UNIVERSITY OF OKLAHOMA
GRADUATE COLLEGE

DIGITAL BEAMFORMING APPLICATIONS AND DEMONSTRATIONS OF AN RF SYSTEM-ON-A-CHIP

A DISSERTATION
SUBMITTED TO THE GRADUATE FACULTY
in partial fulfillment of the requirements for the
Degree of
DOCTOR OF PHILOSOPHY

By

KYLE AARON STEINER
Norman, Oklahoma
2021

DIGITAL BEAMFORMING APPLICATIONS AND DEMONSTRATIONS OF AN RF SYSTEM-ON-A-CHIP

A DISSERTATION APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY THE COMMITTEE CONSISTING OF

Dr. Mark Yeary, Chair

Dr. Caleb Fulton

Dr. Bob Palmer

Dr. Jorge Salazar-Cerreno

Dr. Cameron Homeyer

To my Lord and Savior, Jesus Christ and to my beautiful, intelligent, and caring wife Devin, whose support made this pursuit possible

Acknowledgments

I would like to acknowledge my committee chair, Dr. Yeary, whose enthusiastic support was pivotal in my decision to pursue this degree. His consistent guidance and availability throughout the past several years have been essential. I would also like to thank my committee members, Drs. Bob Palmer, Caleb Fulton, Jorge Salazar, and Cameron Homeyer for their support throughout this endeavor. And lastly, I would like to thank my family. My parents and grandparents have showed unending support of my pursuits throughout my life, and my time at the University of Oklahoma was no different. And my dear wife for being willing to upturn our lives to move out of our home state of Florida. Completing this goal would never have been possible without her support. I am so fortunate to have her by my side.

Contents

Appendix A

List of Tables

List of Figures

Abstract

EM phased array system bandwidth is conventionally constrained by the use of phase shifters for beamsteering, which results in beam squint and pulse dispersion of wideband signals. Wideband antenna performance can be achieved through the use of element-level true time delay (TTD) units, but this is often impractical due to the complexities associated with TTD analog devices. The continued improvement of high-speed analog-to-digital converters (ADC) and digital-to-analog converters (DAC) places digital signal conversion at the element level. This allows TTD beamsteering to be accomplished digitally via a combination of integer-sample delays and fractional-sample delay finite impulse response (FIR) filters, enabling support for wideband communication and radar imaging operating modes.

As phased array systems rely on matched channel characteristics, accurate system calibration is paramount for optimum performance. Narrowband systems which implement beamforming via attenuators and phase shifters often employ lookup tables (LUT) containing a set of correction commands to be superimposed on the desired steering operation. These are commonly dependent on current and desired system characteristics, such as operating frequency, steering direction, power level, and/or temperature conditions. In contrast, wideband systems require higher fidelity compensation techniques capable of correcting imbalanced and dispersive channel effects from element-level electronics.

This dissertation examines deterministic and adaptive beamforming techniques and provides solutions to the aforementioned challenges by contributing the development and demonstration of a wideband digital beamformer with equalization on an RF system-on-a-chip (RFSoC). Performance metrics of the testbed match or exceed current publications of RFSoC based demonstrations. The RFSoC is a unique, state-of-the-art, highly integrated device that incorporates a field programmable gate array (FPGA), high speed ADCs and DACs with a system-on-a-chip (SOC) architecture onto the same silicon fabric. As much of the digital and analog RF circuitry is now integrated into a single package, these devices are revolutionizing radar and communication systems, reshaping phased array system design strategies. This enabling technology facilitates the development of compact all-digital arrays, massively increasing the available degrees of freedom in system control, a paradigm shift in industry and engineering communities.

The beamformer testbed is demonstrated on a sub-Nyquist-sampled 1.6 GHz S-band phased array system implemented using a Xilinx 8-channel 4 GSPS RFSoC. To enable TTD digital beamsteering, each channel is compensated via a conjugate symmetric fractional-sample delay FIR filter bank. By modifying the TTD filter structure to support complex coefficients, channel equalization is integrated with the fractional-sample delays to compensate undesired channel characteristics. To confirm the efficacy of this approach, results are provided for uncalibrated and calibrated system operation. Anechoic chamber measurements are presented as well as the FPGA floorplans showing RFSoC device utilization for both uncalibrated and calibrated configurations.

Chapter 1

Introduction

1.1 Background and RFSoC

Phased array systems provide a highly flexible and rapidly controllable method for interacting with various energy wave domains, such as seismic [1], underwater acoustic [2], aeroacoustic [3], and electromagnetic (EM) [4]. This dissertation focuses exclusively on EM waves, although much of the beamforming discussion is relevant to other applications. The EM spectrum has long been leveraged as a method for conveying and extracting information, be it for communications [5], radar tracking [6] and imaging [7] [8], or signal intelligence [9].

Since their inception, phased array systems have been utilized to interact with the electromagnetic (EM) spectrum due to their many advantages over single-element systems. These include graceful degradation, modularity of their design, controllable degrees of freedom (DOF), rapid steering control, and design flexibility. However, phased array systems also present many challenges, such as higher system complexity, increased development and fabrication costs, calibration challenges, and inherent bandwidth limitations depending on steering control implementation. Bandwidth limitations can be addressed by replacing phase shifters with true time delay (TTD) units [10] to control the alignment of received signals, although this is often precluded at the element level by the complexity associated with analog TTD devices. Additionally, as phased array operation assumes matched channels behavior, inter-channel variations must be accurate characterized and compensated to enable proper system performance.

In 2017, Xilinx announced the radio frequency system-on-a-chip (RFSoC) product line, which incorporates a high performance field programmable gate array (FPGA) together with several gigasample per second (GSPS) data converters and a system-on-a-chip (SoC) architecture in the same integrated circuit (IC) package. The integration of multiple analog-to-digital converter (ADC) and digital-to-analog converter (DAC) channels into the FPGA/SoC device provides a potential footprint reduction of 50% and power reduction of 75% [11]. In brief, these compact devices were geared towards the next generation tele-communication industry, such as 5G and 6G, yet they provide an ideal technology platform for the future of phased array radar for three reasons: sample synchronization across multiple channels is readily possible, low power consumption, and compact form factor. The latter is especially evident, as traditional analog up/down conversion circuits are not needed for radar systems in the S-band owing to the multi-gigasample RF data converters, which enable direct-sampling of incident waveforms. Other researchers have been exploring RFSoC technology for next generation radar applications, and recent breakthroughs include: bistatic radar [12], phased array cost reduction [13], data reduction for digital apertures [14], near-field calibration [15], real-time signal generation [16], or fully-digital radar system development [17].

Given the novelty of the RFSoC, few examples of RFSoC-based phased array systems exist in the current literature. Table 1.1 provides a summary of key performance metrics of several publications, discussed further in Chapter 4. This dissertation presents the development, implementation, and demonstration of a wideband sub-Nyquist-sampled beamformer with equalization using an RFSoC. Performance metrics were chosen to match or exceed those given in current publications. Specifically, the sample frequency and digital bandwidth match the highest given values of 4 GHz [18] and 80% Nyquist zone [19], respectively. Additionally, the results presented herein utilize the highest fractional bandwidth of better than 50% and the highest absolute bandwidth of 1.6 GHz out of the surveyed platforms. It is the

Table 1.1: Literature Summary

	Cost Reduction [13]	TTD Beamformer [18]	Digital Beamformer [19]	IMPACT [20]	Research Testbed
Elements	64	16	16	16	8
Architecture	Direct Conversion	Direct Sampling	Direct Conversion	Direct Conversion	Direct Sampling
Sample Frequency	125 MHz	4 GHz	2 GHz	125 MHz	4 GHz
Carrier Frequency	3 GHz	1 GHz	28 GHz	3.5 GHz 4.9 GHz 9.5 GHz	3 GHz
Bandwidth	8 MHz	100 MHz	800 MHz	500 MHz	1.6 GHz
Notes	MVDR	Four-Beam			Bandpass Sampling, Digital EQ

first RFSoC-based TTD beamformer at full digital bandwidth, the first example of RFSoC-based sub-Nyquist beamforming, and the first wideband digital equalizer on an RFSoC.

1.2 Mathematical Notation

Before beginning the discussion on phased arrays, a short discussion on mathematical notation is merited. Describing phased array systems and adaptive beamforming using compact mathematical notation inevitably gives rise to the topics of linear algebra and multivariable calculus. In addition, adaptive algorithms give rise to statistical analysis using these equations. Potentially further obfuscating mathematical notation is the use of complex variables to compactly represent signal magnitude and phase. Many adaptive beamforming algorithm derivations seek to optimize system performance by minimizing system error, requiring matrix derivatives with respect to complex-valued vectors. A discussion of the optimization of a real valued function of a complex vector, a situation commonly arising in many adaptive algorithm derivations, is given by Brandwood [21] and is used extensively in Chapter 3. Some texts [22], [23], [24] present derivations using strictly real variables resulting in subtle differences from those that use complex variables [25], [26]. In this dissertation, all derivations are carried out using complex variables.

The matrix notation in this dissertation follows the conventions used in several sources [22], [25], [26]. A non-bold, typically lowercase character, such as s, represents a scalar variable. An exception to this convention occurs in Section 3.3.1, where the signal power is given by $E\{|s(k)|^2\} = S$. A lowercase bold character, such as $\mathbf{x}$, represents a column vector. An N-length vector with enumerated elements x_n is given by

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} . \tag{1.1}$$

An uppercase bold character, such as $\mathbf{A}$, represents a matrix. An $\mathbf{N} \times \mathbf{M}$ example with elements $A_{n,m}$ is given by

$$\mathbf{A} = \begin{bmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,M} \\ A_{2,1} & A_{2,2} & \cdots & A_{2,M} \\ \vdots & \vdots & \ddots & \vdots \\ A_{N,1} & A_{N,2} & \cdots & A_{N,M} \end{bmatrix} . \tag{1.2}$$

The variable $\mathbf{H}$ is an exception given in Section 4.2, where it is used to represent a column vector of frequency-domain data. The corresponding time domain column vector is denoted given by $\mathbf{h}$.

1.3 Dissertation Organization

This dissertation is organized as follows. Chapter 2 provides a description of phased array systems, discussing common system components and fundamental control theory. A discussion on the differences between narrowband and wideband performance is provided along with an exploration of null formation and arbitrary beamshaping. It also touches on direct-sampled beamforming and presents considerations for sub-Nyquist sampling of received waveforms. Chapter 3 covers adaptive beamforming within the context of this emerging class of digital arrays, providing both the derivations and comparison of several adaptive beamforming algorithms. Adaptive algorithm simulations are also included and discussed. Wideband digital beamforming demonstrations are provided in Chapter 4. The RFSoC testbed is covered in detail as well as the design and implementation of a fractional-sample delay filter bank for digital TTD support. Simulations and chamber measurement results for uncalibrated performance are provided. Least-squares calibration techniques and the implementation of a wideband equalizer are also provided. Various equalization synthesis techniques are discussed in detail and demonstrated through chamber measurements.

1.4 Primary Contribution

For the first time, this dissertation presents the development, implementation, and demonstration of a wideband sub-Nyquist-sampled beamformer with equalization using an RFSoC. Sub-Nyquist sampling is the unique concept that allows the a true-time delay based beamformer to operate in the S-band, and this sampling approach is made possible by the multi-gigasample RF data converters on the same substrate as the FPGA, which enable direct-sampling of incident waveforms. Sub-Nyquist sampling is a relatively new concept within the radar community; more importantly, its union with digital true-time delay beamforming and integrated equalization on an RFSoC is not available in the current literature.

Chapter 2

Phased Array Processing

Fundamentally, a phased array system is composed of an array of sensors, a method for time or phase shifting sensor signals, and a method for splitting and/or combining sensor signals [4]. The ability to control sensor signal amplitude is a common feature, and many EM systems also require a method for modulating the signal bandwidth between baseband frequency and the carrier frequency [27]. In most practical systems, some level of amplification [28] is required for both transmit and receive. However, because the focus of this dissertation is the electronic steering of such systems, power amplifiers (PA) and low noise amplifiers (LNA) will not be discussed further. In the beamforming demonstrations provided in Chapter 4, frequency conversion is implemented via digital downconversion (DDC) and digital upconversion (DUC) as well as inherently through aliasing associated with sub-Nyquist-sampled beamforming, a topic covered in Section 2.7.

Section 2.1 provides a brief overview of common phased array components. Section 2.2 discusses the relationship between spatial domain and time domain filtering and their application to phased array systems. Section 2.3 discusses classical beamforming, presenting a mathematical representation of a phased array as the basis for derivations of narrowband and wideband Fourier beamforming. A comparison of these two cases is also provided. Methods for deterministic null formation and arbitrary beam shaping are provided in Section 2.4 and Section 2.5, respectively. Lastly, a study of sub-Nyquist-sampled beamforming is presented in Section 2.7, which discusses beamforming implications for bandpass sampling the incident waveform at the element level.

2.1 Phased Array System Components

This section discusses various phased array building blocks, including antenna elements, signal control techniques, the summation node, and digital downconversion. Increasingly, digitization is moving closer to the sensor array, with the goal of providing each channel with its own ADC and DAC [29]. This provides incredible system flexibility due to the accessibility of individual element signals for digital processing. Applications to digital adaptive beamforming are explored in Chapter 3, and considerations for ADC sample frequency are presented in Section 2.7.

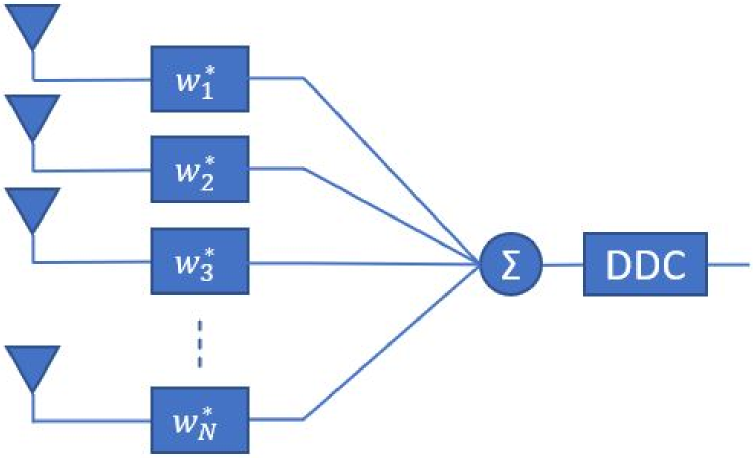

Figure 2.1: Linear Phased Array

The block diagram given in Figure 2.1 shows an array of sensors, each with individual complex-valued weight w_n^*, a summation node, and a DDC. Although many block diagrams found in textbooks do not include a method for upconversion or downconversion, as it is not fundamentally required in a phased array system, it is common in EM phased array systems and a relevant part of the digital architecture implemented in Chapter 4. Frequency shifting can be accomplished via an analog mixer [30] or numerically in the digital domain [31] via DDC/DUC. Given that this dissertation focuses more on the receive case in the digital domain, the inclusion of a DDC functional block was deemed prudent. Details for an efficient DDC technique outlined in [7] is presented in Section 2.1.4.1. Although this scheme was originally planned for use in the testbed demonstrations provided in Chapter 4, it was ultimately set aside in favor of the dedicated DDC hardware within the RFSoC.

2.1.1 Antenna Element

The properties of the antenna elements within an array have a direct impact on the array performance. Chapter 5 of Hansen [32] provides an overview of several common phased array elements such as the thin dipole, waveguide slot, and transverse electromagnetic (TEM) Horn. These are amongst those covered in Chapter 5 of Mailloux [4] with the addition of the patch antenna.

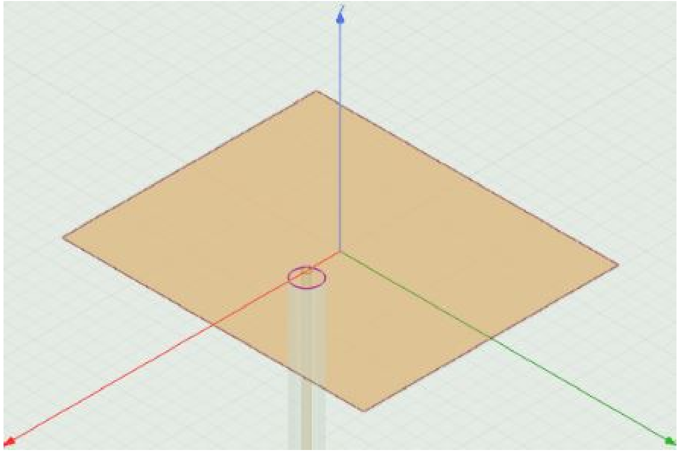

Figure 2.2: Patch Antenna Simulation

Because the elements in a phased array are physically fixed when scanning electronically, one of the primary antenna element design goals is to achieve a wide embedded element pattern. This minimizes pattern loss over steering angle reducing loss at the array level. Hence elements are typically designed to be more isotropic, having low directivity. As an example, Figure 2.2 shows an HFSS model of a patch antenna polarized in along the x-axis. Its corresponding patterns in the E-plane and H-plane are given in the left and right plots of Figure 2.3, respectively. Solid lines show the co-polarization while dashed lines show cross-polarization. The loss at ±45° for both planes is on the order of 3 dB.

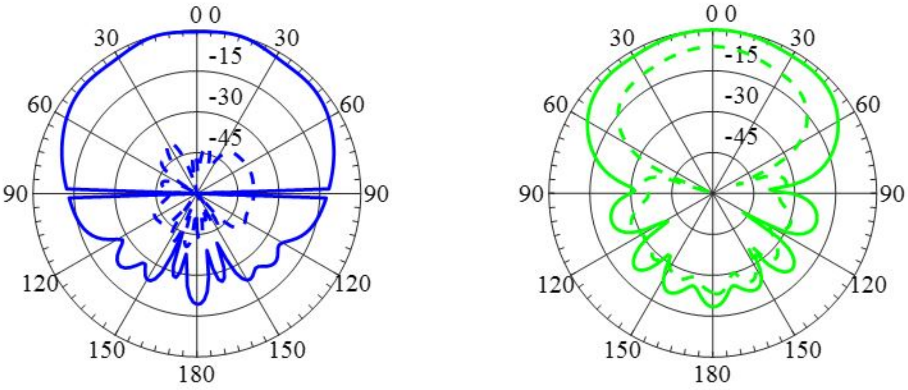

Figure 2.3: Patch Antenna Pattern

Additionally, because access to polarimetric data is often advantageous for weather radar applications [33], dual-polarized elements are beneficial. The University of Oklahoma (OU) has developed a dual-polarized patch element with wide scan angle and low cross-polarization [34], which is currently being integrated into a state-of-the-art all-digital phased array.

2.1.2 Signal Compensation

One of the primary benefits of phased array systems is the Signal-to-Noise Ratio (SNR) improvement achieved by coherently summing correlated signals amongst uncorrelated noise, where correlation is considered both spatially and temporally. Coherency is achieved through spatial filtering of element signals via phase shifters, TTD units, digital filters, or a combination thereof. Adaptive beamforming, covered in Chapter 3, can be employed to suppress undesired correlated signals and often requires channel amplitude control.

Digitally-controlled analog phase shifters offering 5.625° resolution and attenuators offering 0.5 dB have been quite common on the market for some time now. These devices provide approximately constant phase shifts and attenuation over frequency, giving rise to narrowband performance as explored in the following sections. Section 2.3.2 outlines the benefits of beamsteering using TTD units as this mitigates narrowband issues that arise from conventional phase shifters. See Figure 2.10 for a comparison of phase shifter and TTD unit behavior in the frequency and time domains. For all-digital arrays, phase and amplitude control can be accomplished digitally via either multiplication by a complex value or through filtering.

2.1.3 Summation Node

The summation node is used to combine individual channel signals in receive mode or split the transmit signal between each channel. A multi-port Wilkinson power divider [30] is an example of a band-limited analog splitter/combiner. Additionally, as the design is based on quarter wavelength dimensions, it may be too large for lower frequency applications. Summation in the digital domain addresses bandwidth concerns inherent in many analog solutions if channel-level digital control is available.

2.1.4 Digital Downconversion

Downconversion allows received signals to be processed at complex baseband. Although this has historically been implemented via analog mixers, digital processing techniques are becoming more prevalent. Figure 2.4 shows a signal flow block diagram for a DDC. A received signal is shifted in frequency, filtered to remove the conjugate symmetric spectrum, and downsampled. Although the downsampling sub-block is not necessarily required, it is included here as it is common on many DDC schemes and architectures.

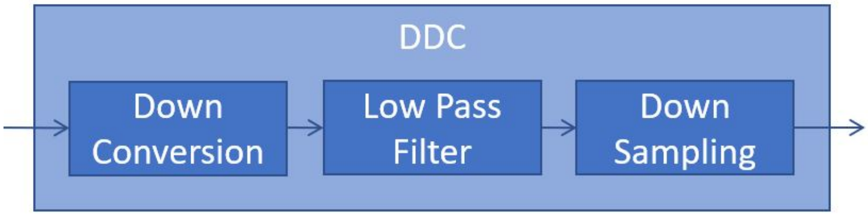

Figure 2.4: Digital Downconversion Block Diagram

2.1.4.1 Downconversion

A DDC scheme given in Chapter 3 of Richards [7] is presented here, wherein the sample frequency F_{s}, and carrier frequency f_{c}, satisfy the following relationship,

$$\frac{F_{\mathrm{s}}}{4} = f_{\mathrm{c}} \bmod \frac{F_{\mathrm{s}}}{2} \ . \tag{2.1}$$

To aid in the discussion of sub-Nyquist beamforming in Chapter 2.7, an undersampled case where $F_{\mathrm{c}} = 3$ GHz and $F_{\mathrm{s}} = 4$ GHz is discussed here. This frequency plan satisfies Eq. (2.1) as demonstrated by

$$\begin{aligned} F_{\mathrm{c}} \bmod \frac{F_{\mathrm{s}}}{2} &= 3 \text{ GHz} \bmod \frac{4 \text{ GHz}}{2} \\ &= 3 \text{ GHz} \bmod 2 \text{ GHz} \\ &= 1 \text{ GHz} \\ &= \frac{F_{\mathrm{s}}}{4} \ . \end{aligned} \tag{2.2}$$

The received signal or its conjugate spectrum aliases to $\frac{F_s}{4}$ in the digital domain, centering it in the first Nyquist zone. When the real-valued signal is sampled in an even Nyquist zone, as is the case here, frequency folding places the conjugate spectrum at $\frac{F_s}{4}$ and the original spectrum at -$\frac{F_s}{4}$.

DDC relies on the frequency shift property of the Fourier transform [35], which states that a frequency shift of F_{a} in the function $G(F)$ corresponds to a phase shift in the time domain, as given by

$$\mathcal{F}^{-1}\{G(F - F_{\mathrm{a}})\} \triangleq e^{j2\pi F_{\mathrm{a}} t} g(t) \ . \tag{2.3}$$

Thus, by multiplying the receive signal by a complex exponential, the original spectrum can be shifted from -$\frac{F_s}{4}$ to complex baseband.

However, because the received signals are represented as samples in the digital domain, the right-hand-side (RHS) of Eq. (2.3) should be rewritten in the discrete domain noting the following sampling relations: the time domain axis t relates to the sample index m by $t = mT_\mathrm{s}$ and the continuous time frequency F relates to the normalized digital frequency f by $f = \frac{F}{F_\mathrm{s}}$. The sample period and sample frequency are given by T_s and F_s respectively. Making the appropriate substitutions, the RHS of Eq. (2.3) can be rewritten as

$$e^{j2\pi F_\mathrm{a} t} g(t) = e^{j2\pi f_\mathrm{a} m} g(mT_\mathrm{s}) \ . \tag{2.4}$$

As we seek to shift the original spectrum from -$\frac{F_s}{4}$, the normalized frequency shift f_a is given by

$$f_\mathrm{a} = \frac{-\frac{F_\mathrm{s}}{4}}{F_\mathrm{s}} = -\frac{1}{4} \ , \tag{2.5}$$

which, when substituted into the complex exponential term of Eq. (2.4), provides the following simplification.

$$e^{j2\pi \frac{-1}{4} m} = e^{-j\frac{\pi}{2} m} = j^{-m} \tag{2.6}$$

Hence, a conjugate symmetric spectrum centered at -$\frac{F_\mathrm{s}}{4}$ can be shifted to baseband simply by applying the appropriate sign change and redefining every other signal sample as a fully imaginary value.

2.1.4.2 Filtering

Following downconversion to baseband, the conjugate symmetric spectrum, which was located at -$\frac{F_\mathrm{s}}{4}$, will be split between $\pm\frac{F_\mathrm{s}}{2}$. The 16-tap magnitude response recommended by Richards [7] is shown in Figure 2.5. If ≈ -35 dB of stopband rejection is insufficient, a new filter can be designed with the necessary attenuation [36].

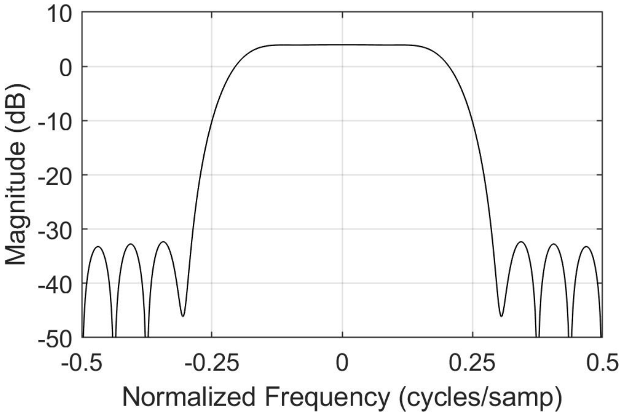

Figure 2.5: DDC Low-pass Filter Magnitude Response

2.1.4.3 Down-sampling

Filtering mitigates the issues of aliasing that arise due to down-sampling. Thus, the signal can be down-sampled by a factor of two, which spreads the spectrum in the digital frequency domain.

2.1.4.4 DDC Example

Figure 2.6 shows the signal spectrum magnitude at various stages of the DDC block diagram given in Figure 2.4 for a 3 GHz-centered, 1000 MHz LFM transmit waveform. The upper plot shows the conjugate symmetric transmit spectrum centered at ±5 GHz. The mid-left plot shows the sub-Nyquist sampled receive signal which has aliased to $\pm\frac{F_\mathrm{s}}{4} = \pm 1$ GHz. Downconversion shifts the spectrum to the left by $\frac{F_\mathrm{s}}{4}$, wrapping the lower spectrum such that it is split between $\pm\frac{F_\mathrm{s}}{2}$. This is shown along with the filter magnitude response in the mid-right plot. The lower left plot provides the filtered downconverted spectrum, showing a reduction in the upper and lower half-spectrums by about 35 dB. Downsampling by a factor of two reduces the signal voltage by 6 dB and causes the reduced upper and lower half-spectrums to alias into the main band. The final baseband spectrum is shown in the lower-right plot, and comprises up to 80% of the digital frequency domain for which $F_\mathrm{s} = 2$ GHz.

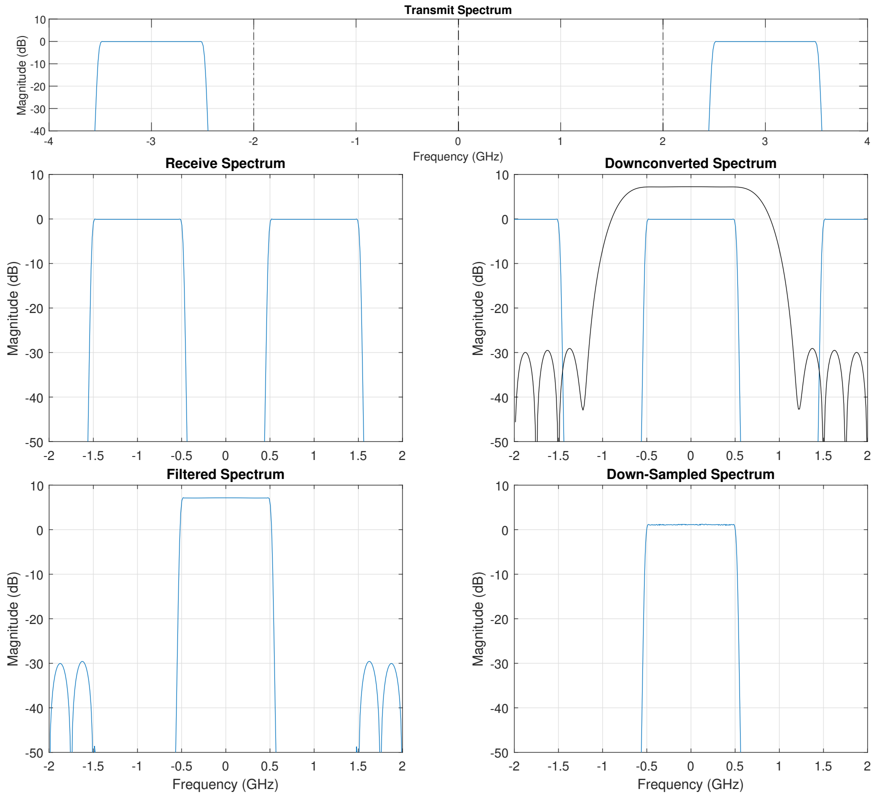

Figure 2.6: Signal Spectrum: Top: Transmit Spectrum; Mid-Left: Receive Spectrum; Mid-Right: Downconverted Spectrum, Low-Left: Filtered Spectrum; Low-Right: Down-Sampled Spectrum

2.2 Spatial and Time Domain Processing

Fourier processing is fundamental to the operation of phased array systems. A phased array seeks to filter incident signals in the spatial domain such that they can be coherently processed in the time domain. In the author's experience, much of the engineering curriculum is devoted to time domain filtering and the temporal frequency domain representation of signals. The concept of spatial frequency may not arise until several semesters after the Fourier transform is introduced. This results in a deeper intuition of time domain processing as compared to spatial domain processing.

The reason for this may be due to the ease with which time domain analog and digital filters can be implemented in hardware and explored in a lab environment. The complexity of a phased array system often precludes its availability in a lab. And the concepts required for image processing, such as multi-dimensional filtering are best taught upon a background of single-dimensional processing, a foundation more easily introduced in the time domain than the spatial domain. The ubiquitousness of music also contributes to a greater intuition in time domain filtering. While spatial filters have applications in video games, image processing, and optics, the average person may not interact with these topics to the degree required to develop an intuitive understanding of spatial filtering.

Fortunately, given the commonality in the underlying mathematics, there is massive overlap between time domain and spatial domain processing. Some of the derivations provided in Chapter 3, as applied to adaptive beamforming, are presented in textbooks focused solely on time domain processing. Concepts such as over-sampling and under-sampling, the Fourier transform properties, aliasing, and physicality constraints are all applicable to both time domain and spatial domain processing. Additionally, processing wave-based signals with a phased array system inherently involves both time domain and spatial domain processing, given the fundamental relation between space in time in the manifestation of wave phenomena.

One constraint in time domain processing not present in spatial domain processing is that of causality. Temporal sequences have an inherent directional constraint that is not present in spatial sequences. In the author's experience, this caused much frustration at the introduction of the concept of negative frequencies. However, negative frequencies arise naturally in spatial domain processing in an intuitive manner that is lacking in time domain processing. Interaction with spatial filtering concepts can clarify the notion of negative frequency and its application in time domain filtering.

2.3 Classical Beamforming

The fundamental intent in phased array beamforming is to determine sensor signal compensation which causes signals to add coherently for a given steering direction. Van Veen calls this *classical beamforming* [37]. To maximize SNR, other beamsteering techniques seek to mitigate the effects of directional interferers through deterministic or adaptive directional nulling [25]. However, the focus in this section is on the coherent summation of signals for a given direction.

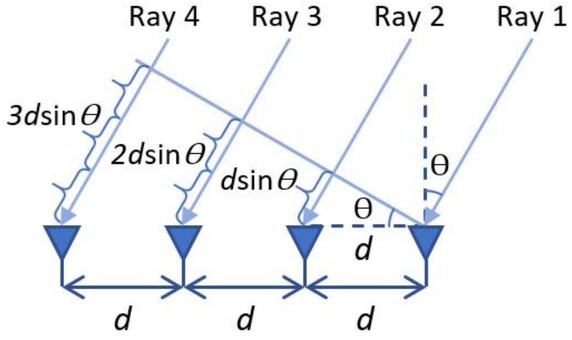

Figure 2.7: Ray Tracing Geometry for 4-Element ULA

Consider the diagram in Figure 2.7, which shows a planewave phase front impinging on a 4-element uniform linear array (ULA) with element spacing d, from angle θ. Note the additional travel distance to each subsequent element of $d\sin\theta$. Given that EM waves in free space travel at the speed of light c, the phase front propagation time from element-to-element is given by $\Delta t_\theta = \frac{d\sin\theta}{c}$. Therefore, the time difference between the first element and element n is given by

$$\Delta t_{\mathrm{n},\theta} = \frac{(n-1)d\sin\theta}{c} \; . \tag{2.7}$$

The time shift property of the Fourier transform [35] states that a time shift of t_a in the function $g(t)$ corresponds to a phase shift in the frequency domain as given by

$$\mathcal{F}\{g(t-t_\mathrm{a})\} \triangleq e^{-j2\pi f t_\mathrm{a}} G(f) \; . \tag{2.8}$$

This property can be used to succinctly describe the relative phase shift in the frequency domain seen by any element n from any incident angle θ at any frequency f. This will be referred to as the measured phase response, and is given by

$$\begin{aligned} \mathbf{S}(\theta, n, f) &= e^{-j2\pi f \Delta t_{\mathrm{n},\theta}} \\ &= e^{-j2\pi f \frac{(n-1)d\sin\theta}{c}} \\ &= \begin{bmatrix} S_1(\theta, f) & S_2(\theta, f) & \cdots & S_N(\theta, f) \end{bmatrix} . \end{aligned} \tag{2.9}$$

Note that for the reference element $n = 1$, the relative phase shift is 0°. The reference element may be selected arbitrarily and is commonly chosen for mathematical convenience.

