output SNR can further be improved by including any eigenvectors corresponding to eigenvalues that are greater than or equal to one.

As described in Section II.B.1, the eigenvectors $\boldsymbol{\Phi}_i$ form an orthonormal basis. According to Bouffaut [5], we can form another basis using the eigenvectors of $(\boldsymbol{\Gamma}_{\boldsymbol{\eta}_0}{}^{-1}\boldsymbol{\Gamma}_{\boldsymbol{s}_0})^T$, which we denote $\boldsymbol{\Psi}_i$. Then

$$\boldsymbol{\Psi} = \boldsymbol{\Gamma}_{\boldsymbol{\eta}_0} \boldsymbol{\Phi}, \tag{31}$$

where the matrices $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$, have column vectors $\boldsymbol{\Phi}_i$ and $\boldsymbol{\Psi}_i$ respectively, which define an orthogonal space in which we can expand the observation so that the signal can be differentiated from the noise, maximizing the SNR quantity [5].

B. RANDOM SIGNAL EXPANSION

A noisy stochastic observation signal can be expanded into a sum of deterministic basis vectors weighted by uncorrelated random variables [2]. Those basis vectors can be chosen so that a number of random variables contain more signal information than noise. The Karhunen-Loève Expansion (KLE) described in Section II.B.2 meets these criteria. Let us define the vector $\boldsymbol{z}(k)$ from (25) to be a discrete, zero-mean, second-order stationary random signal sequence of N samples, and $z_{i,k}, i \in \{1, \dots, N\}$ is a zero-mean, uncorrelated random variable sequence [2]. Thus, $E\{z_{i,k}\} = 0$, $E\{z_{i,k}, z_{j,k}, k\} = \delta_{i,j} E\{z_{i,k}^2\}$, and

$$\boldsymbol{z}(k) = \sum_{i=1}^{N} z_{i,k} \boldsymbol{\Psi}_i, \tag{32}$$

where $z_{i,k}$ are the random variable coefficients of the decomposition of $\boldsymbol{z}(k)$, and $\boldsymbol{\Psi}_i$ are basis vectors previously introduced in Section III.A. The random variable coefficients $z_{i,k}$ can be expressed as the inner product of the observation $\boldsymbol{z}(k)$ and the basis vector $\boldsymbol{\Phi}_i$ [5]:

$$z_{i,k} = \boldsymbol{z}(k)^T \boldsymbol{\Phi}_i. \tag{33}$$

According to Cavassilas [1], coefficients $z_{i,k}$ are uncorrelated with unit power when the noise is expanded on the basis $\boldsymbol{\Psi}$, while the signal expanded on the same basis will yield uncorrelated coefficients with power equal to the corresponding eigenvalue λ_i, as illustrated in (19). Then it follows that there is a number of expansion coefficients that contain mostly signal information and would contribute to a good approximation of the signal of interest. Using the dimension reduction properties of KLE discussed in Section II.B.3, we can reconstruct the signal of interest with the following expression:

$$\tilde{\boldsymbol{s}}(k) = \sum_{i=1}^{Q} z_{i,k} \boldsymbol{\Psi}_i \,, \tag{34}$$

where $Q \leq N$ is the dimension of the basis $\boldsymbol{\Psi}$ that minimizes the MSE between the signal of interest $\boldsymbol{s}(k)$ and the approximation $\tilde{\boldsymbol{s}}(k)$ [5].

C. TIME-VARYING FILTER

The discrete observation $\boldsymbol{z}(k)$ can be expressed as a vector of length N, processed using sliding windows:

$$\boldsymbol{z}(k) = \left\{ z\left[k - \frac{L-1}{2}\right], \dots, z[k], \dots, z\left[k + \frac{L-1}{2}\right] \right\}, \tag{35}$$

where L represents the odd-numbered length of the sample window, and $k = 1, \dots N$ is the center time sample of the window [2]. The signal $\boldsymbol{s}(k)$ and noise $\boldsymbol{\eta}(k)$ must both be considered second-order stationary in order to apply the SMF effectively. To ensure this stationarity characteristic in practical applications, L is chosen to be small enough that the autocovariance of the sequence within the time window is wide sense stationary (WSS) [3]. Overlapping the windows by $L - 1$ allows us to process one data point $z[k]$ for each time sample k (the center sample of each window), and adding $\frac{L-1}{2}$ zeroes to both the beginning and end of the data sequence $\boldsymbol{z}(k)$ gives us N windows, and therefore N data points $z[k]$ [2]. Input SNR and the dimension of the basis $\boldsymbol{\Psi}$ also depend on the time index k, and are denoted by $\rho[k]$ and $Q[k]$ respectively. Then the approximation of the signal given in (34) can be rewritten in terms of k and truncated to order $Q[k]$:

$$\tilde{\boldsymbol{s}}_{Q[k]}[k] = \sum_{i=1}^{Q[k]} z_{i,k}\, \boldsymbol{\Psi}_i \left[\frac{L+1}{2}\right], \tag{36}$$

where $\tilde{\boldsymbol{s}}_{Q[k]}[k]$ represents the kth data point of the approximated signal of interest, and $\boldsymbol{\Psi}_i \left[\frac{L+1}{2}\right]$ is the center point of the ith basis vector $\boldsymbol{\Psi}$ [5]. We can also consider $Q[k]$ to be the number of eigenvalues λ_i, that when multiplied by the SNR for each sample window of $\boldsymbol{z}(k)$, are greater than or equal to one [2]:

$$\lambda_i \rho[k] \geq 1. \tag{37}$$

The approximation in (36) can be rewritten using (33):

$$\tilde{\boldsymbol{s}}_{Q[k]}[k] = \boldsymbol{z}(k)^T \sum_{i=1}^{Q[k]} \boldsymbol{\Psi}_i \left[\frac{L+1}{2}\right] \boldsymbol{\Phi}_i \,, \tag{38}$$

where $\boldsymbol{\Psi}_i$ and $\boldsymbol{\Phi}_i$ are L-dimensional bases derived from $L \times L$ reduced covariance matrices $\boldsymbol{\Gamma}_{\boldsymbol{s}_0}$ and $\boldsymbol{\Gamma}_{\boldsymbol{\eta}_0}$ [2]. Then the value $Q[k]$ will be between 1 and L. Let $\boldsymbol{h}_{Q[k]}$ be the impulse response for the corresponding filter (referred to as the filter vector) such that

$$\boldsymbol{h}_{Q[k]} = \sum_{i=1}^{Q[k]} \boldsymbol{\Psi}_i \left[\frac{L+1}{2}\right] \boldsymbol{\Phi}_i \,. \tag{39}$$

Then the expression shown in (38) can be expressed as the inner product of the observation vector and the filter vector for a specific value $Q[k]$, resulting in:

$$\tilde{\boldsymbol{s}}_{Q[k]}[k] = \boldsymbol{z}(k)^T \boldsymbol{h}_{Q[k]}. \tag{40}$$

Note the filter vector $\boldsymbol{h}_1$ corresponds to the eigenvector associated with the largest eigenvalue λ_1 and maximizes the SNR quantity. Bouffaut [5] suggests choosing a maximum value for $Q[k] \leq L$, using a priori knowledge of the signal of interest frequencies in order to decrease processing time. Let this value be called Q_{max}. Then the filter vector $\boldsymbol{h}_{Q_{max}}$ is defined as the superposition of all filter vectors up to Q_{max}, and it encompasses all the frequencies contained in the signal of interest. We use this process to define Q_{max} individual filter vectors which are associated with frequency bands of the signal of interest. The value $Q[k]$ is determined by the estimated input SNR value at time k, as defined by (37). In turn, the value $Q[k]$ determines which specific filter impulse expression is used at time k, resulting in a time-varying filter. This filter bank can be stored and used for real-time signal processing [12], which will be discussed in the next section.

D. REAL-TIME ANALYSIS

For military applications, it is important that signal detection occurs in real-time. In [5], Bouffaut describes a technique for real-time analysis that uses a priori knowledge of the signal of interest to first conduct offline calculations in order to process noisy observations as they occur in real-time applications. In this approach, we first create a bank

of time-varying filters, as discussed in the previous section. Then we apply those filters selectively, choosing the filter that will optimize the output SNR value at each sample time. This section will first discuss the process of building the filter bank, which we call offline analysis. Then we will describe the steps for online analysis, which is the real-time processing of the observation and approximation of the signal [12].

1. Offline Analysis

To create the filter bank, we must estimate the signal and noise autocovariance functions. Since the signal of interest has known characteristics, its autocovariance matrix can be calculated. For the purpose of calculating the filter bank, the noise autocovariance is estimated using simulated noise with a fixed variance [12]. To reduce the number of filter vectors from L in order to decrease processing time, we initially choose the maximum filter parameter Q_{max} to represent only the highest 10% of eigenvalues λ_i (Q_{max} equals the number of eigenvalues in the highest 10th percentile) [5]. As discussed in Section II.B.2, the signal can be accurately represented by a small number of eigenvectors associated with the largest eigenvalues. Figure 3 shows a block diagram of the resulting offline analysis algorithm.

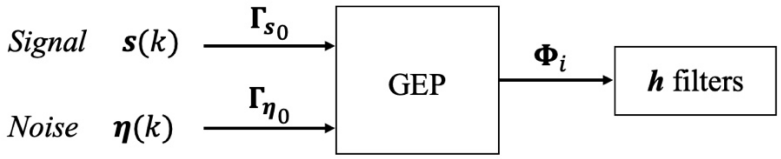

Figure 3. Offline SMF processing diagram. Adapted from [5].

2. Online Analysis

Once the filter bank is created during offline analysis, we can process received data to determine whether the signal of interest is present in real-time. First, we estimate the noise autocovariance of the observed data so it can be used to calculate the online SNR value for each window [5]. However, signal may be present in any of these windows [12],

which complicates the noise covariance estimation process. In windows where signal may be present, signal contributions can be separated from the noise floor using the short-time Fourier transform (STFT) [13]. The STFT is a tool that analyzes the signal local frequency behavior by taking windows in the time-domain and performing a fast Fourier transform on the data within each window [14]. Window parameters are determined by sampling frequency, observation duration, and expected signal of interest duration [5]. Using weighted overlapping windows to take the STFT of the observation results in a spectrogram [12]. The STFT separates the observation into frequency bins, some of which are expected to contain signal information based on the signal of interest known frequency profile, and some of which are expected to contain only noise.

Let $\boldsymbol{\gamma_z}(k', f)$ be the STFT of the received observation $\boldsymbol{z}(k)$, where k' represents the time index used in the time-frequency expansion [14]. Then its squared modulus $|\boldsymbol{\gamma_z}(k', f)|^2$ is the spectrogram, a time-frequency representation of the observation power spectral density (PSD). To obtain a good estimate of the noise floor, we apply a median filter to each frequency sub-band of the STFT of the observation. Since a median filter replaces outliers with the median of the neighboring samples, choosing a filter-order much longer than the signal duration will smooth out the signal contributions if any are present within the median filter window. Let the resulting time-frequency estimate of the background noise be denoted $\left|\boldsymbol{\gamma}_{\tilde{\boldsymbol{\eta}}}(k', f)\right|$ [5]. By the Wiener-Khintchine theorem, the Fourier transform of the correlation function of a random process is its PSD [6]. As a result, an estimate of the noise autocorrelation function $r_z(k')$ may be obtained as the inverse Fourier transform of the PSD of the observation after it is median-filtered [5]. Then we can estimate the noise autocovariance matrix $\boldsymbol{\Gamma}_{\tilde{\boldsymbol{\eta}}_0}$ by creating a Toeplitz matrix out of the autocorrelation function $r_z(k')$.

The goal of online analysis is to get a time-dependent estimate of the SNR $\rho[k]$ in order to determine $Q[k]$, the number of terms in the overall SMF filter expression to apply to the observation at time k; however, this is challenging in a noisy environment, as false signal detection may occur due to the presence of unpredictable wideband transient signals in the observation. These transients are neither noise, nor the signal of interest, but may trigger signal detection since they may occur within and near the signal of interest

bandwidth. Limiting false alarm occurrences can be accomplished by comparing both the observation $\boldsymbol{\gamma_z}(k', f)$ and the estimated noise $\left|\boldsymbol{\gamma_{\tilde{\eta}}}(k', f)\right|$ in the time-frequency domain [12]. Using the STFT, we divided the observation into frequency bins based on the known frequency profile of the signal of interest. Let set A contain the frequency bins where signal would be expected (signal bandwidth), and let set B contain the frequency bins where the signal would not be expected [5].

To estimate the online SNR value, there are three steps following the STFT and median filtering of the observation. First, we estimate the presence of the signal of interest for each time sample k' within the signal bandwidth A by comparing the maximum value of the modulus of the observation against the mean value of the noise estimate [12]:

$$SOI[k'] = \frac{\max_f\{|\boldsymbol{\gamma_z}(k', f)|\}}{\overline{\left|\boldsymbol{\gamma_{\tilde{\eta}}}(k', f)\right|}}, f \in A. \tag{41}$$

If the expression shown in (41) is greater than 1, we can assume the presence of something other than noise; however, we cannot not yet distinguish between transients and the signal of interest [5]. Second, we estimate the presence of these transients by finding the maximum ratio between the modulus of the time-frequency observation and the noise estimate within the frequency bins adjacent to the signal of interest frequency range [12]:

$$trans[k'] = \max_f \left|\frac{\boldsymbol{\gamma_z}(k', f)}{\boldsymbol{\gamma_{\tilde{\eta}}}(k', f)}\right|, f \in B. \tag{42}$$

Then, we determine the time-varying SNR $\rho[k']$ in decibels by taking the ratio of the two previous expressions $SOI[k']$ and $trans[k']$ [5]:

$$\rho[k'] = 20 \log\left(\frac{SOI[k']}{trans[k']}\right). \tag{43}$$

Finally, we interpolate our results from the time-frequency domain (k', f) back into the time domain (k) in order to determine the estimated SNR in real time $\rho[k]$ [12].

E. SUMMARY

In this section, we described the process used to estimate the online SNR value $\rho[k]$ and to evaluate $Q[k]$, the number of terms present in the SMF impulse response expression derived for each time sample of the observation. Then we can approximate the signal by applying the appropriate number of terms in the filter expression created offline, allowing us to determine the start point and duration of the signal of interest present within the observation [5]. Figure 4 shows a block diagram of the online analysis algorithm. The next chapter discusses the conduct of the experiment, to include the simulation of the noisy observation and the parameters used to build the filters and approximate the signal of interest.