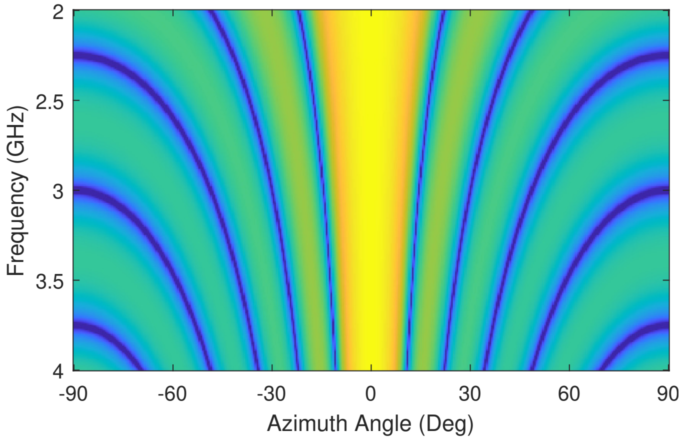

Figure 2.8: 8-Element ULA Boresight Array Factor

To form the array factor, one sums the tensor given in Eq. (2.9) along the element dimension to yield a complex-valued response over incident angle and frequency, as given by

$$\mathbf{S_0}(\theta, f) = \sum_{n=1}^{N} \mathbf{S}(\theta, n, f) = \sum_{n=1}^{N} e^{-j2\pi f \frac{(n-1)d\sin\theta}{c}} , \tag{2.10}$$

where N is the number elements. Figure 2.8 shows a plot of the magnitude of $\mathbf{S_0}(\theta, f)$ for an 8-element ULA with element spacing $d = \frac{\lambda_\mathrm{c}}{2}$ at $f_\mathrm{c} = 3$ GHz, from 2 to 4 GHz. The main beam at $\theta = 0°$ indicates that for a planewave incident at mechanical boresight, sensor signals sum coherently without the need for compensation.

2.3.1 Narrowband Beamforming

Classical narrowband beamforming is accomplished by selecting for each element a single phase shift with which to compensate the corresponding sensor signals. For the purposes of this discussion, element weight magnitudes are normalized to unitary. To minimize steering error over the signal bandwidth, it is prudent to select the frequency-invariant phase shifts at the waveform center frequency f_{c}. Although this is commonly defined as the arithmetic mean of f_{high} and f_{low}, the geometric mean results in an accurate steering angle. This distinction is of little consequence except for wide bandwidths at wide steering angles. The N-element measured phase response at the center frequency f_{c} is given by

$$\mathbf{S}(\theta) = \begin{bmatrix} S_1(\theta) & S_2(\theta) & \cdots & S_N(\theta) \end{bmatrix} , \tag{2.11}$$

where

$$S_n(\theta) = e^{-j2\pi f_{\mathrm{c}} \frac{(n-1)d}{c} \sin\theta} . \tag{2.12}$$

The steered array factor [38] is given by the inner product between the complex steering weight vector $\mathbf{w}$ and the measured phase response $\mathbf{S}$. In this dissertation, phased array compensation is discussed using the convention found in certain adaptive filtering texts [25], [39], where the application of the weight vector is given by the Hermitian transpose, as described by

$$y(\theta) = \mathbf{w}^H \mathbf{S}(\theta) = \begin{bmatrix} w_1^* & w_2^* & \cdots & w_N^* \end{bmatrix} \begin{bmatrix} S_1(\theta) \\ S_2(\theta) \\ \vdots \\ S_N(\theta) \end{bmatrix} , \tag{2.13}$$

where $w_n^* = a_n e^{j\phi_n}$. The variable $a_n = 1$ since weight magnitudes are currently constrained to unitary. Although merely a labeling convention, this facilitates the adaptive beamforming derivations provided in Chapter 3, which rely on complex-valued matrix calculus. In these derivations, row vectors are typically represented by the Hermitian transpose of the corresponding column vector. Some texts [23], [24] appear to focus primarily on real-valued filtering applications and thus use real-valued functions rather than the more general complex-valued case. Other texts [26] explore complex-valued filtering using the direct transpose of the filter coefficients rather than the Hermitian transpose.

The complex inner product in Eq. (2.13) yields the array factor, given by

$$y(\theta) = w_1^* S_1(\theta) + w_2^* S_2(\theta) + \cdots + w_N^* S_N(\theta) \ , \tag{2.14}$$

where

$$w_n^* S_n(\theta) = e^{-j(2\pi f_{\mathrm{c}} \frac{(n-1)d}{c} \sin\theta - \phi_n)} \ . \tag{2.15}$$

The maximum magnitude of Eq. (2.14) is achieved when terms are summed coherently. This occurs for the direction θ_{st} at which all sensor signals have the same phase, a relationship defined by

$$2\pi f_{\mathrm{c}} \frac{(n-1)d}{c} \sin\theta_{\mathrm{st}} - \phi_n = C \ . \tag{2.16}$$

The appropriate phase shifts to steer a beam to angle θ_{st} are determined by solving Eq. (2.16) for ϕ_n, namely

$$\phi_n = 2\pi f_{\mathrm{c}} \frac{(n-1)d}{c} \sin\theta_{\mathrm{st}} \ . \tag{2.17}$$

Note that the constant C does not impact the relative phase between elements and can be absorbed into the unknown phase constant of the impinging wave. For mathematical ease, C is set to 0.

Thus, the weighting component for element n which steers the main beam to θ_{st} is given by

$$w_n^* = a_n e^{j2\pi f_{\mathrm{c}} \frac{(n-1)d}{c} \sin\theta_{\mathrm{st}}} \ . \tag{2.18}$$

Although there is no constraint on amplitude distribution to coherently sum sensor signals for a single direction of interest, amplitude distributions can serve to reduce sidelobe levels at the expense of gain, a trade-off deemed beneficial in many receive cases. Hansen [32] discusses several distributions including the Dolph-Chebyshev, Taylor One-Parameter, and Taylor $\bar{n}$ distributions. Additional constraints on the weight vector $\mathbf{w}$, such as multiple beams or null constraints, can also impact the amplitude distribution [25]. As these amplitude distributions are not the subject of this section, classical beamforming is assumed and a uniform distribution applied to compensation weights. Null formation is discussed in Section 2.4 and adaptive beamforming in Chapter 3.

2.3.2 Wideband Beamforming

Narrowband beamforming assumes signal bandwidth does not extend appreciably from the waveform center frequency. Two system bandwidth considerations [40] are discussed in Sections 2.3.3.1 and 2.3.3.2. Frequency-dependent compensation values enable wider system bandwidths. Consider the elements of Eq. (2.11) with frequency variation as given by

$$S_n(\theta, f) = e^{-j2\pi f \frac{(n-1)d}{c} \sin\theta} \ . \tag{2.19}$$

The wideband antenna pattern is given by

$$y(\theta, f) = \mathbf{w}^H \mathbf{S}(\theta, f) = \begin{bmatrix} w_1^*(f) & w_2^*(f) & \cdots & w_N^*(f) \end{bmatrix} \begin{bmatrix} S_1(\theta, f) \\ S_2(\theta, f) \\ \vdots \\ S_N(\theta, f) \end{bmatrix} , \tag{2.20}$$

After following the sequence outlined in the previous section, Eq. (2.17) becomes frequency dependent, as given by

$$\phi_n(f) = 2\pi f \frac{(n-1)d}{c} \sin\theta_{\mathrm{st}} \ . \tag{2.21}$$

Thus, the weighting component for element n which steers the main beam to θ_{st} over frequency is given by

$$w_n^*(f) = a_n e^{j2\pi f \frac{(n-1)d}{c} \sin\theta_{\mathrm{st}}} \ . \tag{2.22}$$

The phase shift necessary to maintain a particular steering angle over frequency is a linear function of frequency. Solving Eq. (2.21) for θ_{st} shows the nonlinear frequency dependence of steering angle assuming frequency-invariant phase shifting. Compensation phases must track with frequency in order to maintain steering direction for wideband signals.

$$\theta_{\mathrm{st}}(f) = \sin^{-1}\left(\frac{c}{2\pi(n-1)d}\frac{\phi_n}{f}\right) \tag{2.23}$$

2.3.3 Narrowband/Wideband Comparison

A comparison of several ideal array factors is given in Figure 2.9. These array factors are simulated for an 8-element ULA with $d = \frac{\lambda_{\mathrm{c}}}{2}$ at $f_{\mathrm{c}} = 3$ GHz. The left column corresponds to $\theta_{\mathrm{st}} = 15^\circ$, the center column to $\theta_{\mathrm{st}} = 30^\circ$, and the right column to $\theta_{\mathrm{st}} = 45^\circ$. The top row shows narrowband beamforming, for which a beam is steered using frequency-invariant phase shifts provided using Eq. (2.17). The bottom row

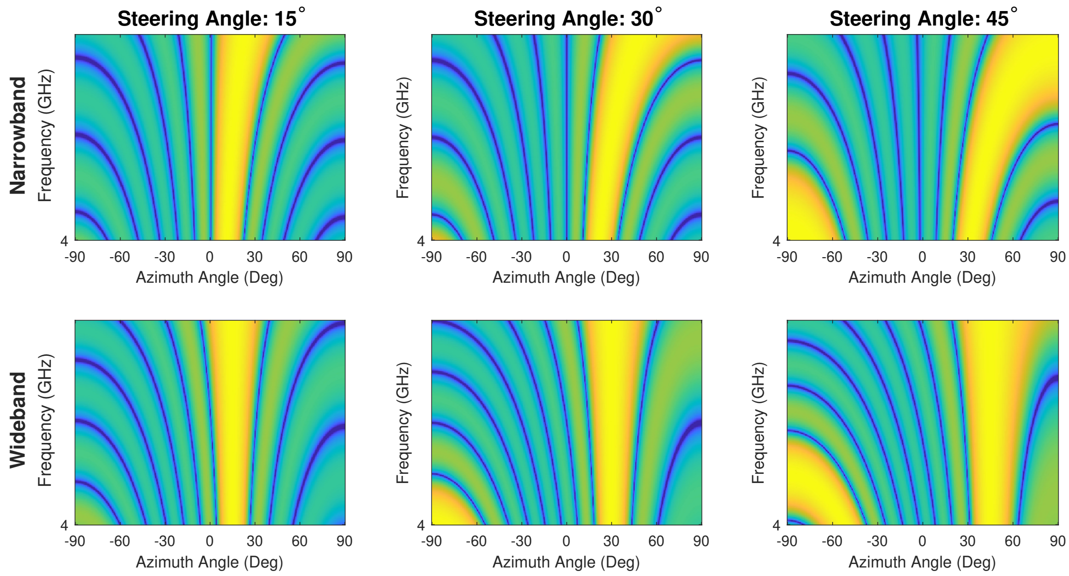

Figure 2.9: Array Factors for 8-Element ULA with $d = \frac{\lambda_{\mathrm{c}}}{2}$ at $f_{\mathrm{c}} = 3$ GHz, Left Column: Narrowband, Right Column: Wideband, Top Row: $\theta_{\mathrm{st}} = 15°$, Middle Row: $\theta_{\mathrm{st}} = 30°$, Bottom Row: $\theta_{\mathrm{st}} = 45°$

shows wideband beamforming, which is steered using the frequency-variant phase response described by Eq. (2.21), where 2 GHz $\leq f \leq$ 4 GHz.

2.3.3.1 Beam Squint

Note the inaccuracy of the beam position in the top row, which is correct only at the center frequency $f_{\mathrm{c}} = 3$ GHz. As described by Eq. (2.23), for a phase shift ϕ_n given by Eq. (2.17), beam accuracy progressively degrades with increasing $\Delta f = |f - f_{\mathrm{c}}|$. This phenomena is known as beam squint [41], and becomes more drastic with increasing steering angles. System bandwidth can be approximated by determining the frequency range over which the main beam response is within 3 dB of the peak response for the given steering direction [32]. System bandwidth may be asymmetric about the steering frequency due to the nonlinear nature of Eq. (2.23) coupled with the array element pattern [4]. Compensation may be required to ensure the 3-dB system bandwidth is symmetric about the waveform center frequency. Larger arrays have smaller beamwidths reducing system bandwidth [10].

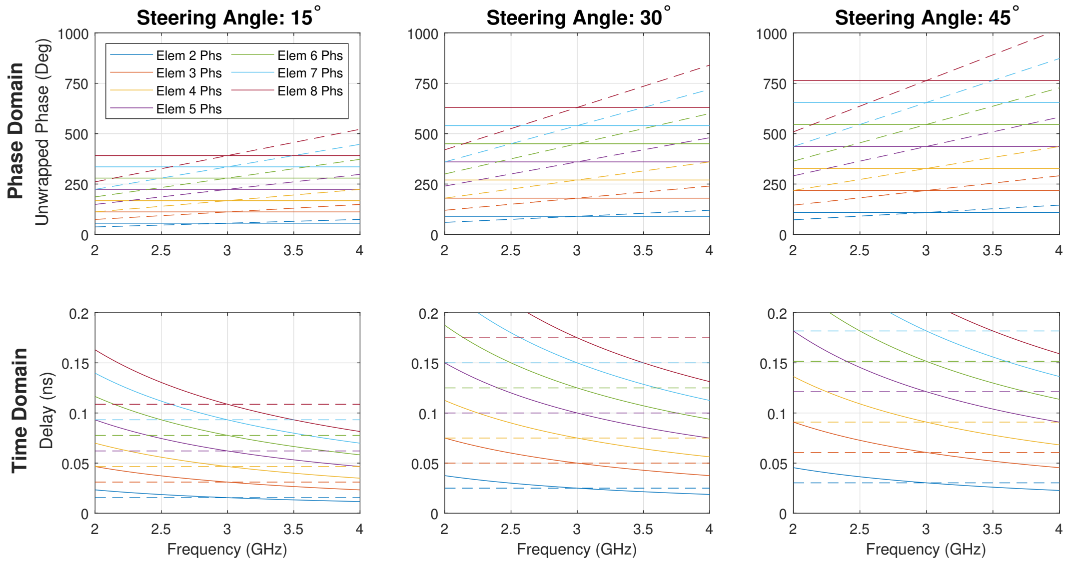

Figure 2.10: Compensation Delays for 8-Element ULA with $d = \frac{\lambda_c}{2}$ at $f_c = 3$ GHz, Left Column: Phase Delays, Right Column: Time Delays, Top Row: $\theta_{st} = 15°$, Middle Row: $\theta_{st} = 30°$, Right Row: $\theta_{st} = 45°$

In contrast, steering accuracy is maintained in the bottom row. All plots have frequency-varying sidelobe behavior, which can be mitigated through frequency-dependent amplitude tapering if required [42]. This is explored further in Section 2.5, which discusses arbitrary beamshaping. Narrowband and wideband array factors are equivalent at the center frequency cut $f = 3$ GHz for a given steering angle. Grating lobes can be seen at increased steering angles and increased electrical element displacement at the higher end of the spectrum.

Figure 2.10 shows the unwrapped ϕ_n responses used to compute the array factors in Figure 2.9. As in Figure 2.9, the left column corresponds to $\theta_{st} = 15°$, the center column to $\theta_{st} = 30°$, and the right column to $\theta_{st} = 45°$. Phase delays are shown in the top row while the corresponding time delays are shown in the bottom row, computed using the relation

$$\Delta t = \frac{\Delta \phi_n}{360°} \frac{c}{f} \ . \tag{2.24}$$

Because element 1 was selected as the phase reference, it is not plotted as it has no associated delay.

Solid lines show the frequency-invariant ϕ_n responses corresponding to the left column of Figure 2.9. Dashed lines show the frequency-variant ϕ_n responses corresponding to the right column of Figure 2.9. The linear phase response produces a constant time delay, electrically shifting mechanical boresight to the desired steering direction. Increased steering angles require increased delays as can be seen in the array geometry given in Figure 2.7.

2.3.3.2 Pulse Dispersion

Conventional phase shifters, commonly used for narrowband beamforming, support maximum delays of one cycle. When a phase shift of greater than one cycle is required, it will be wrapped to within 0° to 360°. This results in pulse dispersion, an example of which is shown in Figure 2.11.

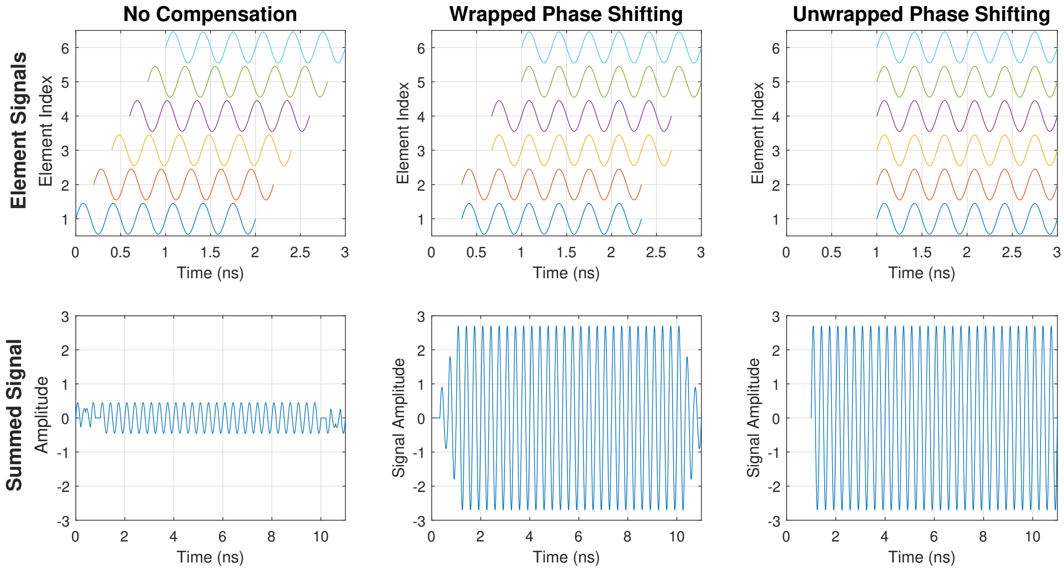

Figure 2.11: Wrapped and Unwrapped Phase Compensation Demonstrating Pulse Dispersion

The top left plot shows six periods of a sinusoid at $f_c = 3$ GHz as received by six elements spaced $d = \frac{\lambda_c}{2}$ apart. The propagation delay between elements is approximately 0.2 ns for the given incident angle of $\theta = 37°$. The middle left plot shows sensor signals when using conventional wrapped phase shifting. Because the relative delays between element 6 and elements 4 to 1 are greater than one cycle, compensated signals are coherent but not fully aligned. The bottom left plot shows sensor signals when adjustments of greater than one cycle are possible, which results in coherent and fully aligned sensor signals.

The right column shows signal summation of 30 cycles for the uncompensated, wrapped phase shifting, and unwrapped phase shifting cases. Wrapped phase shifting and unwrapped phase shifting both result in a fully collimated response when sensor signals fully overlap. The ramp times shown in the middle right plot from about 0.33-1.0 ns and 10.33-11.0 ns indicate pulse distortion. This ramp time is known as the aperture fill time [43] t_{fill}, and corresponds to the phase front propagation time delay across the array between the first and the last elements. It provides an estimate of the system bandwidth [43] for a given steering angle, as given by $B = \frac{1}{t_{\text{fill}}}$, although less conservative than the beam squint approach previously discussed [40]. Larger arrays have a larger aperture fill time, decreasing system bandwidth due to pulse distortion.

2.4 Null Formation

Commonly, it is desirable to spatially cancel interfering signals from a given direction. This can be accomplished deterministically by applying additional constraints to the beamforming calculations outlined in Section 2.3. Recall the steered array factor given in Eq. (2.20): $y(\theta, f) = \mathbf{w}(f)^H \mathbf{S}(\theta, f)$. The intent of deterministic beamforming is to solve Eq. (2.20) for $\mathbf{w}$ under one or more directional constraints. In the simplest case, classical beamforming [37] applies a distortionless constraint in the desired look direction θ_0, namely

$$\begin{aligned} 1 &= \mathbf{w}^H \mathbf{a}_0 \\ &= w_1^* + w_2^* e^{-jkd\sin(\theta_0)} + \cdots + w_N^* e^{-jkdN\sin(\theta_0)} \ , \end{aligned} \tag{2.25}$$

where $k = \frac{2\pi f}{c} = \frac{2\pi}{\lambda}$ is known as the wavenumber and λ is the wavelength. As this directional constraint is normalized by the array factor main beam magnitude, the resulting weight vector magnitude should be normalized to $\frac{1}{N}$. The column vector $\mathbf{a}_0 = \mathbf{S}(\theta_0)$ is the array manifold vector [44] for the direction θ_0. This vector provides the complex-valued adjustment as a function of incident angle between an incident waveform and the element signals due to array geometry. As noted in Section 2.3.1, the normalized weight vector which satisfies the above constraint cancels the complex component of each element such that

$$w_n^* = e^{jkd(n-1)\sin(\theta_0)} \ . \tag{2.26}$$

Null constraints, which constrain the array factor to zero for a given look direction θ_m, can be added to Eq. (2.25) as given by

$$\begin{aligned} \begin{bmatrix} 1 & 0 \cdots 0 \end{bmatrix} &= \mathbf{w}^H \begin{bmatrix} \mathbf{a}_0 & \mathbf{a}_1 & \cdots \mathbf{a}_M \end{bmatrix} \\ \mathbf{g}^T &= \mathbf{w}^H \mathbf{C} \ , \end{aligned} \tag{2.27}$$

where M is the number of desired nulls. In general, the number of nulls will be such that $\mathbf{C}$ is not square and an explicit matrix inverse will not exist. Godara [45] utilizes the Moore-Penrose Pseudoinverse [46] $\mathbf{C}^+$, defined in Eq. (2.28), to approximate $\mathbf{C}^{-1}$ through the method of least squares.

$$\mathbf{C}^+ = \mathbf{C}^H(\mathbf{C}\mathbf{C}^H)^{-1} \tag{2.28}$$

Barata [46] notes that although $(\mathbf{C}\mathbf{C}^H)^{-1}$ may not exist, the addition of a small offset term $\mu\mathbf{I}$ removes the singularity associated with the matrix inverse. Thus $(\mathbf{C}\mathbf{C}^H + \mu\mathbf{I})^{-1}$ does exist for all non-vanishing μ, and as such

$$\mathbf{C}^+ \approx \mathbf{C}^H(\mathbf{C}\mathbf{C}^H + \mu\mathbf{I})^{-1} \tag{2.29}$$

Hence, the final weight vector is approximated by

$$\begin{aligned} \mathbf{w}^H &\approx \mathbf{u}^T \mathbf{C}^+ \\ &\approx \mathbf{u}^T \mathbf{C}^H (\mathbf{C}\mathbf{C}^H + \mu \mathbf{I})^{-1} \ , \end{aligned} \tag{2.30}$$

where μ is chosen to be small compared to $\mathbf{C}\mathbf{C}^H$. Although each additional null constraint utilizes system DOFs, degrading the ability to coherently combine signals in the main beam, this is often a worthwhile trade off given the greater reduction in spatially correlated interferer energy.

2.4.1 Paley-Wiener Theorem

It is prudent to consider the quality of a prescribed null in how it manifests in the antenna pattern. The Paley-Wiener theorem [47] is defined as

$$\int_{-\infty}^{\infty} \frac{|\ln |F(\omega)||}{1+\omega^2} d\omega < \infty \ . \tag{2.31}$$

This criteria is often used to test system causality [35]. As Kak states [48], "If a square-integrable magnitude function $|F(\omega)|$ satisfies the Paley-Wiener condition, a suitable phase function can be associated with it to give a physically realisable system."

Alternatively, any physically realizable system will satisfy the Paley-Wiener theorem. Therefore, the antenna pattern for any physical array is constrained by the Paley-Wiener criteria. In noting that the natural logarithm is undefined for an argument of zero, one observes that the antenna magnitude response for any physical phased array system cannot, even in the ideal case, achieve a true null over any nonzero bandwidth. In practice, system imperfections further degrade null depths over both spatial frequency and temporal frequency.

2.5 Arbitrary Beamshaping

Although TTD beamforming completely mitigates beam squint and pulse dispersion, the resulting array factor remains frequency-variant, as shown in Figure 2.8. Frequency variation in beamwidth can have degrading effects on wideband synthetic aperture radar (SAR) given the inconsistent illumination area of the imaging target. An FIR filter bank can be synthesized to mitigate beam squint as well as provide a frequency-invariant system response using a desired template pattern [42].

We seek an FIR filter for each element whose combined output produces the desired array factor for the given bandwidth. To determine the desired filter responses, first consider the frequency-dependent array factor for an N-element ULA, as described in Eq. (2.10). It is repeated here for reference: $\mathbf{S_0}(\theta, f) = \sum_{n=1}^{N} \mathbf{S}_n(\theta, f)$, where $\mathbf{S}_n(\theta, f)$ is defined in Eq. (2.9). The magnitude response of this function is provided in Figure 2.8. The steered response $y(\theta, f)$ is given by the application of a weight vector $\mathbf{w}(f)$ to the measured phase response $\mathbf{S}(\theta, f)$, as defined in Eq. (2.20). Rather than determining the weight vector as described in Sections 2.3.1, 2.3.2, or 2.4, we specify a template pattern which is used as a target for a least-squares optimization over the desired bandwidth.

By prescribing a desired template array factor y_{T} over frequency and steering angle, one can determine the element weight vectors $\mathbf{w}$, which minimize the cost function,

$$\min_{\mathbf{w}} ||\mathbf{S}_f \mathbf{w}_f - y_{\mathrm{T},f}||_2^2 \tag{2.32}$$

where $\mathbf{S}_f$, $\mathbf{w}_f$, and $y_{\mathrm{T},f}$ are frequency cuts of $\mathbf{S}[\theta, n, f]$, $\mathbf{w}[n, f]$, and $y_{\mathrm{T}}[\theta, f]$ respectively. The closed form solution to this least squares problem is given by

$$\mathbf{w}_f = (\mathbf{S}_f^H \mathbf{S}_f)^{-1} \mathbf{S}_f^H y_{\mathrm{T},f} \ . \tag{2.33}$$

The frequency domain of the matrix $\mathbf{w}[n, f]$ is populated by solving Eq. (2.33) for each desired frequency cut. Channel filters can be synthesized using the resulting weight vectors $\mathbf{w}_n$. Filter synthesis of arbitrary frequency responses is covered in detail in Section 4.2.1.2. The resulting channel filters are denoted by the variable $\mathbf{h}_n$.

A beamshaping example steered to θ_{st} = -27° is given in Figure 2.12. The simulated aperture is an 8-element ULA with element spacing $d = \frac{\lambda_{f_{\mathrm{mid}}}}{2}$ for an instantaneous bandwidth of 1600 MHz centered at 3 GHz. The vertical dashed lines denote the steering angle while the horizontal dashed lines denote the desired bandwidth. The template array factor was calculated at $f_{\mathrm{low}} = 2.2$ GHz and applied to a bandwidth of 2 GHz.

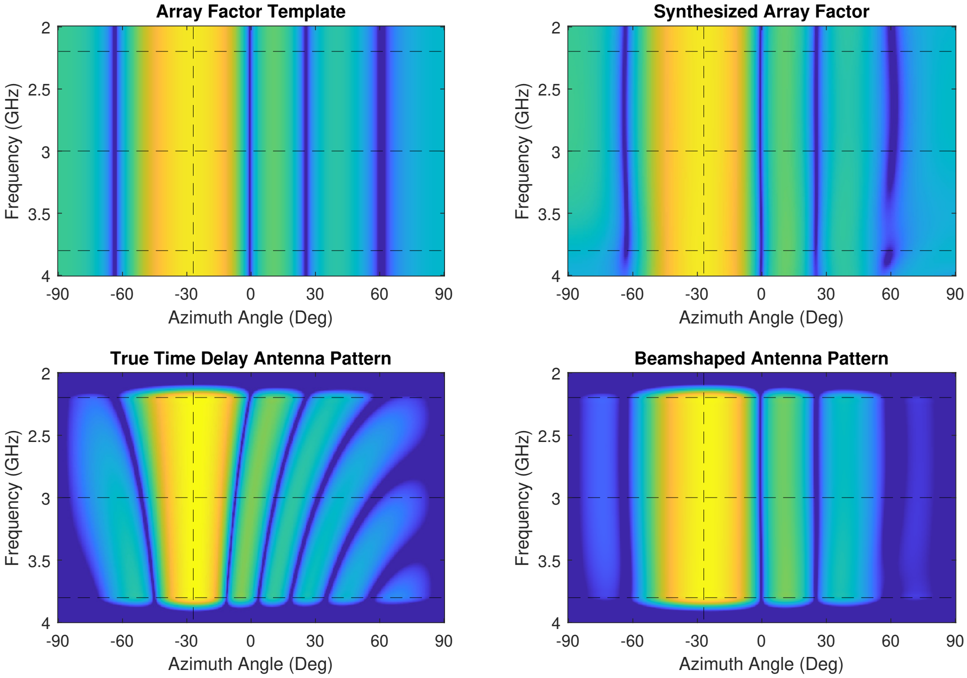

Figure 2.12: Upper Left: Pattern Template, y_{T}, Upper Right: Pattern Template over frequency, $y_{\mathrm{T},f}$, Lower Left: TTD Pattern, Lower Right: Beamshaped Pattern

The upper left plot shows the template pattern across the instantaneous bandwidth. The synthesized array factor $\tilde{y} = \mathbf{w}^H \mathbf{S}$ is given in the upper right plot, which approximately equals the template array factor y_{T}. Subtle variations can be seen towards the upper end of the band. The beamshaped antenna pattern, determined by passing the element signals through the synthesized weighting filters $\mathbf{h}_{\mathrm{n}}$, is given in the bottom right. A TTD wideband antenna pattern is given in the bottom left for reference. The antenna patterns in the bottom row assume a $\cos\theta$ element pattern.

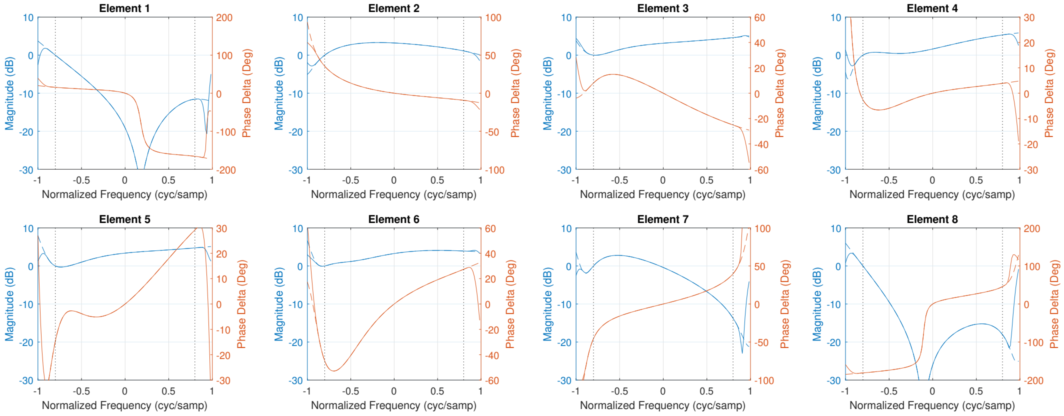

Figure 2.13: Beamshaping Compensation Filters

Figure 2.13 shows the frequency response for each channel filter. Desired filter responses $\mathbf{w}_n$ are given by solid traces and the synthesized filter response from the FIR filters $\mathbf{h}_n$ are given by dashed traces. Synthesized filters utilized 31 taps. Vertical dashed lines denote the bandwidth of interest, outside of which filter agreement is not necessitated. As expected, element tapering is induced in order to maintain the wider beamwidth as the array becomes electrically smaller at higher frequencies. Additionally, the center element filters tend to increase in magnitude in order to maintain a consistent gain across the bandwidth. Figure 2.14 shows the magnitude and phase error between the desired and synthesized filter responses. Good agreement is shown within the band of interest, with worst case errors of 0.02 dB and 0.1° present for the edge elements.