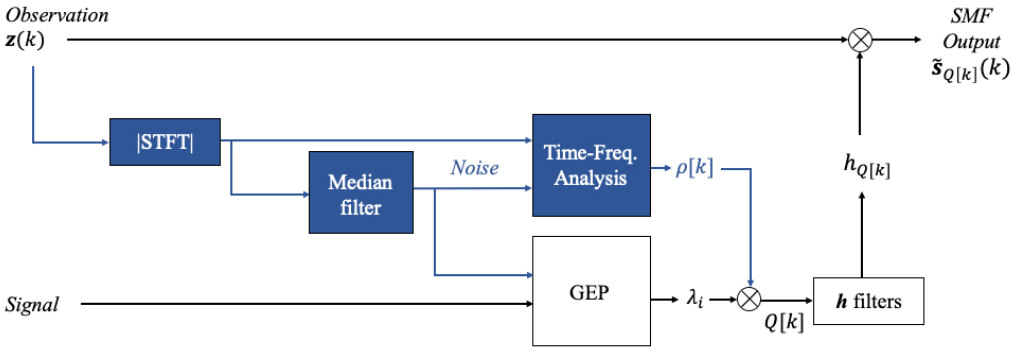

Figure 4. Online SMF processing diagram. Adapted from [5].

IV. EXPERIMENT

In this section we present the details of our implementation of the SMF discussed earlier as applied to the detection and reconstruction of tonal signals. Using MATLAB, we generated the signals and embedded them in white noise at fixed SNR values. We then conducted offline analysis to create the filter bank for each signal of interest. Finally, we processed the noisy signal using online analysis, resulting in an approximation of the original signal. Though the goal was detection of the signal, recreating the signal also allowed us to determine its start time and duration when present amongst noise. First, we discuss the experimental set up, starting with signal and noise simulation parameters. Next, we present the approach to building the filter bank. Finally, we describe the process applied to approximate the signal.

A. NOISY SIGNAL SIMULATION

1. Simulated Signals

In this experiment we tested the SMF on cosine tones and three types of chirps to simulate basic signals that may be encountered in military acoustic and radar applications. The simulated cosine signal contained a single frequency and is represented by a sequence of M samples:

$$\boldsymbol{s}(k) = \alpha \cos\left(2\pi f_0 \frac{k}{f_s}\right), \tag{44}$$

where $k = 1, \dots, M$ and $M = 400$. We fixed the noise power level and calculated the signal amplitude α to achieve the signal power that would result in a specific SNR value. We used base frequencies $f_0 = 0.2\,Hz$ and $0.8\,Hz$, which are scaled down by two orders of magnitude from a typical acoustic signal frequency to keep the processing time manageable for the simulation. We chose a fairly high sampling rate of $f_s = 25 f_0$ to ensure good signal resolution.

Next, we considered chirp signals, which are nonstationary tones with changing frequencies, presenting a challenge to the SMF. In this experiment, we used linear,

quadratic, and exponential chirps simulated using the MATLAB function *chirp.m*. Once again, the signal amplitude α was determined according to the desired SNR level. We set the chirps to have a maximum frequency $f_0 = 0.2\ Hz$, and a minimum frequency f_Δ that was either 20% lower than f_0 ($f_\Delta = 0.1667\ Hz$), or 80% lower than f_0 ($f_\Delta = 0.1111\ Hz$), in order to test the effect of chirp bandwidth on the SMF. Since f_0 is the highest frequency present in the chirp signal, we again selected a sampling frequency $f_s = 25f_0$.

2. Simulated Noisy Observation

We simulated a white noise sequence $\boldsymbol{\eta}(k)$ by creating a vector of $N = 2000f_s$ normally distributed random variables with a mean of 0 and variance of 1 using the MATLAB function *randn.m* to add noise to the signal. We then embedded the signal $\boldsymbol{s}(k)$ of length $M = 400f_s$ in the noise sequence so that the observation was much longer than the signal of interest. We set the signal to start near the center of the sequence, $k = 1000f_s$, where it was combined with the noise sequence for its duration M. As described in Section III.C, the resulting observation signal $\boldsymbol{z}(k)$, a sequence of N samples was broken down into windows of length $L = 101$ to ensure low levels of frequency variations within a given time window. According to Bonnal, Danes, and Renaud [3], we can consider the signal as wide sense stationary (WSS) within each of these windows. Each odd-length window was indexed by its center point $z[k]$,

$$\boldsymbol{z}(k) = \left\{z\left[k - \frac{L-1}{2}\right], \dots, z[k], \dots, z\left[k + \frac{L-1}{2}\right]\right\}. \tag{45}$$

Breaking the observation into overlapping windows will allow us to eventually evaluate its properties in real-time. First, however, we use the characteristics of the known signal of interest $\boldsymbol{s}(k)$ and modeled noise $\boldsymbol{\eta}(k)$ to conduct offline analysis.

B. OFFLINE ANALYSIS

1. Determining Second-Order Signal Characteristics

As discussed in Section III.D.1, the first step in creating the filter bank is estimating the signal and noise autocovariances using a priori knowledge of the signal and noise characteristics. In this experiment, we used the MATLAB function *xcorr.m* to calculate the

autocorrelation sequence of each signal type. We chose a maximum lag equal to window length $L = 101$, so that only the first 101 points were used. The sequence was normalized so that at a lag of 0, the autocorrelation was equal to 1. We also subtracted the signal mean from the signal $\boldsymbol{s}(k)$ when calculating the autocorrelation, which resulted in the autocovariance sequence (note that the mean of the signal is very close to zero to start with). Since the behavior of the signal within each window is approximated as wide sense stationary (WSS), the signal autocovariance matrix $\boldsymbol{\Gamma_s}$ is a $L \times L$ Toeplitz matrix generated from the autocovariance sequence [8]. Next, we normalized the covariance matrix with respect to its trace by dividing each of its elements by the sum of its own eigenvalues, which were calculated using the MATLAB function *eig.m*, to produce the reduced signal autocovariance matrix $\boldsymbol{\Gamma}_{\boldsymbol{s}_0}$.

To calculate the reduced noise autocovariance matrix $\boldsymbol{\Gamma}_{\boldsymbol{\eta_0}}$, we followed the same procedure as above, with the simulated noise sequence $\boldsymbol{\eta}(k)$. We used the $L \times L$ matrix $\boldsymbol{\Gamma}_{\boldsymbol{\eta_0}}$ to create the filter bank during offline analysis. During online processing, the estimated reduced noise covariance matrix, denoted $\boldsymbol{\Gamma}_{\tilde{\boldsymbol{\eta}}_0}$, will be calculated again as the observation is processed in real-time (see Section IV.C.1 for details).

2. Building the Filter Bank

As discussed in Section III.D.1, we set the maximum value of $Q[k]$ to a predetermined length equal to the number of eigenvalues of the noise autocovariance that are in the top ten percentile [5]. Let this predetermined offline maximum value of $Q[k]$ be defined as $Q[k]_{max_0}$, the maximum dimension of the subspace which optimizes the SNR. Next, applying the reduction properties of the KLE as discussed in Section II.B.2, we projected the signal and noise autocovariance matrices $\boldsymbol{\Gamma}_{\boldsymbol{s}_0}$ and $\boldsymbol{\Gamma}_{\boldsymbol{\eta_0}}$ onto the optimal reduced-dimension signal subspace by following the transformation approach proposed by Bouffaut [15], defined as:

$$\begin{cases} \tilde{\boldsymbol{\Gamma}}_{\boldsymbol{s}_0} = \boldsymbol{V}_{\boldsymbol{s},l}^T \boldsymbol{\Gamma}_{\boldsymbol{s}_0} \boldsymbol{V}_{\boldsymbol{s},l} \\ \tilde{\boldsymbol{\Gamma}}_{\boldsymbol{\eta}_0} = \boldsymbol{V}_{\boldsymbol{s},l}^T \boldsymbol{\Gamma}_{\boldsymbol{\eta}_0} \boldsymbol{V}_{\boldsymbol{s},l} \end{cases}, \forall\, l = 1, \dots, Q[k]_{max_0}, \tag{46}$$

where $\tilde{\boldsymbol{\Gamma}}_{\boldsymbol{s}_0}$ and $\tilde{\boldsymbol{\Gamma}}_{\boldsymbol{\eta}_0}$ are the projected $Q[k]_{max_0} \times Q[k]_{max_0}$ signal and noise autocovariance matrices, respectively. In addition, $\boldsymbol{V}_{\boldsymbol{s},l}$ is defined as a matrix of $L \times 1$ eigenvectors obtained from the signal autocovariance matrix, with its columns sorted by the associated eigenvalues in descending order from 1 to L. Note that in (46), we truncate $\boldsymbol{V}_{\boldsymbol{s},l}$, taking only the signal eigenvectors that correspond to the $Q[k]_{max_0}$ highest eigenvalues, which define the optimal signal subspace [10]. Next, we used the projected signal and noise autocovariance matrices $\tilde{\boldsymbol{\Gamma}}_{\boldsymbol{s}_0}$ and $\tilde{\boldsymbol{\Gamma}}_{\boldsymbol{\eta}_0}$ to solve for the GEP defined in (30), which becomes

$$\tilde{\boldsymbol{\Gamma}}_{\boldsymbol{s}_0} \tilde{\boldsymbol{\Phi}}_l = \lambda_l \tilde{\boldsymbol{\Gamma}}_{\boldsymbol{\eta}_0} \tilde{\boldsymbol{\Phi}}_l, \forall\, l = 1, \dots, Q[k]_{max_0}, \tag{47}$$

where $\tilde{\boldsymbol{\Phi}}_l$ is a matrix of $Q[k]_{max_0} \times 1$ eigenvectors obtained from (47) sorted by the associated eigenvalues λ_l in descending order from 1 to L, which we again truncate to $Q[k]_{max_0}$. Next, let $\boldsymbol{\Phi}$ be defined as

$$\boldsymbol{\Phi} = \boldsymbol{V}_{\boldsymbol{s},l} \tilde{\boldsymbol{\Phi}}_l, \forall\, l = 1, \dots, Q[k]_{max_0}, \tag{48}$$

where $\boldsymbol{\Phi}$ is the matrix made up of L column basis vectors $\boldsymbol{\Phi}_l$, of which we consider only the first $Q[k]_{max_0}$ vectors. Each basis vector $\boldsymbol{\Phi}_l$ is normalized as follows:

$$\frac{1}{\sqrt{\boldsymbol{\Phi}_l^T \boldsymbol{\Gamma}_{\boldsymbol{\eta}_0} \boldsymbol{\Phi}_l}} \boldsymbol{\Phi}_l, \forall\, l = 1, \dots, Q[k]_{max_0}. \tag{49}$$

Using (31), we created basis vectors $\boldsymbol{\Psi}_l, \forall\, l = 1, \dots, Q[k]_{max_0}$, which are deterministic vectors of the noisy signal expansion [2]:

$$\boldsymbol{\Psi}_l = \boldsymbol{\Gamma}_{\boldsymbol{\eta}_0} \boldsymbol{\Phi}_l. \tag{50}$$

Finally, we built the filter bank by defining a filter $\boldsymbol{h}_{Q[k]}$ for each value of $Q[k]$ from 1 to $Q[k]_{max_0}$ using (39), which leads to:

$$\boldsymbol{h}_{Q[k]} = \sum_{l=1}^{\mathrm{Q[k]}} \boldsymbol{\Psi}_l \left[\frac{L+1}{2}\right] \boldsymbol{\Phi}_l. \tag{51}$$

As we process the signal in real-time, we will determine a $Q[k]$ value for each window of the observation $\boldsymbol{z}(k)$, and then apply the optimal filter $\boldsymbol{h}_{Q[k]}$ to that portion of the online signal.

C. ONLINE ANALYSIS

1. Estimating Online Noise Covariance

When analyzing a noisy observation, we must understand noise characteristics to process the data and detect the signal of interest. During offline processing, we built a filter bank using the autocovariance function of a white noise model. For online processing, noise parameters are unknown until processing begins. The goal of estimating the online noise autocovariance is to determine the filter parameter $Q[k]$ needed to process each data sample as it becomes available in real-time. Recall we do not know whether the signal is present at a given time or whether we have noise only in real-time scenarios. Since the signal duration is short compared to the total duration of the observation, and we know the signal of interest frequencies, we can estimate the noise characteristics using median filtering and time-frequency analysis as discussed in Section III.C.2.

In order to get an estimate of the noise characteristics, we chose our parameters for the STFT and median filters as follows. First, we normalized the received observation $\boldsymbol{z}(k)$ by dividing each element by the maximum value of $\boldsymbol{z}(k)$. Then we performed an STFT on the data using the MATLAB function *spectrogram.m*. We chose a window length $L = 101$ to match the length of the signal autocovariance sequence. We used a Hanning window to reduce spectral leakage, and set the window overlap to 98% to avoid information loss between windows. We zero-padded the Fast Fourier transform to 1024 to ensure high spectral resolution [14]. The resulting STFT is a representation of the observation in the time-frequency domain $\boldsymbol{\gamma}_z(k', f)$, where k' is a time sample in the time-frequency domain and f is the frequency. An example of such a spectrogram obtained for a cosine signal centered at $f_0 = 0.2\ Hz$ with an SNR value of 6 dB is illustrated in Figure 5a. In order to mitigate any contributions from the signal that would distort the characteristics of the noise, we used the function *medfilt1.m* to process the absolute value of the STFT $|\boldsymbol{\gamma}_z(k', f)|$ through a median filter twice as long as the signal of interest, ensuring any transients caused

by signal presence would be smoothed out. This filter of length $2M = 800$ was applied to each frequency sub-band through time. The resulting estimation of the time-frequency behavior of the noise is $\left|\boldsymbol{\gamma}_{\tilde{\boldsymbol{\eta}}}(k', f)\right|$. A plot of the noise estimation of an observation with SNR value of 6 dB, and cosine signal centered at $f_0 = 0.2\ Hz$ is illustrated in Figure 5b. Note that the 0.2Hz tone frequency has been smoothed out by the median filter.

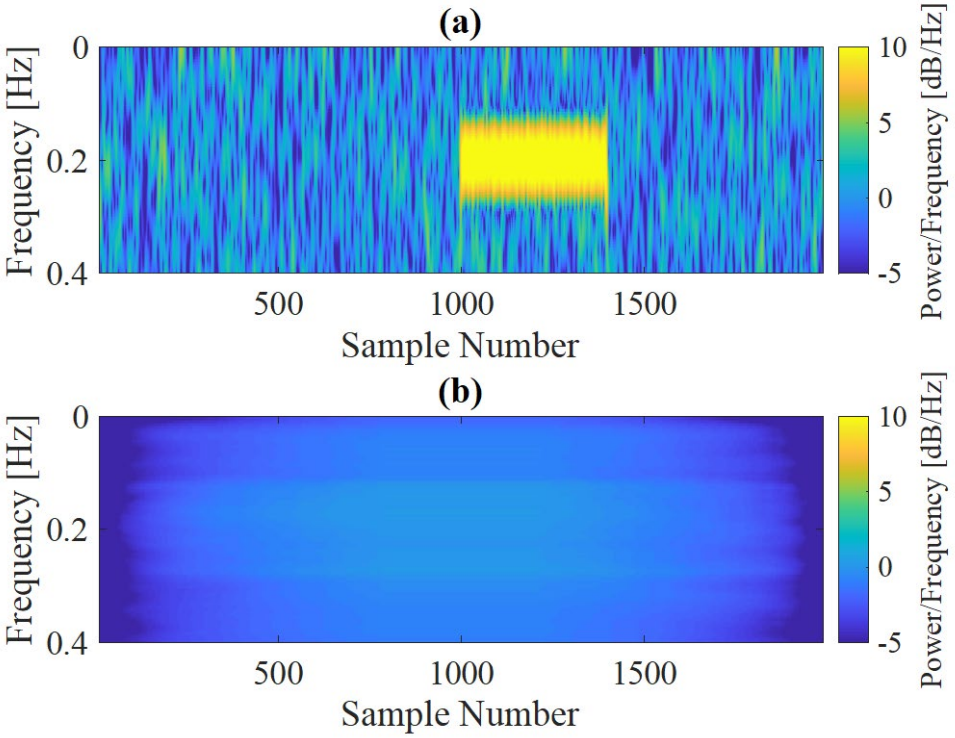

Figure 5. Spectrogram plots of a cosine signal centered at 0.2 Hz embedded in noise at 6dB SNR. (a) Noisy observation. (b) Estimated noise.

By the Wiener-Khintchine theorem, we can calculate the autocorrelation function of a WSS process by taking the inverse discrete Fourier transform of the power spectral density (PSD) [8]. The PSD of the estimated noise can be expressed as $\left|\boldsymbol{\gamma}_{\tilde{\boldsymbol{\eta}}}(k', f)\right|^2$, the magnitude squared of the median filtered observation STFT. We then apply the Wiener-Khintchine theorem by taking the inverse Fourier transform of the PSD. The result is $\boldsymbol{\Gamma}_{\tilde{\boldsymbol{\eta}}}$,

the autocovariance matrix of the STFT estimated online noise, where each row corresponds to a frequency f and each column corresponds to a time sample k'.

Because we assume the observation is WSS in each window, the autocovariance matrix of the online noise, which we will call $\boldsymbol{\Gamma}_{\tilde{\boldsymbol{\eta}}_0}$, must be a Toeplitz matrix where the diagonal elements represent the covariances between the noise samples with the same lag [6]. By summing each row of $\boldsymbol{\Gamma}_{\tilde{\boldsymbol{\eta}}}$, we combined all the elements of each frequency f, collapsing $\boldsymbol{\Gamma}_{\tilde{\boldsymbol{\eta}}}$ into an autocovariance sequence evaluated at each time sample k'. We then interpolated the sequence to evaluate the autocovariance at L frequencies from 0 to $f_s/2$. Using the function *toeplitz.m*, we created the estimated online noise autocovariance matrix $\boldsymbol{\Gamma}_{\tilde{\boldsymbol{\eta}}_0}$ (a $L \times L$ toeplitz matrix) out of the autocovariance sequence. Using *eig.m*, we computed the eigenvalues $\lambda_{\tilde{\boldsymbol{\eta}}}$ of the noise autocovariance matrix $\boldsymbol{\Gamma}_{\tilde{\boldsymbol{\eta}}_0}$, and we let $Q[k]_{max}$ be the number of eigenvalues $\lambda_{\tilde{\boldsymbol{\eta}}} > 1$, since this choice results in improving the SNR value, and set that value to be the high limit of the value $Q[k]$. Note that the maximum value of $Q[k]$ derived from online processing, which we called $Q[k]_{max}$, was set to be less than or equal to the maximum $Q[k]$ derived from offline preprocessing, which we called $Q[k]_{max_0}$[5].

Next, we projected the $L \times L$ estimated online noise autocovariance matrix $\boldsymbol{\Gamma}_{\tilde{\boldsymbol{\eta}}_0}$ onto the signal subspace, using the same signal eigenvector matrix $\boldsymbol{V}_{\boldsymbol{s},l}$ that we used to build the filter bank in Section IV.B.2 [15]:

$$\tilde{\boldsymbol{\Gamma}}_{\tilde{\boldsymbol{\eta}}_0} = \boldsymbol{V}_{\boldsymbol{s},l}^T \boldsymbol{\Gamma}_{\tilde{\boldsymbol{\eta}}_0} \boldsymbol{V}_{\boldsymbol{s},l}\,, \forall\, l = 1, \dots, Q[k]_{max}, \tag{52}$$

where $\tilde{\boldsymbol{\Gamma}}_{\tilde{\boldsymbol{\eta}}_0}$ is the resulting $Q[k]_{max} \times Q[k]_{max}$ estimated online noise autocovariance matrix. Then we solved the GEP with the projected online estimated noise autocovariance matrix $\tilde{\boldsymbol{\Gamma}}_{\tilde{\boldsymbol{\eta}}_0}$,

$$\tilde{\boldsymbol{\Gamma}}_{\boldsymbol{s}_0} \boldsymbol{V}_{\boldsymbol{s}} = \tilde{\lambda}_l \tilde{\boldsymbol{\Gamma}}_{\tilde{\boldsymbol{\eta}}_0} \boldsymbol{V}_{\boldsymbol{s}}, \tag{53}$$

where $\tilde{\lambda}_l$ are the sorted eigenvalues and $\boldsymbol{V}_{\boldsymbol{s}}$ is the matrix of associated eigenvectors that solve the GEP. These eigenvalues, along with the online SNR estimates, determine the filter parameter $Q[k]$ for each time sample k of the observation during real-time processing, allowing us to choose the optimal filter for each window.

2. Estimating Online SNR

Once we determined the eigenvalues from (53) using the estimated noise autocovariance, we applied the three steps described in Section III.D.2 to estimate the SNR of each sample window. Since we had completed time-frequency analysis of the observation, we first estimated whether or not the signal frequency was present within each time window. Next, we estimated the presence of unwanted transient signals in each window. Finally, we compared these results to estimate the SNR for each time window.

Time-frequency analysis using the STFT separated the noise from the signal by dividing the observation into overlapping time windows that were also divided by frequency from 0 to $f_s/2$, resulting in data separated into consecutive frequency "bins." We chose the frequency bins that contained the frequencies of our signal of interest and called them $Set\ A$. We assumed the remaining frequency bins contained frequencies that only contributed to noise or other transient signals present in the environment. We called this group of frequencies $Set\ B$. In the single frequency cosine signal scenario, we let $Set\ A$ contain the range of frequencies 10% lower than the tone frequency f_0 up to frequencies 10% higher than f_0, while $Set\ B$ contained the range of frequencies 50% lower than f_0 and below, as well as frequencies 50% higher than f_0 and above:

$$\begin{matrix} A \in [f_0 - 0.1f_0, f_0 + 0.1f_0] \\ B \in [0, f_0 - 0.5f_0] \cap [f_0 + 0.5f_0, f_s/2] \end{matrix}. \tag{54}$$

In the chirp signal scenario, we let $Set\ A$ contain the range of frequencies 10% lower than f_Δ, the lowest frequency in the chirp bandwidth, to frequencies 10% of f_Δ higher than f_0, while $Set\ B$ contained the range of frequencies 50% lower than f_Δ and below, as well as frequencies 50% of f_Δ higher than f_0 and above:

$$\begin{matrix} A \in [f_\Delta - 0.1f_\Delta, f_0 + 0.1f_\Delta] \\ B \in [0, f_\Delta - 0.5f_\Delta] \cap [f_0 + 0.5f_\Delta, f_s/2] \end{matrix}. \tag{55}$$

Separating the observation into $Set\ A$ and $Set\ B$, or signal-only frequencies and noise-only frequencies respectively, we estimated the presence of the signal by maximizing the STFT of the observation $|\boldsymbol{\gamma}_z(k', f)|$ and dividing by the mean noise estimate

$\overline{|\boldsymbol{\gamma}_{\tilde{\eta}}(k', f)|}$ for each frequency bin where signal could be expected ($Set\ A$), as referenced in (41),

$$SOI[k'] = \frac{\max_f\{|\boldsymbol{\gamma}_z(k', f)|\}}{\overline{|\boldsymbol{\gamma}_{\tilde{\eta}}(k', f)|}}, f \in A. \tag{56}$$

We then estimated where unwanted transient signals would possibly occur by maximizing the ratio of the STFT of the observation $\boldsymbol{\gamma}_z(k', f)$ and the noise estimate $\boldsymbol{\gamma}_{\tilde{\eta}}(k', f)$ for each frequency bin that would not contain frequencies found in the signal of interest ($Set\ B$), as illustrated by (42),

$$trans[k'] = \max_f \left| \frac{\boldsymbol{\gamma}_z(k', f)}{\boldsymbol{\gamma}_{\tilde{\eta}}(k', f)} \right|, f \in B. \tag{57}$$

Both $SOI[k']$ and $trans[k']$ expressions were median filtered using a filter order equal to 20% of the length of the sequence to smooth out any outliers and interpolated from the time-frequency domain (k', f) back into the original time domain (k) [12]. The online SNR value $\rho[k]$ was estimated as the ratio of $SOI[k]$ to $trans[k]$ defined previously by (43).

$$\rho[k] = 20 \log\left(\frac{SOI[k]}{trans[k]}\right). \tag{58}$$

Thus, we created an estimated SNR $\rho[k]$ expressed in decibels for each time sample k, where k represents the center sample of the k^{th} time window, as discussed in Section III.D.2. Figure 6 shows a plot of the estimated online SNR values for a cosine signal centered at $f_0 = 0.2\ Hz$ with an input SNR value of 6 dB.

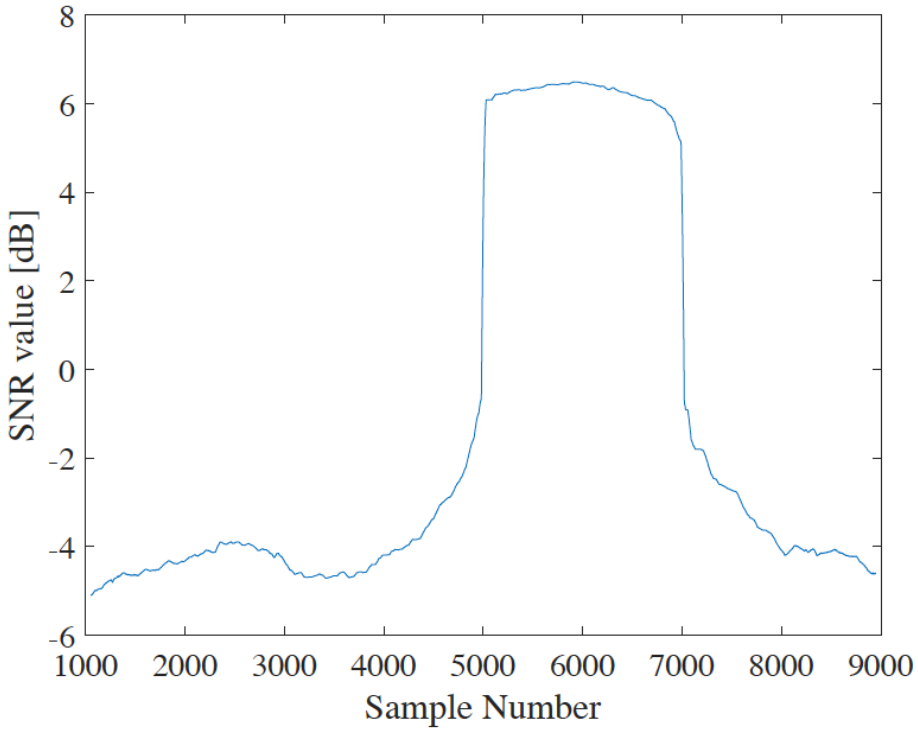

Figure 6. Estimated online SNR for 0.2 Hz cosine signal at SNR of 6dB

D. APPROXIMATING THE SIGNAL OF INTEREST

Once we determined the online SNR for each time sample, we used the inequality (37) to find the parameter $Q[k]$, which represents the number of eigenvalues $\tilde{\lambda}_l$ such that

$$\rho[k]\tilde{\lambda}_l > 1, \tag{59}$$

for each time sample k [2]. As discussed in Section III.C, the parameter $Q[k]$ determines which time-varying filter $\boldsymbol{h}_{Q[k]}$ from the offline bank we will apply to each window of the observation in order to maximize the SNR at the given time sample k. Then an approximation of the signal of interest obtained at time k can be obtained from the inner product of the window of the observation $\boldsymbol{z}(k)$ at sample k and the filter vector $\boldsymbol{h}_{Q[k]}$ as in (40):

$$\tilde{\boldsymbol{s}}_{Q[k]}[k] = \boldsymbol{z}(k)^T \boldsymbol{h}_{Q[k]}. \tag{60}$$

Here $\tilde{\boldsymbol{s}}_{Q[k]}[k]$ is a scalar approximation of the signal at time sample k, and $\boldsymbol{h}_{Q[k]}$ is the $L \times 1$ filter vector taken from the offline filter bank that corresponds to the appropriate value of $Q[k]$:

$$\boldsymbol{h}_{Q[k]} = \sum_{i=1}^{Q[k]} \boldsymbol{\Psi}_i \left[\frac{L+1}{2}\right] \boldsymbol{\Phi}_i \,. \tag{61}$$

This process ensures the optimal filter is used on each window with the goal of maximizing the SNR within the window [5]. Though our purpose for using the SMF is to determine whether a signal is present at a certain time rather than to approximate the signal itself, signal approximation allows us to evaluate the effectiveness of the SMF on different signals at varying SNR values. In the next section, we compare the approximated signal to the transmitted signal to determine the accuracy with which the SMF detects the signal presence. At each chosen SNR value, we will examine the similarities between the recreated signal and the original transmitted signal, as well as evaluate the proportion of the signal recreated using the SMF.