To further demonstrate synthesis of an arbitrary response, Figure 2.15 shows a wideband pattern with the OU logo nulled into the array factor sidelobes. Using the procedure presented in Section 2.4, separate array factor templates were specified over frequency containing nulls at prescribed locations. To account for the additional degrees of freedom (DOF) required for null synthesis, the number elements was increased to 32. The center of the band, where the ‘O’ and ‘U’ overlap, required

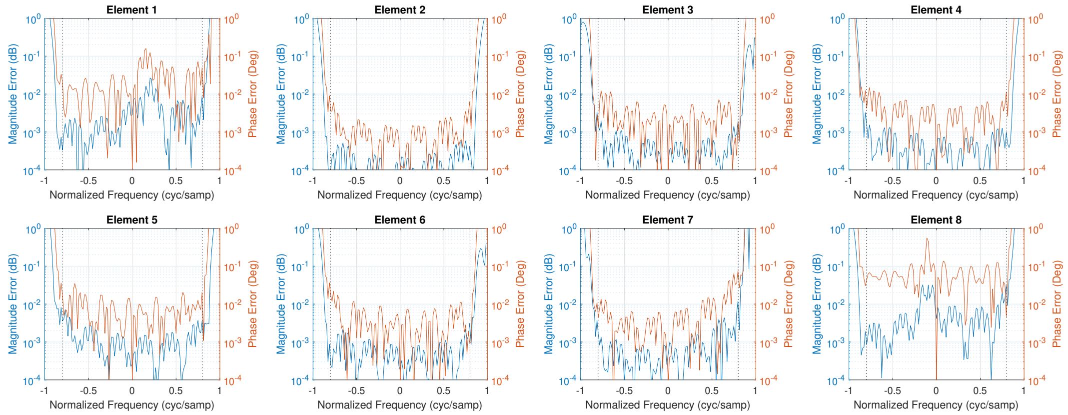

Figure 2.14: Beamshaping Compensation Filter Error

synthesis of four nulls and the main beam. Good agreement is shown between the template array factor y_T and the synthesized array factor $\tilde{y}$. The desired channel weights $\mathbf{w}$ contain high transient behavior near the transitions between varying frequency cuts. Thus, there is greater degradation between the synthesized array factor and the beamshaped antenna pattern shown in the lower right.

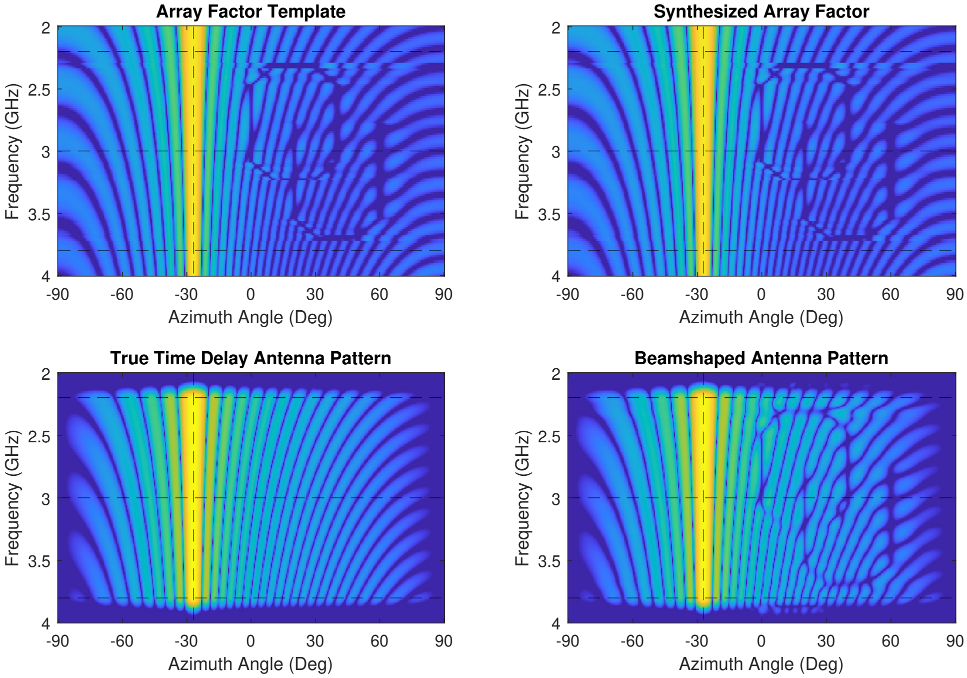

Figure 2.15: Arbitrary Pattern with OU Logo: Upper Left - Pattern Template, y_T, Upper Right - Pattern Template over frequency, $y_{\mathrm{T},f}$, Lower Left - TTD Pattern, Lower Right - Beamshaped Pattern

2.6 Beamforming Summary

In summary of the aforementioned beamforming techniques, Figure 2.16 shows a comparison of narrowband, wideband, and beamshaped antenna patterns. Frequency cuts at band edges and center show beam squint in the narrowband case and frequency varying sidelobe behavior in the wideband case. Antenna patterns in the beamshaping case show almost complete overlap.

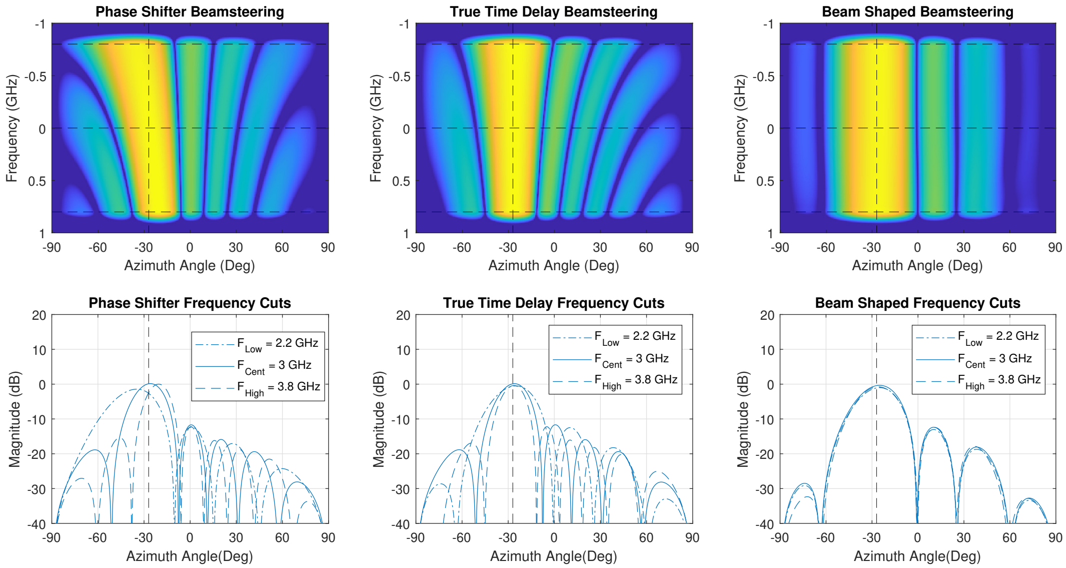

Figure 2.16: Beamforming Comparison of Phase Shifter, True Time Delay, and Beamshaping Beamforming

2.7 Sub-Nyquist-Sampled Beamforming

The rapid increase in ADC sampling rates has made practical the direct sampling of incident RF waveforms using an RFSoC, which can serve to reduce system complexity by mitigating the need for analog downconversion. However, given that the third generation of Xilinx's UltraScale+ RFSoC line supports ADC sample rates of up to 5.0 GHz, sampling above the Nyquist frequency is only practical into the lower portion of the S-band. Waveforms above 2.5 GHz require either some form of downconversion or bandpass sampling. Additionally, the proportional relationship between sampling frequency and power consumption [49] provides a compelling reason to limit the ADC sample rate to that which is required for system operation.

Sub-Nyquist sampling, or bandpass sampling, enables digital representation of bandpass signals sampled below the Nyquist rate. Sampled signals alias from higher Nyquist zones to within the digital spectrum without requiring frequency conversion [36]. Although this is theoretically valid for arbitrarily high Nyquist zones, the 3-dB cut off for the RFSoC ADCs used in this dissertation constrains the input spectrum to 4 GHz. RFSoC devices with broader input frequency ranges could enable bandpass sampling of higher frequency carriers, mitigating the need for external downconversion hardware. As outlined in Section 4.1.2, the beamformer testbed is designed to operate at a sample frequency of 4 GHz and support a 1.6 GHz bandwidth centered in the second Nyquist zone at 3 GHz. The sampled waveform aliases into the first Nyquist zone.

This section discusses the Shannon sampling theorem and presents sub-Nyquist sampling implications to beamforming. As given in Chapter 14 of [8], the Nyquist Sampling Theorem answers the question, "how frequently must samples be taken to adequately represent the analog signal?" Subsection 2.7.1 presents the sampling theorem and discusses its implications for digitizing carrier-modulated receive waveforms, also known as bandpass sampling [36]. Subsection 2.7.3 discusses implications to phased array beamforming due to time domain aliasing of the receive signals. A mathematical explanation and simulation results are provided.

2.7.1 Shannon Sampling Theorem

The well known Nyquist sampling theorem, described in several texts [8], [39], [50], is given by $f_\mathrm{s} > 2F_\mathrm{high}$, where F_high is the highest frequency in the waveform of interest. The variable $f_\mathrm{s} = 2F_\mathrm{high}$ is known as the critical sampling frequency [50]. However, it has just as profound of implications in the spatial domain as the critical element spacing for phased arrays [4], [32] is given by

$$d = \frac{\lambda}{2} \; . \tag{2.34}$$

Section 2.2 discusses some of the relationships between the time domain and spatial domain processing. Following the discussion in Chapter 14 of [8], a band-limited analog signal $x_{\mathrm{a}}(t)$ and its Fourier transform $X_{\mathrm{a}}(F)$ are presented and compared to the sampled signal $x_{\mathrm{s}}(t)$ and its Fourier transform $X_{\mathrm{s}}(F)$. The sampled signal can be derived from the continuous signal through multiplication with an impulse train, as given by

$$\begin{aligned} x_{\mathrm{s}}(t) &= x_{\mathrm{a}}(t)\left(\sum_{n=-\infty}^{\infty} \delta_{\mathrm{D}}(t - nT_{\mathrm{s}})\right) \\ &= \sum_{n=-\infty}^{\infty} x[n]\delta_{\mathrm{D}}(t - nT_{\mathrm{s}}) \; , \end{aligned} \tag{2.35}$$

where $x_{\mathrm{a}}(nT_{\mathrm{s}}) = x[n]$ and $\delta_{\mathrm{D}}(t)$ is the *Dirac delta function*. Morrison [50] notes that a signal which is discrete in one domain has a periodic Fourier transform. Hence, $X_{\mathrm{s}}(F)$ is given by

$$X_{\mathrm{s}}(F) = \frac{1}{T_{\mathrm{s}}} \sum_{k=-\infty}^{\infty} X_{\mathrm{a}}(F - kF_{\mathrm{s}}) \; . \tag{2.36}$$

Figure 2.17 shows a time domain sinc function with bandwidth $B = 1$ GHz and its corresponding *Rect* function Fourier spectrum. The continuous case and two discrete cases with different sample frequencies are presented. As described by Eq. (2.36), the Fourier transform shown in the upper right plot is replicated at every integer multiple of F_{s} for the middle right and lower right plots. In comparing the two sampled cases, one can see that as the sample frequency drops, the space between the spectrum replicas decreases. When the sample frequency is equal to twice the highest frequency in the original spectrum, $F_{\mathrm{high}} = \frac{B}{2} = 500$ MHz for the given example, the

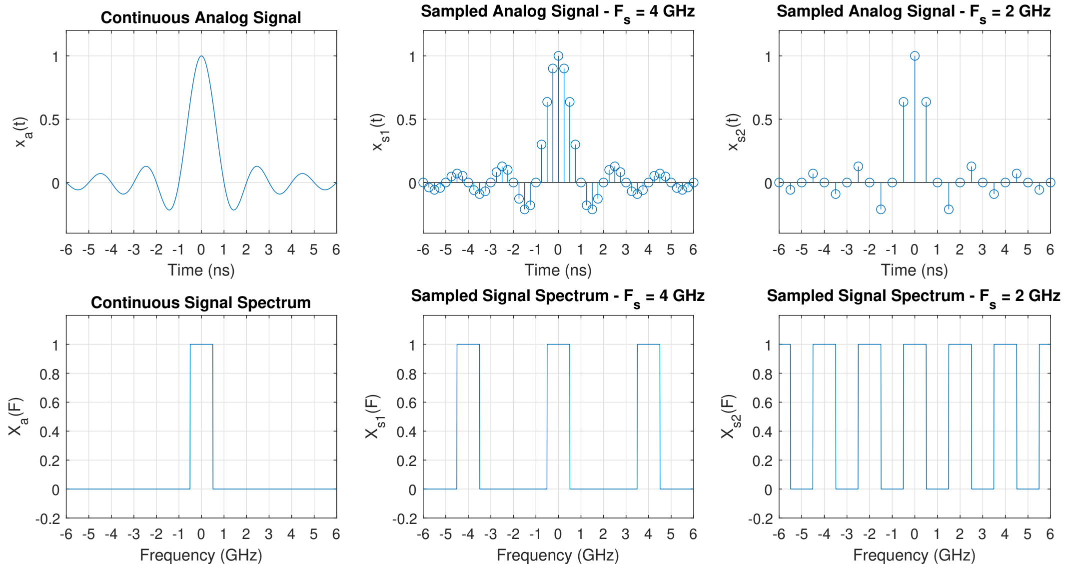

Figure 2.17: Time and Frequency Domain Representation of Continuous and Discrete 1 GHz Sync Functions at Baseband

spectrums will begin to overlap. This gives rise to the Nyquist sampling theorem, which ensures no overlap between repeated spectrums. In practice, because the signal spectrum rolls off more gradually than indicated in the above example, margin must be added to ensure aliasing does not occur (e.g. $f_s = 2.5F_{\text{high}}$ [7]).

2.7.2 Sub-Nyquist Sampling

Consider now a sinc function modulated to a carrier frequency $F_c = 4$ GHz, as shown in Figure 2.18. The corresponding 1 GHz spectrum is now centered around ±4 GHz. Sampling at $2F_{\text{high}} = 2(F_c + \frac{B}{2}) = 9$ GHz would enable full representation of the signal at the carrier frequency resulting in a pair of 1 GHz spectral replicas at every 9 GHz.

As demonstrated in Figure 2.17, reducing the sample frequency brings these replicas closer together. Assuming no other spectral content within the signal of interest, the sample frequency can be reduced well below the Nyquist rate of 9 GHz, until just before the replicas begin to overlap. This occurs at $F_s = B = 1$ GHz [8]. Sampling this modulated signal results in the same discrete samples, assuming coherency, as in

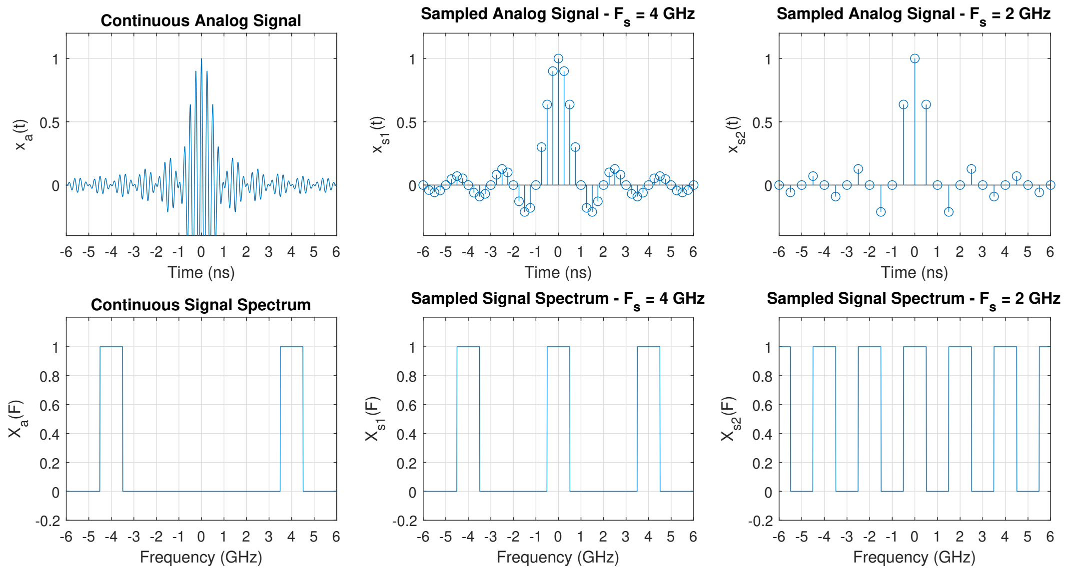

Figure 2.18: Time and Frequency Domain Representation of Continuous and Discrete 1 GHz Sync Functions at 4 GHz

Figure 2.17. Hence, as long as the sample frequency is selected to ensure the signal spectrum replicas do not overlap, the signal of interest can be fully represented from its discrete samples, regardless of carrier frequency. Care must be taken to ensure that no other spectral content aliases into the digitized waveform [36]. Hardware imperfections can make it difficult to ensure spectral content is confined within the signal of interest in practical systems, requiring great care to be taken when designing such systems. Specifically, nonlinear amplifier effects can manifest in transmit operation where PAs are commonly run in saturation, causing aliased signal spurs to degrade the waveform.

2.7.3 Sub-Nyquist-Sampled Beamforming Implications

Spectrum aliasing due to sub-Nyquist sampling affects the relative phase of received signals, impacting beamforming compensation. This phenomena is also present when beamforming at complex baseband due to frequency shifting [18]. Consider two discrete signals offset by a given time difference Δt, which corresponds to a phase difference $\Delta\phi$ at a given frequency F. When said signals alias to a new frequency

F_{a}, although the relative time difference is unaffected, the relative phase difference changes to $\Delta\phi_{\mathrm{a}}$ due to the new period $T_{F_{\mathrm{a}}}$.

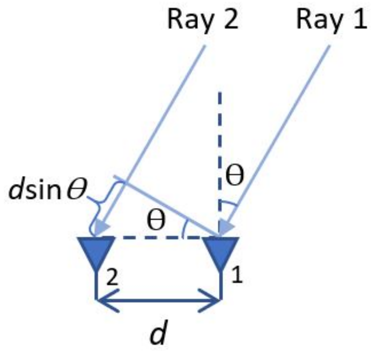

Figure 2.19: Ray Tracing Geometry for 2-Element Array

To further explore this, consider a monochromatic planewave of frequency F defined by

$$x(t) = \cos\left(2\pi F t\right) \ . \tag{2.37}$$

And suppose this planewave impinges on a two element array from an arbitrary angle θ, as shown in Figure 2.19. For mathematical simplicity, the impinging phase front is defined to have a phase offset of 0° at the first array element. The tone incident on the nearest element can be represented by Eq. (2.37) while the delayed tone is given by

$$x_{\mathrm{d}}(t) = \cos(2\pi F(t - \Delta t)) \ , \tag{2.38}$$

where $\Delta t = \frac{d \sin\theta}{c}$, as described in Section 2.3.

Sampling each signal at the sample frequency F_{s} yields the following two discrete functions:

$$\begin{aligned} x[m] &= \cos\left(2\pi \frac{f}{T_{\mathrm{s}}} m T_{\mathrm{s}}\right) \\ &= \cos\left(2\pi f m\right) \end{aligned} \tag{2.39}$$

and

$$\begin{aligned} x_{\mathrm{d}}[m] &= \cos\left(2\pi\frac{f}{T_{\mathrm{s}}}\left(mT_{\mathrm{s}}-\Delta t\right)\right) \\ &= \cos\left(2\pi f m - 2\pi\frac{f}{T_{\mathrm{s}}}\Delta t\right)\ . \end{aligned} \tag{2.40}$$

The time domain axis t relates to the sample index m by $t = mT_{\mathrm{s}}$ and the continuous time frequency F relates to the normalized discrete frequency f by $F = fF_{\mathrm{s}} = \frac{f}{T_{\mathrm{s}}}$. The sample period and sample frequency are given by T_{s} and F_{s} respectively.

Because sinusoids are 2π periodic, if the magnitude of f is greater than 0.5, the time-varying component of the sinusoidal argument will alias to a new normalized frequency f' such that $-0.5 \leq f' < 0.5$. Aliased representations of Eqs. (2.39) and (2.40) are given by

$$x[m] = \cos\left(2\pi f' m\right) \tag{2.41}$$

and

$$x_{\mathrm{d}}[m] = \cos\left(2\pi f' m - 2\pi\frac{f}{T_{\mathrm{s}}}\Delta t\right)\ . \tag{2.42}$$

The constant phase term in Eq. (2.42) contains the original discrete frequency f rather than the aliased frequency f'.

To align the two receive tones, the first sensor signal must be delayed by the phase front propagation time between the two elements, namely Δt. To apply a TTD to the discrete signal, a delay sample index can be defined as $m_{\mathrm{d}} = m - m_0$, where the delay sample offset $m_0 = \frac{\Delta t}{T_\mathrm{s}}$. The delay sample offset will generally not be an integer. Substituting the delay sample index into Eq. (2.39) yields

$$\begin{aligned} x[m] &= \cos\left(2\pi f m_\mathrm{d}\right) \\ &= \cos\left(2\pi f\left(m - m_0\right)\right) \\ &= \cos\left(2\pi f\left(m - \frac{\Delta t}{T_\mathrm{s}}\right)\right) \\ &= \cos\left(2\pi f m - 2\pi f\frac{\Delta t}{T_\mathrm{s}}\right) \ , \end{aligned} \tag{2.43}$$

which is equivalent to the second received tone described by Eq. (2.40). As a result, the two signals can be summed coherently.

Substituting the delayed index m_d into the aliased signal in Eq. (2.41) yields

$$\begin{aligned} x[m] &= \cos\left(2\pi f' m_\mathrm{d}\right) \\ &= \cos\left(2\pi f' m - 2\pi f'\frac{\Delta t}{T_\mathrm{s}}\right) \ . \end{aligned} \tag{2.44}$$

Note that the phase offset term in Eq. (2.44) differs from that in Eq. (2.42).

Eq. (2.44) can be further compensated with a correction phase ϕ_corr as given by

$$x[m] = \cos\left(2\pi f' m - 2\pi f'\frac{\Delta t}{T_\mathrm{s}} + \phi_\mathrm{corr}\right) \ . \tag{2.45}$$

Equating the cosine arguments of Eq. (2.42) and Eq. (2.45), one solves for ϕ_corr to yield

$$\begin{aligned} \phi_{corr} &= -2\pi\frac{\Delta t}{T_\mathrm{s}}(f - f') \\ &= -2\pi\Delta t(F - F') \ . \end{aligned} \tag{2.46}$$

This correction phase depends on the difference between the original frequency and aliased frequency and the time delay Δt between the current element and reference element. Note that pulse distortion should be considered if this correction phase grows beyond 1 cycle.

2.7.4 Phase Compensation Sign

Frequency folding must also be accounted for in array control for incident signals in an even Nyquist zone. Figure 2.20 shows two examples for a system with sample frequency $F_\mathrm{s} = 4$ GHz. The left plot shows the transmitted signal spectrum centered in the second Nyquist zone ($\frac{F_\mathrm{s}}{2} \leq f \leq F_\mathrm{s}$); the right plot shows the transmitted signal spectrum centered in the third Nyquist zone ($F_\mathrm{s} \leq f \leq \frac{3F_\mathrm{s}}{2}$). Under the DDC scheme outlined in Section 2.1.4, the signal spectrum located at $\frac{F_\mathrm{s}}{4}$ is shifted to baseband while the that at -$\frac{F_\mathrm{s}}{4}$ is filtered out.

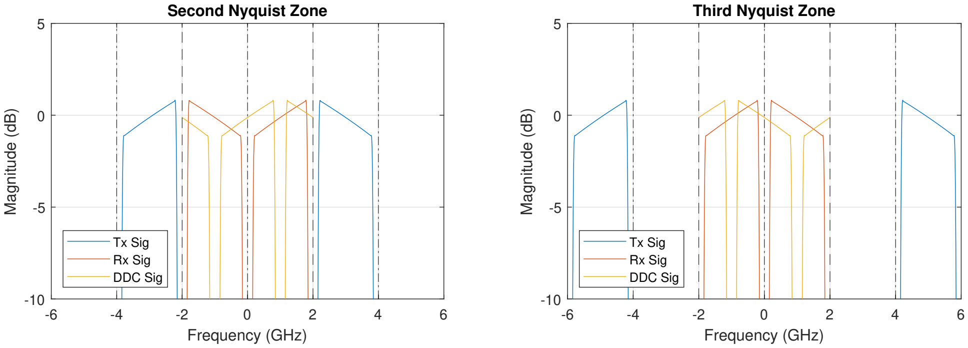

Figure 2.20: Frequency Folding

In the right plot, the orientation of the yellow baseband signal matches that of the positive frequency blue transmit signal, while it is reversed in the left plot. Due to frequency folding from the second Nyquist zone to the first, the conjugate spectrum ends up shifted to baseband. This manifests as a negation of the phase argument in the measured phase response, given in Eq. (2.9). This can be mitigated by negating the command steering angle θ_st or modifying the DDC to shift the -$\frac{F_\mathrm{s}}{4}$ spectrum to baseband instead. Either way, steering commands for a system sampling receive signals at a sub-Nyquist rate must account for frequency folding.

2.7.5 System Examples

This section presents antenna patterns for a narrowband and wideband system employing sub-Nyquist sampling. The wideband system is presented with and without the correction phase discussed in Section 2.7.3. Antenna pattern results are provided at various stages of the DDC to illustrate the beamsteering effect of each stage.

2.7.5.1 Phase Shifter Sub-Nyquist Sampled Example

The block diagram given in Figure 2.21 shows a narrowband system in which each channel has an ADC and phase shifter and the DDC is placed after the summation node. Figure 2.22 shows the frequency-dependent antenna pattern at various stages of the DDC for an 8-element array steered to $\theta_{\mathrm{st}} =$ -27°. The incident waveform is a 1.6 GHz wide chirp centered at 3 GHz and the sample frequency is $F_s = 4$ GHz. The vertical dashed line denotes the steering angle and the horizontal dashed lines denote the waveform bandwidth and the center frequency. The DDC follows the scheme in Section 2.1.4, in which the spectrum that has aliased to $\frac{F_s}{4} = 1$ GHz is downconverted to baseband.

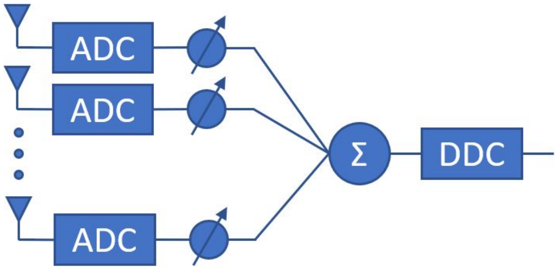

Figure 2.21: Phase Shifter Compensated System Block Diagram

Figure 2.23 shows an angle cut at $\theta_{\mathrm{st}} =$ -27° of the DDC spectrums given in Figure 2.22. Amplitude variation due to beam squint is readily apparent. The blue trace corresponds to the upper left plot of Figure 2.22, the red trace to the upper right plot, the yellow trace to the lower left plot, and the purple trace to the lower right plot. The asymmetry in the conjugate spectrums is due to the incoherency of

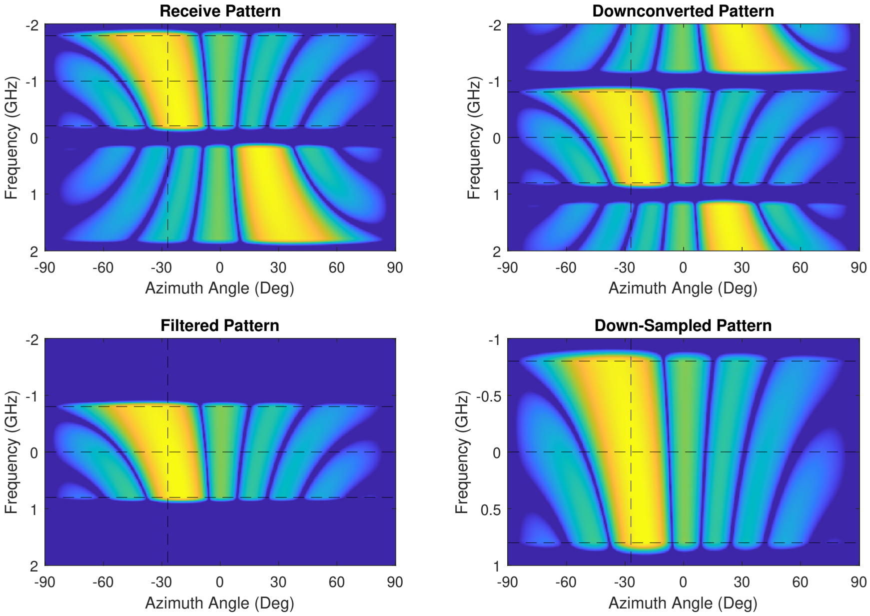

Figure 2.22: Narrowband Array Factor within DDC

the conjugate symmetric signals in the steering direction. In fact, as discussed in Section 2.7.4, a main beam appears in the conjugate spectrum at $\theta = 27°$. Hence, for the given system parameters, if the incident chirp was centered at 5 GHz instead of 3 GHz, the resulting phase commands would need to be negated.

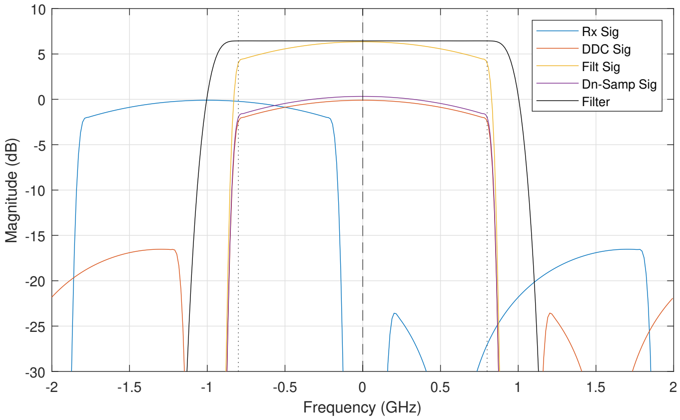

Figure 2.23: Narrowband Array Factor within DDC

2.7.5.2 TTD Sub-Nyquist Sampled Example

A block diagram for a wideband system without phase correction is given in Figure 2.24. Relevant system parameters are outlined in Section 2.7.5.1. The TTD units are implemented using sinc-based fractional delay FIR filters coupled with integer sample delays, discussed in detail in Chapter 4.

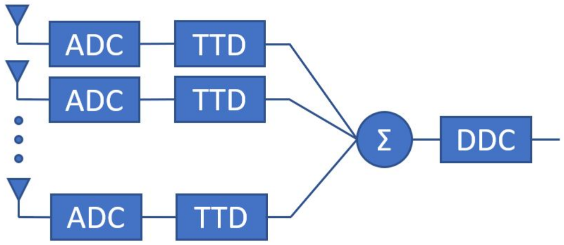

Figure 2.24: TTD Compensated System Block Diagram

The corresponding DDC patterns are shown in Figure 2.25. Although time shifts for a beam steered to $\theta_{\mathrm{st}} =$ -27° were appropriately calculated using Eq. (2.7), the resulting patterns show significant steering errors. Note that at $\theta_{\mathrm{st}} =$ -27°, the pattern response does not vary over frequency. This indicates that mechanical boresight has been shifted to the steering direction due to the channel delays, despite the main beam error. Figure 2.26 shows the steering angle cut corresponding to the plots in Figure 2.25. Due to TTD signal compensation, conjugate symmetry is present along the given steering angle.

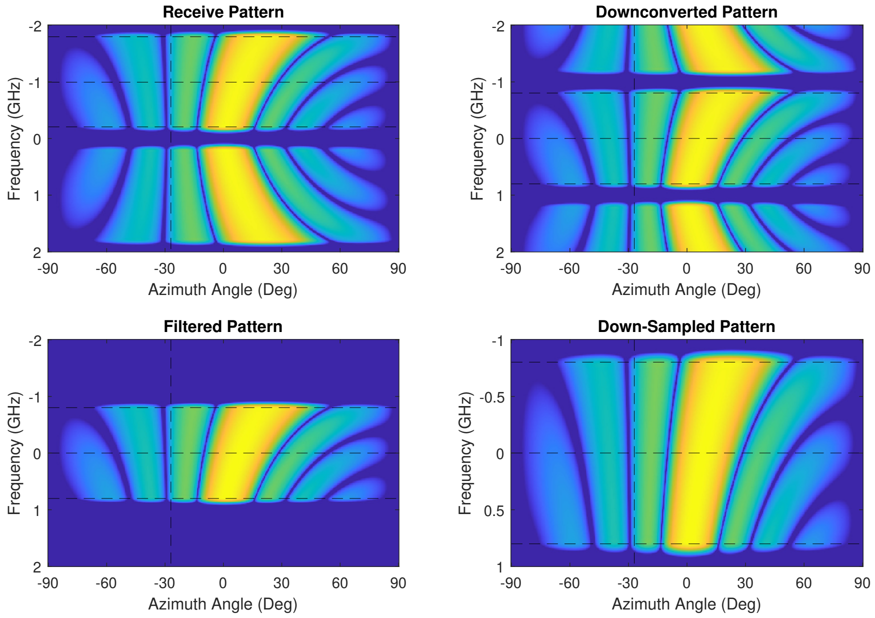

Figure 2.25: Wideband Array Factor within DDC - No Phase Correction

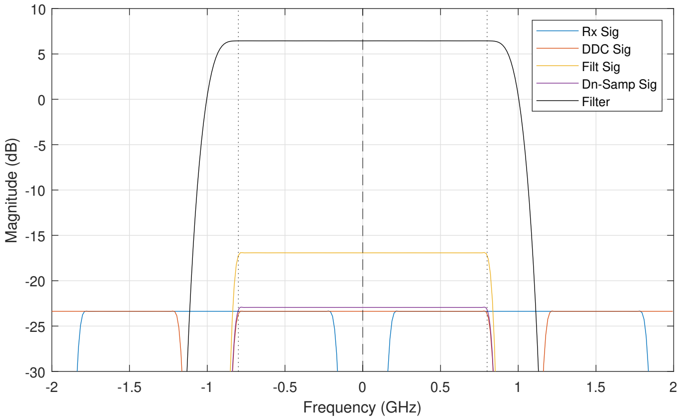

Figure 2.26: Narrowband Array Factor within DDC

2.7.5.3 Wideband Sub-Nyquist Sampled Example

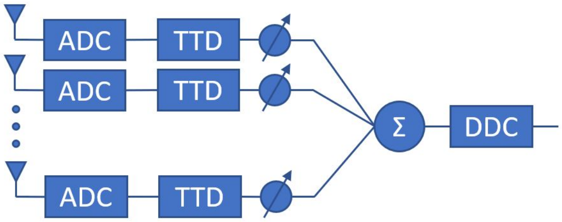

Figure 2.27: TTD Compensated System Block Diagram with Phase Correction

The block diagram is Figure 2.27 shows a TTD system with phase correction. The corresponding antenna patterns are shown in Figure 2.28 in which the main beam is appropriately located. The amplitude roll off shown in Figure 2.23 due to beam squint has now been corrected. The conjugate symmetry shown in Figure 2.26 is no longer present, as applying the phase correction shifts the conjugate spectrum in the opposite direction of the main beam. However, as mechanical boresight has been shifted to the steering angle due to TTD, the spectral response along the θ_{st} is flat in both the primary spectrum and the conjugate spectrum. Applying the TTD in conjunction with the appropriate phase shift results in a wideband response steered to the desired direction of $\theta =$ -27°.