THIS PAGE INTENTIONALLY LEFT BLANK

V. EXPERIMENTAL RESULTS

The previous chapter provided an overview of our experimental process for signal reconstruction using the stochastic matched filter. This chapter will summarize the results of the experiment, including a discussion of the criteria and methods used to measure the effectiveness of the SMF for detecting signals in various SNR levels, followed by graphical depictions of our experimental results.

A. EVALUATION METHODS

First, we compared the transmitted signal $\boldsymbol{s}(k)$ to the signal approximated by the SMF, $\tilde{\boldsymbol{s}}_{Q[k]}[k]$, at various SNR values to evaluate the effectiveness of the SMF for signal detection, and chose two criteria to measure. Then, we evaluated the level of difference between the approximated signal and the original transmitted signal by measuring their similarity. Next, we determined the level of transmitted signal accurately generated, i.e., detected, by the SMF. Both evaluations were accomplished using a Monte Carlo simulation which we ran 100 times at SNR values from -6 dB to 15 dB, with means and 95% confidence intervals calculated using 5000 bootstrap data samples.

1. Evaluating Approximated Signal Accuracy

It is important to evaluate the accuracy of the SMF in recreating the transmitted signal because it is a measure of the capability of the SMF to recognize and detect the specific signal of interest. We accomplished this goal by running the SMF algorithm 100 times for each SNR value and taking the mean of the absolute value squared of the difference between the transmitted signal and the approximated signal:

$$\overline{\left|\boldsymbol{s}(k) - \tilde{\boldsymbol{s}}_{Q[k]}[k]\right|^2}. \tag{62}$$

Then we divided each difference by the transmitted signal power to normalize and subtracted it from 1 to get the percentage of similarity between the two signals. We used these values to calculate the means and 95% confidence intervals for 5000 bootstrap samples with the MATLAB functions *bootstrp.m* and *bootci.m*. For example, Figure 7

illustrates the differences present between transmitted and reconstructed approximated signal values obtained at a relatively high SNR value of 6 dB, though the entire duration of the signal is approximated accurately.

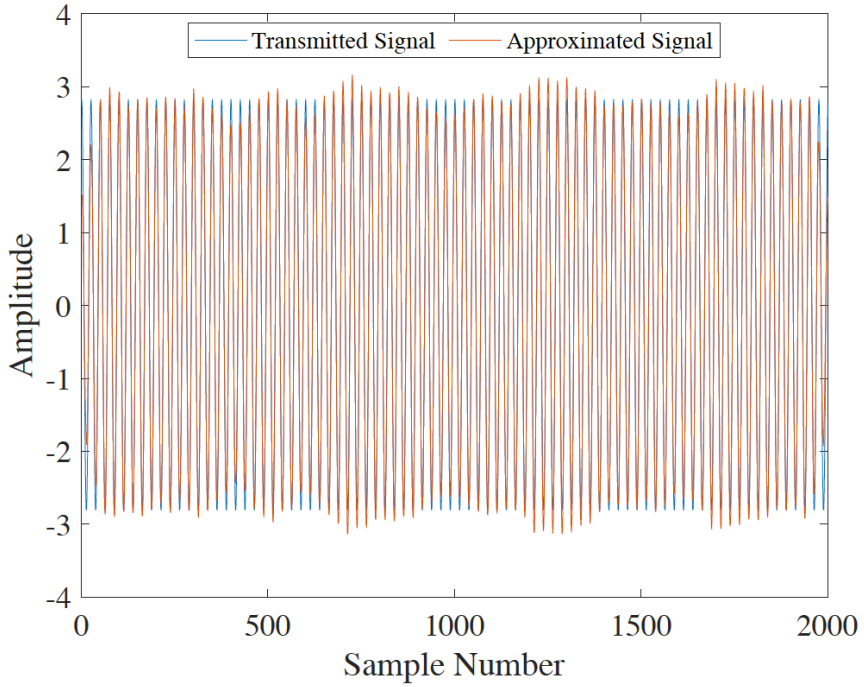

Figure 7. Comparison of transmitted cosine signal with $f_0 = 0.2\ Hz$ to SMF approximated signal at SNR value of 6 dB

2. Evaluating Approximated Signal Length

The second criterion selected to evaluate the SMF performance considers the proportion of the signal recovered by the SMF at various SNR values, and the proportion of the signal which may be missed by the SMF, which indicates a delay in signal detection. Again, we run the SMF algorithm 100 times at each SNR value, this time determining the duration of the approximated signal using the *find.m* function in MATLAB to detect the first and last indices where $Q > 0$, indicating the start and end of the approximated signal. We then divided the approximated signal length by the length of the transmitted signal to find the percentage of signal detected, and once again calculated means and 95% confidence intervals using the bootstrap method described above. Figure 8 illustrates that

only part of the signal has been approximated and thus detected for a relatively low SNR value of -4dB.

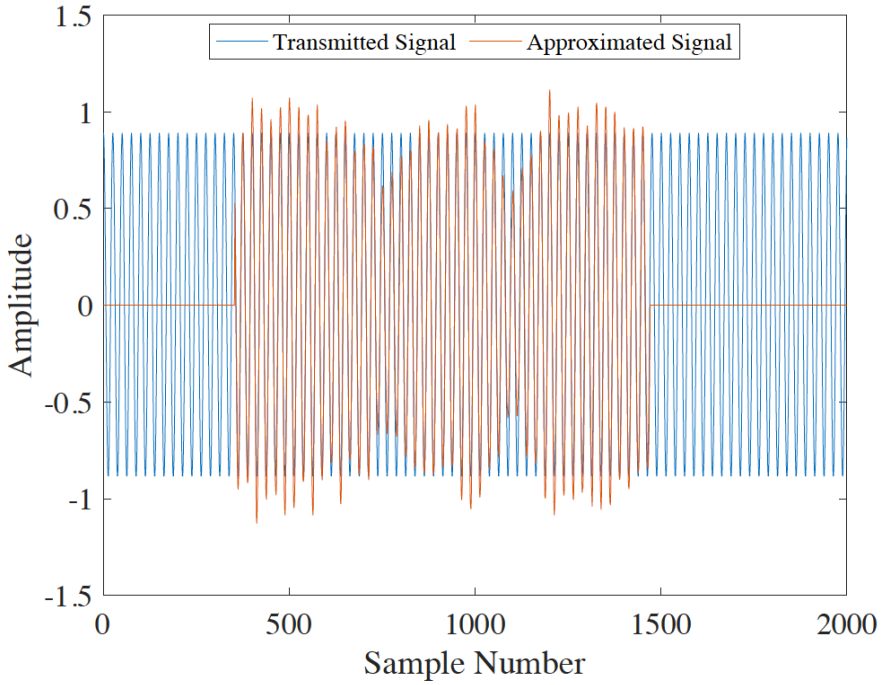

Figure 8. Comparison of transmitted cosine signal with $f_0 = 0.2\ Hz$ to SMF approximated signal at SNR value of -4 dB

B. RESULTS

Using the criteria and methods described in the previous section, we plotted the means and 95% confidence intervals for cosine signal and chirp signal approximations compared to transmitted signals. This section will display the resulting plots beginning with cosine signals centered at $f_0 = 0.2\ Hz$ and $f_0 = 0.8\ Hz$, followed by a brief discussion about the chirp signals we used in our experiment, and resulting plots.

1. Comparing Cosine Signal Results

When testing the SMF on simple cosine signals, we chose tone frequencies of $f_0 = 0.2\ Hz$ and $f_0 = 0.8\ Hz$ to compare SMF performance between signals with lower and higher frequencies, at a sampling frequency of $f_s = 25 f_0$. Figure 9 illustrates the accuracy

of the SMF in approximating cosine signals centered at both frequencies. At SNR values of -5 dB and under, the SMF approximation does not perform particularly well, but is more accurate for lower frequency cosine signals. However, we see slightly better results with fewer variations when approximating the higher frequency cosine signals at SNR values of -4 dB and higher. Figure 10 illustrates the detection capability of the SMF for cosine signals at both tone frequencies. Again, the SMF does not perform well at relatively low SNR values, but we see that it is slightly more successful with lower frequency signals at the lowest SNR values considered in the study. Interestingly, we also see that the lower frequency signals are more completely detected at higher SNR values as well. Note that the percent of signal detected is slightly higher than 100% at high SNR values. This is due to the use of overlapping windows. Residual indications of signal are present in the windows just before the signal begins and just after it ends.

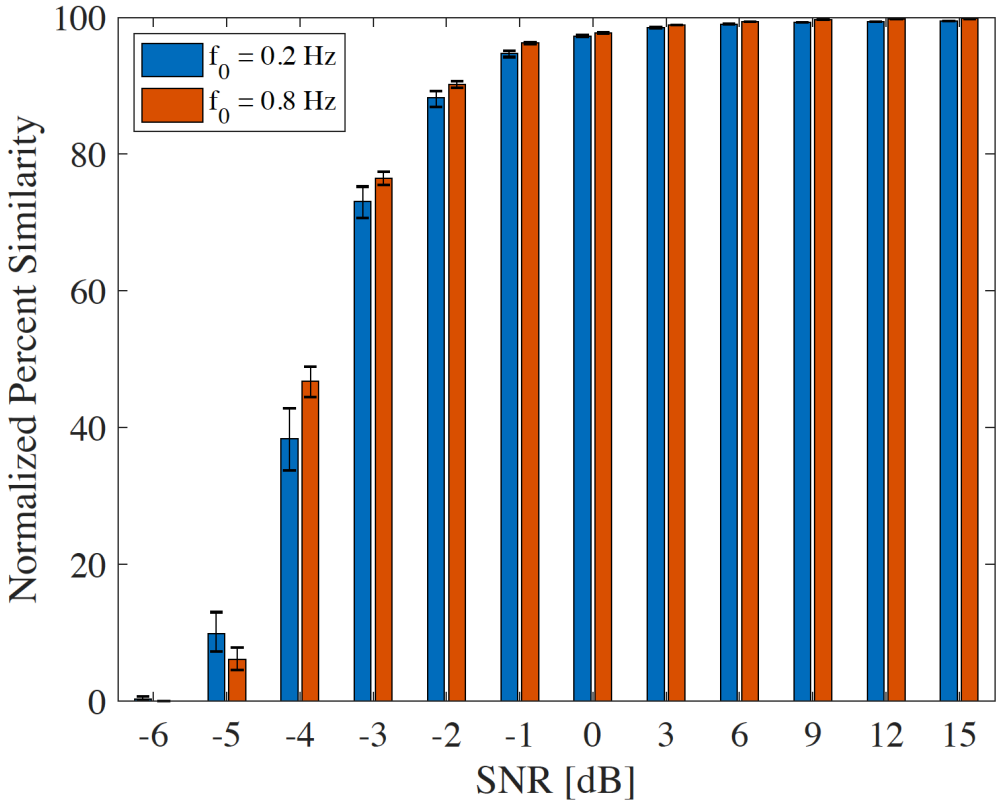

Figure 9. Normalized similarity between transmitted and reconstructed cosine signals - means and 95% confidence intervals

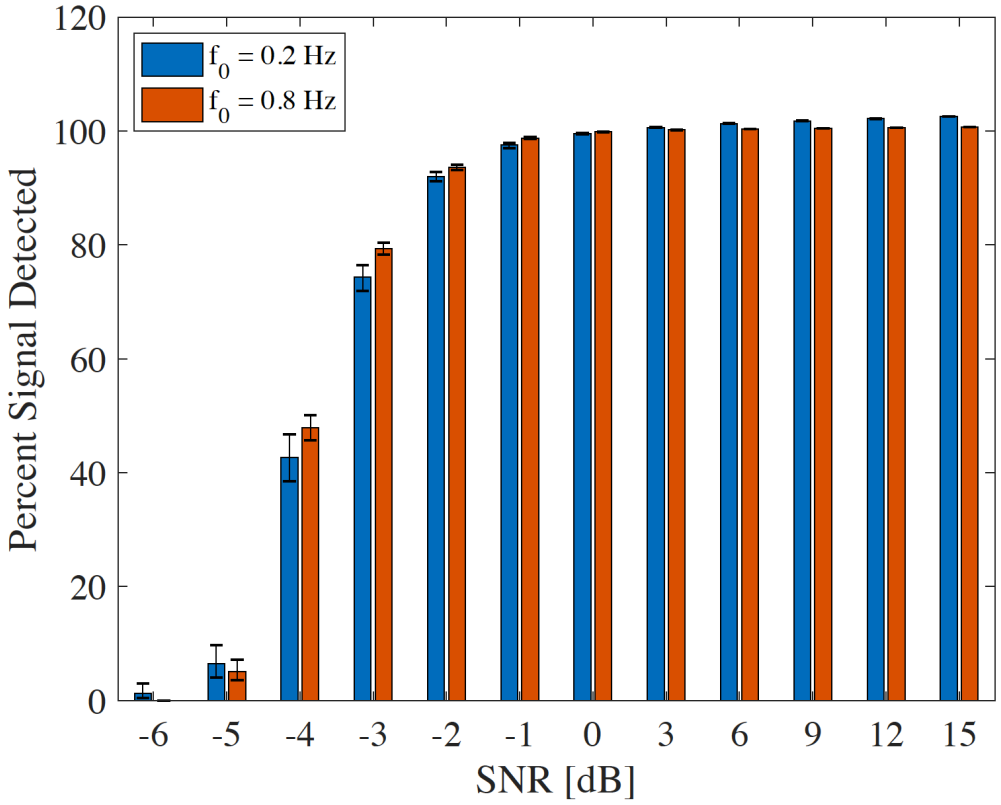

Figure 10. Percentage of cosine signal length detected using the SMF implementation - means and 95% confidence intervals

2. Comparing Chirp Signal Results

For this experiment, we tested chirp signals, also known as swept-frequency cosines, which are tones with time-varying frequency values. This change in frequency is referred to as the bandwidth of the signal. We tested three types of chirp signals: those that increased in frequency linearly, quadratically, and logarithmically. As described in Section IV.A.1, we chose $f_0 = 0.2\,Hz$ to be the highest frequency and set the lowest frequencies to be either 20% lower or 80% lower to test the effect of signal bandwidth on the SMF; the sampling frequency remains at $f_s = 25f_0$.

Results shown in Figures 11 and 12 indicate the SMF is decidedly more successful at approximating and detecting signals with smaller frequency variation (or smaller bandwidth) at SNR levels between -4 dB and 9 dB for linear chirp signals.

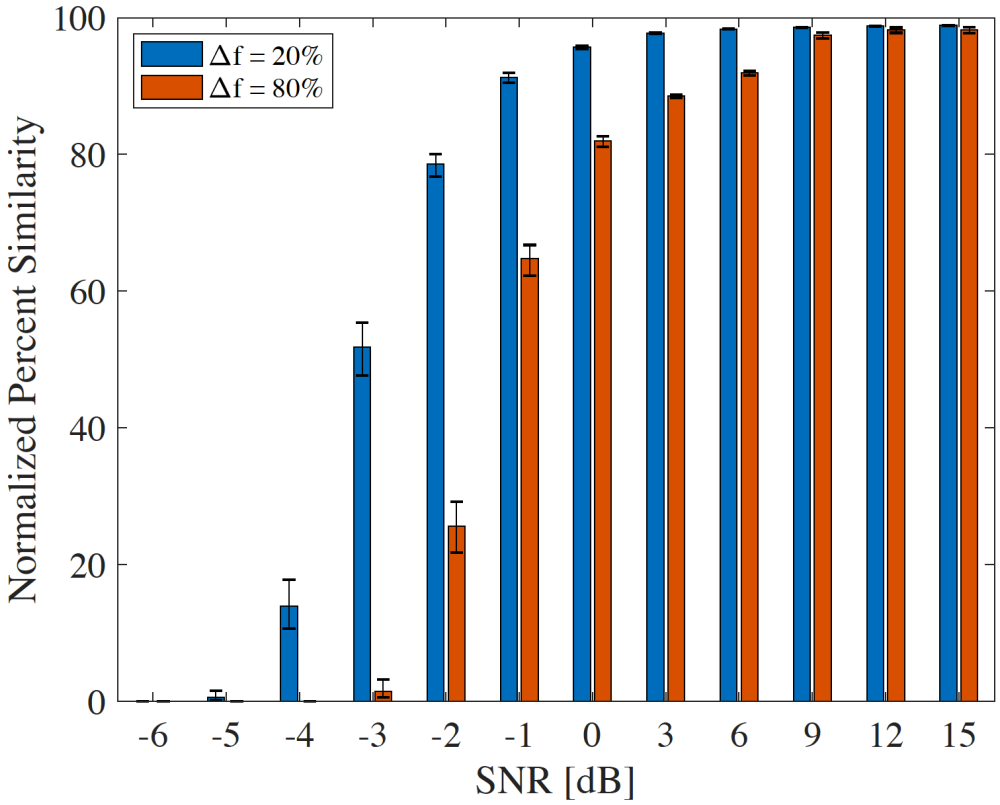

Figure 11. Normalized similarity between transmitted and reconstructed linear chirp signals - means and 95% confidence intervals

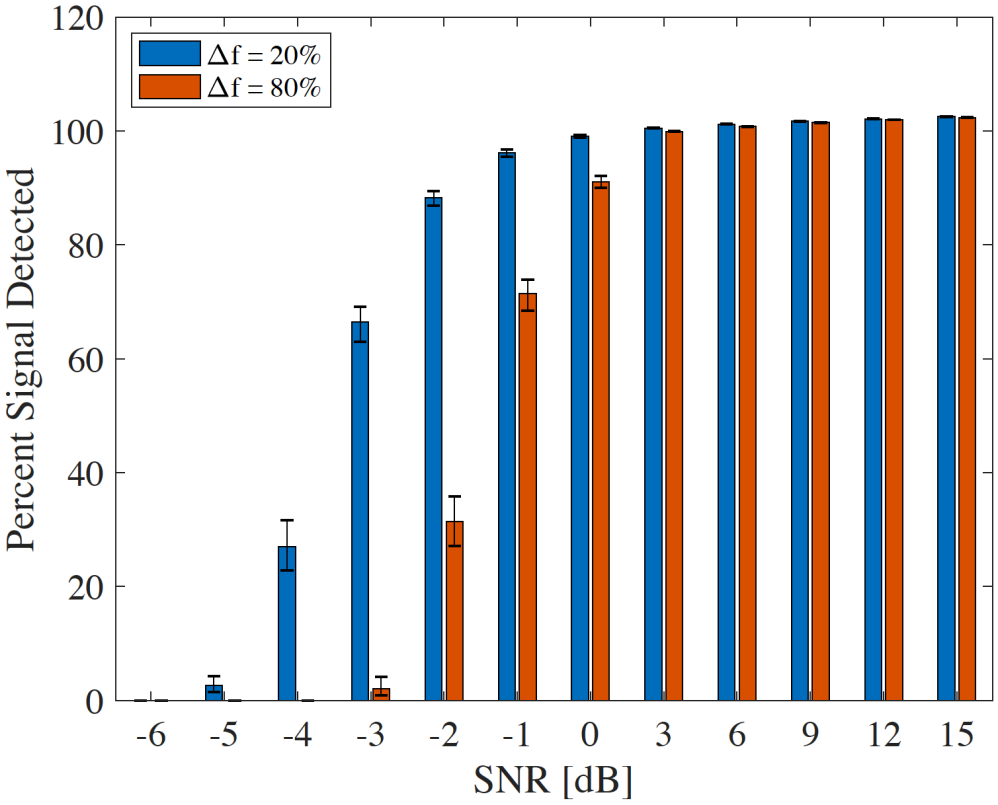

Figure 12. Percentage of linear chirp signal length detected using the SMF implementation - means and 95% confidence intervals

Similarly, we see a decrease in SMF performance when increasing the bandwidth of quadratic chirp signals, as illustrated in Figures 13 and 14.

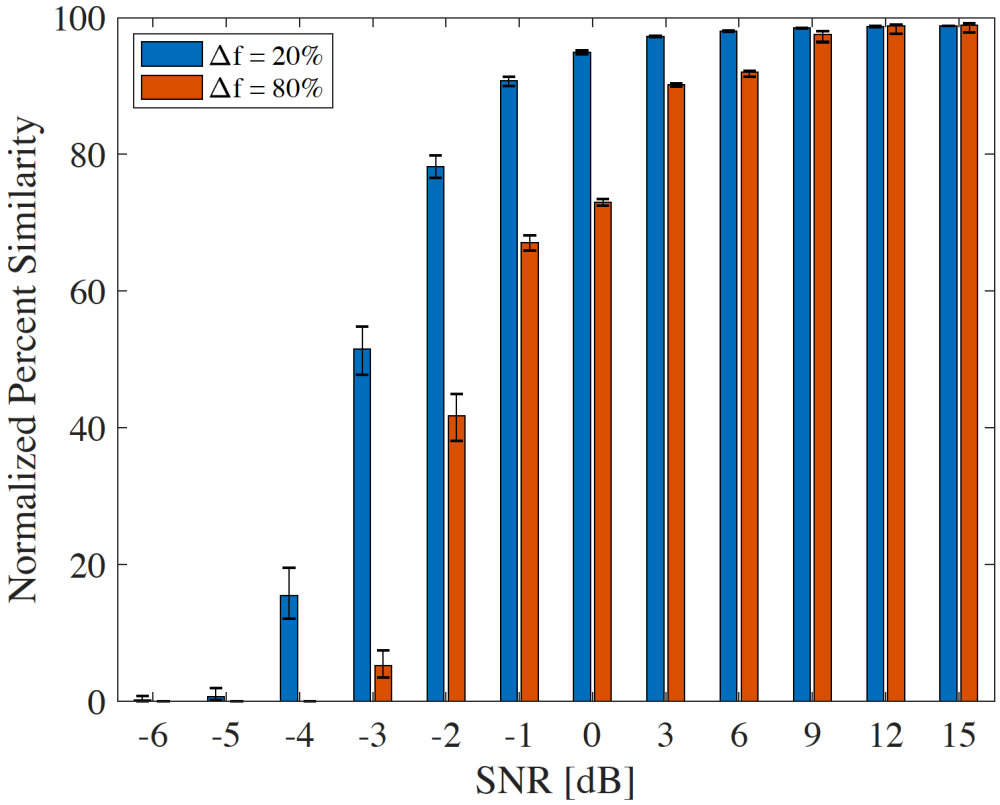

Figure 13. Normalized similarity between transmitted and reconstructed quadratic chirp signals - means and 95% confidence intervals

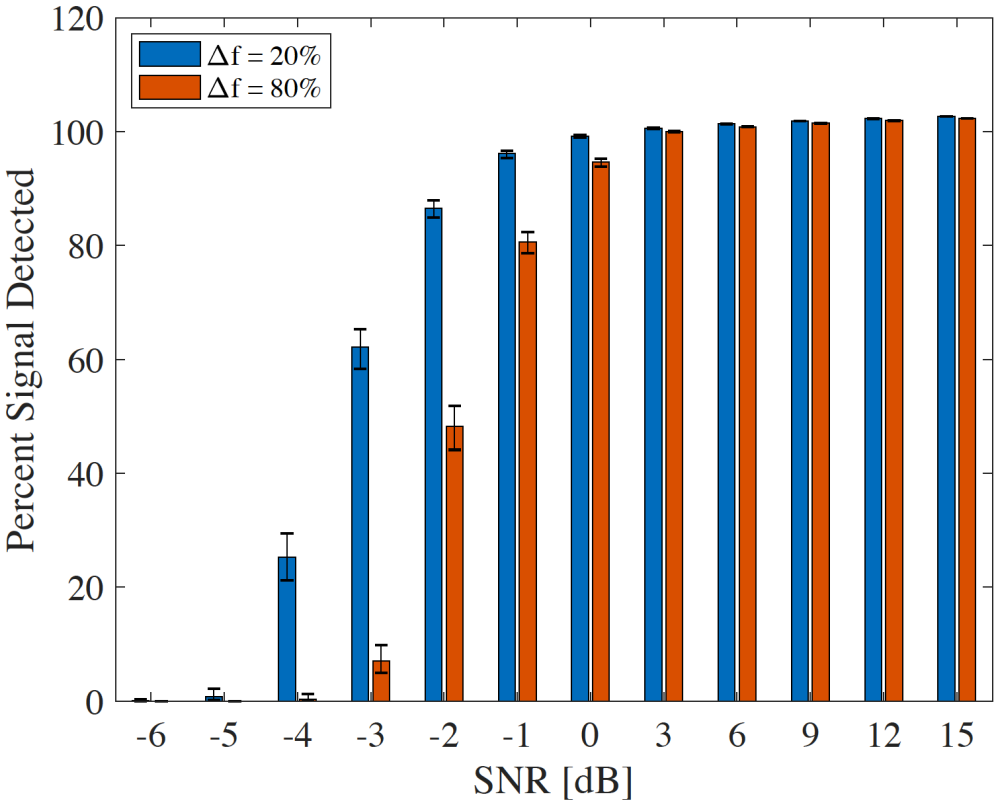

Figure 14. Percentage of quadratic chirp signal length detected using the SMF implementation - means and 95% confidence intervals

Figures 15 and 16 present the results of SMF approximation and detection of logarithmic chirp signals. As expected, the SMF performance in the mid-range SNR values is reduced for logarithmic chirp signals with higher bandwidths.

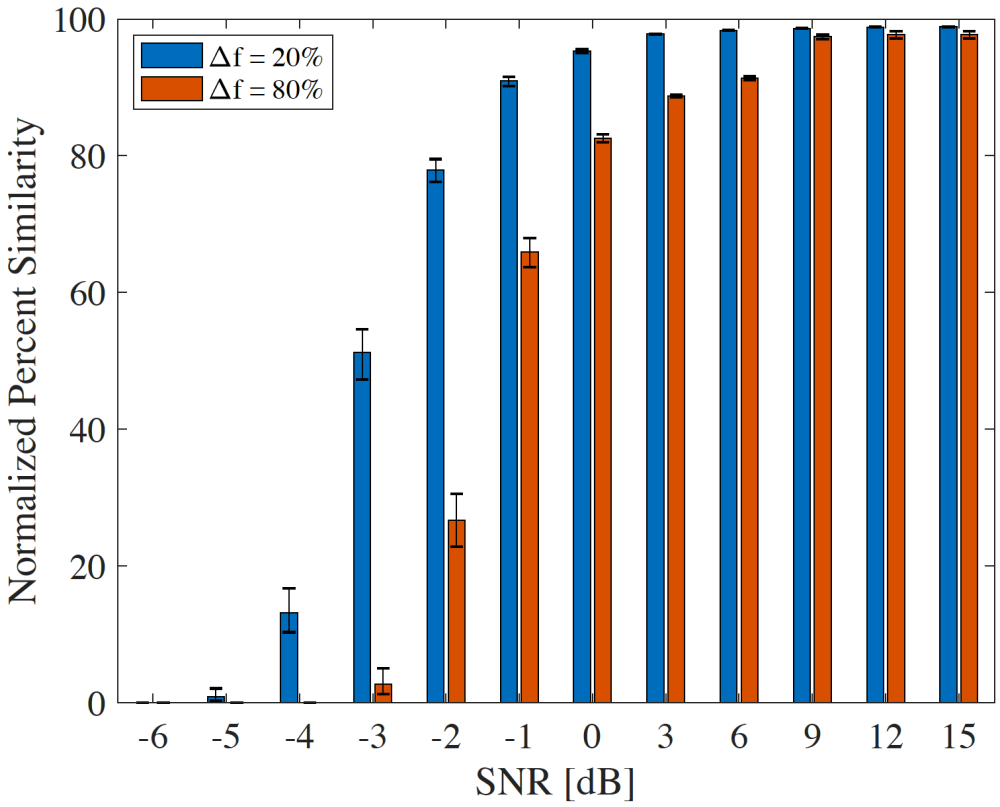

Figure 15. Normalized similarity between transmitted and reconstructed logarithmic chirp signals - means and 95% confidence intervals