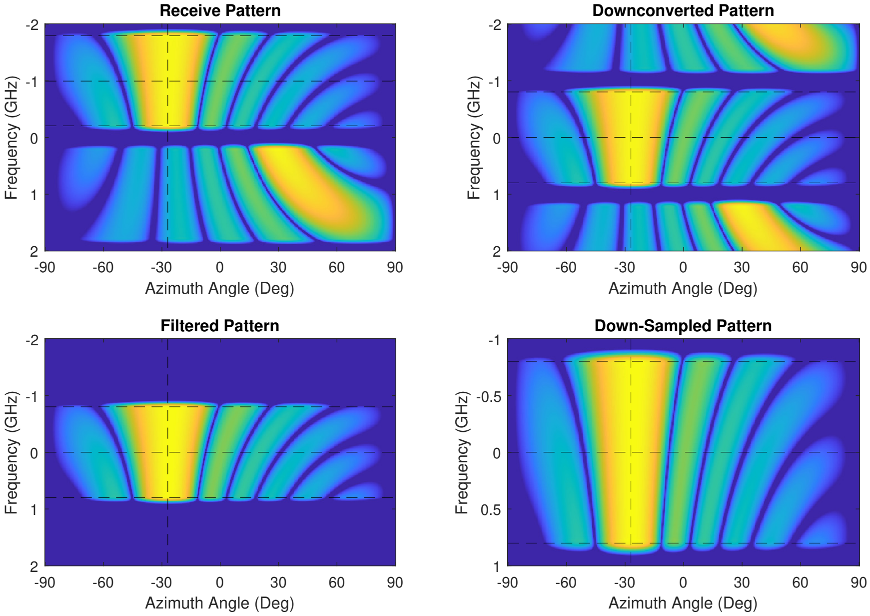

Figure 2.28: Wideband Array Factor within DDC - Phase Correction

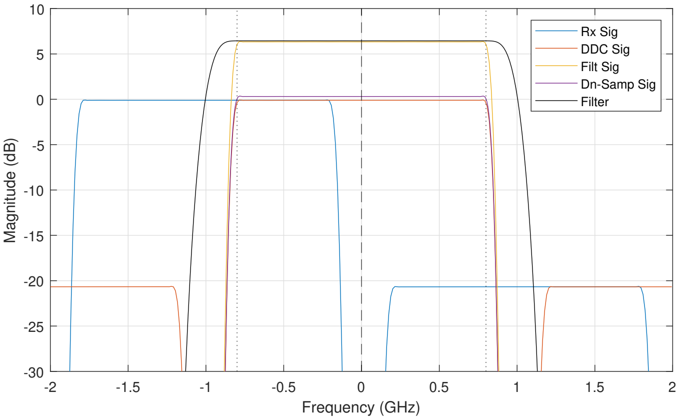

Figure 2.29: Narrowband Array Factor within DDC

Chapter 3

Adaptive Beamforming

Adaptive beamforming enables array control based on current or recently received system data seeking to discriminate between desired and undesired signals. Mozingo [22] notes that fundamental differences between desired and undesired signals can be exploited to aid SNR improvement. Spatial and temporal signal information provide estimates of statistical parameters that can be exploited to improve system performance. Often, seeking to enhance the desired signal while simultaneously degrading interfering signals involves trade-offs. Adaptive theory provides methods by which one can optimize these goals improving system performance.

In this chapter, Section 3.1 presents the mathematical framework from which the adaptive algorithms are derived. Array representation is provided for ULA's as well as arbitrary geometry. Section 3.2 presents a technique for estimating environment statistics for use in adaptive algorithms. Several derivations of iterative and statistically optimum adaptive algorithms are presented in Section 3.3. And finally, algorithm comparisons and simulations are provided in Section 3.4.

3.1 System Representation

This section presents a mathematical representation of a phased array system from which adaptive beamforming algorithms can be derived. For the sake of simplicity, array discussions presented in this chapter pertain to the ULA, for which a steering direction is fully defined by a single angle θ. Section 3.1.1 provides the framework to expand array representation and control to 2-dimensional and 3-dimensional arrays.

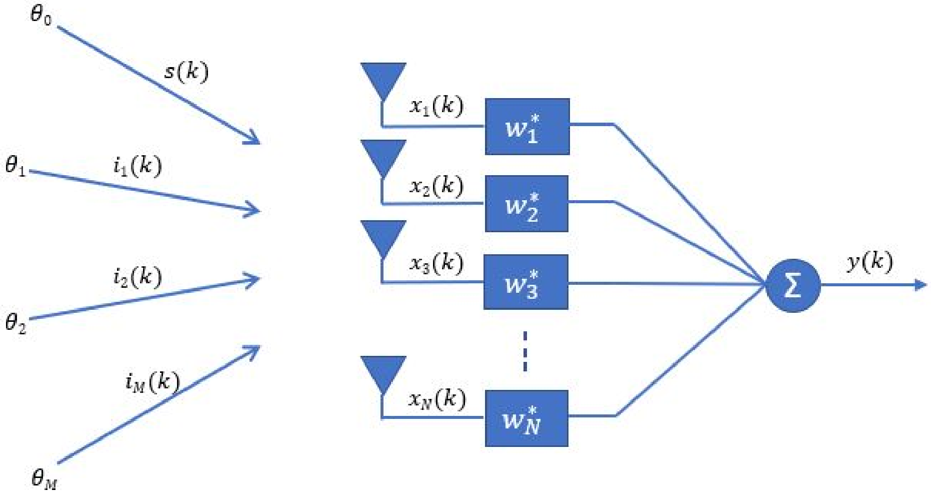

Figure 3.1: Linear Array Block Diagram

Figure 3.1 provides a block diagram of a ULA to aid in the discussion of a phased array mathematical representation. As previously mentioned, this dissertation follows the convention of using the element weight conjugate w_n^* in the mathematical system representation, as employed in [25] and [39]. As depicted, the array output is given by

$$y(k) = \mathbf{w}^H \mathbf{x}(k) \ , \tag{3.1}$$

where the weight vector $\mathbf{w}$ containing the complex array weights is given by

$$\mathbf{w}^H = \begin{bmatrix} w_1^* & w_2^* & \cdots & w_N^* \end{bmatrix} \ . \tag{3.2}$$

The receive signal vector $\mathbf{x}(k)$, as described by Eq. (3.3), contains the signal received by each element as a function of time, indexed by k. It is the superposition of the signal of interest vector $\mathbf{x}_s(k)$ and the undesired signal vector $\mathbf{u}(k)$, which is comprised of the interference signal vector $\mathbf{x}_i(k)$ and the noise vector $\mathbf{n}(k)$.

$$\begin{aligned} \mathbf{x}(k) &= \mathbf{x}_s(k) + \mathbf{u}(k) \\ &= \mathbf{x}_s(k) + \mathbf{x}_i(k) + \mathbf{n}(k) \ . \end{aligned} \tag{3.3}$$

The array manifold vector [44] translates an incident signal to the element signals. It is a subset of the measured phase response described in Eq. (2.9). Therefore, the signal of interest $\mathbf{x}_s(k)$ can be represented by the product of the array manifold vector $\mathbf{a}_0$ and the signal $s(k)$ impinging on the array, namely

$$\mathbf{x}_s(k) = \mathbf{a}_0 s(k) \ , \tag{3.4}$$

where an array manifold vector pertaining to the incident angle θ_m for an N-element ULA with element spacing d and wavenumber k is given by

$$\mathbf{a}_m = \begin{bmatrix} 1 \\ e^{-jkd\sin(\theta_m)} \\ \vdots \\ e^{-jkdN\sin(\theta_m)} \end{bmatrix} \ . \tag{3.5}$$

The undesired signal is comprised of the incident interferers and noise received by the array, as given by

$$\mathbf{u}(k) = \mathbf{x}_i(k) + \mathbf{n}(k) \ , \tag{3.6}$$

where

$$\mathbf{x}_i(k) = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \cdots & \mathbf{a}_M \end{bmatrix} \begin{bmatrix} i_1(k) \\ i_2(k) \\ \vdots \\ i_M(k) \end{bmatrix} \tag{3.7}$$

and

$$\mathbf{n} = \begin{bmatrix} n_1 \\ n_2 \\ \vdots \\ n_N \end{bmatrix} \ . \tag{3.8}$$

Elements of the noise vector $\mathbf{n}(k)$ are assumed to be uncorrelated zero-mean Gaussian noise.

3.1.1 Multidimensional Arrays

In order to describe a planar or conformal phased array system, one must modify the array manifold vector to account for additional dimensions in both arbitrary element positions and incident angle directions. For a multidimensional array, element positions are described by x, y, and z components and steering directions require two angles, typically θ and ϕ. Consider the directional wavenumber $\mathbf{k}$, which describes the spatial frequency projected along the direction defined by θ and ϕ. It can be described by

$$\mathbf{k} = \frac{2\pi}{\lambda}\begin{bmatrix}\sin\theta\cos\phi\\ \sin\theta\sin\phi\\ \cos\theta\end{bmatrix} = k\begin{bmatrix}\sin\theta\cos\phi\\ \sin\theta\sin\phi\\ \cos\theta\end{bmatrix}. \tag{3.9}$$

As expected, the magnitude of the directional wavenumber is $\frac{2\pi}{\lambda} = k$. If the position for element n is given by

$$\mathbf{p}_n = \begin{bmatrix}p_{x_n}\\ p_{y_n}\\ p_{z_n}\end{bmatrix}, \tag{3.10}$$

then the multidimensional array manifold vector can be written as

$$\mathbf{a}_m = \begin{bmatrix}e^{-j\mathbf{k}^T_{\theta_m,\phi_m}\mathbf{p}_0}\\ e^{-j\mathbf{k}^T_{\theta_m,\phi_m}\mathbf{p}_1}\\ \vdots\\ e^{-j\mathbf{k}^T_{\theta_m,\phi_m}\mathbf{p}_N}\end{bmatrix}. \tag{3.11}$$

Note that for a ULA with element spacing d oriented along the x-axis, for which $\phi = 0°$, Eqs. (3.9) and (3.10) reduce to

$$\mathbf{k} = k \begin{bmatrix} \sin\theta \\ 0 \\ \cos\theta \end{bmatrix} \tag{3.12}$$

and

$$\mathbf{p}_n = \begin{bmatrix} p_{x_n} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} nd \\ 0 \\ 0 \end{bmatrix} , \tag{3.13}$$

respectively. In substituting Eqs. (3.12) and (3.13), Eq. (3.11) simplifies down to Eq. (3.5) as given by

$$\mathbf{a}_m = \begin{bmatrix} e^{-jk0d\sin\theta_m} \\ e^{-jk1d\sin\theta_m} \\ \vdots \\ e^{-jkNd\sin\theta_m} \end{bmatrix} = \begin{bmatrix} 1 \\ e^{-jkd\sin\theta_m} \\ \vdots \\ e^{-jkNd\sin\theta_m} \end{bmatrix} . \tag{3.14}$$

3.2 Sample Matrix Inversion

As will be shown in Section 3.3, many adaptive algorithms require knowledge of the array correlation matrix. Gross [25] notes that much of the literature uses the array covariance matrix instead of the array correlation matrix, which is valid when the corresponding signals are zero mean. This will be the assumption carried throughout this chapter. Relevant covariance matrices include the total signal covariance matrix

$\mathbf{R}_{xx}$, the desired signal covariance matrix $\mathbf{R}_{ss}$, the undesired signal covariance matrix $\mathbf{R}_{uu}$, the interferer covariance matrix $\mathbf{R}_{ii}$, and the noise covariance matrix $\mathbf{R}_{nn}$, where

$$\begin{aligned}\mathbf{R}_{xx} &= \mathbf{R}_{ss} + \mathbf{R}_{uu} \\ &= \mathbf{R}_{ss} + \mathbf{R}_{ii} + \mathbf{R}_{nn} \ .\end{aligned} \tag{3.15}$$

In practice, because the signal statistics are not known, they must be estimated based on current array samples [25]. An estimate of the covariance matrix can be computed by

$$\hat{\mathbf{R}} = \frac{1}{K}\sum_{k=1}^{K}\mathbf{x}(k)\mathbf{x}^{H}(k) \ , \tag{3.16}$$

where $\mathbf{x}(k)$ is an N-element vector containing the k^{th} array sample and K is the number of samples in the covariance matrix estimate. The sample covariance matrix $\hat{\mathbf{R}}$ can be used in place of the true covariance matrix.

3.2.1 Reed-Mallett-Brennan Rule

Naturally, when estimating the covariance matrix, there is a trade-off between minimizing sample size and maximizing accuracy. The Reed-Mallett-Brennan (RMB) rule [51] provides the following relationship to aid in this trade-off decision.

$$loss = -10\log_{10}\left(\frac{K+2-N}{K+1}\right) \tag{3.17}$$

where K, as defined in Eq. (3.16), is the number of signal vector $\mathbf{x}$ samples and N is the number of array elements. Eq. (3.17) can be rewritten as

$$loss = -10\log_{10}\left(\frac{(K'-1)N+2}{K'N+1}\right) \tag{3.18}$$

where $K'N = K$. The RMB rule predicts the loss in dB from the ideal SNR as a function of the number of elements N and the number of samples $K'N$. Figure 3.2

shows a plot of Eq. (3.18) as a function of the number of elements N for multiple values of K'.

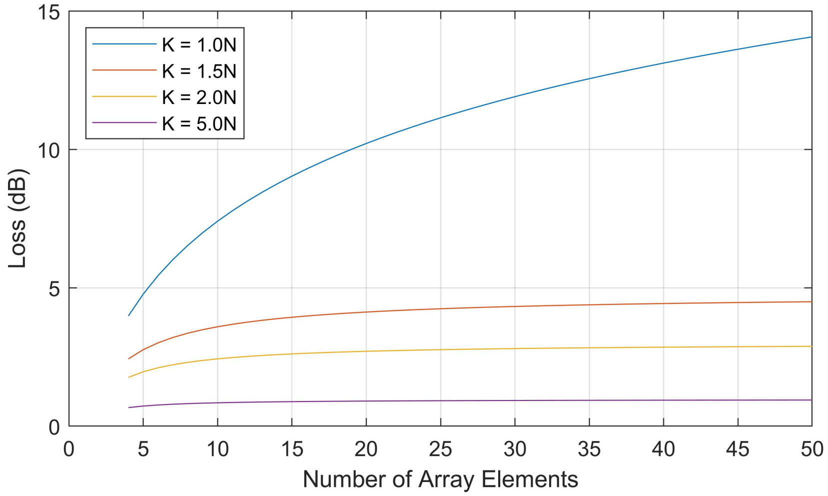

Figure 3.2: Loss from optimum response when using covariance matrix estimate

This shows that for $K' = 2$, the sample covariance matrix estimate results in a loss of about 3 dB as compared to if the true statistics were known. Sensor signals are assumed to be stationary over the sample period and operation time of the adaptive array. It should be noted that Eq. (3.18) appears to be valid for values of $K' > 1$ given that values of $K' < 1$ predict lower loss than when $K' = 1$. Additionally, the $K' = 1$ case does not show the asymptotic response shown for cases of $K' > 1$. This indicates that the RMB rule should only be applied to sample sizes greater than the number of array elements N.

3.3 Adaptive Algorithms

The following section provides derivations for several adaptive algorithms. As will be shown, the Minimum Mean Square Error (MMSE) [25] derivation results in the *Wiener-Hopf equations* [26]. Based on the MMSE derivation, the iterative Least Mean Squares (LMS) [25] and Recursive Least Squares (RLS) [52] algorithms are derived. Subsequently, the Maximum Signal to Interference/Noise (SINR) [25] and Maximum

Likelihood (ML) [25] algorithms are presented. And lastly, the Minimum Variance Distortionless Response (MVDR) [44] and Linearly Constrained Minimum Variance (LCMV) [44] algorithms are given.

3.3.1 Minimum Mean Squared Error

One way to determine the ideal filter weights $\mathbf{w}_{opt}$ is to minimize the mean squared error (MSE) between the received signal and the desired signal. This is the method by which the *Wiener Filter* is derived [26]. Figure 3.3 shows an adaptive array for which the current desired signal sample is represented by $d(k)$ and the current array output signal sample, an estimate of the desired signal, is represented by $y(k)$.

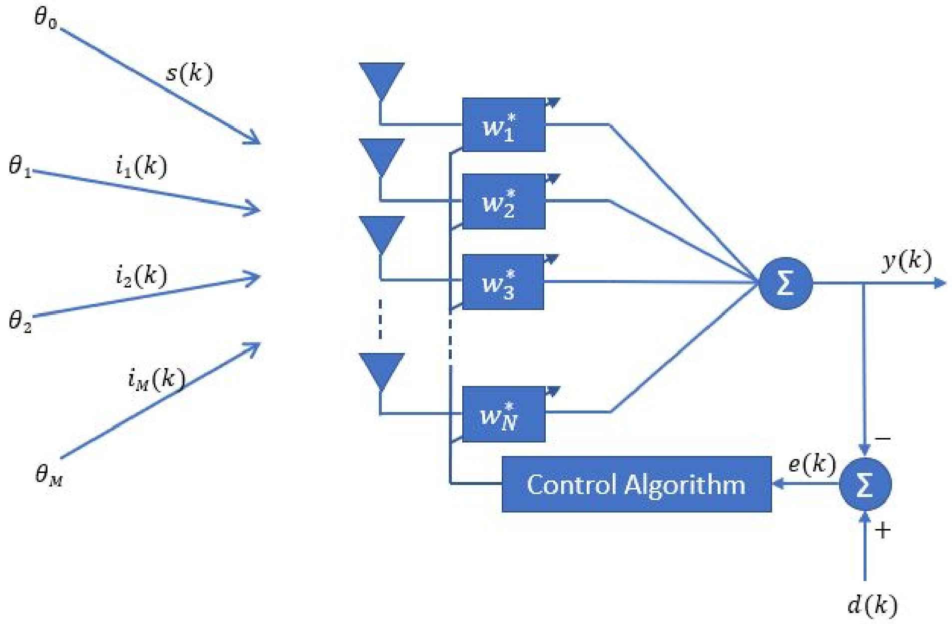

Figure 3.3: Adaptive Array Block Diagram

The current error sample is given by

$$\begin{aligned} e(k) &= d(k) - y(k) \\ &= d(k) - \mathbf{w}_k^H \mathbf{x}(k) \end{aligned} \tag{3.19}$$

and the squared error magnitude as

$$\begin{aligned}|e(k)|^2 &= |d(k)-y(k)|^2 \\ &= \big(d(k)-y(k)\big)\big(d(k)-y(k)\big)^* \\ &= \big(d(k)-y(k)\big)\big(d^*(k)-y^*(k)\big) \\ &= d(k)d^*(k)-d(k)y^*(k)-d^*(k)y(k)+y(k)y^*(k) \\ &= |d(k)|^2-d(k)\mathbf{x}^H(k)\mathbf{w}_k-d^*(k)\mathbf{w}_k^H\mathbf{x}(k)+\mathbf{w}_k^H\mathbf{x}\mathbf{x}^H\mathbf{w}_k \ .\end{aligned} \tag{3.20}$$

Some texts [22], [24] present the squared error assuming real vector values, such that $e(k)^2 = d(k)^2 - 2d(k)\mathbf{x}^T(k)\mathbf{w}_k + \mathbf{w}_k^T\mathbf{x}\mathbf{x}^T\mathbf{w}_k$, while others [52], [39], [44] present the more general and applicable complex case given above. Gross [25] obscured the matter as he appeared to carry complex notation throughout the derivation while combining the inner and outer terms as if they were real. This resulted in the following equation $|e(k)|^2 = |d(k)|^2 - 2d(k)\mathbf{x}^H(k)\mathbf{w}_k + \mathbf{w}_k^H\mathbf{x}\mathbf{x}^H\mathbf{w}_k$, which the author of this dissertation was unable to reconcile.

Given that $x(k)$ and $d(k)$ are nonstationary random processes, the mean of the squared error, or the MSE, can be found by taking the expectation of Eq. (3.20), as given by

$$\begin{aligned}\xi &= E\{|e(k)|^2\} \\ &= E\{|d(k)|^2\} - E\{d\mathbf{x}^H(k)\mathbf{w}\} - E\{d^*\mathbf{w}^H\mathbf{x}(k)\} + E\{\mathbf{w}^H\mathbf{x}(k)\mathbf{x}^H(k)\mathbf{w}\} \\ &= E\{|d(k)|^2\} - E\{d\mathbf{x}^H(k)\}\mathbf{w} - \mathbf{w}^H E\{d^*\mathbf{x}(k)\} + \mathbf{w}^H E\{\mathbf{x}(k)\mathbf{x}^H(k)\}\mathbf{w} \\ &= E\{|d(k)|^2\} - \mathbf{r}_{xd^*}^H\mathbf{w} - \mathbf{w}^H\mathbf{r}_{xd^*} + \mathbf{w}^H\mathbf{R}_{xx}\mathbf{w} \ ,\end{aligned} \tag{3.21}$$

where because the weight vector $\mathbf{w}$ is deterministic, it can be pulled out of the expectation operator. Note that the time dependence of the weight vector $\mathbf{w}$ has been removed given that Eq. (3.21) describes the mean system response. $\mathbf{R}_{xx}$ represents the total signal covariance matrix as defined by $E\{\mathbf{x}(k)\mathbf{x}^H(k)\}$ while $\mathbf{r}_{xd^*}$ represents the cross covariance vector between the desired signal and the current signal vector as defined by $E\{d^*\mathbf{x}(k)\}$.

The weight vector $\mathbf{w}_{\text{opt}}$ which minimizes the convex function given by Eq. (3.21) can be determined by setting the gradient of ξ with respect to the weight vector equal to zero. As described by Brandwood [21], because ξ is a real valued function of a complex vector and its Hermitian transpose, the minimum can be found by setting the gradient equal to zero with respect to either $\mathbf{w}^H$ or $\mathbf{w}$. Both cases are presented in Eqs. (3.22) and (3.23) respectively.

$$\nabla_{\mathbf{w}^H}\xi = -\mathbf{r}_{xd^*} + \mathbf{R}_{xx}\mathbf{w}_{\text{opt}} = 0 \tag{3.22}$$

$$\begin{aligned}\nabla_{\mathbf{w}}\xi &= -\mathbf{r}^*_{xd^*} + (\mathbf{R}^H_{xx}\mathbf{w}_{\text{opt}})^* = 0\\ &= -\mathbf{r}^*_{xd^*} + \mathbf{R}^T_{xx}\mathbf{w}^*_{\text{opt}} = 0\end{aligned} \tag{3.23}$$

Solving Eq. (3.23) for $\mathbf{w}_{\text{opt}}$ yields

$$\begin{aligned}\mathbf{w}^*_{\text{opt}} &= (\mathbf{R}^T_{xx})^{-1}\mathbf{r}^*_{xd^*}\\ \mathbf{w}_{\text{opt}} &= (\mathbf{R}^H_{xx})^{-1}\mathbf{r}_{xd^*}\\ &= \mathbf{R}^{-1}_{xx}\mathbf{r}_{xd^*}\ ,\end{aligned} \tag{3.24}$$

where for a Hermitian matrix, $A^H = A$. Eq. (3.24) clearly matches the solution to Eq. (3.22) and is the well known *Wiener-Hopf equations* [26] applied to a spatial filter. Note that although Eqs. (3.22) and (3.23) have the same solution $\mathbf{w}_{\text{opt}}$, $\nabla_{\mathbf{w}^H}\xi(k)$ and $\nabla_{\mathbf{w}}\xi(k)$ are themselves conjugates of each other.

In the ideal case, the reference signal $d(k)$ is equal to the desired signal $s(k)$ impinging on the array. Recalling the definition of the received signal $\mathbf{x}(k)$ given in

Eq. (3.3), and assuming that $s(k)$ is uncorrelated with the undesired interferers and noise, the cross correlation vector $\mathbf{r}_{xd^*}$ can be rewritten as

$$\begin{aligned}\mathbf{r}_{xd^*} &= E\{s^*(k)\mathbf{x}(k)\} \\ &= E\{s^*(k)(\mathbf{x}_s(k) + \mathbf{x}_i(k) + \mathbf{n}(k))\} \\ &= E\{s^*(k)\mathbf{x}_s(k)\} + E\{s^*(k)\mathbf{x}_i(k)\} + E\{s^*(k)\mathbf{n}(k)\} \\ &= E\{s^*(k)\mathbf{a}_0 s(k)\} + 0 + 0 \\ &= E\{|s(k)|^2\}\mathbf{a}_0 \\ &= S\mathbf{a}_0\end{aligned} \tag{3.25}$$

where because the array manifold vector $\mathbf{a}_0$ is a deterministic quantity, it can be pulled out of the expectation operator. $E\{|s(k)|^2\} = S$ is the signal power, which for the purposes of identifying the weight vector $\mathbf{w}$ can be taken as an arbitrary constant μ. This results in

$$\mathbf{w}_{\text{opt}} = \mu\mathbf{R}_{xx}^{-1}\mathbf{a}_0 \ , \tag{3.26}$$

which has the same form as the Applebaum solution [53] given by

$$\mathbf{w}_{\text{opt}} = \mu\mathbf{R}_{nn}^{-1}\mathbf{a}_0 \ . \tag{3.27}$$

As will be shown in Section 3.3.4, the Applebaum solution maximizes the Signal to Noise Ratio (SNR) or the SINR depending on the which covariance matrix is available.

3.3.2 Least Mean Squares Algorithm

The least mean squares (LMS) algorithm seeks to minimize the MSE between the desired signal and the filtered signal by iteratively manipulating the corresponding filter weights. A block diagram of an LMS adaptive array is given in Figure 3.4. As described in Section 9.2 of Hayes [26], the LMS algorithm is an approximation of the steepest descent adaptive filter, which minimizes the MSE by taking small iterative steps along the error function in the direction of steepest descent.

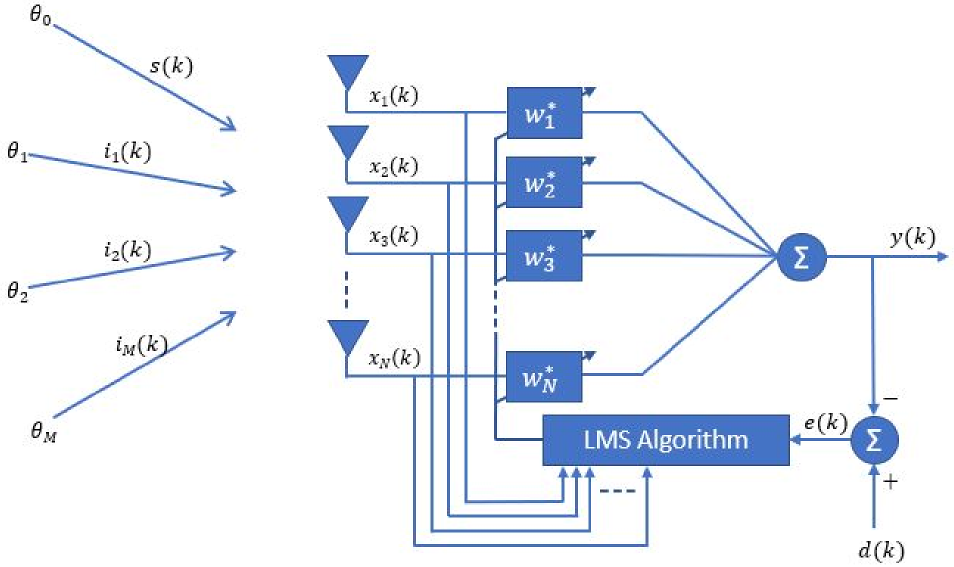

Figure 3.4: LMS Adaptive Array

3.3.2.1 Method of Steepest Descent

Referring back to Eq. (3.24), note that the MMSE algorithm requires an inversion of the covariance matrix, an operation which is known to have $\mathcal{O}(n^3)$ complexity [54]. The method of steepest descent iterates to the ideal weight vector in computing the next weight vector by taking a small step μ along the error surface in the negative gradient direction away from the current weight vector. Figure 3.5 shows a simulated MSE surface for two real-valued weights. Because the MSE surface is a quadratic convex function, there exists a global minimum, towards which the gradient will always point.

The steepest descent algorithm is described mathematically by

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu \nabla_{\mathbf{w}} \xi \tag{3.28}$$

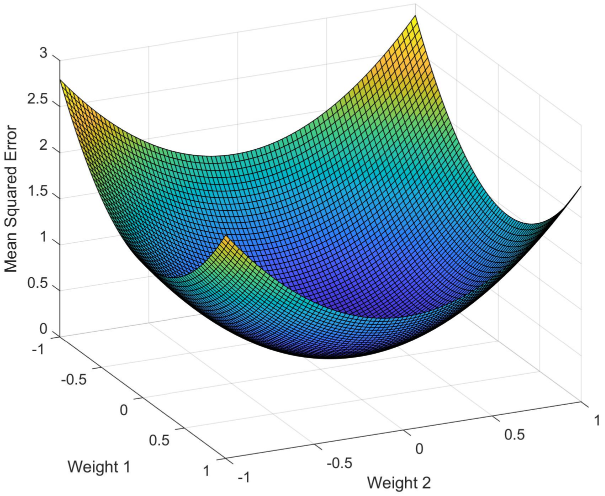

Figure 3.5: LMS MSE Surface

or

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu \nabla_{\mathbf{w}^H} \xi \ . \tag{3.29}$$

Rewriting Eqs. (3.22) and (3.23) as

$$\begin{aligned} \nabla_{\mathbf{w}} \xi &= \mathbf{r}_{xd^*} - \mathbf{R}_{xx}\mathbf{w}(k) \\ &= E\{d^*\mathbf{x}(k) - \mathbf{x}(k)\mathbf{x}^H(k)\mathbf{w}(k)\} \\ &= E\{(d^* - \mathbf{x}^H(k)\mathbf{w}(k))\mathbf{x}(k)\} \\ &= E\{e^*\mathbf{x}(k)\} \ , \end{aligned} \tag{3.30}$$

and

$$\begin{aligned} \nabla_{\mathbf{w}^H} \xi &= \mathbf{r}^*_{xd^*} - (\mathbf{R}_{xx}\mathbf{w}(k))^* \\ &= E\{e\mathbf{x}^*(k)\} \ , \end{aligned} \tag{3.31}$$

and substituting respectively into Eqs. (3.28) and (3.29) yields

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu E\{e^*\mathbf{x}\} \tag{3.32}$$

and

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu E\{e\mathbf{x}^*\} \ . \tag{3.33}$$

Both Eqs. (3.32) and (3.33) converge to the ideal weight vector $\mathbf{w} = \mathbf{R}_{xx}^{-1}\mathbf{r}_{xd^*}$, albeit along different paths.

3.3.2.2 LMS Approximation

The LMS algorithm approximates the expectations in Eqs. (3.32) and (3.33) using current signal vector sample $\mathbf{x}(k)$ and weight vector $\mathbf{w}(k)$, as given by

$$E\{e^*\mathbf{x}\} \approx \left(d^*(k) - \mathbf{x}^H(k)\mathbf{w}(k)\right)\mathbf{x}(k) \tag{3.34}$$

and

$$E\{e\mathbf{x}^*\} \approx \left(d(k) - \mathbf{w}^H(k)\mathbf{x}(k)\right)\mathbf{x}^*(k) \ , \tag{3.35}$$

respectively. Thus, the LMS algorithm can be implemented through either of the following equations.