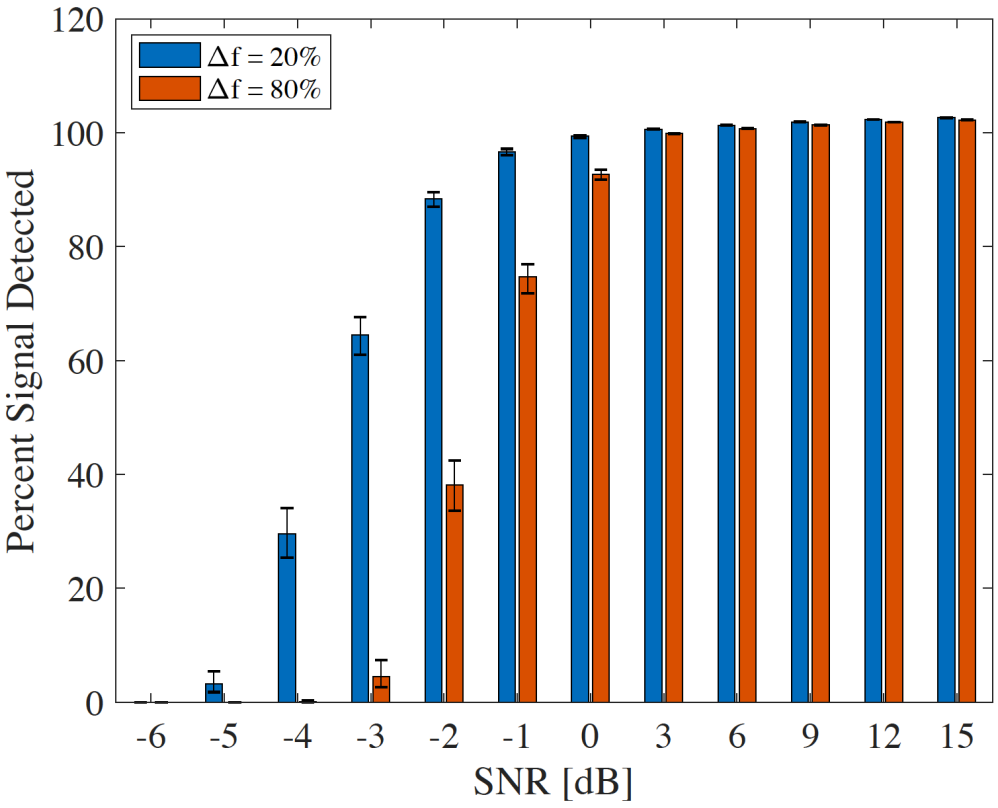

Figure 16. Percentage of logarithmic chirp signal length detected using the SMF implementation - means and 95% confidence intervals

Recognizing that bandwidth affects SMF performances for chirp signals, we also investigated whether chirp types would affect results. Figures 17 and 18 compare SMF performances for signals of each chirp type with a 20% change in frequency. For chirp signals with a smaller bandwidth, the chirp type had little effect on either accuracy or detection capability.

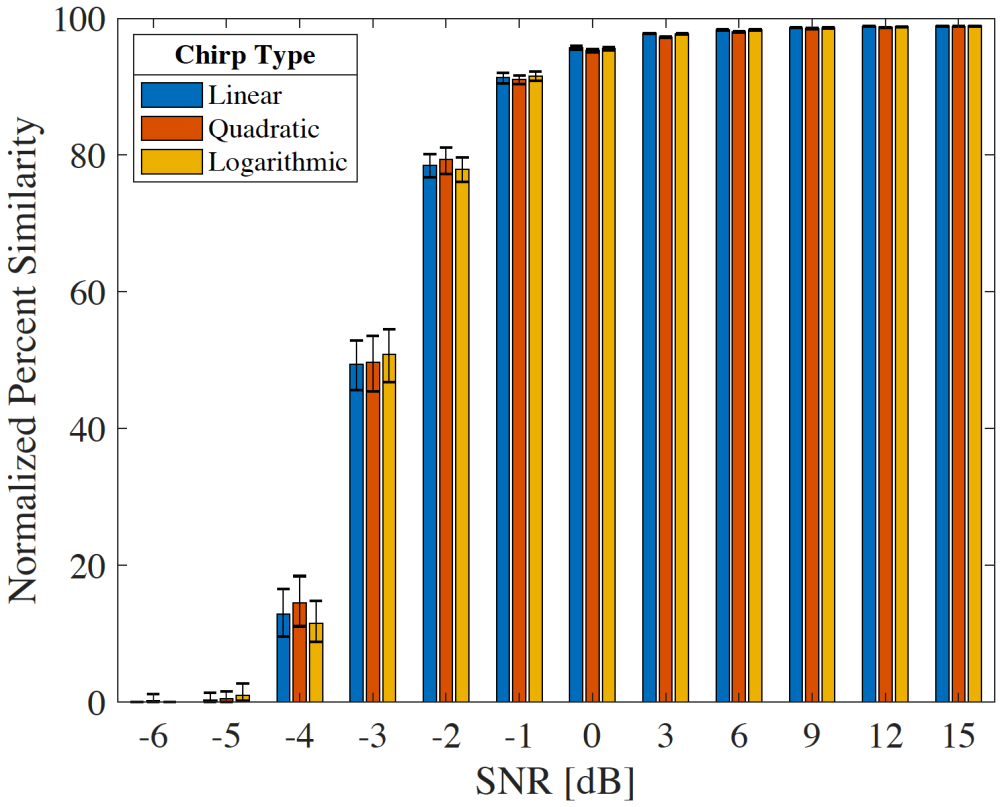

Figure 17. Normalized similarity between transmitted and reconstructed chirp signals with 20% frequency change- means and 95% confidence intervals

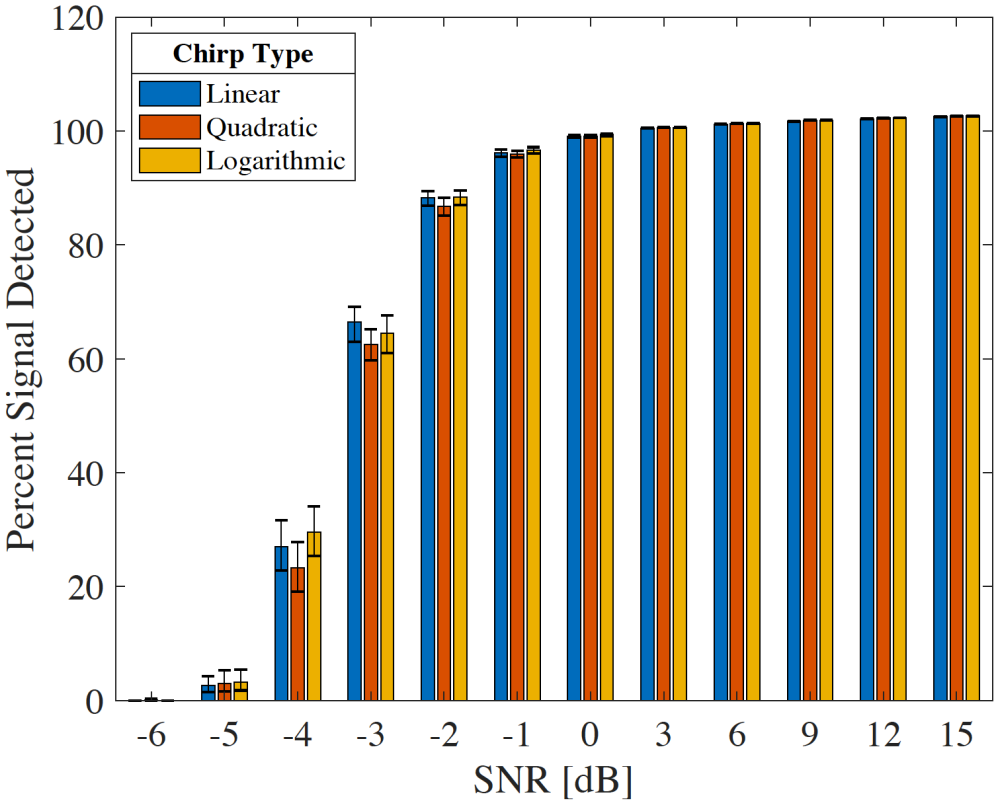

Figure 18. Percentage of signal length detected using the SMF implementation for chirp signals with 20% frequency change - means and 95% confidence intervals

Figures 19 and 20 illustrate results obtained for chirp signals with 80% change in frequency. Results show that quadratic chirps showed moderately better results at relatively lower SNR values than both other chirp types, especially for detection.

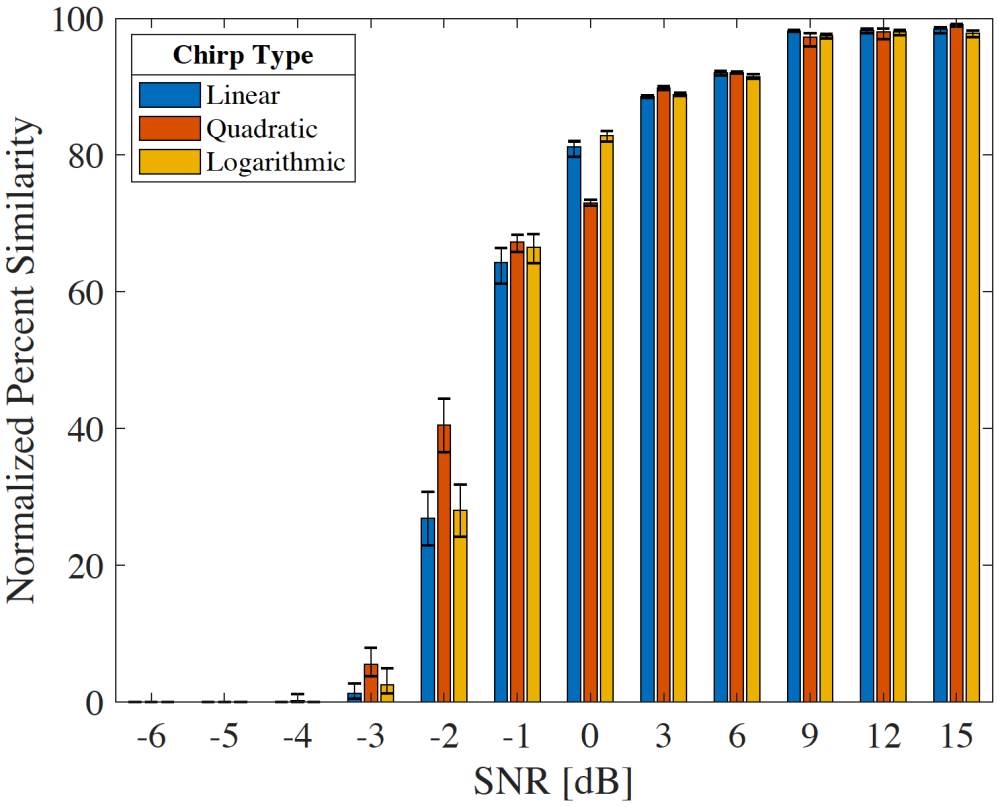

Figure 19. Normalized similarity between transmitted and reconstructed chirp signals with 80% frequency change- means and 95% confidence intervals

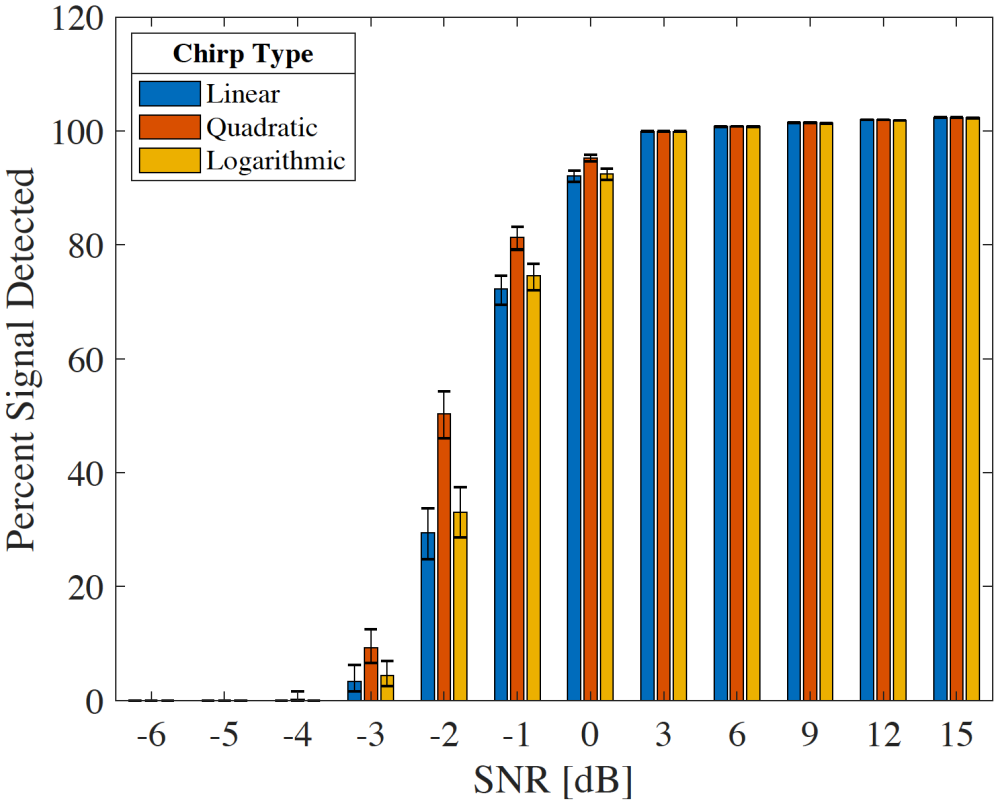

Figure 20. Percentage of signal length detected using the SMF implementation for chirp signals with 80% frequency change - means and 95% confidence intervals

C. SUMMARY OF RESULTS

In order to determine whether or not the SMF is effective for the signals we tested, we must consider the likelihood of the algorithm to detect and recognize these signals in real-time. Simulation results show the SMF is most effective against simple cosine signals, and performed best at SNR values of -3 dB and above, approximating and detecting the transmitted signal with at least 75% accuracy. Results also show the tone frequency had a small impact. Instead, the major factor in SMF accuracy and detection capabilities is the signal bandwidth for chirp types considered in the study. The SMF was much more successful against signals with lower bandwidths at slightly higher SNR values. Even at the lower of the two bandwidths tested, the SMF exceeded 75% accuracy and detection

ability at -2 dB and above for all chirp types. For higher bandwidth signals, the SMF performed similarly at SNR levels of -1 or 0 dB.

Overall, results show that the SMF is an effective detection tool for simple one-tone cosine signals, as well as multiple types of chirp signals, at moderate SNR levels. We found that the SMF can detect parts of certain signals at SNR values as low as -5dB. The SMF shows more reliable results for most signals tested at SNR values between -3dB and -2dB. While these SNR values are not particularly low, the SMF shows promise as a signal detection tool in difficult environments, warranting further research and improvement of the algorithm. The next chapter will discuss the conclusions of this study and present opportunities for future work.

VI. CONCLUSIONS AND FUTURE RESEARCH OPPORTUNITIES

This study explored the potential military application of the stochastic matched filter, a less known and less often researched adaptation of the classic matched filter. In the course of our research, we outlined the theory behind SMF and built an algorithm for approximating simple signals embedded in noise. Signal detection has always been an important challenge for the military, especially in challenging noise environments such as the undersea domain. Implementing the SMF could potentially enhance U.S. Navy passive signal detection capability.

Our study considered basic signals at medium to high SNR values to determine practicality of the SMF technique. Results suggest that the SMF could be a viable technique for military acoustic signal detection as it could be employed for passive, real-time detection of signals in challenging noise environments [5]. In addition, the SNR estimation algorithm investigated as part of the study, could be refined for use against more complex signals, such as multi-tone cosine signals and high-bandwidth chirp signals.

At this point, there has been very little research applying the SMF for military scenarios. A next step could be to build a filter bank for real-world military signals of interest and test the SMF on actual observed data, vice simulated signals and noise. Finally, the SMF has also shown promise as a de-noising application and has been used in two dimensions to de-noise imagery, including synthetic aperture sonar (SAS) [2]. An additional application that could be investigated is de-noising synthetic aperture radar (SAR) imagery.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX

This appendix presents the MATLAB code used to build SMF algorithms for simple cosine and chirp signals. These algorithms were adapted from code [15] developed by Dr. Léa Bouffaut in her 2019 PhD dissertation [d]. The code used to conduct the simulations and compare results may be provided by request of the author (michelle.welch@nps.edu) or thesis advisor (fargues@nps.edu).

A. SMF ALGORITHM FOR SIMPLE COSINE SIGNALS

```
% SMF algorithm for simple cosine signals

format compact
format shortG
% clear,close all

% ---------------------- CREATE NOISY SIGNAL ------------------------

% build signal of interest
f0 = 0.2; % <-----------------------------------------------------
target freq
fs = 25*f0; % sampling freq
ts = 1/fs;
Tsig = 400;
tsig = 0:ts:Tsig-ts;

N = 2000*fs; % length of total received signal
%rng(5);
w = sqrt(1).*randn(1,N); % fixed noise sequence var = 1;
Pw = var(w); % noise power

iSNRdB = -60 %   <-------------------------------- * CHOOSE SNR in dB
*
iSNRw = 10^(iSNRdB/10); % convert SNR to W/W

Ps = Pw*iSNRw; % calculate desired signal power
s_k1 = cos(2*pi*f0.*tsig); % basic cosine signal for calculations
ns = length(s_k1); % signal length

A = sqrt(Ps/(sum(s_k1.^2)./ns)); % calculate signal amplitude
s_k = A.*s_k1; % embedded signal with correct amplitude

% create noisy observed signal
I = 1000*fs:1000*fs+ns-1; % signal indices
Signal_Indices = [I(1) I(end)]
sig = zeros(1,N);
sig(1,I(1,:)) = s_k;
```

```
z = sig+w; % signal embedded in noise (observed signal)

Tobs = (length(z)-1)/fs; % observation duration
tobs = 0:1/fs:Tobs; % observation time vector

% ---------------------- OFFLINE ANALYSIS ----------------------

% signal autocorrelation sequence
L = 101; % Window length

[rs,lags] = xcorr(s_k1-mean(s_k1),L,'coeff'); % experimental
autocorrelation sequence of signal
rs = rs(L+1:end-1);

gams0 = toeplitz(rs);

% signal covariance from estimated signal expression (normalized with
respect to trace)
gams0 = gams0/sum(eig(gams0));

[svec,lams] = eig(gams0);
lams = diag(lams);
[lams,pos] = sort(lams,'descend');
svec = svec(:,pos); % Sorted signal eigenvectors

% modeled noise covariance
w1 = w/max(abs(w));
rn0 = zeros(1,2*L-1);
for i=1:length(w1)-L+1
   A = w1(i:i+L-1);
   B = xcorr(A-mean(A),'coeff'); % centered
   rn0 = rn0+B/max(B); % reduced
end

rn0 = rn0/max(rn0); % reduced
gamn0 = toeplitz(rn0(L:end)); % Noise covariance matrix
gamn0 = gamn0/sum(eig(gamn0)); % normalized with respect to the trace


% noise covariance matrix analysis to evaluate Qmax
% keep highest 10% eigenvalues -- most of the noise energy
[~,lamn] = eig(gamn0);
lamn = diag(lamn);
lamn = sort(lamn);
Qmax0 = sum(lamn > lamn(fix(0.9*length(lamn))));

% Karhunen-Loeve theorem
sproj = svec(:,1:Qmax0)'*gams0*svec(:,1:Qmax0); % Projection of the
signal covariance matrix onto signal subspace
nproj0 = svec(:,1:Qmax0)'*gamn0*svec(:,1:Qmax0); % Projection of the
noise covariance matrix onto signal subspace

% GEP
```

```
[vec0,lam0] = eig(sproj/nproj0);
lam0 = abs(diag(lam0));
[lam0,pos] = sort(lam0,'descend'); % Sort by descending eigenvalues
vec0 = vec0(:,pos);

% find phi and psi
phi = svec(:,1:Qmax0)*vec0;
for i = 1:size(phi,2)
   phi(:,i) = phi(:,i)/sqrt(phi(:,i)'*gamn0*phi(:,i)); %normalize
end

psi = gamn0*phi;

% create filter bank
h = zeros(size(phi));
[np,~] = size(phi);
cp = fix((np+1)/2);
for i = 2:Qmax0+1
   h(:,i) = h(:,i-1) + psi(cp,i-1).*phi(:,i-1);
end
h=h(:,2:i);

% ---------------- ONLINE (TIME-FREQUENCY) ANALYSIS -------------

% STFT of observation
%L = 101; % stft time window length
ovlp = floor(.98*L); % stft window overlap length
win = hann(L);  % stft window
nfft = 2^10; % number of fft sampling points

[Yz,fz,tz,psd] = spectrogram(z/max(z),win,ovlp,nfft,fs); % stft, freqs,
and time vector

figure(1)
subplot(221)
%spectrogram(z/max(z),win,ovlp,nfft,fs)
imagesc(tz*fs,fz,abs(Yz))
ax = gca; ax.YDir = 'reverse';ax.YLim = [0 f0*2];
xlabel('Time Index');ylabel('Frequency');
title('Spectrogram of Observation')

Yn = abs(Yz); % |Yz(k',f)| STFT modulus
nmf = 2*ns; % length of median filter
Yn = medfilt1(Yn,nmf,[],2); % Yn(k',f) noise estimate

figure(2)
subplot(211)
mesh(Yn)
title('Noise Estimate in TF Domain')

% zcall [k']
fz_lo = find(fz >= f0-(f0*0.1),1); % set freq range for signal presence
fz_hi = find(fz >= f0+(f0*0.1),1);
```

```
zcall = max(abs(Yz(fz_lo:fz_hi,:))./mean(Yn(fz_lo:fz_hi,:),1)); % calc
zcall [k'] [3.39]
[~,fzi] = max(abs(Yz(fz_lo:fz_hi,:))); % indices within freq range that
maximize zcall
fz_max = fz(fz_lo+fzi); % frequencies where zcall is maximized
fz_max = interp1(tz,fz_max,tobs); % interpolate to time domain
zcall = interp1(tz,zcall,tobs); % interpolate to time domain
zcall = medfilt1(zcall,round(N/5)); % smooth outliers via median
filtering

figure(1)
subplot(2,2,2)
plot(zcall)
title('zcall [k]')
xlim([0 length(zcall)])

% trans [k']
transk = abs(Yz./Yn); % calc trans [k'] [3.40]
fn_lo = find(fz >= f0-(f0*0.5),1); % set freq range for noise-only
observation
fn_hi = find(fz >= f0+(f0*0.5),1);
transk(fn_lo:fn_hi,:) = zeros(size(transk(fn_lo:fn_hi,:))); % zero out
signal freqs
transk = max(transk,[],1); % find max
transk = interp1(tz,transk,tobs); % interpolate to time domain
transk = medfilt1(transk,round(N/5)); % smooth outliers via medain
filtering

subplot(2,2,3)
plot(transk)
title('trans [k]')
xlim([0 length(transk)])

% online SNR estimate
rhok = (zcall./transk).^2; % calc online SNR est
%rhokdB = 20*log10(zcall./transk); % calc online SNRdB est [3.41]
rhokdB = 10*log10(zcall./transk);

subplot(2,2,4)
plot(rhokdB)
title('\rho [k] Estimate in dB')
xlim([0 length(rhokdB)])

% estimate the noise covariance (online)
rn = real(ifft(Yn.^2)); % autocorrelation of noise (Wiener-Khintchine
Theorem)
rn = sum(rn,2);
f_e = linspace(0,0.5*fs,L); % frequencies for evaluation
rn = interp1(fz,rn,f_e);

figure(2)
subplot(212)
plot(rn)
title('Estimated Autocovariance of Noise \Gamma_N')
```

```
gamn = toeplitz(rn); % noise covariance matrix
gamn = gamn-min(min(gamn));
gamn = gamn/max(max(gamn));

% online estimate of Qmax
[~,lam] = eig(gamn); % GEP
lam = diag(lam);
lam = sort(lam);
Qmax = sum(lam > 1); % max filter order

%limit number of filters
b = size(h,2);
if b < Qmax
   Qmax = b; % Qmax = Qmax0 most likely
end

% online GEP
sproj = svec(:,1:Qmax)'*gams0*svec(:,1:Qmax); % Projection of the
signal covariance matrix onto signal subspace
nproj = svec(:,1:Qmax)'*gamn*svec(:,1:Qmax); % Projection of the online
estimated noise covariance matrix onto signal subspace

[~,lamz] = eig(sproj/nproj);
lamz = abs(diag(lamz));
lamz = sort(lamz,'descend'); % Sort by descending eigenvalues

figure(3)
subplot(211)
stem(lamz)
title('Eigenvalues \lambda_z [k]')

% ----------------------- SIGNAL RECONSTRUCTION -----------------------
--
% in real time
nz = length(z);
sQk = [];
Q = [];

z0 = zeros(1,L+nz-1); % zero pad the observation
if mod(L,2)==0 % even or odd
   z0(L/2:end-L/2) = z;
else
   z0((L+1)/2:end-(L-1)/2) = z;
end

for i = 1:nz
   zk(i,:) = z0(i:i+L-1);
   Q(i) = sum((abs(lamz)*rhok(i)) >= 1);
   if Q(i) <= 0
       sQk(i) = 0;
   else sQk(i) = zk(i,:)*h(:,Q(i));
   end
end
```

```
figure(3)
subplot(212)
plot(rhok)
title('Online Estimated SNR \rho [k]')

figure(4)
subplot(211)
stem(Q)
title('Filter Order Q[k]')
subplot(212)
plot(z)
hold on
plot(sQk)
hold off
title('Reconstructed Signal s_Q[k] vs. Observation z [k]')
legend('Observed Signal z [k]','Reconstructed Signal
s_Q[k]','Location','southwest')

figure(5)
si = Signal_Indices;
plot(s_k)
hold on
plot(sQk(si(1):si(2)))
hold off
xlabel('Sample Number');ylabel('Amplitude');
ax = gca;
%ax.Title.FontSize = 18;
ax.XAxis.FontSize = 18;
ax.YAxis.FontSize = 18;
set(gca, 'FontName', 'Times New Roman')
legend({'Transmitted Signal','Approximated
Signal'},'FontSize',14,'FontName','Times New
Roman','NumColumns',2,'Location','best')
%title('Estimated Signal over Transmitted Signal')

%-------------CHAPTER 4 FIGURES--------------------

figure(10)
%imagesc(tz,fz,db(abs(Yz).^2))
imagesc(tz,fz,abs(Yz))
ax = gca;
ax.YLim = [0 f0*2];%ax.YDir = 'reverse';
xlabel('Sample Number');ylabel('Frequency [Hz]')
ax.XAxis.FontSize = 18;
ax.YAxis.FontSize = 18;
c = colorbar;
% set(gca,'ColorScale','log');c.Label.String = 'Amplitude [dB]';
% c.Label.FontSize = 18;
set(gca, 'FontName', 'Times New Roman')
%title('Spectrogram of Observation')

figure(11)
%mesh(Yn)
```

```
% imagesc(tz,fz,db(abs(Yn)))
imagesc(tz,fz,Yn)
ax = gca;
ax.YLim = [0 f0*2];%ax.YDir = 'reverse';
xlabel('Sample Number');ylabel('Frequency [Hz]');%zlabel('Amplitude');
ax.XAxis.FontSize = 18;
ax.YAxis.FontSize = 18;
ax.ZAxis.FontSize = 18;
c = colorbar;
% set(gca,'ColorScale','log');c.Label.String = 'Amplitude [dB]';
% c.Label.FontSize = 18;
set(gca, 'FontName', 'Times New Roman')


figure(12)
stem(lamz,'filled','MarkerSize',10)

figure(13)
plot(rhokdB)
xlabel('Sample Number');ylabel('SNR value [dB]')
ax = gca;
ax.XAxis.FontSize = 18;
ax.YAxis.FontSize = 18;
set(gca, 'FontName', 'Times New Roman')
```