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu e^*(k)\mathbf{x}(k) \tag{3.36}$$

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{x}^*(k) \tag{3.37}$$

Note that $e(k) = d(k) - \mathbf{w}_k^H\mathbf{x}(k)$ is known as the *a posteriori estimation error* [39]

3.3.3 Recursive Least Squares Algorithm

The Recursive Least Squares (RLS) algorithm [39] extends the LMS algorithm, seeking to minimize the squared error recursively. The cost function estimates the MSE ξ given in Eq. (3.21) using a weighted sum of the previous error samples, as given by

$$\epsilon(k) = \sum_{i=1}^{k} \lambda^{k-i}|e(i)|^2 \ . \tag{3.38}$$

λ is a positive value just below 1 (e.g. $\lambda = 0.998$), known as the forgetting factor [52]. When $\lambda = 1$, the RLS algorithm is said to be the *Ordinary Least Squares algorithm* [25] or the *Growing Window RLS algorithm* [26]. The weighting function is applied to each error sample such that the i^{th} sample is scaled by λ^{k-i}.

Recall from Section 3.3.1, the weight vector $\mathbf{w}$ which minimizes the MSE is $\mathbf{w}_{\text{opt}} = \mathbf{R}_{xx}^{-1}\mathbf{r}_{xd^*}$, as defined in Eq. (3.24). The RLS algorithm approximates the covariance matrix $\mathbf{R}_{xx}$ and the cross covariance vector $\mathbf{r}_{xd^*}$ as weighted sums of past sample-based statistical estimates, as given by

$$\hat{\mathbf{R}}_{xx}(k) = \sum_{i=1}^{k} \lambda^{k-i}\mathbf{x}(i)\mathbf{x}^H(i) \tag{3.39}$$

and

$$\hat{\mathbf{r}}_{xd^*}(k) = \sum_{i=1}^{k} \lambda^{k-i}\mathbf{x}(i)d^*(i) \tag{3.40}$$

respectively. Note that Eqs. (3.38) - (3.40) are defined as continuing summations starting with the first sample of operation. This differs from the sample covariance matrix estimate given in Section 3.2, which is computed over a block of K samples.

The current statistical estimates in Eqs. (3.39) and (3.40) can be rewritten recursively as functions of the forgetting factor and the current array sample vector $\mathbf{x}(k)$, as given by

$$\begin{aligned}\hat{\mathbf{R}}_{xx}(k) &= \sum_{i=1}^{k-1} \lambda^{k-i}\mathbf{x}(i)\mathbf{x}^H(i) + \lambda^0\mathbf{x}(k)\mathbf{x}^H(k)\\ &= \lambda\sum_{i=1}^{k-1} \lambda^{k-1-i}\mathbf{x}(i)\mathbf{x}^H(i) + \mathbf{x}(k)\mathbf{x}^H(k)\\ &= \lambda\hat{\mathbf{R}}_{xx}(k-1) + \mathbf{x}(k)\mathbf{x}^H(k)\end{aligned} \tag{3.41}$$

and

$$\hat{\mathbf{r}}_{xd^*}(k) = \lambda\hat{\mathbf{r}}_{xd^*}(k-1) + \mathbf{x}(k)d^*(k)\ . \tag{3.42}$$

Now that the estimates $\hat{\mathbf{R}}_{xx}(k)$ and $\hat{\mathbf{r}}_{xd^*}(k)$ are defined, the current weight vector can be written as $\mathbf{w}(k) = \hat{\mathbf{R}}_{xx}^{-1}(k)\hat{\mathbf{r}}_{xd^*}(k)$. However, this still requires inverting a matrix.

An alternative method is to rewrite $\hat{\mathbf{R}}_{xx}^{-1}(k)$ in terms of λ and $\mathbf{x}$. A simplified version of the Sherman Morrison-Woodbury (SMW) theorem [55] defines the inverse of a matrix of the form in Eq. (3.41) as

$$(\mathbf{A} + \mathbf{z}\mathbf{z}^H)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{z}\mathbf{z}^H\mathbf{A}^{-1}}{1 + \mathbf{z}^H\mathbf{A}^{-1}\mathbf{z}} \ , \tag{3.43}$$

assuming $1 + \mathbf{z}^H\mathbf{A}^{-1}\mathbf{z} \neq 0$. Making the appropriate substitutions, the inverse of Eq. (3.41) can be written as

$$\hat{\mathbf{R}}_{xx}^{-1}(k) = \lambda^{-1}\hat{\mathbf{R}}_{xx}^{-1}(k-1) - \frac{\lambda^{-2}\hat{\mathbf{R}}_{xx}^{-1}(k-1)\mathbf{x}(k)\mathbf{x}^H(k)\hat{\mathbf{R}}_{xx}^{-1}(k-1)}{1 + \lambda^{-1}\mathbf{x}^H(k)\hat{\mathbf{R}}_{xx}^{-1}(k-1)\mathbf{x}(k)} \ . \tag{3.44}$$

As shown in [25], [39], and [37], it is common to define the gain vector $\mathbf{g}(k)$ as

$$\mathbf{g}(k) = \frac{\lambda^{-1}\hat{\mathbf{R}}_{xx}^{-1}(k-1)\mathbf{x}(k)}{1 + \lambda^{-1}\mathbf{x}^H(k)\hat{\mathbf{R}}_{xx}^{-1}(k-1)\mathbf{x}(k)} \ . \tag{3.45}$$

This allows Eq. (3.44) to be rewritten as

$$\hat{\mathbf{R}}_{xx}^{-1}(k) = \lambda^{-1}\hat{\mathbf{R}}_{xx}^{-1}(k-1) - \lambda^{-1}\mathbf{g}(k)\mathbf{x}^H(k)\hat{\mathbf{R}}_{xx}^{-1}(k-1) \ . \tag{3.46}$$

At this point, the current weight vector $\mathbf{w}$ can be written as

$$\begin{aligned} \mathbf{w}(k) &= \hat{\mathbf{R}}_{xx}^{-1}(k)\hat{\mathbf{r}}_{xd^*}(k) \\ &= \lambda\hat{\mathbf{R}}_{xx}^{-1}(k)\hat{\mathbf{r}}_{xd^*}(k-1) + \hat{\mathbf{R}}_{xx}^{-1}(k)\mathbf{x}(k)d^*(k) \ , \end{aligned} \tag{3.47}$$

where $\hat{\mathbf{r}}_{xd^*}(k)$ is defined in Eq. (3.42). Substituting Eq. (3.46) into the first term of Eq. (3.47) yields

$$\begin{aligned} \mathbf{w}(k) &= \hat{\mathbf{R}}_{xx}^{-1}(k-1)\hat{\mathbf{r}}_{xd^*}(k-1) - \mathbf{g}(k)\mathbf{x}^H(k)\hat{\mathbf{R}}_{xx}^{-1}(k-1)\hat{\mathbf{r}}_{xd^*}(k-1) \\ &\quad + \hat{\mathbf{R}}_{xx}^{-1}(k)\mathbf{x}(k)d^*(k) \\ &= \mathbf{w}(k-1) - \mathbf{g}(k)\mathbf{x}^H(k)\mathbf{w}(k-1) + \hat{\mathbf{R}}_{xx}^{-1}(k)\mathbf{x}(k)d^*(k) \ . \end{aligned} \tag{3.48}$$

To further simplify Eq. (3.48), Eq. (3.45) is rewritten as

$$\begin{aligned} \mathbf{g}(k) &= \left[\lambda^{-1}\hat{\mathbf{R}}_{xx}^{-1}(k-1)\mathbf{x}(k) - \lambda^{-1}\mathbf{g}(k)\mathbf{x}^H(k)\hat{\mathbf{R}}_{xx}^{-1}(k-1)\right]\mathbf{x}(k) \\ &= \hat{\mathbf{R}}_{xx}^{-1}(k)\mathbf{x}(k) \ . \end{aligned} \tag{3.49}$$

Substituting Eq. (3.49) into Eq. (3.48) yields the final weight vector relation,

$$\begin{aligned} \mathbf{w}(k) &= \mathbf{w}(k-1) - \mathbf{g}(k)\mathbf{x}^H(k)\mathbf{w}(k-1) + \mathbf{g}(k)d^*(k) \\ &= \mathbf{w}(k-1) + \mathbf{g}(k)(d^*(k) - \mathbf{x}^H(k)\mathbf{w}(k-1)) \\ &= \mathbf{w}(k-1) + \mathbf{g}(k)e^*(k) \ , \end{aligned} \tag{3.50}$$

where $e(k) = d(k) - \mathbf{w}^H(k-1)\mathbf{x}(k)$ is known as the *a priori estimation error* [39]. Note the subtle difference from the *a posteriori estimation error* which is used in the LMS algorithm.

Table 3.1 summarizes the RLS equations, where $\mathbf{P} = \hat{\mathbf{R}}_{xx}^{-1}$. Because the weight vector $\mathbf{w}$ and covariance matrix inverse $\mathbf{P}$ are recursively updated, initialization values are required. Hayes [26] notes that initial estimates of the covariance matrix and cross covariance vector can be computed using Eqs. (3.39) and (3.40) and used to determine values for $\mathbf{w}$ and $\mathbf{P}$. Alternatively, $\mathbf{w}$ can simply be initialized to a vector of zeros and $\mathbf{P}$ initialized to $\delta^{-1}\mathbf{I}$ where δ is a small positive constant.

Table 3.1: RLS Equations

Description	Equation
Error Sample	$e(k) = d(k) - \mathbf{w}^H(k-1)\mathbf{x}(k)$
Gain Vector	$\mathbf{g}(k) = \frac{\lambda^{-1}\mathbf{P}(k-1)\mathbf{x}(k)}{1+\lambda^{-1}\mathbf{x}^H(k)\mathbf{P}(k-1)\mathbf{x}(k)}$
Weight Vector	$\mathbf{w}(k) = \mathbf{w}(k-1) + \mathbf{g}(k)e^*(k)$
Covariance Inverse	$\mathbf{P}(k) = \lambda^{-1}\mathbf{P}(k-1) - \lambda^{-1}\mathbf{g}(k)\mathbf{x}^H(k)\mathbf{P}(k-1)$

3.3.4 Maximum Signal to Interference/Noise Ratio

Another method for determining the optimum weight vector $\mathbf{w}_{\text{opt}}$ is to maximize the SINR. This is the basis for the Applebaum beamformer [53]. One must first describe the desired signal power σ_s^2 and the undesired signal power σ_u^2. The signal power is defined as

$$\begin{aligned}\sigma_s^2 &= E\left[|\mathbf{w}^H\mathbf{x}_s(k)|^2\right]\\ &= E\left[(\mathbf{w}^H\mathbf{x}_s(k))(\mathbf{w}^H\mathbf{x}_s(k))^H\right]\\ &= E\left[\mathbf{w}^H\mathbf{x}_s(k)\mathbf{x}_s(k)^H\mathbf{w}\right]\\ &= \mathbf{w}^H E\left[\mathbf{x}_s(k)\mathbf{x}_s(k)^H\right]\mathbf{w}\\ &= \mathbf{w}^H\mathbf{R}_{ss}\mathbf{w}\ .\end{aligned} \tag{3.51}$$

The interference power is given by

$$\begin{aligned}\sigma_i^2 &= E\left[|\mathbf{w}^H\mathbf{x}_i(k)|^2\right]\\ &= \mathbf{w}^H\mathbf{R}_{ii}\mathbf{w}\end{aligned} \tag{3.52}$$

and the noise power is given by

$$\begin{aligned}\sigma_n^2 &= E\left[|\mathbf{w}^H\mathbf{n}(k)|^2\right]\\ &= \mathbf{w}^H\mathbf{R}_{nn}\mathbf{w}\ .\end{aligned} \tag{3.53}$$

As given in Eq. (3.3), the undesired signal is composed of the interference signal and noise. Therefore, the undesired signal power is defined as

$$\begin{aligned}\sigma_u^2 &= \sigma_i^2 + \sigma_n^2 \\ &= \mathbf{w}^H \mathbf{R}_{ii} \mathbf{w} + \mathbf{w}^H \mathbf{R}_{nn} \mathbf{w} \\ &= \mathbf{w}^H \mathbf{R}_{uu} \mathbf{w} \ .\end{aligned} \tag{3.54}$$

This allows the SINR to be given by

$$\text{SINR} = \frac{\sigma_s^2}{\sigma_u^2} = \frac{\mathbf{w}^H \mathbf{R}_{ss} \mathbf{w}}{\mathbf{w}^H \mathbf{R}_{uu} \mathbf{w}} \ . \tag{3.55}$$

The maximum SINR can be determined by setting the derivative of Eq. (3.55) with respect to $\mathbf{w}^H$ equal to zero, as outlined in Chapter 10 of [56]. The result of optimization is given by

$$\begin{aligned}\mathbf{R}_{ss} \mathbf{w} &= \text{SINR} \cdot \mathbf{R}_{uu} \mathbf{w} \\ \mathbf{R}_{uu}^{-1} \mathbf{R}_{ss} \mathbf{w} &= \text{SINR} \cdot \mathbf{w} \ .\end{aligned} \tag{3.56}$$

As described by [25], Eq. (3.56) is an eigenvector equation where the eigenvectors and eigenvalues describe weight vectors $\mathbf{w}$ and corresponding SINR's. Therefore, the maximum SINR can be achieved by selecting as the weight vector $\mathbf{w}_{\text{opt}}$ the eigenvector corresponding to the maximum eigenvalue of the Hermitian matrix $\mathbf{R}_{uu}^{-1} \mathbf{R}_{ss}$.

Knowing the optimum weight vector $\mathbf{w}_{\text{opt}}$, Eq. (3.56) can be rewritten in the form of Eq. (3.26) and (3.27) such that

$$\mathbf{w}_{\text{opt}} = \frac{1}{\text{SINR}} \mathbf{R}_{uu}^{-1} \mathbf{R}_{ss} \mathbf{w} \ . \tag{3.57}$$

Given that

$$\begin{aligned}\mathbf{R}_{ss} &= E\{\mathbf{x}_s\mathbf{x}_s^H\} \\ &= E\{(s(k)\mathbf{a}_0)(s^*(k)\mathbf{a}_0^H)\} \\ &= E\{|s(k)|^2\}\mathbf{a}_0\mathbf{a}_0^H \\ &= S\mathbf{a}_0\mathbf{a}_0^H \ ,\end{aligned} \tag{3.58}$$

Eq. (3.57) can be modified as follows

$$\begin{aligned}\mathbf{w}_{\text{opt}} &= \frac{S}{\text{SINR}}\mathbf{R}_{uu}^{-1}\mathbf{a}_0\mathbf{a}_0^H\mathbf{w} \\ &= \frac{S\mathbf{a}_0^H\mathbf{w}_{\text{opt}}}{\text{SINR}}\mathbf{R}_{uu}^{-1}\mathbf{a}_0 \\ &= \mu\mathbf{R}_{uu}^{-1}\mathbf{a}_0 \ ,\end{aligned} \tag{3.59}$$

which is the Applebaum solution [53] for maximizing SINR. If the undesired signals do not contain spatially correlated interferers (e.g. if $\mathbf{x}_i = 0$), then $\mathbf{R}_{uu} = \mathbf{R}_{nn}$ and maximizing the SINR is equivalent to maximizing the SNR. Unless otherwise noted, the Applebaum algorithm will refer to Eq. (3.59).

3.3.5 Maximum Likelihood

The Maximum Likelihood optimization method assumes that the noise environment consists of zero-mean Gaussian noise. The goal is to determine the optimal weight vector $\mathbf{w}$ from the probability density function (PDF) describing the likelihood that the observed array vector $\mathbf{x}$ occurred given that the desired signal vector $\mathbf{x}_s$ occurred. Given that the noise vector $\mathbf{n}$ is a stationary zero-mean Gaussian random vector with covariance matrix $\mathbf{R}_{nn} = \sigma_n^2\mathbf{I}$ and the array vector $\mathbf{x}$ is a stationary Gaussian random vector with mean $\mathbf{x}_s = s\mathbf{a}_0$ such that $\mathbf{x} = \mathbf{x}_s + \mathbf{n}$, the joint Gaussian conditional PDF is given by

$$\begin{aligned} p(\mathbf{x}|\mathbf{x}_s) &= \frac{1}{\sqrt{2\pi\sigma_n^2}} e^{\left(-(\mathbf{x}-\mathbf{x}_s)^H \mathbf{R}_{nn}^{-1}(\mathbf{x}-\mathbf{x}_s)\right)} \\ &= \frac{1}{\sqrt{2\pi\sigma_n^2}} e^{\left(-(\mathbf{x}-\mathbf{a}_0 s)^H \mathbf{R}_{nn}^{-1}(\mathbf{x}-\mathbf{a}_0 s)\right)} \ . \end{aligned} \tag{3.60}$$

The intent is to maximize Eq. (3.60), which is accomplished by maximizing the argument within the exponent. The log-likelihood function solves for this argument as given by

$$\begin{aligned} L[\mathbf{x}_s] &= -\ln[p(\mathbf{x}|\mathbf{x}_s)] \\ &= c + (\mathbf{x} - \mathbf{a}_0 s)^H \mathbf{R}_{nn}^{-1}(\mathbf{x} - \mathbf{a}_0 s) \ , \end{aligned} \tag{3.61}$$

where $c = -\ln\left(\frac{1}{\sqrt{2\pi\sigma_n^2}}\right)$, which is independent of s and $\mathbf{x}$. There exists a signal estimate $\hat{s}$ that maximizes Eq. (3.61) which can be determined by setting the derivative with respect to s^* equal to 0.

$$\begin{aligned} \frac{\partial L}{\partial s^*} &= \frac{\partial}{\partial s^*}\Big(c(\mathbf{x} - \mathbf{a}_0 s)^H \mathbf{R}_{nn}^{-1}(\mathbf{x} - \mathbf{a}_0 s)\Big) \\ &= \frac{\partial}{\partial s^*}\Big(\mathbf{x}^H \mathbf{R}_{nn}^{-1}\mathbf{x} - s^*\mathbf{a}_0^H \mathbf{R}_{nn}^{-1}\mathbf{x} - s\mathbf{x}^H \mathbf{R}_{nn}^{-1}\mathbf{a}_0 + (ss^*)\mathbf{a}_0^H \mathbf{R}_{nn}^{-1}\mathbf{a}_0\Big) \\ &= 0 - \frac{\partial}{\partial s^*}\Big(s^*\mathbf{a}_0^H \mathbf{R}_{nn}^{-1}\mathbf{x}\Big) - \frac{\partial}{\partial s^*}\Big(s\mathbf{x}^H \mathbf{R}_{nn}^{-1}\mathbf{a}_0\Big) + \frac{\partial}{\partial s^*}\Big((ss^*)\mathbf{a}_0^H \mathbf{R}_{nn}^{-1}\mathbf{a}_0\Big) \\ &= -\mathbf{a}_0^H \mathbf{R}_{nn}^{-1}\mathbf{x} - \left(\frac{\partial}{\partial s}\Big(s\mathbf{x}^H \mathbf{R}_{nn}^{-1}\mathbf{a}_0\Big)\right)^* + \hat{s}\mathbf{a}_0^H \mathbf{R}_{nn}^{-1}\mathbf{a}_0 \\ &= -\mathbf{a}_0^H \mathbf{R}_{nn}^{-1}\mathbf{x} - \Big(\mathbf{x}^H \mathbf{R}_{nn}^{-1}\mathbf{a}_0\Big)^* + \hat{s}\mathbf{a}_0^H \mathbf{R}_{nn}^{-1}\mathbf{a}_0 \\ &= -\mathbf{a}_0^H \mathbf{R}_{nn}^{-1}\mathbf{x} - \mathbf{a}_0^H \mathbf{R}_{nn}^{-1}\mathbf{x} + \hat{s}\mathbf{a}_0^H \mathbf{R}_{nn}^{-1}\mathbf{a}_0 \\ &= -2\mathbf{a}_0^H \mathbf{R}_{nn}^{-1}\mathbf{x} + 2\hat{s}\mathbf{a}_0^H \mathbf{R}_{nn}^{-1}\mathbf{a}_0 \end{aligned} \tag{3.62}$$

Note that Eq. (5) of Brandwood [21] states that $\frac{\partial}{\partial z} f(z, z^*) = \left(\frac{\partial}{\partial z^*} f(z, z^*)\right)^*$. Setting Eq. (3.62) equal to 0 and solving for the signal estimate $\hat{s}$ yields

$$\hat{s} = \frac{\mathbf{a}_0^H \mathbf{R}_{nn}^{-1}}{\mathbf{a}_0^H \mathbf{R}_{nn}^{-1} \mathbf{a}_0} \mathbf{x} \tag{3.63}$$

Recalling from Eq. (3.1) that the array output is given by $y = \mathbf{w}^H \mathbf{x}$, the weight vector which maximizes the likelihood that the array output y corresponds to the signal estimate $\hat{s}$ is given by

$$\mathbf{w}_{ML} = \frac{\mathbf{R}_{nn}^{-1} \mathbf{a}_0}{\mathbf{a}_0^H \mathbf{R}_{nn}^{-1} \mathbf{a}_0} \tag{3.64}$$

3.3.6 Minimum Variance Distortionless Response

The minimum variance distortionless response (MVDR), also known as the Capon beamformer [57], seeks minimize the signal variance while requiring a distortionless signal response in a prescribed direction. The array manifold vector corresponding to the prescribed direction θ_0 is $\mathbf{a}_0$ as described by Eq. (3.5) and the distortionless constraint, as given in Eq. (2.25), is $\mathbf{w}^H \mathbf{a}_0 = 1$. Rewriting the array output described in Eq (3.1) under the distortionless constraint yields

$$y = s + \mathbf{w}^H \mathbf{u} \ . \tag{3.65}$$

Given that $E\{y\} = s$ and s is uncorrelated with the undesired signals and noise, the signal variance is given by

$$\begin{aligned} \sigma_y^2 &= E\{y^2\} - E\{y\}^2 \\ &= E\{(s + \mathbf{w}^H \mathbf{u})^* (s + \mathbf{w}^H \mathbf{u})\} - |s|^2 \\ &= E\{|s|^2\} + E\{s^* \mathbf{w}^H \mathbf{u}\} + E\{s \mathbf{u}^H \mathbf{w}\} + E\{\mathbf{w}^H \mathbf{u}\mathbf{u}^H \mathbf{w}\} - |s|^2 \\ &= |s|^2 + 0 + 0 + E\{\mathbf{w}^H \mathbf{u}\mathbf{u}^H \mathbf{w}\} - |s|^2 \\ &= \mathbf{w}^H \mathbf{R}_{uu} \mathbf{w} \\ &= \sigma_u^2 \ . \end{aligned} \tag{3.66}$$

The method of Lagrange [58] can be employed to determine a weight vector $\mathbf{w}$ which minimizes the signal variance under the distortionless constraint. First, define the following cost function

$$\begin{aligned} J(\mathbf{w}) &= \sigma_u^2 + \lambda(1 - \mathbf{w}^H \mathbf{a}_0) \\ &= \mathbf{w}^H \mathbf{R}_{uu} \mathbf{w} + \lambda(1 - \mathbf{w}^H \mathbf{a}_0) \ , \end{aligned} \tag{3.67}$$

where λ is a Lagrange multiplier. Setting the cost function gradient with respect to the weight vector $\mathbf{w}^H$ equal to 0 yields

$$\nabla_{\mathbf{w}^H} J(\mathbf{w}) = \mathbf{R}_{uu} \mathbf{w}_{\text{mvdr}} - \lambda \mathbf{a}_0 = 0 \ , \tag{3.68}$$

which when solved for the weight vector results in

$$\mathbf{w}_{\text{mvdr}} = \lambda \mathbf{R}_{uu}^{-1} \mathbf{a}_0 \ . \tag{3.69}$$

To determine the Lagrange multiplier λ, recall the distortionless constraint given in Eq. (2.25). Solving for the weight vector $\mathbf{w}$ yields

$$\mathbf{w} = \frac{1}{\mathbf{a}_0^H} \ . \tag{3.70}$$

Substituting Eq. (3.70) into Eq. (3.69) and solving for λ yields

$$\lambda = \frac{1}{\mathbf{a}_0^H \mathbf{R}_{uu}^{-1} \mathbf{a}_0} \ . \tag{3.71}$$

Finally, Eq. (3.71) is substituted back into Eq. (3.69) to give the MVDR beamforming weight equation,

$$\mathbf{w}_{\text{mvdr}} = \frac{\mathbf{R}_{uu}^{-1} \mathbf{a}_0}{\mathbf{a}_0^H \mathbf{R}_{uu}^{-1} \mathbf{a}_0} \ . \tag{3.72}$$

Eq (3.72) is predicated on the assumption that noise statistics are known or can be estimated in the absence of the signal of interest. If the noise environment is not available without the signal of interest, than $\mathbf{R}_{uu}$ must be replaced with $\mathbf{R}_{xx}$. Van Trees [44] distinguishes the latter case from the former, calling it the Minimum Power Distortionless Response (MPDR). Hence, the MPDR weight vector is given by

$$\mathbf{w}_{\text{mpdr}} = \frac{\mathbf{R}_{xx}^{-1} \mathbf{a}_0}{\mathbf{a}_0^H \mathbf{R}_{x}^{-1} \mathbf{a}_0} \ . \tag{3.73}$$

He specifies that the MVDR and MPDR beamformers are often interchanged in the literature. Simulation results in Section 3.4.2 present the MPDR beamformer.

3.3.7 Linearly Constrained Minimum Variance

The Linearly Constrained Minimum Variance (LCMV) beamformer expands on the MVDR beamformer, allowing for additional constraints on top of the distortionless constraint given in Eq. (2.25). Van Trees [44] outlines several types of constraints including directional constraints, derivative constraints, and eigenvector constraints.

Directional constraints have been explored in Sections 2.4 and 3.3.6 and are utilized in the simulation results in Section 3.4.2.

Distortionless constraints and null constraints are both examples of directional constraints, which constrains the array factor response to 1 or 0 for a given direction. The LCMV constraint equations are given by

$$\mathbf{w}^H\mathbf{C} = \mathbf{g}^H \ , \tag{3.74}$$

where

$$\mathbf{C} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \cdots & \mathbf{a}_M \end{bmatrix} \tag{3.75}$$

and

$$\mathbf{g}^H = \begin{bmatrix} g_1^* & g_2^* & \cdots & g_M^* \end{bmatrix} \ . \tag{3.76}$$

This matches the constraints in Eq. (2.27). The vectors in $\mathbf{C}$ are the array manifold vectors $\mathbf{a}_n$ corresponding to the given constraint direction θ_n. The elements of $\mathbf{g}$ are the corresponding desired array factor responses. For the purposes of this derivation, it is assumed that the first constraint is distortionless.

As with the MVDR beamformer, the LCMV beamformer seeks to minimize the signal variance σ_u^2 under the constraints given in Eq. (3.74). Following the sequence outlined in Section 3.3.6, a cost function containing the noise power and given constraints can be defined as

$$\begin{aligned} J(\mathbf{w}) &= \sigma_u^2 + (\mathbf{g}^H - \mathbf{w}^H\mathbf{C})\boldsymbol{\lambda} \\ &= \mathbf{w}^H\mathbf{R}_{uu}\mathbf{w} + (\mathbf{g}^H - \mathbf{w}^H\mathbf{C})\boldsymbol{\lambda} \ , \end{aligned} \tag{3.77}$$

where $\boldsymbol{\lambda}$ is now a vector of Lagrange multipliers. Again, the optimum weight vector can be found by setting the cost function gradient with respect to the weight vector $\mathbf{w}^H$ equal to 0, as given by

$$\nabla_{\mathbf{w}^H} J(\mathbf{w}) = \mathbf{R}_{uu}\mathbf{w}_{\text{lcmv}} - \mathbf{C}\boldsymbol{\lambda} = 0 \; . \tag{3.78}$$

Solving for the weight vector yields

$$\mathbf{w}_{\text{lcmv}} = \mathbf{R}_{uu}^{-1}\mathbf{C}\boldsymbol{\lambda} \; . \tag{3.79}$$

To determine the Lagrange multiplier vector $\boldsymbol{\lambda}$, first solve the constraint equations given in Eq. (3.74) for $\mathbf{w}$ as given by

$$\mathbf{w} = (\mathbf{C}^H)^{-1}\mathbf{g} \; . \tag{3.80}$$

Then substitute $\mathbf{w}$ into Eq. (3.79) to give

$$\boldsymbol{\lambda} = (\mathbf{C}^H\mathbf{R}_{uu}^{-1}\mathbf{C})^{-1}\mathbf{g} \; . \tag{3.81}$$

Lastly, substitute Eq. (3.81) into Eq. (3.79) to give the LCMV beamforming weight equation,

$$\mathbf{w}_{\text{lcmv}} = \mathbf{R}_{uu}^{-1}\mathbf{C}(\mathbf{C}^H\mathbf{R}_{uu}^{-1}\mathbf{C})^{-1}\mathbf{g} \; . \tag{3.82}$$

As in Section 3.3.6, Van Trees [44] denotes the case when $\mathbf{R}_{uu}$ is to be replaced with $\mathbf{R}_{xx}$ as the Linearly Constrained Minimum Power (LCMP) beamformer, which is given by

$$\mathbf{w}_{\text{lcmp}} = \mathbf{R}_{xx}^{-1}\mathbf{C}(\mathbf{C}^H\mathbf{R}_{xx}^{-1}\mathbf{C})^{-1}\mathbf{g} \; . \tag{3.83}$$

Simulation results in Section 3.4.2 present the LCMP beamformer for which the first constraint is distortionless and the remaining constraints assign nulls.

The LCMV beamformer simplifies to the MVDR beamformer when $\mathbf{g} = 1$ and $\mathbf{C} = \mathbf{a}_0$, as given by

$$\begin{aligned} \mathbf{w}_{\text{mvdr}} &= \mathbf{R}_{uu}^{-1}\mathbf{C}(\mathbf{C}^H\mathbf{R}_{uu}^{-1}\mathbf{C})^{-1}\mathbf{g} \\ &= \mathbf{R}_{uu}^{-1}\mathbf{a}_0(\mathbf{a}_0^H\mathbf{R}_{uu}^{-1}\mathbf{a}_0)^{-1} \\ &= \frac{\mathbf{R}_{uu}^{-1}\mathbf{a}_0}{\mathbf{a}_0^H\mathbf{R}_{uu}^{-1}\mathbf{a}_0} \ . \end{aligned} \tag{3.84}$$

3.4 Algorithm Comparison

A summary of the governing equations for adaptive algorithms is given in Table 3.2. Because several of the algorithms derived in the previous section are identical to each other in form, only one of a given form is listed in the table. Algorithms of the form $\mathbf{w} = \mu\mathbf{R}_{uu}^{-1}\mathbf{a}_0$ include the MMSE from Section 3.3.1 and Maximum SINR from Section 3.3.4. Algorithms of the form $\mathbf{w} = \frac{\mathbf{R}_{xx}^{-1}\mathbf{a}_0}{\mathbf{a}_0^H\mathbf{R}_{xx}^{-1}\mathbf{a}_0}$ include the ML from Section 3.3.5 and MVDR/MPDR from Section 3.3.6. The LMCV and LMCP algorithms are of the form $\mathbf{w} = \mathbf{R}_{xx}^{-1}\mathbf{C}(\mathbf{C}^H\mathbf{R}_{xx}^{-1}\mathbf{C})^{-1}\mathbf{g}$. Algorithms of a given form differ in implementation solely on the definition of the covariance matrix.

The first primary distinction between the equations given in Table 3.2 is with regards to iterative versus statistically optimum algorithms. The LMS and RLS algorithms both iterate to an optimum solution based on learning parameters (μ, λ), the current sample of the array vector $\mathbf{x}(k)$, and an estimate of the desired sample $d(k)$. Their derivations were developed in an effort to avoid the costly inversion of the covariance matrix $\mathbf{R}$. Rather than requiring a look direction, the iterative algorithms require an estimate of the signal of interest.

The remaining algorithms are considered statistically optimum and, despite a varying number and type of constraints, are of the *Wiener Filter* form. As described in Section 3.3.1, assuming that the signal of interest is uncorrelated with the undesired noise and interferers, the cross covariance vector reduces to the array manifold vector

Table 3.2: Adaptive Algorithm Equations

Algorithm	Description	Equation
LMS	Error Sample	$e(k) = d(k) - \mathbf{w}^H(k)\mathbf{x}(k)$
	Weight Vector	$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{x}^*(k)$
RLS	Error Sample	$e(k) = d(k) - \mathbf{w}^H(k-1)\mathbf{x}(k)$
	Gain Vector	$\mathbf{g}(k) = \frac{\lambda^{-1}\mathbf{P}(k-1)\mathbf{x}(k)}{1+\lambda^{-1}\mathbf{x}^H(k)\mathbf{P}(k-1)\mathbf{x}(k)}$
	Weight Vector	$\mathbf{w}(k) = \mathbf{w}(k-1) + \mathbf{g}(k)e^*(k)$
	Covariance Inverse	$\mathbf{P}(k) = \lambda^{-1}\mathbf{P}(k-1) - \lambda^{-1}\mathbf{g}(k)\mathbf{x}^H(k)\mathbf{P}(k-1)$
Applebaum	Weight Vector	$\mathbf{w} = \mu\mathbf{R}_{uu}^{-1}\mathbf{a}_0$
MPDR	Weight Vector	$\mathbf{w} = \frac{\mathbf{R}_{xx}^{-1}\mathbf{a}_0}{\mathbf{a}_0^H\mathbf{R}_{xx}^{-1}\mathbf{a}_0}$
LCMP	Weight Vector	$\mathbf{w} = \mathbf{R}_{xx}^{-1}\mathbf{C}(\mathbf{C}^H\mathbf{R}_{xx}^{-1}\mathbf{C})^{-1}\mathbf{g}$

times a scalar, as given in Eq. (3.25). In this case, the *Wiener Filter* given in Eq. (3.24) reduces to the Applebaum form. Each statistically optimum algorithm requires the covariance matrix inverse and the desired steering direction, but differ in the definition of the scalar and whether additional constraints are required.

The aforementioned statistically optimum adaptive algorithms were developed with the intention of detecting point targets [59] and rely on strong spatial correlation between receive signals. Coherent signals from different angles of arrival (AOA) will also degrade performance. A quadratic weight vector constraint can be levied on the LCMV beamformer to prevent signal cancellation in the presence of a coherent interferer [60]. Algorithm performance is also sensitive to errors in the look direction. The Robust Capon Beamformer [61] seeks to address this sensitivity by allowing the look direction of the distortionless constraint to reside within an error ellipsoid.

Many applications, such as weather radar, rely on distributed targets rather than point targets. Yoshikawa [59] presents an iterative MMSE approach which provides reasonable estimation of radar variables. Nai [57] discusses an adaptive beamspace processing algorithm, which employs an adaptive algorithm behind a set of deterministic beams. It is operationally similar to the Capon beamformer, but translated from element space to beamspace. Additionally, clutter returns for airborne radar degrade sensitivity often masking weak targets in the presence of strong scatterers. The LCMV algorithm with appropriately selected null locations can provide attenuation of clutter returns. The sparse LCMV beamformer given in [62] provides a method for determining the appropriate constraints to suppress ground clutter based on minimization of the MSE while searching a limited number of degrees of freedom.

3.4.1 Computational Complexity

This section seeks to compare the computational costs of the algorithms presented in Section 3.3. For the purposes of this section, algorithmic complexity is taken to be the number of arithmetic operations required for a given algorithm to compute the weight vector, or in the case of the iterative algorithms, to reach steady state. This analysis does not account for memory requirements, although that is a common cost metric [8]. Optimization techniques that exploit matrix topology, such as QR Decomposition [13], [63], are also not accounted for in these estimates. However, assuming that a given optimization technique provides the same improvement for all algorithms to which the technique applies, the relative costs given herein should still be useful.

Table 3.3 summarizes the estimated computation cost for the various adaptive algorithm forms presented in this chapter, where N is the number of array elements, S is the number of samples in the reference waveform for the iterative algorithms, K is the number of samples used to estimate the covariance matrix for the statistically optimum algorithms, and M is the number of constraints for the LCMV and LCMP algorithms. Estimated costs are based on the assumption that multiplication of an $N \times M$ matrix with an $M \times P$ matrix has a computational complexity of $\mathcal{O}(NMP)$ [64]

Table 3.3: Adaptive Algorithm Computational Complexity

Equation Form	Complexity
$e(k) = d(k) - \mathbf{w}^H(k)\mathbf{x}(k)$ $\mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{x}^*(k)$	$\mathcal{O}\left(3NS\right)$
$e(k) = d(k) - \mathbf{w}^H(k-1)\mathbf{x}(k)$ $\mathbf{g}(k) = \frac{\lambda^{-1}\mathbf{P}(k-1)\mathbf{x}(k)}{1+\lambda^{-1}\mathbf{x}^H(k)\mathbf{P}(k-1)\mathbf{x}(k)}$ $\mathbf{w}(k) = \mathbf{w}(k-1) + \mathbf{g}(k)e^*(k)$ $\mathbf{P}(k) = \lambda^{-1}\left(\mathbf{I} - \mathbf{g}(k)\mathbf{x}^H(k)\right)\mathbf{P}(k-1)$	$\mathcal{O}\left(5N^2S + 5NS\right)$
$\mathbf{w}_{\text{opt}} = \mu\mathbf{R}^{-1}\mathbf{a}_0$	$\mathcal{O}\left(N^3 + N^2K + N^2 + N\right)$
$\mathbf{w} = \frac{\mathbf{R}^{-1}\mathbf{a}_0}{\mathbf{a}_0^H\mathbf{R}^{-1}\mathbf{a}_0}$	$\mathcal{O}\left(N^3 + N^2K + N^2 + 2N\right)$
$\mathbf{w} = \mathbf{R}^{-1}\mathbf{C}(\mathbf{C}^H\mathbf{R}^{-1}\mathbf{C})^{-1}\mathbf{g}$	$\mathcal{O}(N^3 + N^2K + N^2M + ...$ $NM^2 + M^3 + M^2 + NM)$

and inversion of an $N \times N$ matrix has a computational complexity of $\mathcal{O}(N^3)$ [54]. Costs for individual operations, summarized in Tables 3.4 and 3.5 for the iterative algorithms and the statistically optimum algorithms respectively, are combined to determine the values given in Table 3.3.

In calculating the computational cost of the iterative algorithms, it was assumed that the algorithm requires the entire waveform estimate to converge to the optimum weight vector. Therefore, operational costs given in Table 3.4 are scaled by the number of waveform samples S for the total algorithm cost reported in Table 3.3. Additionally, the cost of adding two $N\times 1$ vectors is taken to be the same as computing their inner product. For the purposes of comparing the algorithms of interest, this distinction is inconsequential as it is clear that the statistically optimum algorithms scale at a rate of N^3 while the RLS algorithm scales at N^2 and LMS at N. For very large arrays, iterative algorithms will be more practical. The LMS algorithm is the simplest, but offers the least accurate estimate of the covariance matrix. The RLS algorithm provides a compromise between cost and accuracy.

Table 3.4: Iterative Algorithm Operation Cost

Term	Description	Cost
$\mathbf{w}^H(k)\mathbf{x}(k)$ $\mathbf{w}^H(k-1)\mathbf{x}(k)$	$1 \times N$ Vector times $N \times 1$ Vector	$\mathcal{O}(N)$
$e(k)\mathbf{x}^*(k)$ $e^*(k)\mathbf{g}(k)$	Scalar times $N \times 1$ Vector	$\mathcal{O}(N)$
$\mathbf{w}(k) + \mu e(k)\mathbf{x}^*(k)$ $\mathbf{w}(k-1) + \mathbf{g}(k)e^*(k)$	$N \times 1$ Vector plus $N \times 1$ Vector	$\mathcal{O}(N)$
$\mathbf{P}(k-1)\mathbf{x}(k)$	$N \times N$ Matrix times $N \times 1$ Vector	$\mathcal{O}(N^2)$
$\lambda^{-1}\left[\mathbf{P}(k-1)\mathbf{x}(k)\right]$	Scalar times $N \times 1$ Vector	$\mathcal{O}(N)$
$\mathbf{x}^H(k)\left[\mathbf{P}(k-1)\mathbf{x}(k)\right]$	$1 \times N$ Vector times $N \times 1$ Vector	$\mathcal{O}(N)$
$\mathbf{g}(k)\mathbf{x}^H(k)$	$N \times 1$ Vector times $1 \times N$ Vector	$\mathcal{O}(N^2)$
$\mathbf{I} - \mathbf{g}(k)\mathbf{x}^H(k)$	$N \times N$ Matrix minus $N \times N$ Matrix	$\mathcal{O}(N^2)$
$\left[\mathbf{I} - \mathbf{g}(k)\mathbf{x}^H(k)\right]\mathbf{P}(k-1)$	$N \times N$ Matrix times $N \times N$ Matrix	$\mathcal{O}(N^2)$
$\lambda^{-1}\left[\left(\mathbf{I} - \mathbf{g}(k)\mathbf{x}^H(k)\right)\mathbf{P}(k-1)\right]$	Scalar times $N \times N$ Matrix	$\mathcal{O}(N^2)$

Table 3.5: Statistically Optimum Algorithm Operation Cost

Term	Description	Cost
$\hat{\mathbf{R}}(k) = \mathbf{x}(k)\mathbf{x}^H(k)$	$N \times 1$ Vector times $1 \times N$ Vector	$\mathcal{O}\left(N^2\right)$
$\hat{\mathbf{R}} = \frac{1}{K}\sum_{k=1}^{K}\hat{\mathbf{R}}(k)$	K Covariance Matrix Samples	$\mathcal{O}\left(N^2K\right)$
$\hat{\mathbf{R}}^{-1}$	Inverse of an $N \times N$ Matrix	$\mathcal{O}\left(N^3\right)$
$\hat{\mathbf{R}}^{-1}\mathbf{a}_0$	$N \times N$ Matrix times $N \times 1$ Vector	$\mathcal{O}\left(N^2\right)$
$\mu\left[\hat{\mathbf{R}}^{-1}\mathbf{a}_0\right]$	Scalar times $N \times 1$ Vector	$\mathcal{O}\left(N\right)$
$\mathbf{a}_0^H\left[\hat{\mathbf{R}}^{-1}\mathbf{a}_0\right]$	$1 \times N$ Vector times $N \times 1$ Vector	$\mathcal{O}\left(N\right)$
$\hat{\mathbf{R}}^{-1}\mathbf{C}$	$N \times N$ Matrix times $N \times M$ Matrix	$\mathcal{O}\left(N^2M\right)$
$\mathbf{C}^H\left[\hat{\mathbf{R}}^{-1}\mathbf{C}\right]$	$M \times N$ Matrix times $N \times M$ Matrix	$\mathcal{O}\left(NM^2\right)$
$\left(\mathbf{C}^H\hat{\mathbf{R}}^{-1}\mathbf{C}\right)^{-1}$	Inverse of an $M \times M$ Matrix	$\mathcal{O}\left(M^3\right)$
$\left[\left(\mathbf{C}^H\hat{\mathbf{R}}^{-1}\mathbf{C}\right)^{-1}\right]\mathbf{g}$	$M \times M$ Matrix times $M \times 1$ Vector	$\mathcal{O}\left(M^2\right)$
$\left[\hat{\mathbf{R}}^{-1}\mathbf{C}\right]\left[\left(\mathbf{C}^H\hat{\mathbf{R}}^{-1}\mathbf{C}\right)^{-1}\mathbf{g}\right]$	$N \times M$ Matrix times $M \times 1$ Vector	$\mathcal{O}\left(NM\right)$

3.4.2 Simulations

Simulation results of the reported algorithms are presented below. The aperture consists of a 16-element ULA of isotropic elements with half wavelength spacing. The simulation signal environment includes signals from various sources at various power levels in the presence of noise. Each signal is a spatially correlated vector of noise. The signal of interest is incident at $\theta = 27°$ at a SNR of 0 dB. Interferers were placed at $\theta =$ -15°, 48° and -30° at SNR's of 10.4 dB, 13.6 dB, and 0 dB respectively. These are denoted with dashed vertical black lines. Additional nulls were prescribed at $\theta =$ -38° and $\theta = 44°$ for the LCMP algorithm and are denoted with solid vertical black lines.

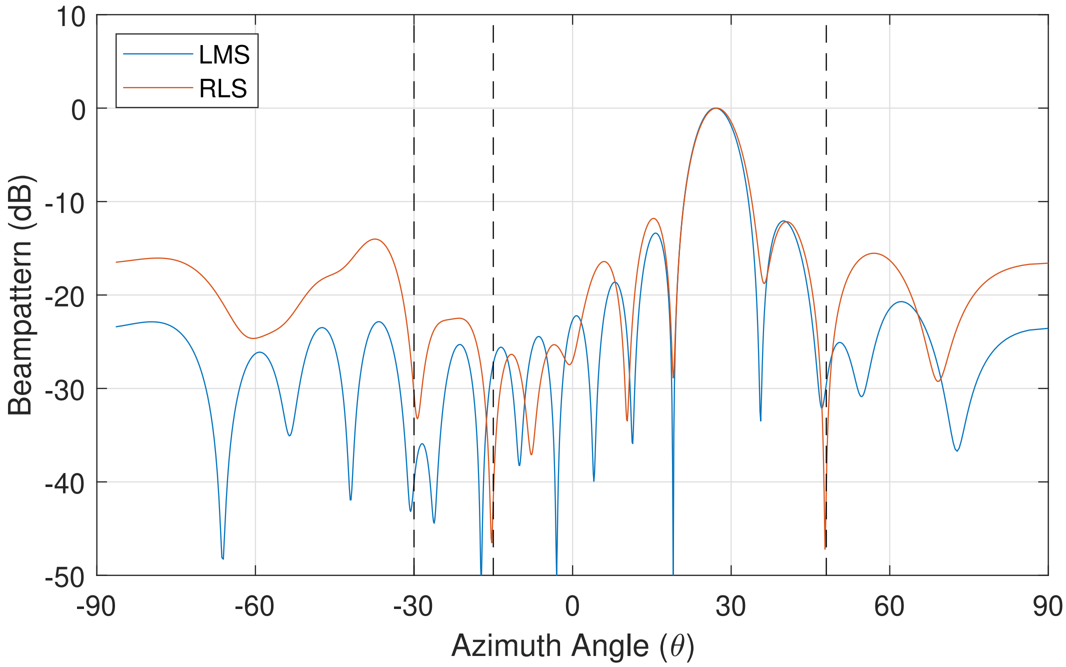

Figure 3.6: Simulated Patterns - Iterative Algorithms

The normalized patterns from the iterative algorithms are given in Figure 3.6. The RLS algorithm provides deeper nulls as compared to the LMS algorithm but results in higher sidelobes at almost everywhere else. The LMS algorithm does not appear to respond to either of the two weaker interferers at $\theta =$ -30° or $\theta =$ -15°. Both algorithms provide an accurate main beam position at $\theta = 27°$.

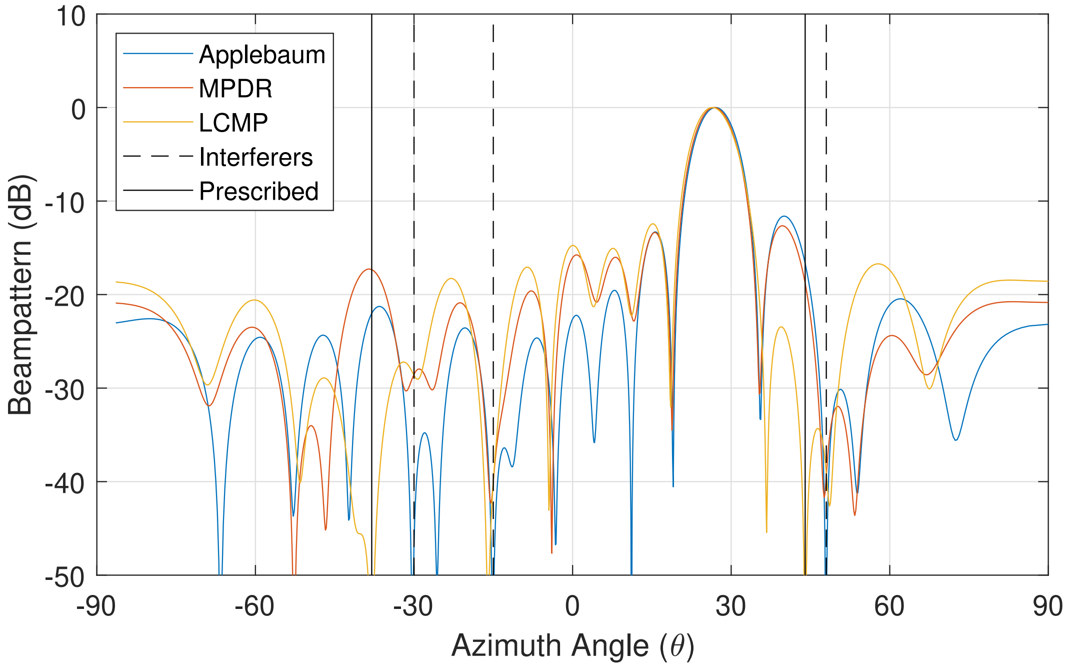

Figure 3.7: Simulated Patterns - Statistical Optimum Algorithms

Figure 3.7 show the statistically optimum patterns. The Applebaum pattern was computed using $\mathbf{R}_{uu}$, which includes the environment noise and interferers. The MPDR and LCMP patterns utilize $\mathbf{R}_{xx}$, which contain the signal of interest as well as environment noise and interferers. The Applebaum pattern effectively isolates all three interferers regardless of strength. The MPDR and LCMP patterns struggle with placing a null for the θ = -30° interferer. Additionally, the LCMP pattern shows noticeable inaccuracy in the null position for the θ = -15°. The LCMP pattern effectively implements the prescribed nulls at θ = -38° and 44°.

Chapter 4

Wideband Digital Beamforming on an RFSoC

Analog phase shifters have conventionally provided a narrowband method for collimating signals. However, with the increasing speed of ADCs, DACs, and digital processors, all-digital systems have become realistic. As discussed in [29], digitizing signals at the element level of the array enables unprecedented capabilities. The development of the RF system-on-a-chip (RFSoC), which supports multiple-GHz sample rates, has enabled the integration of fully digital direct-sampling phased array systems. This chapter provides demonstrated operation of an 8-element fully digital sub-Nyquist-sampled wideband receive array utilizing an RFSoC with and without equalization.

Current literature [18], [13], [19], [65] provides several examples demonstrating sample rates of 1–4 GHz in various system architectures. However, most incorporate direct conversion rather than direct sampling. The authors in [18] demonstrate a multi-beam digital beamformer which employs direct sampling of 1 GHz, 100 MHz bandwidth quadratic-amplitude modulation (QAM) signals using an ADC sample rate of 4 GSPS. In [13], a minimum-variance distortionless response (MVDR) beamformer is digitally implemented on an RFSoC with ADCs operating at 125 MHz. Its system architecture incorporates an RF Downconverter with 8 MHz passband. The authors in [19] demonstrate a 16-element phased array with ADCs operating at 2 GSPS which facilitates an 800 MHz bandwidth. However, at a carrier frequency of 28 GHz, narrowband beamforming was sufficient. Beamforming measurements summarized in [65] utilize the Integrated Multi-use Phased Array Common Tile (IMPACT) [20] which supports an instantaneous bandwidth of about 500 MHz. In this paper, beamforming for a tightly coupled dipole array (TCDA) is demonstrated at 3.5 GHz, 4.9 GHz, and 9.5 GHz, without mention of test signal bandwidth. The authors in [66] present narrowband measurements and discuss a wideband beamforming engine with equalization.

The demonstrated 1.6 GHz system bandwidth at a sample rate of 4 GHz presented in this dissertation utilizes the fill digital bandwidth (80% Nyquist Zone), which is not the case in most of the papers surveyed. Additionally, sub-Nyquist direct sampling of the incident waveform was not incorporated in any of the surveyed architectures, which can potentially mitigate the need for external downconversion hardware. Lastly, although wideband digital equalization was demonstrated theoretically [66], experimental measured results were lacking. This dissertation contributes to the engineering community wideband digital beamforming demonstrations with equalization using an RFSoC.

Uncalibrated and calibrated results are presented. Section 4.1 introduces the RFSoC testbed and provides simulated and measured results. An overview of the fractional-sample delay filter bank and its embedded system implementation are provided. Hardware, firmware, and software details of the research testbed are provided as well. Section 4.2 presents derivations for least-squares equalizer synthesis techniques and their implementation with the wideband beamformer. Wideband channel characterization and chamber measurements are a provided, demonstrating wideband digital beamforming with equalization.

4.1 Uncalibrated Wideband Digital Beamforming

This section presents uncalibrated operation of an 8-element fully digital sub-Nyquist-sampled wideband receive array utilizing an RFSoC. The incident waveform is a 1.6 GHz linear chirp signal centered at 3 GHz. Array sensor signals are directly sampled by 4 GSPS ADCs such that the chirp waveform is centered in the second Nyquist zone, and upon sampling folds into the first Nyquist zone. Signal compensation is applied at complex baseband following the digital downconverter (DDC).

Section 4.1.1 covers the design of finite impulse response (FIR) fractional-sample delay filters for an embedded system. An overview of the wideband beamforming testbed is provided in Section 4.1.2, including the hardware, firmware, and software. Section 4.2.3 presents the results of simulations, over-the-wire (OTW) bench testing, and over-the-air (OTA) far-field anechoic chamber measurements. Comparisons between narrowband and wideband beamsteering measurements are presented as well as measured versus simulated wideband beamsteering performance.

4.1.1 True Time Delay Units

This section discusses digital true time delay (TTD) units and their application to wideband beamforming. TTD units can be implemented digitally through a combination of integer and fractional-sample delays. While integer-sample delays can be trivially implemented by shifting digital signal samples, fractional-sample delays require a digital filter. This section provides a design method for fractional-sample delay FIR filter synthesis and discusses considerations for embedded system applications.

4.1.1.1 Ideal Fractional-Sample Delay Filter

The ideal fractional-sample delay filter has a frequency response with unity gain and linear phase, as prescribed by the time shift property of the Fourier transform. Thus, the ideal frequency response for a delay t_{a}, is given by

$$H_{\mathrm{d}}(F) = e^{-j2\pi F t_{\mathrm{a}}} \;, \tag{4.1}$$

where the magnitude and phase responses are $|H_{\mathrm{d}}(F)| = 1$ and $\angle H_{\mathrm{d}}(F) = -2\pi F t_{\mathrm{a}}$, respectively. Different fractional-sample delays correspond to different phase slopes.

Because a digital FIR filter is composed of discrete filter taps, the filter has a discrete time domain representation with sample frequency F_s. This results in a periodic frequency domain representation with period F_s. Thus, Eq. (4.1) is rewritten to be periodic, as given by

$$H_\mathrm{d}(f) = \sum_{k=-\infty}^{\infty} H_{\mathrm{d}_{1/T_\mathrm{s}}}(f + kF_\mathrm{s}),\ |f| \leq \frac{F_\mathrm{s}}{2}\ , \tag{4.2}$$

where one period is described by

$$H_{\mathrm{d}_{1/T_\mathrm{s}}}(F) = e^{-j2\pi F t_\mathrm{a}},\ |F| \leq \frac{F_\mathrm{s}}{2}\ . \tag{4.3}$$

The corresponding set of filter taps can be computed by taking the inverse discrete-time Fourier transform, given in Eq. (4.4), of the desired periodic frequency response.

$$h_\mathrm{d}[n] \triangleq T_\mathrm{s} \int_{1/T_\mathrm{s}} H_{\mathrm{d}_{1/T_\mathrm{s}}}(f) e^{j2\pi f n T_\mathrm{s}} df\ . \tag{4.4}$$

Substituting Eq. (4.3) into Eq. (4.4) yields the following relation,

$$\begin{aligned} h_\mathrm{d}[n] &= T_\mathrm{s} \int_{1/T_\mathrm{s}} e^{-j2\pi f t_\mathrm{a}} e^{j2\pi f n T_\mathrm{s}} df \\ &= T_\mathrm{s} \int_{1/T_\mathrm{s}} e^{j2\pi f (nT_\mathrm{s} - t_\mathrm{a})} df \\ &= \frac{T_\mathrm{s}}{j2\pi(nT_\mathrm{s} - t_\mathrm{a})} e^{j2\pi f (nT_\mathrm{s} - t_\mathrm{a})} \Bigg|_{-\frac{1}{2T_\mathrm{s}}}^{\frac{1}{2T_\mathrm{s}}} \\ &= \frac{1}{\pi(n - \frac{t_\mathrm{a}}{T_\mathrm{s}})} \frac{e^{j\pi(n - \frac{t_\mathrm{a}}{T_\mathrm{s}})} - e^{-j\pi(n - \frac{t_\mathrm{a}}{T_\mathrm{s}})}}{2i} \\ &= \frac{\sin(\pi(n - \frac{t_\mathrm{a}}{T_\mathrm{s}}))}{\pi(n - \frac{t_\mathrm{a}}{T_\mathrm{s}})} \\ &\triangleq \mathrm{sinc}\left(\pi\left(n - \frac{t_\mathrm{a}}{T_\mathrm{s}}\right)\right)\ . \end{aligned} \tag{4.5}$$

As the resulting sinc response is both infinite and non-causal, it is not possible to implement the ideal fractional-sample delay filter.

4.1.1.2 Approximate Fractional-Sample Delay Filter

In order to design a finite, causal approximation, the ideal response must be appropriately truncated and shifted. In general, merely truncating the response to some finite length l_{len} produces undesirable ripple in the frequency domain. A symmetric window function improves the ripple response appreciably at the acceptable cost of a small reduction in magnitude. Although many window functions exist, a Blackman window was utilized for the fractional-sample delay filter bank in this demonstration. It is recommended to select an odd filter length given the majority of the sinc function energy is concentrated near the center of the response. Additionally, limiting the fractional-sample delay to $-0.5 < \frac{t_{\mathrm{a}}}{T_{\mathrm{s}}} < 0.5$ minimizes filter asymmetry.

To achieve causality, the finite windowed sinc response must shifted such that the first sample corresponds to $n = 0$. This is accomplished by shifting the response to the right by $\frac{l_{\mathrm{len}}-1}{2}$, assuming an odd length l_{len}. Thus, the final taps for a t_{a}-delay filter with odd length l_{len}, designed to operate on a signal sampled at frequency $F\mathrm{s} = \frac{1}{T_{\mathrm{s}}}$, are given by

$$h_{\mathrm{d}}[n] = w[n]\mathrm{sinc}\left(\pi\left(n - \frac{t_{\mathrm{a}}}{T_{\mathrm{s}}} - \frac{l_{\mathrm{len}} - 1}{2}\right)\right) , \tag{4.6}$$

where $w[n]$ is some window function, in this case a Blackman window, of length l_{len} and n is subject to the constraint $0 \leq n \leq l_{\mathrm{len}} - 1$.

4.1.1.3 Filter Length

Considerations must be made for embedded system implementations given limited system resources. Figure 4.1 provides a comparison of the magnitude response and group delay for $\frac{1}{2}$-sample delay filters of various lengths. As longer filters provide

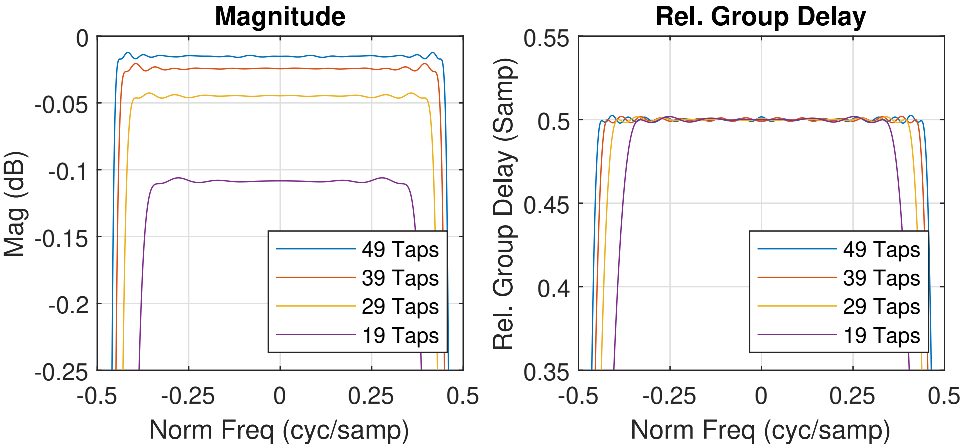

Figure 4.1: Filter Length Comparison: $\frac{1}{2}$-Sample Delay, 16-bit Fixed-Point

increased bandwidth, one must to ensure sufficient hardware resources to maintain accuracy in high digital bandwidth applications.

4.1.1.4 Fixed-Point Quantization

FIR filters are typically implemented using fixed-point numerical representation rather than floating point due to computational efficiency. Figure 4.2 shows degradation in the frequency response due to different fixed-point precisions for a 49-tap $\frac{1}{2}$-sample delay filter. Black traces show the float point response. Subsequent traces show the 16-bit, 14-bit, and 12-bit responses, respectively. Reduced coefficient precisions cause increased ripple.

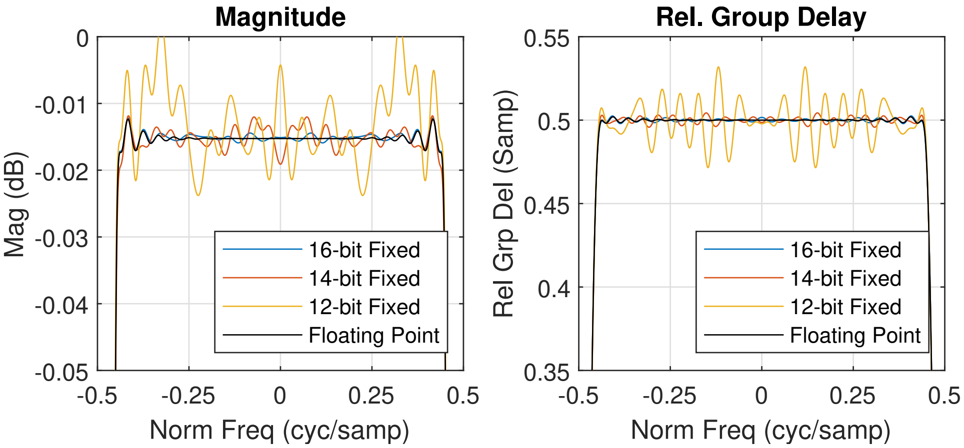

Figure 4.2: Fixed-Point Precision Comparison: $\frac{1}{2}$-Sample Delay, 49-Taps

4.1.1.5 Fractional-Sample Delay Resolution

In real-time applications, filter coefficients are typically pre-computed for a finite set of prescribed delays rather than computed in real-time. Figure 4.3 shows an example filter bank with a $\frac{1}{6}$-sample resolution. The filter bank consists of six 49-tap filters with 16-bit fixed-point coefficients. The set of filter coefficients defined for a 0-fractional-sample delay results in an integer-sample delay equivalent to the filter group delay. This ensures that the fractional-sample filter bank group delay is applied regardless of whether a nonzero fractional-sample delay is required for a given steering operation.

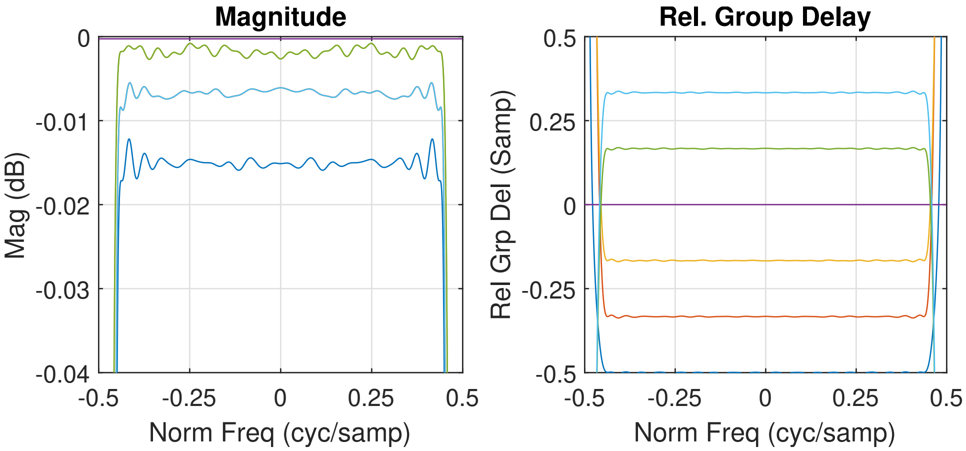

Figure 4.3: Filter Bank: $\frac{1}{6}$-Sample Resolution, 49-Tap, 16-bit Fixed-Point

Quantization lobes due to finite fractional-sample delay resolution degrade system SNR [67]. In this dissertation, phase shifter quantization was used as a baseline for TTD unit quantization analysis. Common digitally-controlled analog phase shifters have 6 control bits, providing a resolution of $\frac{360°}{2^6} = 5.625°$. This provides a maximum time shift resolution given by

$$t_{\mathrm{res}} = \frac{\phi_{\mathrm{res}}}{360°} T_{f_{\mathrm{high}}} = \frac{T_{f_{\mathrm{high}}}}{2^6} \ , \tag{4.7}$$

where f_{high} is the highest frequency in the signal of interest and $T_{f_{\mathrm{high}}}$ is its corresponding period. Thus, the fractional-sample resolution can be defined as $m_{\mathrm{res}} = \frac{t_{\mathrm{res}}}{T_{\mathrm{s}}}$, where T_{s} is the sample period. The fractional-sample resolution which maintains, at minimum, an equivalent resolution using TTD units as a 6-bit phase shifter for a signal whose highest frequency is $F_{\mathrm{high}} = 3.8$ GHz directly sampled at $F_{\mathrm{s}} = 4$ GHz, is given by

$$m_{\mathrm{res}} = \frac{T_{f_{\mathrm{high}}}}{2^6\ T_{\mathrm{s}}} = \frac{4\mathrm{GHz}}{2^6\ 3.8\mathrm{GHz}} \approx 0.0165 \text{ samp} . \tag{4.8}$$

Thus, the desired resolution is about $\frac{1}{60}$-sample, or that the filter bank would contain 60 separate sets of coefficients.

4.1.2 Uncalibrated RFSoC Testbed

An image of the wideband beamforming research testbed mounted in OU's far-field anechoic chamber is given in Figure 4.4. The case at the base of the black high density polyethylene (HDPE) frame is a Pentek 3U VPX chassis which houses the Xilinx RFSoC. Eight MMCX-to-SMA cables provide RF interfaces from the antenna elements to the RFSoC ADC channels for receive operation. The channel 1 DAC provides an analog RF interface for the beamformer output following the digital upconverter (DUC). Beamformer control is applied through a Secure Shell (SSH) connection to the Petalinux operating system (OS) running on the RFSoC real-time processor.