B. SMF ALGORITHM FOR CHIRP SIGNALS

```
% SMF algorithm for chirp signals

format compact
format shortG
% clear,close all

% ---------------------- CREATE NOISY SIGNAL ------------------------

% build signal of interest
f1 = 0.2; % max chirp freq
f0 = f1/1.8; % min chirp freq <---------------------------- choose
bandwidth, 1.2 for 20%, 1.8 for 80%
fs = 25*f1; % sampling freq
ts = 1/fs;
Tsig = 400;
tsig = 0:ts:Tsig-ts;

N = 2000*fs; % length of total received signal
%rng(5);
w = sqrt(1).*randn(1,N); % fixed noise sequence var = 1;
Pw = var(w); % noise power

iSNRdB = -2 %   <--------------------------- * CHOOSE SNR in dB *
iSNRw = 10^(iSNRdB/10); % convert SNR to W/W
```

```
Ps = Pw*iSNRw; % calculate desired signal power

% Chirp Signal
type = 1;   %<-------------------------------------------------------
-- choose type: 0, 1, or 2
if type == 0
       s_k1 = chirp(tsig,f0,Tsig,f1,'linear');
       display('linear')
   elseif type == 1
       s_k1 = chirp(tsig,f0,Tsig,f1,'quadratic');
       display('quadratic')
   elseif type == 2
       s_k1 = chirp(tsig,f0,Tsig,f1,'logarithmic');
       display('logarithmic')
   else
       display('Choose 0 for linear, 1 for quadratic, or 2 for
logarithmic')
end

ns = length(s_k1); % signal length

A = sqrt(Ps/(sum(s_k1.^2)./ns)); % calculate signal amplitude
s_k = A.*s_k1; % embedded signal with correct amplitude

% create noisy observed signal
I = 1000*fs:1000*fs+ns-1; % signal indices
Signal_Indices = [I(1) I(end)]
sig = zeros(1,N);
sig(1,I(1,:)) = s_k;
z = sig+w; % signal embedded in noise (observed signal)

Tobs = (length(z)-1)/fs; % observation duration
tobs = 0:1/fs:Tobs; % observation time vector

% ----------------------- OFFLINE ANALYSIS -----------------------

% signal autocorrelation sequence
L = 101; % Window length

[rs,lags] = xcorr(s_k1-mean(s_k1),L,'coeff'); % experimental
autocorrelation sequence of signal
rs = rs(L+1:end-1);

gams0 = toeplitz(rs);

% signal covariance from estimated signal expression (normalized with
respect to trace)
gams0 = gams0/sum(eig(gams0));

[svec,lams] = eig(gams0);
lams = diag(lams);
[lams,pos] = sort(lams,'descend');
svec = svec(:,pos); % Sorted signal eigenvectors
```

```
% modeled noise covariance
w1 = w/max(abs(w));
rn0 = zeros(1,2*L-1);
for i=1:length(w1)-L+1
   A = w1(i:i+L-1);
   B = xcorr(A-mean(A),'coeff'); % centered
   rn0 = rn0+B/max(B); % reduced
end

rn0 = rn0/max(rn0); % reduced
gamn0 = toeplitz(rn0(L:end)); % Noise covariance matrix
gamn0 = gamn0/sum(eig(gamn0)); % normalized with respect to the trace


% noise covariance matrix analysis to evaluate Qmax
% keep highest 10% eigenvalues -- most of the noise energy
[~,lamn] = eig(gamn0);
lamn = diag(lamn);
lamn = sort(lamn);
Qmax0 = sum(lamn > lamn(fix(0.9*length(lamn))));

% Karhunen-Loeve theorem
sproj = svec(:,1:Qmax0)'*gams0*svec(:,1:Qmax0); % Projection of the
signal covariance matrix onto signal subspace
nproj0 = svec(:,1:Qmax0)'*gamn0*svec(:,1:Qmax0); % Projection of the
noise covariance matrix onto signal subspace

% GEP
[vec0,lam0] = eig(sproj/nproj0);
lam0 = abs(diag(lam0));
[lam0,pos] = sort(lam0,'descend'); % Sort by descending eigenvalues
vec0 = vec0(:,pos);

% find phi and psi
phi = svec(:,1:Qmax0)*vec0;
for i = 1:size(phi,2)
   phi(:,i) = phi(:,i)/sqrt(phi(:,i)'*gamn0*phi(:,i)); %normalize
end

psi = gamn0*phi;

% create filter bank
h = zeros(size(phi));
[np,~] = size(phi);
cp = fix((np+1)/2);
for i = 2:Qmax0+1
   h(:,i) = h(:,i-1) + psi(cp,i-1).*phi(:,i-1);
end
h=h(:,2:i);

% ---------------- ONLINE (TIME-FREQUENCY) ANALYSIS -------------
```

```
% STFT of observation
%L = 101; % stft time window length
ovlp = floor(.98*L); % stft window overlap length
win = hann(L);  % stft window
nfft = 2^10; % number of fft sampling points

[Yz,fz,tz,psd] = spectrogram(z/max(z),win,ovlp,nfft,fs); % stft, freqs,
and time vector

figure(1)
subplot(221)
%spectrogram(z/max(z),win,ovlp,nfft,fs)
imagesc(tz*fs,fz,abs(Yz))
ax = gca; ax.YDir = 'reverse';ax.YLim = [0 f1*2];
xlabel('Time Index');ylabel('Frequency')
title('Spectrogram of Observation')

Yn = abs(Yz); % |Yz(k',f)| STFT modulus
nmf = 2*ns; % length of median filter
Yn = medfilt1(Yn,nmf,[],2); % Yn(k',f) noise estimate

figure(2)
subplot(211)
mesh(Yn)
title('Noise Estimate in TF Domain')

% zcall [k']
fz_lo = find(fz >= f0-(f1*0.1),1); % set freq range for signal presence
fz_hi = find(fz >= f1+(f1*0.1),1);
zcall = max(abs(Yz(fz_lo:fz_hi,:))./mean(Yn(fz_lo:fz_hi,:),1)); % calc
zcall [k'] [3.39]
[~,fzi] = max(abs(Yz(fz_lo:fz_hi,:))); % indices within freq range that
maximize zcall
fz_max = fz(fz_lo+fzi); % frequencies where zcall is maximized
fz_max = interp1(tz,fz_max,tobs); % interpolate to time domain
zcall = interp1(tz,zcall,tobs); % interpolate to time domain
zcall = medfilt1(zcall,round(N/5)); % smooth outliers via median
filtering

figure(1)
subplot(2,2,2)
plot(zcall)
title('zcall [k]')
xlim([0 length(zcall)])

% trans [k']
transk = abs(Yz./Yn); % calc trans [k'] [3.40]
fn_lo = find(fz >= f0-(f1*0.5),1); % set freq range for noise-only
observation
fn_hi = find(fz >= f1+(f1*0.5),1);
transk(fn_lo:fn_hi,:) = zeros(size(transk(fn_lo:fn_hi,:))); % zero out
signal freqs
transk = max(transk,[],1); % find max
transk = interp1(tz,transk,tobs); % interpolate to time domain
```

```
transk = medfilt1(transk,round(N/5)); % smooth outliers via medain
filtering

subplot(2,2,3)
plot(transk)
title('trans [k]')
xlim([0 length(transk)])

% online SNR estimate
rhok = (zcall./transk).^2; % calc online SNR est
rhokdB = 20*log10(zcall./transk); % calc online SNRdB est [3.41]

subplot(2,2,4)
plot(rhokdB)
title('\rho [k] Estimate in dB')
xlim([0 length(rhokdB)])

% estimate the noise covariance (online)
rn = real(ifft(Yn.^2)); % autocorrelation of noise (Wiener-Kintchine
Theorem: IF(PSD) = autocorr func)
rn = sum(rn,2);
f_e = linspace(0,0.5*fs,L); % frequencies for evaluation
rn = interp1(fz,rn,f_e);

figure(2)
subplot(212)
plot(rn)
title('Estimated Autocovariance of Noise \Gamma_N')

gamn = toeplitz(rn); % noise covariance matrix
gamn = gamn-min(min(gamn));
gamn = gamn/max(max(gamn));

% online estimate of Qmax
[~,lam] = eig(gamn); % GEP
lam = diag(lam);
lam = sort(lam);
Qmax = sum(lam > 1); % max filter order

%limit number of filters
b = size(h,2);
if b < Qmax
   Qmax = b; % Qmax = Qmax0 most likely
end

% online GEP
sproj = svec(:,1:Qmax)'*gams0*svec(:,1:Qmax); % Projection of the
signal covariance matrix onto signal subspace
nproj = svec(:,1:Qmax)'*gamn*svec(:,1:Qmax); % Projection of the online
estimated noise covariance matrix onto signal subspace

[~,lamz] = eig(sproj/nproj);
lamz = abs(diag(lamz));
```

```
lamz = sort(lamz,'descend'); % Sort by descending eigenvalues

figure(3)
subplot(211)
stem(lamz)
title('Eigenvalues \lambda_z [k]')

% ----------------------- SIGNAL RECONSTRUCTION -----------------------
--
% in real time
nz = length(z);
sQk = [];
Q = [];

z0 = zeros(1,L+nz-1); % zero pad the observation
if mod(L,2)==0 % even or odd
   z0(L/2:end-L/2) = z;
else
   z0((L+1)/2:end-(L-1)/2) = z;
end

for i = 1:nz
   zk(i,:) = z0(i:i+L-1);
   Q(i) = sum((abs(lamz)*rhok(i)) >= 1);
   if Q(i) <= 0
       sQk(i) = 0;
   else sQk(i) = zk(i,:)*h(:,Q(i));
   end
end

figure(3)
subplot(212)
plot(rhok)
title('Online Estimated SNR \rho [k]')

figure(4)
subplot(211)
stem(Q)
title('Filter Order Q[k]')
subplot(212)
plot(z)
hold on
plot(sQk)
hold off
title('Reconstructed Signal s_Q[k] vs. Observation z [k]')
legend('Observed Signal z [k]','Reconstructed Signal
s_Q[k]','Location','southwest')

figure(5)
si = Signal_Indices;
plot(s_k)
hold on
plot(sQk(si(1):si(2)))
hold off
```

```
xlabel('Sample Number');ylabel('Amplitude');
ax = gca;
%ax.Title.FontSize = 18;
ax.XAxis.FontSize = 18;
ax.YAxis.FontSize = 18;
set(gca, 'FontName', 'Times New Roman')
legend({'Transmitted Signal','Approximated
Signal'},'FontSize',14,'FontName','Times New
Roman','NumColumns',2,'Location','best')
%title('Estimated Signal over Transmitted Signal')
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] J.-F. Cavassilas, "Stochastic matched filter," in *Proceedings of the Institute of Acoustics (International Conference on Sonar Signal Processing)*, vol. 13, part 9, 1991, pp. 194–199.

[2] P. Courmontagne, "The stochastic matched filter and its applications to detection and de-noising," in *Stochastic Control*, C. Myers, Ed. London: IntechOpen, 2010 [Online]. Available: https://www.intechopen.com/books/3748 doi: 10.5772/260

[3] J. Bonnal, P. Danès, and M. Renaud, "Detection of acoustic patterns by stochastic Matched Filtering," presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 2010, pp. 1970–1975.

[4] J. L. Mori and P. Gounon, "The use of stochastic matched filter in active sonar," *10th European Signal Processing Conference*, Tampere, Finland, 2000, pp. 1–4.

[5] L. Bouffaut, "Detection and classification in passive acoustic contexts: application to blue whale low-frequency signals," PhD dissertation, PhD School of Marine and Coastal Sciences, University of West Brittany, Brest, France, 2019.

[6] C. W. Therrien, Discrete Random Signals and Statistical Signal Processing. Englewood Cliffs, NJ: Prentice-Hall, 1992.

[7] D. Manolakis, I. Vinay, and S. Kogon, Statistical and Adaptive Signal Processing: Spectral Estimation, Signal Modeling, Adaptive Filtering and Array *Processing*. Norwood, MA, USA: Artech House, 2005 [Online]. Available: ProQuest Ebook Central.

[8] S. Haykin, *Adaptive Filter Theory*. Upper Saddle River, NJ, USA: Pearson, 2014.

[9] C. Maccone, "The KLT (Karhunen–Loève Transform) to extend SETI searches to broad-band and extremely feeble signals," *Acta Astronautica*, vol. 67, no. 11, pp. 1427–1439, 2010 [Online]. Available: https://doi.org/10.1016/ j.actaastro.2010.05.002

[10] B. Borloz, B, Xerri. Subspace SNR maximization: The constrained stochastic matched filter. *IEEE Transactions on Signal Processing, Institute of Electrical and Electronics Engineers,* 2011, 59 (4), pp.1346 - 1355. [Online]. Available: https://doi.org/10.1109/TSP.2010.2102755

[11] M. Chagmani, B. Xerri, and B. Borloz, "Application of the constrained stochastic matched filter subspace tracking to submarine acoustic signals," in *Proceedings of the 2nd Mediterranean Conference on Pattern Recognition and Artificial Intelligence* (MedPRAI '18), Rabat, Morocco, 2018, pp 33–37.

[12] L. Bouffaut, R. Dréo, V. Labat, A. Boudraa, and G. Barruol, “Passive stochastic matched filter for Antarctic blue whale call detection,”*Journal of the Acoustical Society of America,* vol. 144, no. 955, pp. 955–965, Aug. 2018. [Online]. Available: https://doi.org/https://doi.org/10.1121/1.5050520

[13] F. Socheleau, E. Leroy, A.C. Pecci, F. Samaran, J. Bonnel, and J. Royer, “Automated detection of Antarctic blue whale calls,” *Journal of the Acoustical Society of America*, vol. 138, pp. 3105–3117, Nov. 2015. [Online]. Available: https://doi.org/10.1121/1.4934271

[14] R. Cristi, *Modern Digital Signal Processing.* Pacific Grove, CA, USA: Thomson/ Brooks/Cole, 2004.

[15] L. Bouffaut, (2020) SMF_package (Version 1.0) [Source code]. https://github.com/leabouffaut.SMF_package/tree/v1.0

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California