The upper chassis frame supports the linear array and the phase-amplitude control (PAC) board. The PAC board consists of eight analog channels, each providing 6-bit amplitude and phase control at 0.5 dB and 5.625° resolution respectively. Used for beamsteering experiments throughout OU's Radar Innovations Laboratory (RIL), the PAC board is used in this testbed to facilitate baseline narrowband beamsteering measurements provided in Section 4.1.3. A USB battery box and Raspberry Pi provide power and control to the PAC board.

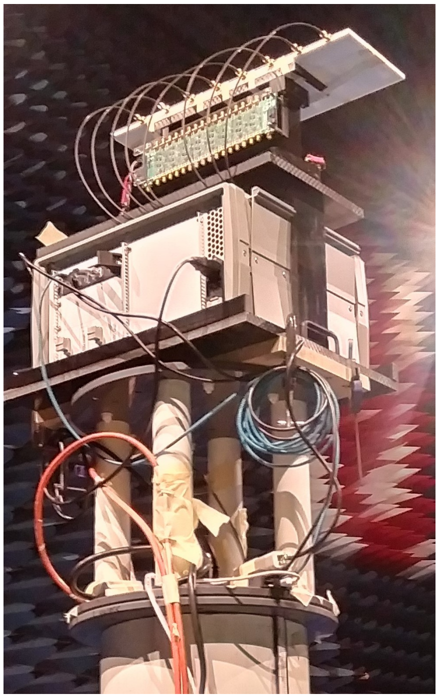

Figure 4.4: Research Testbed Mounted in Far-Field Anechoic Chamber

4.1.2.1 Antenna Array

The antenna aperture shown in Figure 4.5 is a wideband Vivaldi array developed for demonstration of the RFSoC wideband beamformer. Specifically designed for this research endeavor, the 8-element horizontally polarized linear array supports better than a 2 GHz bandwidth centered at 3 GHz, providing at least a 66% fractional bandwidth. This high fractional bandwidth precludes the use of narrowband beamsteering. The digital beamformer was designed to utilize 80% of the Nyquist zone, providing a 1.6 GHz bandwidth at a sample frequency of 4 GHz.

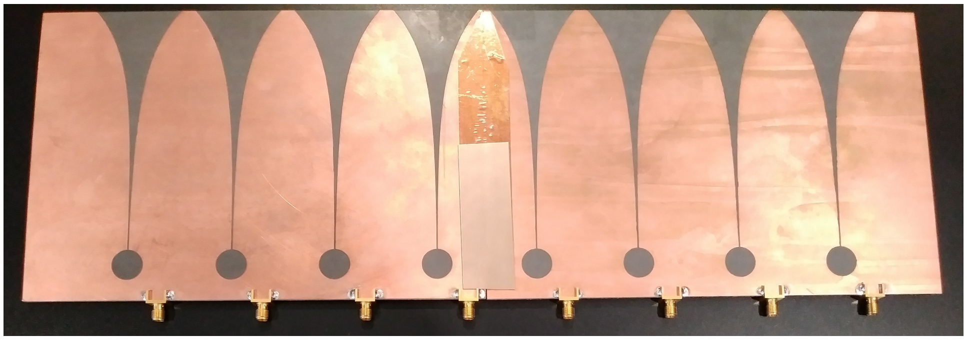

Figure 4.5: Wideband Vivaldi Array used in the Testbed in Figure 4.4

4.1.2.2 RFSoC Hardware

The research testbed is constructed around the Pentek Quartz Model 5950. This is a 3U OpenVPX carrier card which integrates Pentek's Model 6001 QuartzXM eXpress Module and provides MMCX RF interfaces to the eight 4 GSPS ADC's and eight 6.4 GSPS DAC's. The eXpress Module integrates the Xilinx Zynq UltraScale+ RFSoC and houses additional resources such as DDR4 Random Access Memory (RAM) and power management. The Model 5950 board is housed in the Pentek VPX chassis shown in Figure 4.4, which provides power conversion, cooling, and interface access via the Pentek Rear Transition Module (RTM).

4.1.2.3 RFSoC Firmware

The Xilinx RFSoC is built upon an FPGA fabric allowing for extensive customization. Pentek provided the Vivado project for their FPGA Design Kit (FDK), which enables operation directly out of the box. This firmware incorporates the Xilinx real-time processor and RF data converter IP cores, and the necessary logic for various interfaces on the Pentek hardware including 100 GigE UDP, PCI Express, and DDR4 RAM. It also supports direct recording of RF ADC data. Much of this interface logic was subsequently removed as it is not required for the beamforming testbed.

The functional block diagram for the FPGA image is provided in Figure 4.6. As discussed in Sections 4.1.1 and 2.7, sub-Nyquist-sampled true time delay beamforming requires an integer-sample delay, fractional-sample delay, and phase shifter for channel compensation. Xilinx's RF data converter IP core provides user control of the ADC and DAC, and subsequently DDC and DUC, resources. In the DDC for a given channel, the sampled signal, represented by real-valued samples at 4 GSPS, is decimated by a factor of 2 and frequency shifted to complex baseband. The decimated spectrum at complex baseband is represented by complex-valued samples at 2 GSPS.

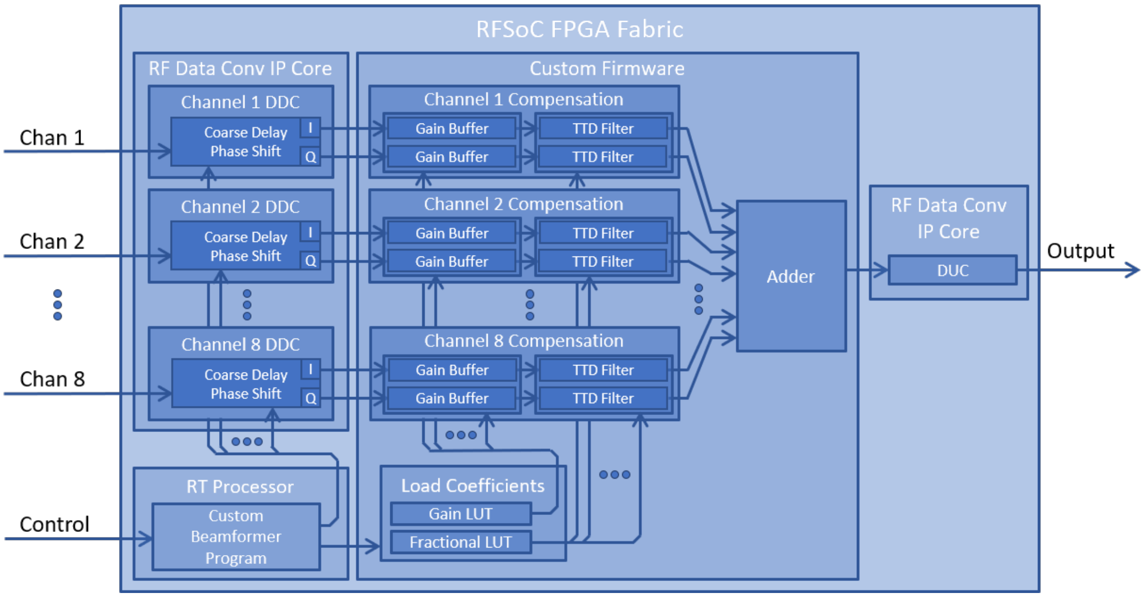

Figure 4.6: Wideband Beamformer Functional Block Diagram

The DDC provides an integer-sample delay of up to seven samples as well as phase offset control of the digital mixer's numerically controlled oscillator (NCO). This allows the integer-sample delay and relative phase shift to be implemented through software control of the DDC for single beam applications rather than through custom FPGA firmware. Fractional-sample delays are implemented via instantiation of FIR filters. The conjugate-symmetric fractional-sample delay filter is implemented via real-valued coefficients allowing for independent filtering of the I and Q data streams. Finally, the compensated channel signals are summed in the adder and passed to the DUC, which interpolates by a factor of 2 and frequency shifts the beamformer output to the carrier frequency for transmission via the DAC interface.

Fractional-Sample Filter Bank Implementation In order to utilize the full digital bandwidth, the fractional-sample filter bank was designed to process multiple samples in parallel. While the complex baseband spectrum is represented by complex-valued samples at 2 GSPS, the fractional-sample filter bank is designed to operate at 250 MHz. This requires parallel processing of eight samples for both the I and Q data streams for each of the eight channels.

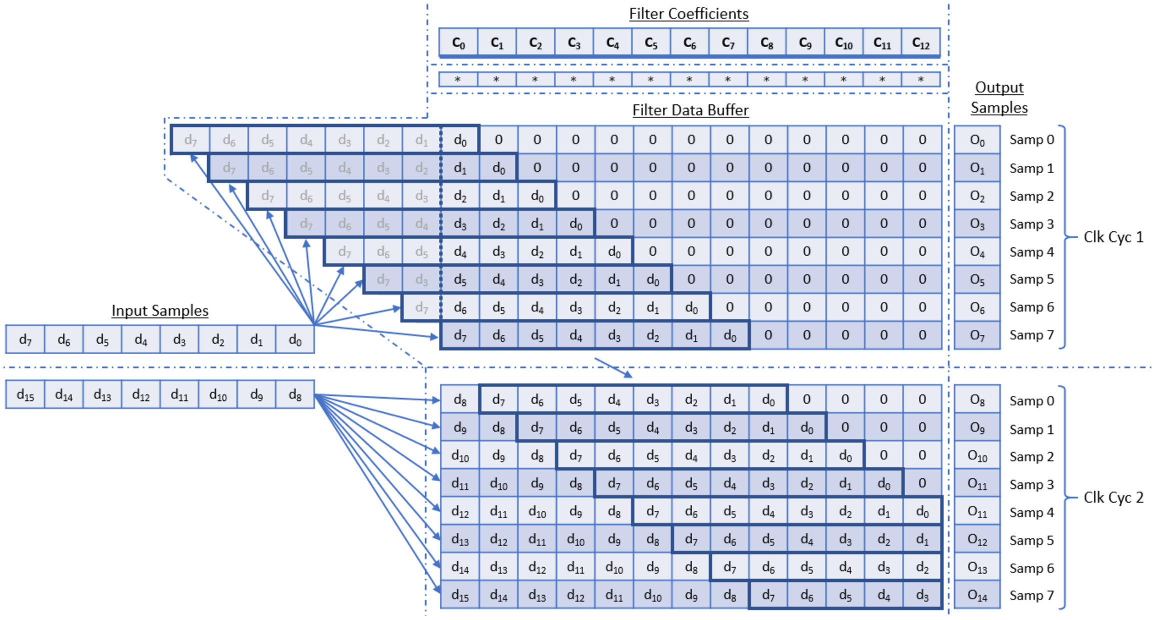

Figure 4.7: Interleaved FIR Filter Diagram

Figure 4.7 shows an interleaved real-valued FIR filter implementation which processes eight samples per clock cycle. In order to maintain data throughput without decimation, the filter must compute an output sample for each possible buffer offset. This necessitates resources for eight simultaneous filter instantiations. For each clock cycle, data in a given row is shifted eight samples to the right to make room for the next input sample set. The greyed-out samples represent registers which hold current input samples not required for the corresponding output sample but which must be stored for the upcoming clock cycle. This architecture requires eight multipliers per filter tap to produce eight output samples per clock cycle. Two interleaved filter instantiations are required for each of the eight channels, requiring 128 multipliers per fractional-sample delay filter tap.

Due to timing issues during FPGA place and route, the fractional-sample delay filter length was set to 13 taps, requiring 1664 DSP units for the fractional-sample delay filter banks. Although difficult to determine from the uncalibrated wideband measurements presented in Section 4.1.3.3, this results in some degradation at the band edges. Filter coefficients utilized 16-bit fixed point precision and were computed for eight fractional-sample delay steps providing 0.125-sample resolution.

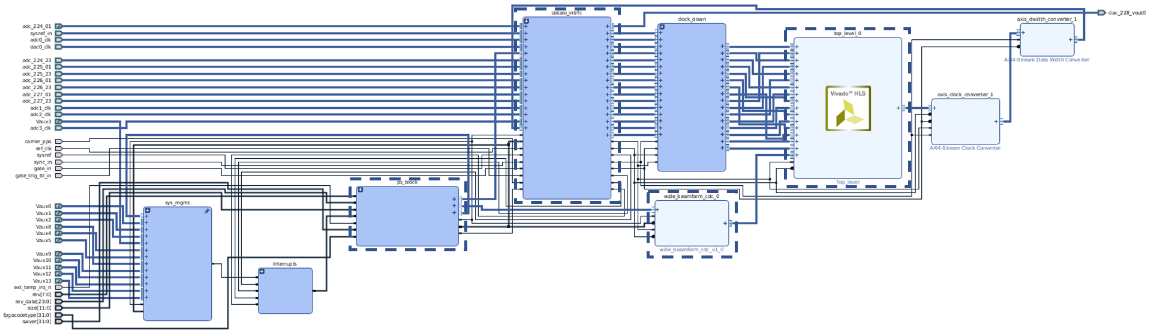

Figure 4.8: Wideband Beamformer Vivado Block Design

Resource Utilization The Vivado FPGA project Block Design is given in Figure 4.8. Much of the original Pentek FDK has been removed to facilitate routing of the beamformer. Various components of interest are outlined, such as the real-time processor, RF IP core, custom beamformer, and clock domain crosser. Device utilization is provided in Table 4.1 and the design layout is given in Figure 4.9. The primary resource of interest is DSP utilization, requiring 1664 DSPs for the fractional-sample delay filter banks, 128 DSPs for the gain buffers, 240 DSPs for the 10-tap complex coefficient output compensation filter, and 3 for miscellaneous applications. A discussion on the output compensation filter is provided in Section 4.1.3.2.

Table 4.1: Uncalibrated Wideband Beamformer FPGA Resource Utilization

Resource	Utilization	Available	%
LUT	63808	425280	15.0
LUTRAM	3215	213600	1.5
FF	106831	850560	12.6
BRAM	28	1080	2.6
DSP	2035	4272	47.6
IO	35	347	10.1
BUFG	8	696	1.2
MMCM	1	8	12.5

The custom designed Vivado HLS block in Figure 4.8 was developed in C++ using Vivado HLS, and contains the fractional-sample delay filter bank and summation

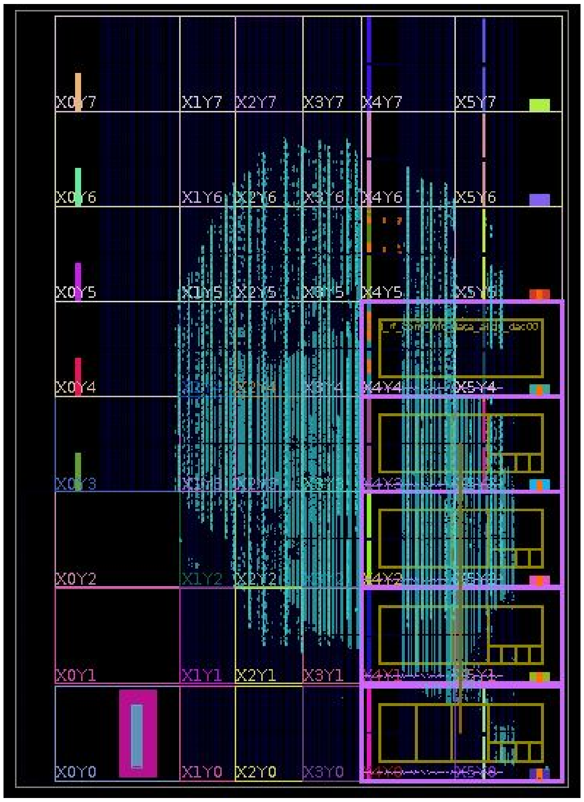

Figure 4.9: Wideband Beamformer Design Layout

node. Vivado HLS provides an environment which facilitates rapid development of register transfer logic (RTL) designs using higher level programming languages such as C and C++. HLS allows the user to dictate how hardware resources are allocated when implementing functional operations. This provides the ability to parallel loop iterations, pipeline data flow, and control how arrays are managed in memory or registers. As discussed in the previous section, parallel resource allocation is necessary to support the required data rates.

4.1.2.4 RFSoC Software

The Pentek Model 5950 also included a Board Support Package (BSP), which provides a series of example programs and an application programming interface (API) to the lower level FPGA logic. Because the example programs were written to interface with Pentek's default FDK FPGA project, much of which is stripped away to make room for the beamformer design, the example program was modified to remove calls to hardware which is no longer present. A custom program was written which memory maps the beamformer IP core, computes the necessary steering commands, and provides control data to the DDC, gain buffers, and fractional-sample delay filter bank.

4.1.3 Uncalibrated Results

This section presents simulated and measured results of the uncalibrated beamformer performance. For reference, a comparison of ideal wideband and narrowband array factors can be found in Figure 2.9 in Section 2.3.3. Results are provided as follows. Section 4.1.3.1 shows simulation results for ideal and measured channel waveforms. Section 4.1.3.2 discusses bench level OTW measurements which facilitated beamformer implementation and demonstrated the need for an output compensation filter. Following the completion of bench testing, far-field chamber OTA measurements were captured for several steering locations for both narrowband and wideband operating modes. These are presented and discussed in Section 4.1.3.3.

4.1.3.1 Simulation Results Uncalibrated

Early in development, a Matlab model was developed to explore various performance parameters. This allowed for comparisons of fractional-sample delay filter lengths, fixed point resolutions, and fractional-sample delay resolutions. Bit-accuracy was incorporated to aid in the HLS development of beamformer firmware, allowing for the generation of test vectors for use with the HLS testbench. Using Pentek FDK support for chirp generation and digital RF data capture, a chirp waveform spanning 80% of the Nyquist zone was externally looped back to each ADC channel, allowing for frequency response characterization. This captured data was fed into the Matlab model to estimate the expected uncalibrated response of the RFSoC wideband beamformer.

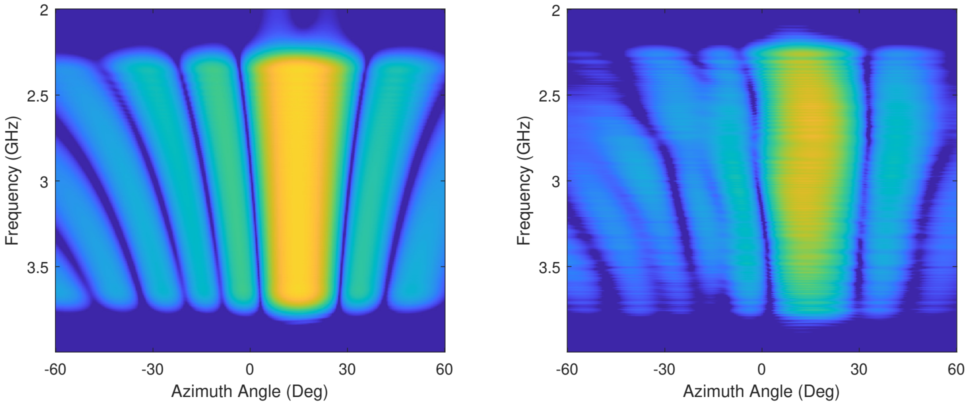

Figure 4.10: Wideband Matlab Simulation Results Steered to $\theta_{\text{st}} = 15°$, Left: Ideal Input Channel Data, Right: Measured Channel Data

Figure 4.10 shows a comparison of simulation results when sourced with ideal channel data and measured channel data.

4.1.3.2 Over-the-Wire Results

Bench testing revealed appreciable gain variation in the RFSoC frequency response. OTW measurements were taken using a network analyzer capturing the gain response of the wideband beamformer. The network analyzer source port fed an 8-1 splitter which served as the simultaneous signal source to each ADC channel. The RFSoC channel 1 DAC was connected to the sink port of the network analyzer and configured to operate in the second Nyquist one. In this setting, the output signal is mixed with the sample frequency in order to shift signal energy into the second Nyquist zone at the expense of the first. Bench testing demonstrated the effectiveness of the beamformer prior to OTA measurements, effectively collimating a beam from an arbitrary distribution of channel delays.

Initial results showing the network analyzer magnitude response are shown in Figure 4.11. The gold trace shows the initial sweep, which captured what is effectively the mainbeam response to a boresight signal. In this configuration, each channel delay is approximately equal. A set of arbitrary cables was characterized and then distributed amongst the channel paths. The blue sweep shows the uncompensated response due

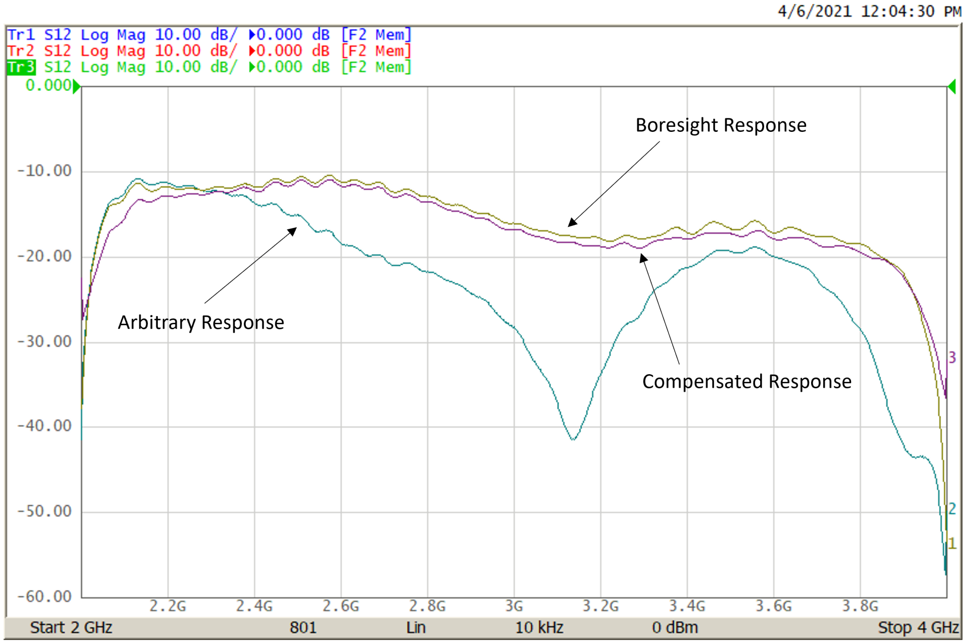

Figure 4.11: Arbitrary Delay OTW Bench Test

to the arbitrary cable delays. The magenta trace shows the final sweep taken after updating the beamformer with the appropriate compensation delays. This shows strong agreement with the initial boresight sweep indicating effective beamsteering.

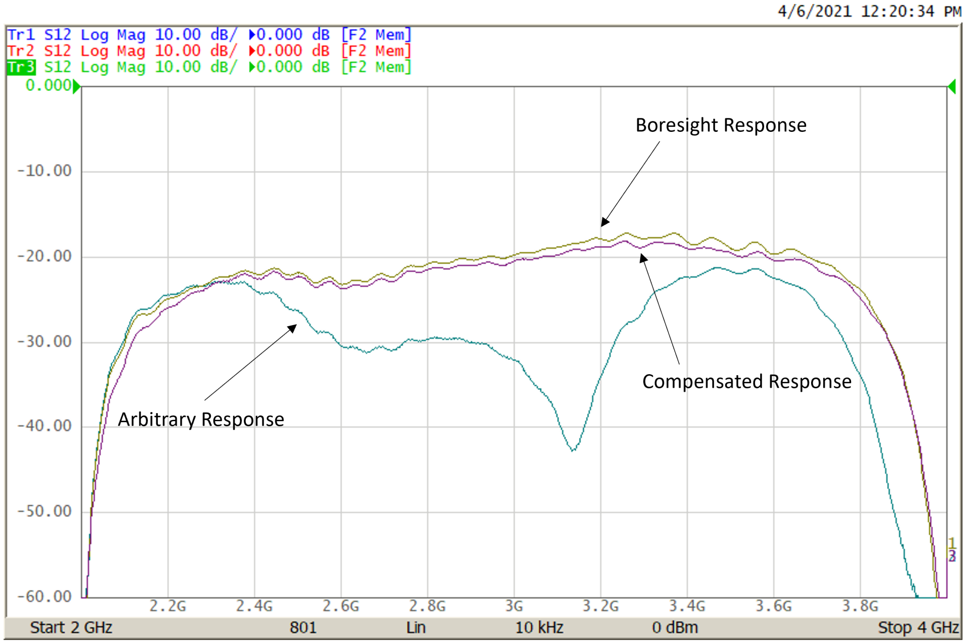

Figure 4.12: Arbitrary Delay OTW Bench Test with Output Compensation

Of note is the strong frequency dependence of the magnitude response. Initial OTA results revealed additional effects on the magnitude frequency response which

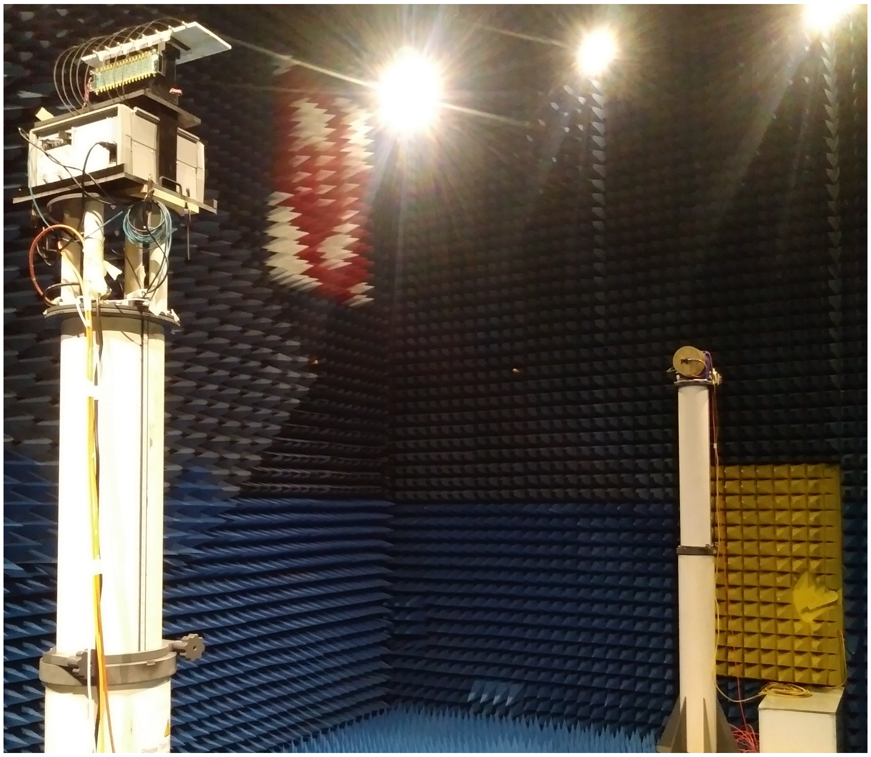

Figure 4.13: Beamforming Testbed Mounted in Far-field Anechoic Chamber

compounded the effects shown in Figure 4.11. A complex-valued output filter was synthesized and implemented to flatten out the magnitude response over frequency. Figure 4.12 shows the magnitude response of the RFSoC beamformer with the output compensation filter, in which the lower frequencies are reduced. This output compensation filter was used in the final system OTA measurements presented in the following section.

4.1.3.3 Over-the-Air Results

OTA results were measured in OU's far-field anechoic chamber, shown in Figure 4.13. Measurements were taken for three test cases, narrowband measurements using OU's PAC board, narrowband measurements using the RFSoC, and wideband measurements using the RFSoC. For each operating mode, the main beam was steered to 15°, 30°, and 45° at the center frequency of 3 GHz.

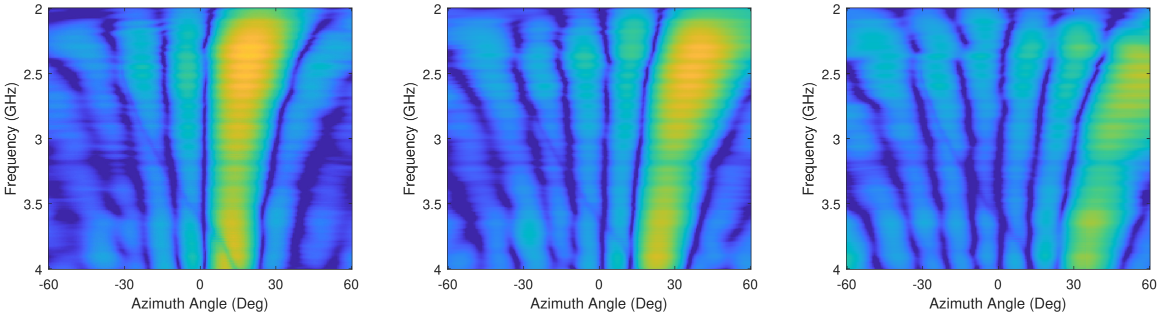

Figure 4.14: Narrowband Measurements using OU's PAC Board

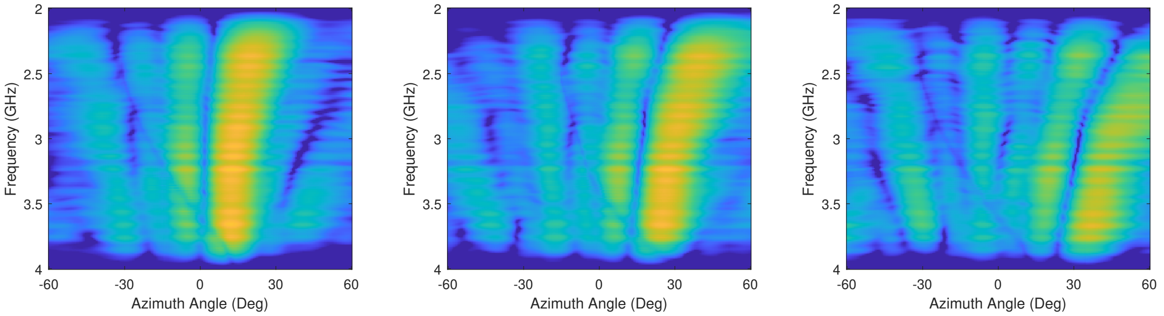

Figure 4.15: Narrowband Measurements using RFSoC Digital Beamforming

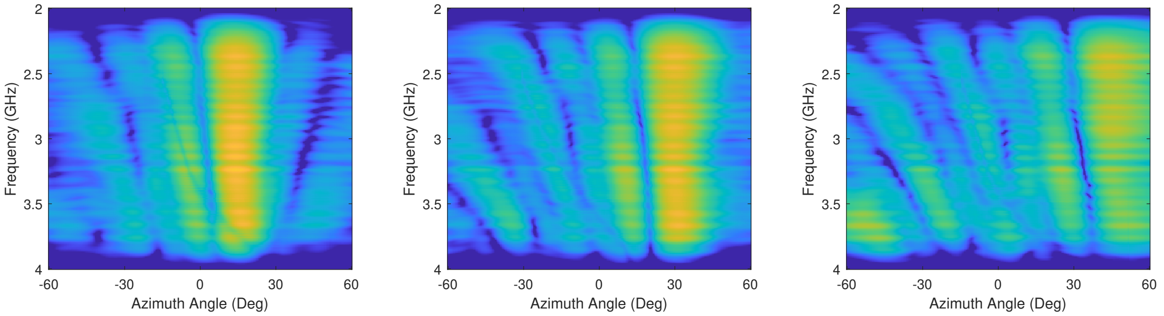

Figure 4.16: Wideband Measurements using RFSoC Digital Beamforming

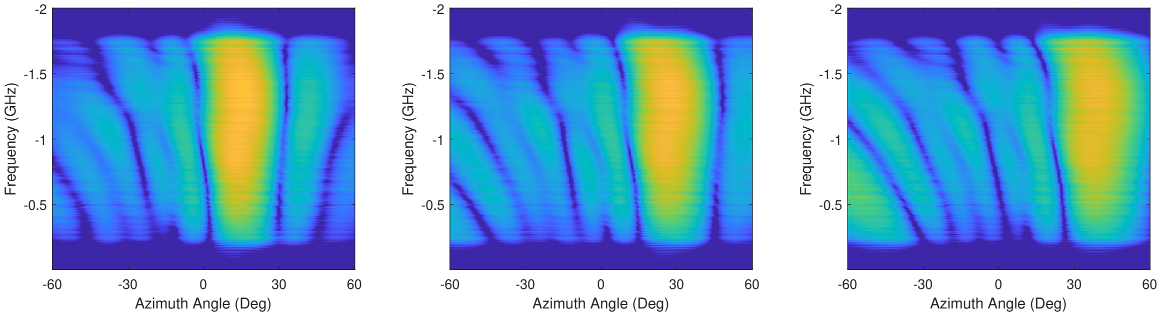

Figure 4.17: Simulation Results using Measured Channel Data

Narrowband measurements acquired using OU's PAC board, shown in Figure 4.14, provide a baseline for the RFSoC results. The RFSoC beamformer implementation supports narrowband beamsteering via phase control of the digital mixer's NCO.

These results are shown in Figure 4.15. Although the main beam position and shape match generally well between PAC board and RFSoC narrowband measurements, nulls are substantially more defined in Figure 4.14 indicating that the PAC board is well calibrated. Appreciable RFSoC channel-to-channel errors degrade sidelobe performance, as predicted in Figure 4.10. Both narrowband test cases show appreciable beam squint over the measurement bandwidth.

Uncalibrated wideband antenna patterns are provided in Figure 4.16. Mainbeam accuracy is maintained throughout the signal bandwidth despite degraded sidelobe performance. Although demonstration of wideband channel equalization is part of future work, measured channel data was incorporated into Matlab simulations. Simulated results are provided in Figure 4.17 and show reasonable agreement with Figure 4.16. Similar sidelobe patterns indicate that effective wideband equalization will substantially improve sidelobe performance.

4.2 Least-Squares Digital Equalizers

Because accurate array performance relies on matched channel behavior [4], substantial inter-channel magnitude and phase errors present in most practical phased array systems often preclude uncalibrated operation. Frequency-varying amplitude and dispersive phase behavior [68], which varies in general between channels, inhibit the coherent summation of element signals, degrading performance characteristics such as main beam steering position and null depth. This is of particular concern in adaptive beamforming applications, which seek to place deep, spatially accurate nulls based on current element signals samples [69].

Given that many narrowband systems implement channel compensation via attenuators and phase shifters [5], simple adjustments to channel control signals are commonly sufficient to mitigate inter-channel imbalances for such applications. This assumes that channel variations are steady enough over the narrow instantaneous bandwidth to be corrected by a constant shift in attenuator and phase commands, effectively a 1-tap complex-coefficient finite impulse response (FIR) filter. As instantaneous bandwidth is often configurable within a larger system bandwidth, a series of frequency-dependent compensation values is often required, often stored in a lookup table (LUT). System bandwidth may be partitioned into several frequency bins such that compensation values can be represented by zeroth or first order approximations, reducing calibration table memory requirements.

In contrast, wideband applications require higher fidelity compensation over the band of interest. As the channel frequency responses for most practical systems have appreciable frequency dependence, zeroth or first order approximations fail to accurately represent channel characteristics. FIR filters have been shown to effectively compensate over a prescribed bandwidth and can be implemented at the element level in fully digital architectures [29].

The authors in [70] demonstrate an equalization approach for both deterministic and adaptive applications. They provide adaptive nulling demonstrations with instantaneous bandwidths of 1 MHz, a considerable achievement given the technology available at the publication of this seminal work. The deterministic approach was leveraged in [69] to determine the impact of equalization on adaptive digital beamforming (ADBF) for a 4-element array. Channel data was represented at 125 MHz, consisting of a 124 MHz chirp. Using measured channel data in the frequency domain, the authors in [71] synthesize channel equalizers for a bandwidth of about 500 MHz sampled at 1.5 GHz. Verification was ultimately carried out in the MATLAB environment. Wideband calibration is implemented using the time reversal technique in [72] and demonstrated [68] for a 16-element array with 500 MHz of instantaneous bandwidth sampled at 1.333 GHz. Antenna patterns at various discrete frequencies are provided at multiple steering angles.

This section applies the equalizer techniques discussed in [69], [70], and [71] to the uncalibrated 8-element RFSoC-based sub-Nyquist-sampled wideband beamformer [73]. For reference, ADC channels operate at 4 GSPS on a 1.6 GHz bandwidth captured using a wideband Vivaldi array which supports better than 2 GHz of bandwidth centered at 3 GHz. The beamformer output is provided through a DAC for chamber measurement. At the center of the testbed is the Pentek Quartz Model 5950, a 3U OpenVPX carrier card which integrates a Xilinx Zynq UltraScale+ RFSoC, providing MMCX RF interfaces to the eight ADCs and eight DACs. Pentek factory software and firmware provided the necessary functionality for waveform generation and RF data capture and a solid base on which to design and implement the equalizer and beamformer.

This section is organized as follows. The relevant background, including channel characterization, equalizer synthesis techniques, equalizer performance assessments, and fractional-sample delay filter bank implementation, is presented in Section 4.2.1. An overview of the wideband equalizer and calibrated beamformer testbed is provided in Section 4.2.2, discussing equalizer design decisions and implementation of a wideband configurable complex-coefficient FIR filter bank. Section 4.2.3 provides simulation results and far-field anechoic chamber measurements for various steering directions and system configurations.

4.2.1 Equalizer Formulation and Design Approach

This section discusses the theory behind the equalizer synthesis techniques explored herein and the design approach for the wideband beamformer demonstrated in Section 4.2.3. Wideband equalization is implemented via configurable complex-coefficient FIR filters synthesized using measured channel characterization data. This approach assumes a linear system response, and although extensive work in nonlinear equalization (NLEQ) [74], [75] and digital pre-distortion (DPD) [76], [77] explores techniques for addressing nonlinear system responses, they are not discussed or applied within.

Section 4.2.1.1 discusses capture of channel characterization data. Filter synthesis techniques are presented in Section 4.2.1.2, which include filter synthesis from an arbitrary frequency response. Section 4.2.1.3 discusses equalizer performance metrics used to quantify equalizer performance sensitivity to group delay and filter length. Lastly, to support true time delay (TTD) beamsteering [10], Section 4.2.1.4 discusses integration of the fractional-sample delay filter bank with the synthesized equalizers.

4.2.1.1 Characterization Data Capture

High accuracy in channel characterization is paramount for effective equalizer implementation. Characterization can be accomplished either in the frequency domain [72] or the time domain [69], [71]. The channel response to a series of continuous wave (CW) tones uniformly distributed across the system bandwidth provides a frequency domain estimate of the channel characteristics. Alternatively, if time domain channel data can be captured directly, the response to a reference waveform can serve to characterize channel behavior. This requires the selection and generation of a reference waveform with which to excite the channel under test. Although several waveforms have been utilized for channel characterization, which may be dependent on the specific system application, the intent is that all frequencies within the band of interest are uniformly excited [69].

Pentek factory firmware support for waveform capture and generation was integral to channel characterization efforts. Figure 4.18 shows a block diagram describing the relevant firmware components for channel characterization. Factory firmware was modified with the addition of the two AXI4-Stream components shown in Figure 4.18, enabling internal digital loopback of the reference waveform. The channel 1 DAC serves as the external reference source, looped back through an 8-to-1 splitter to the

8 ADCs. RF data was captured concurrently for all channels, recording several chirp pulses for characterization. A linear frequency modulated (LFM) chirp waveform spanning 90% of the Nyquist zone served as the reference waveform in this application. The chirp generator within the Pentek RFSoC firmware was configured to output a 1.8 GHz chirp at 4 GSPS centered in the second Nyquist zone. This provided adequate spectral coverage of the 1.6 GHz bandwidth received from the 3 GHz centered array during system operation.

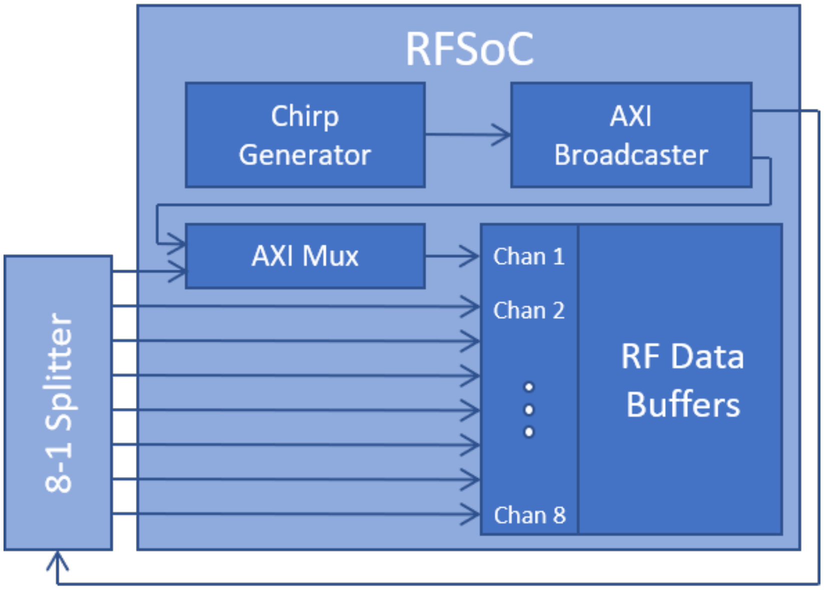

Figure 4.18: RFSoC Channel Characterization Block Diagram: External loopback provides the reference waveform to all 8 channels simultaneously, Internal loopback allows for capture of the reference waveform

Notably, this characterization method neglects the inter-channel differences within the aperture as well as mutual coupling affects. Additional work would center on capturing aperture characteristics by routing the reference waveform over the air through the antenna chamber, potentially characterizing mutual coupling behavior at various azimuth angles. Characterization of the radiation path and chamber hardware would be required to appropriately de-embed these effects from channel characterization.

Measured characterization data is represented as M-length column vectors containing the time domain samples. The internally looped back reference waveform is represented by the column vector $\mathbf{y} = [y_0, y_1, \cdots, y_{\mathrm{M}-1}]^{\mathrm{T}}$. The channel response data is stored in a series of column vectors $\mathbf{x}_{\mathrm{c}} = [x_0, x_1, \cdots, x_{\mathrm{M}-1}]^{\mathrm{T}}$ where the subscript c denotes the channel number. These are summarized in Table 4.2. A Hann window was applied to the initial and final 2.5% of each vector.

Table 4.2: Channel Characterization Variables

Variable	Description
$\mathbf{y}$	Time Domain Reference Waveform Vector
$\mathbf{x}_{\mathrm{c}}$	Time Domain Channel Response Vector

4.2.1.2 Equalizer Filter Synthesis

This section outlines two methods for synthesizing equalizer FIR filter coefficients utilizing the reference waveform and channel characterization data. Frequency domain synthesis is discussed in Section 4.2.1.2 and time domain synthesis is discussed in Section 4.2.1.2. Both techniques utilize a least squares approach to minimize the mean error between the equalizer output and the reference waveform. Section 4.2.1.2 discusses the importance of appropriately delaying the reference waveform prior to filter synthesis. By accounting for the group delay associated with the equalizer FIR filter, the reference waveform is time-aligned with the equalizer output, drastically improving performance. Section 4.2.1.3 discusses a few metrics for quantifying equalizer performance.

Frequency Domain Synthesis This approach seeks to synthesize a channel equalizer FIR filter using a given frequency response [71]. A filter whose frequency response most closely approximates the desired response $\mathbf{H}_{\mathrm{des}}$ is synthesized using arbitrary frequency response filter synthesis [71], [78], [79], etc. This desired response is that which compensates the given channel frequency response to match that of the reference waveform, as defined by $\mathbf{H}_{\mathrm{des}}\mathbf{H}_{\mathrm{x}} = \mathbf{H}_{\mathrm{y}}$. The variables $\mathbf{H}_{\mathrm{x}}$ and $\mathbf{H}_{\mathrm{y}}$ are the channel and reference waveform frequency responses respectively. Solving for the desired equalizer frequency response $\mathbf{H}_{\text{des}}$ yields

$$\mathbf{H}_{\text{des}} = \frac{\mathbf{H}_{\text{y}}}{\mathbf{H}_{\text{x}}} \; . \tag{4.9}$$

As part of equalizer design for the RFSoC, consider seminal design techniques for digital filters. Filter synthesis of the arbitrary frequency response $\mathbf{H}_{\text{des}}$ is achievable using the inverse discrete Fourier transform (IDFT) [35], which is defined as

$$h_n \triangleq \frac{1}{N} \sum_{k=0}^{N-1} H_k e^{j\frac{2\pi}{N}kn}, \; n \in [0, N-1] \; . \tag{4.10}$$

This can be rewritten in matrix form, as given by $\mathbf{h} = \mathbf{C}^{-1}\mathbf{H}$, where $\mathbf{H}$ is an N-length column vector of frequency domain H_k values, $\mathbf{h}$ is an N-length column vector of time domain h_n values, and $\mathbf{C}$ is an $N \times N$ Fourier transform kernel matrix whose elements are given by $e^{-j\frac{2\pi}{N}kn}$, or alternatively,

$$\mathbf{C} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & e^{-j\frac{2\pi}{N}} & \cdots & e^{-j(N-1)\frac{2\pi}{N}} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & e^{-j(N-1)\frac{2\pi}{N}} & \cdots & e^{-j(N-1)^2\frac{2\pi}{N}} \end{bmatrix} \; . \tag{4.11}$$

The indices k and n index the rows and columns respectively.

This approach is inefficient as the number of filter taps will be equivalent to the number of specified frequency points. Hence, either the frequency response will have poor resolution or the FIR filter will be prohibitively long. One can approximate the desired frequency response by reducing the filter length to N_{o} yielding the overdetermined system given by

$$\mathbf{H} \approx \mathbf{C}_{\text{o}}\mathbf{h}_{\text{o}} \; , \tag{4.12}$$

where $\mathbf{h}_{\mathrm{o}}$ is an N_{o}-length column vector of time domain h_n values and $\mathbf{C}_{\mathrm{o}}$ is an $N \times N_{\mathrm{o}}$ kernel matrix consisting of the first N_{o} columns of $\mathbf{C}$

Eq. (4.12) can be approximately solved using the method of ordinary least squares, as given by

$$\min_{\mathbf{h}_{\mathrm{o}}} ||\mathbf{C}_{\mathrm{o}}\mathbf{h}_{\mathrm{o}} - \mathbf{H}||, \tag{4.13}$$

the closed form solution of which can be written as

$$\mathbf{h}_{\mathrm{o}} = (\mathbf{C}_{\mathrm{o}}^{\mathrm{H}}\mathbf{C}_{\mathrm{o}})^{-1}\mathbf{C}_{\mathrm{o}}^{\mathrm{H}}\mathbf{H} \ . \tag{4.14}$$

One can apply a frequency-dependent weighting function to prioritize particular sub-bands within the spectrum. An N-length weighting vector can be incorporated in the form of an $N \times N$ diagonal matrix $\mathbf{W}$. To apply the weight vector, Eq. (4.12) is rewritten as

$$\mathbf{W}^{1/2}\mathbf{H} \approx \mathbf{W}^{1/2}\mathbf{C}_{\mathrm{o}}\mathbf{h}_{\mathrm{o}} \ , \tag{4.15}$$

whose normal equation is given by

$$\mathbf{C}_{\mathrm{o}}^{H}\mathbf{W}\mathbf{C}_{\mathrm{o}}\mathbf{h}_{\mathrm{o}} = \mathbf{C}_{\mathrm{o}}^{H}\mathbf{W}\mathbf{H} \ . \tag{4.16}$$

The solution to the normal equation is given by

$$\mathbf{h}_{\mathrm{o}} = (\mathbf{C}_{\mathrm{o}}^{H}\mathbf{W}\mathbf{C}_{\mathrm{o}})^{-1}\mathbf{C}_{\mathrm{o}}^{H}\mathbf{W}\mathbf{H} \tag{4.17}$$

and is known as the weighted least squares.

Substituting the desired frequency response $\mathbf{H}_{\mathrm{des}}$, as defined in Eq. (4.9), for the arbitrary response $\mathbf{H}$ yields the filter coefficients $\mathbf{h}_{\mathrm{o}} = \mathbf{h}_{\mathrm{eq}}$ whose frequency response closely approximates the desired response. Thus, frequency domain equalizer synthesis is given by

$$\mathbf{h}_{\mathrm{eq}} = (\mathbf{C}_{\mathrm{o}}^{H}\mathbf{W}\mathbf{C}_{\mathrm{o}})^{-1}\mathbf{C}_{\mathrm{o}}^{H}\mathbf{W}\mathbf{H}_{\mathrm{des}} \ . \tag{4.18}$$

Table 4.3 summarizes the relevant variables for frequency domain synthesis.

Table 4.3: Frequency Domain Synthesis Variables

Variable	Description
$\mathbf{H}_{\mathrm{y}}$	Frequency Domain Reference Waveform Vector
$\mathbf{H}_{\mathrm{x}}$	Frequency Domain Channel Response Vector
$\mathbf{H}_{\mathbf{des}}$	Frequency Domain Desired Channel Response Vector
$\mathbf{C}_{\mathrm{o}}$	Reduced DFT Kernel Matrix
$\mathbf{W}$	Diagonal Weighting Matrix
$\mathbf{h}_{\mathrm{eq}}$	Equalizer Coefficient Vector

Time Domain Synthesis Equalizer filter coefficients can also be synthesized directly using time domain samples. Key features of the equalization procedure outlined in [70] are applied to our RFSoC-based wideband beamformer. In brief, this procedure utilizes the least squares approach to identify a weight vector which closely converts time domain samples of the channel characterization data to those of the reference waveform, as given by

$$\mathbf{y}_{\mathrm{des}} \approx \mathbf{X}\mathbf{w}_{\mathrm{eq}} \ , \tag{4.19}$$

but rather than downsampling, the entire data vector is used to synthesize the equalizer.

The column vector $\mathbf{y}_{\mathrm{des}}$ is of length $N = (M + L - 1)$ where M is the number of time domain samples in the reference waveform and channel characterization data and L is the equalizer filter length. It contains time domain samples of the reference waveform $\mathbf{y}$ zero-padded by $L - 1$ such that

$$\mathbf{y}_{\mathrm{des}} = \Big[y_0,\ y_1,\ \cdots,\ y_{M-1},\ 0,\ \cdots,\ 0 \Big]^{\mathrm{T}} . \tag{4.20}$$

The column vector $\mathbf{w}_{\mathrm{eq}}$ is of length L and contains the equalizer filter coefficients $\mathbf{h}_{\mathrm{eq}}$ in reverse order such that $w_l = h_{(L-1)-l}$ where $l = 0, 1 \cdots, L - 1$.

Eq. (4.19) implements a convolution between the channel characterization data $\mathbf{x} = [x_0, x_1, \cdots, x_{\mathrm{M}-1}]^{\mathrm{T}}$ and the unknown weight vector $\mathbf{w}_{\mathrm{eq}} = [h_{L-1}, h_{L-2}, \cdots, h_0]^{\mathrm{T}}$.

Thus, the matrix $\mathbf{X}$ is of size $N \times L$ and contains time domain samples of the characterization data for the given channel such that each row consists of L sequential samples. To capture the full convolution, $\mathbf{x}_{\mathrm{ch}}$ is zero-padded with $L-1$ zeros at the beginning and end of the array, as given by

$$\mathbf{X} = \begin{bmatrix} 0 & 0 & \cdots & 0 & x_0 \\ 0 & 0 & \cdots & x_0 & x_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & x_{L-4} & x_{L-3} \\ 0 & x_0 & \cdots & x_{L-3} & x_{L-2} \\ x_0 & x_1 & \cdots & x_{L-2} & x_{L-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{M-L} & x_{M-L+1} & \cdots & x_{M-2} & x_{M-1} \\ x_{M-L+1} & x_{M-L+2} & \cdots & x_{M-1} & 0 \\ x_{M-L+2} & x_{M-L+3} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{M-2} & x_{M-1} & \cdots & 0 & 0 \\ x_{M-1} & 0 & \cdots & 0 & 0 \end{bmatrix} . \tag{4.21}$$

As Eq. (4.19) is of the same form as Eq. (4.12), the ordinary least squares method, given in Eq. (4.13), can be used to solve for $\mathbf{w}$. The solution is given by

$$\mathbf{w}_{\mathrm{eq}} = (\mathbf{X}^H \mathbf{X})^{-1} \mathbf{X}^H \mathbf{y}_{\mathrm{des}} \ . \tag{4.22}$$

Table 4.3 summarizes the relevant variables for frequency domain synthesis.

Table 4.4: Time Domain Synthesis Variables

Variable	Description
$\mathbf{y}_{\mathrm{des}}$	Time Domain Reference Waveform Vector
$\mathbf{X}$	Time Domain Channel Response Matrix
$\mathbf{w}_{\mathrm{eq}}$	Reversed Equalizer Coefficient Vector

Prescribed Group Delay Time alignment between the reference waveform and equalizer output, briefly discussed in [70], is of particular importance in equalizer design, as ignoring the filter's group delay degrades performance considerably. Applying the appropriate delay to the reference waveform ensures that filter synthesis seeks a filter with a practical group delay. This reference waveform delay will be referred to as the equalizer's *prescribed group delay* Δm_{g}, specified in samples and not necessarily an integer. As has been described, a delay can be induced in the time domain through a combination of integer-sample and fractional-sample delays or the frequency domain by adjusting the reference waveform's phase slope. Performance metrics discussed in Section 4.2.1.3 are used to examine the dependence of equalizer performance on prescribed group delay, providing an opportunity for optimization.

In frequency domain synthesis, discussed in Section 4.2.1.2, the reference waveform is incorporated in Eq. (4.9). A delayed desired response is given as $\mathbf{H}_{\mathrm{des,d}} = \frac{\mathbf{H}_{\mathrm{y,d}}}{\mathbf{H}_{\mathrm{x}}}$ where $\mathbf{H}_{\mathrm{y,d}}$ is the frequency domain response of the reference waveform delayed by Δm_{g} samples. Substituting this into Eq. (4.18) yields the corresponding equalizer $\mathbf{h}_{\mathrm{eq,d}}$. Similarly, a delayed time domain reference waveform vector $\mathbf{y}_{\mathrm{des,d}}$ is substituted into Eq. (4.22) to determine the corresponding weight vector $\mathbf{w}_{\mathrm{eq,d}}$.

The governing equations for frequency and time domain equalizer synthesis, given in Eqs. (4.18) and (4.22), can be expanded for simultaneous processing of several delays. A prescribed group delay range of $\Delta m_{\mathrm{g}} \in \{0, L\}$ samples specified with some fractional-sample step size $\delta_{m_{\mathrm{g}}}$ provides $D = \frac{L}{\delta_{m_{\mathrm{g}}}} + 1$ delay steps to explore. The reference waveform vectors associated with each delay are concatenated to form a reference matrix as given by

$$\mathbf{H}_{\mathrm{des},\Delta} \triangleq \begin{bmatrix} \mathbf{H}_{\mathrm{des,0}} & \mathbf{H}_{\mathrm{des,1}} & \cdots & \mathbf{H}_{\mathrm{des,D}} \end{bmatrix} \tag{4.23}$$

for frequency domain synthesis and

$$\mathbf{Y}_{\text{des},\Delta} \triangleq \begin{bmatrix} \mathbf{y}_{\text{des},0} & \mathbf{y}_{\text{des},1} & \cdots & \mathbf{y}_{\text{des,D}} \end{bmatrix} \tag{4.24}$$

for time domain synthesis. Substituting Eqs. (4.23) and (4.24) into Eqs. (4.18) and (4.22), respectively, yields

$$\mathbf{h}_{\text{eq},\Delta} = (\mathbf{C}_{\text{o}}^{H}\mathbf{W}\mathbf{C}_{\text{o}})^{-1}\mathbf{C}_{\text{o}}^{H}\mathbf{W}\mathbf{H}_{\text{des},\Delta} \tag{4.25}$$

and

$$\mathbf{W}_{\text{eq},\Delta} = (\mathbf{X}^{H}\mathbf{X})^{-1}\mathbf{X}^{H}\mathbf{Y}_{\text{des},\Delta} \ . \tag{4.26}$$

The columns of $\mathbf{h}_{\text{eq},\Delta}$ and $\mathbf{W}_{\text{eq},\Delta}$ contain the sets of filter coefficients corresponding to each prescribed group delay. Recall that columns of $\mathbf{W}_{\text{eq},\Delta}$ contain the equalizer coefficients in reverse order.

4.2.1.3 Equalizer Performance

A numerical metric for equalizer performance provides an efficient method for selecting from various equalizer options, whether it be from different synthesis techniques or different prescribed group delays. This section discusses two methods for quantifying equalizer effectiveness, the Channel Pair Cancellation Ratio (CPCR) [80] and the residual tracking error [70].

Channel Pair Cancellation Ratio This metric is defined as the ratio of the equalizer output power to the residual power, as given by

$$\text{CPCR} \triangleq \frac{P_{\mathbf{x}_{\text{out}}}}{P_{\mathbf{x}_{\text{res}}}} \ . \tag{4.27}$$

The signal power in the column vector $\mathbf{s}$ of length N can be computed by normalizing the inner product $\mathbf{s}^H\mathbf{s}$ by the vector length, described by

$$P_{\mathbf{s}} \triangleq \frac{\mathbf{s}^H\mathbf{s}}{N} \ . \tag{4.28}$$

The output signal vector $\mathbf{x}_{\text{out}}$ is the result of filtering the channel characterization data with the synthesized equalizer, as given by $\mathbf{x}_{\text{out}} = \mathbf{x}_{\text{ch}} * \mathbf{h}_{\text{eq}}$, where the symbol $*$ denotes linear convolution. The residual signal vector $\mathbf{x}_{\text{res}}$ is defined as the difference between the desired signal vector and output signal vector, as given by $\mathbf{x}_{\text{res}} = \mathbf{y}_{\text{des}} - \mathbf{x}_{\text{out}}$. Thus, the discrete signal CPCR is given by

$$\text{CPCR} = \frac{{\mathbf{x}_{\text{out}}}^H\mathbf{x}_{\text{out}}}{{\mathbf{x}_{\text{res}}}^H\mathbf{x}_{\text{res}}} \ . \tag{4.29}$$

Low residual power indicates better agreement between the output signal vector and the reference signal vector resulting in a higher CPCR.

Residual Tracking Error Alternatively, the residual tracking error, which can be determined using QR decomposition [55], also provides an indicator of equalizer performance. QR decomposition can be used to efficiently compute the least squares solution in real-time embedded systems. We redefine the least squares equations given in Eqs. (4.15) and (4.19) in terms of residual error matrices $\mathbf{E}$ and update them to include the reference waveform matrices given in Eqs. (4.23) and (4.24) such that

$$\mathbf{E}_{\text{freq}} = \mathbf{W}^{1/2}\mathbf{H}_{\text{des},\Delta} - \mathbf{W}^{1/2}\mathbf{C}_{\text{o}}\mathbf{h}_{\text{eq},\Delta} \tag{4.30}$$

and

$$\mathbf{E}_{\text{time}} = \mathbf{Y}_{\text{des},\Delta} - \mathbf{X}\mathbf{W}_{\text{eq},\Delta} \ . \tag{4.31}$$

To begin, define, for a given system of equations $\mathbf{E} = \mathbf{Y} - \mathbf{X}\mathbf{W}$, the extended matrix $\mathbf{Z} = \begin{bmatrix} \mathbf{X} & \mathbf{Y} \end{bmatrix}$. QR decomposition decomposes $\mathbf{Z}$ into a unitary matrix $\mathbf{Q}$ and an upper triangle matrix $\mathbf{R}$ such that

$$\mathbf{Z}^H\mathbf{Z} = \begin{bmatrix} \mathbf{X}^H\mathbf{X} & \mathbf{X}^H\mathbf{Y} \\ \mathbf{Y}^H\mathbf{X} & \mathbf{Y}^H\mathbf{Y} \end{bmatrix} = \mathbf{R}^H\mathbf{Q}^H\mathbf{Q}\mathbf{R} = \mathbf{R}^H\mathbf{R} \; . \tag{4.32}$$

Several methods exist for computing the QR decomposition, such as the Householder transformations [81], the Gram-Schmidt process [82], or Givens rotations [83], which are useful for implementing QR decomposition in an embedded system. The matrix $\mathbf{R}$ can be partitioned as described by

$$\mathbf{R} = \begin{bmatrix} \mathbf{U} & \mathbf{V} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \; . \tag{4.33}$$

Given that $\mathbf{R}$ is an upper triangle matrix, the matrices $\mathbf{U}$ and $\mathbf{T}$ are also both upper triangle. Thus, computing the matrix inverse $\mathbf{U}^{-1}$, necessary to determine the least squares solution given by $\mathbf{W} = \mathbf{U}^{-1}\mathbf{V}$, is of considerably lower complexity. The smallest residual error, which can be computed for the d^{th} delay as given by

$$|E_{dd}|^2 = \sum_{c=1}^{d} |T_{cd}|^2 \; , \tag{4.34}$$

indicates the most effective equalizer. Table 4.5 provides the dimensions for the relevant matrices for a filter of length L, zero-padded time domain vectors of length N, and D prescribed group delays.

Table 4.5: QR Decomposition Matrix Dimensions

Matrix	Dimension	Matrix	Dimension
$\mathbf{X}$	$N \times L$	$\mathbf{R}$	$(L+D) \times (L+D)$
$\mathbf{W}$	$L \times D$	$\mathbf{U}$	$L \times L$
$\mathbf{Y}$	$N \times D$	$\mathbf{V}$	$L \times D$
$\mathbf{Z}$	$(L+D) \times (L+D)$	$\mathbf{T}$	$D \times D$

4.2.1.4 TTD Fractional-Sample Delay Integration

As described in [84], wideband beamforming is achieved through the use of true time delay (TTD) beamsteering, which can be digitally implemented through a combination of integer-sample delays and fractional-sample delays [71]. Thus, a channel specific fractional-sample delay filter bank which compensates the particular channel frequency response enables calibrated wideband beamforming. Two methods are considered for fractional-sample delay filter integration with the equalizer.

One option requires independent synthesis of the channel equalizer and fractional-sample delay filter bank and relies on the cascade property of FIR filters [36]. Thus, equalizer filter coefficients of length L_{eq} for a given channel are convolved with each set of coefficients of length L_{del} in the fractional-sample delay filter bank. This generates a bank of channel specific equalized fractional-delay filters of length $L_{\mathrm{eq}} + L_{\mathrm{del}} - 1$, as given by

$$\mathbf{h}_{\mathrm{m,d}} = \mathbf{h}_{\mathrm{eq,m}} * \mathbf{h}_{\mathrm{del,d}} \ , \tag{4.35}$$

where $\mathbf{h}_{\mathrm{eq,m}}$ and $\mathbf{h}_{\mathrm{del,d}}$ represent the filter coefficients for the m^{th} channel equalizer and d^{th} fractional-sample delay respectively. This method requires allocation of the limited filter resources between the equalizer filter and the fractional-sample delay filter.

Alternatively, by adjusting the prescribed group delay, the fractional-sample delay can be synthesized directly into the equalizer. By selecting a block of prescribed group delays spanning an integer-sample at the appropriate fractional-sample resolution, one synthesizes a calibrated fractional-sample delay filter bank. This mitigates the need to distribute limited system resources between independent requirements. Equalizer synthesis results and measured chamber results discussed in Sections 4.2.2 and 4.2.3, respectively, utilize this prescribed group delay approach.

4.2.2 Calibrated RFSoC Testbed

Figure 4.19 shows the calibrated testbed mounted in OU's far-field anechoic chamber. For reference mixed-signal processing is accomplished on Pentek hardware which houses a Xilinx RFSoC and interfaces with a wideband Vivaldi antenna array. Received signals are sampled in the second Nyquist zone by ADCs at 4 GSPS and digitally downconverted to complex baseband where they are decimated by a factor or two, processed, and combined. The combined signal is interpolated by a factor of two, digitally upconverted, and sourced by the DAC at 4 GSPS, whose analog output is measured in the anechoic chamber. It is important to note that internal issues with the RFSoC hardware rendered channels 7 and 8 unusable for equalization. The RFSoC equalizer design presented herein supports equalization and beamforming for all 8 channels, but only channels 1 to 6 interface to the Vivaldi aperture for measurements presented in Section 4.2. The two edge elements are terminated.

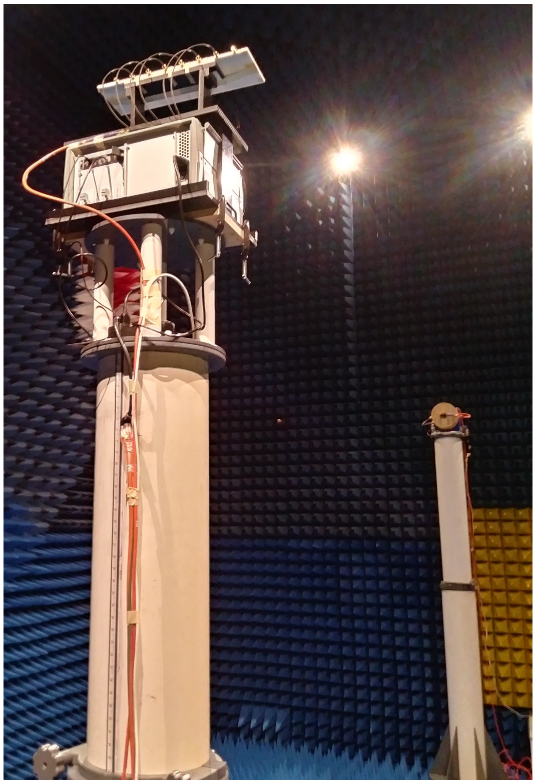

Figure 4.19: Research Testbed Mounted in our Far-Field Anechoic Chamber

4.2.2.1 Channel Characterization

As mentioned in Section 4.2.1.1, the first step in equalizer design is accurate channel characterization. Measured characterization data for channels 1 to 6 is shown in Figure 4.20. The top row shows the real and imaginary time domain data on the left and right, respectively. The black traces show the Hann window function applied to the first and last 2.5% of the time domain data prior to processing.

Frequency domain data is shown in the bottom row, with the magnitude and phase on the left and right, respectively. To emphasize the dispersive channel effects, phase data is presented as relative to the reference waveform. Specifically, traces show the difference between the unwrapped phase of the channel response and that of the reference waveform. Phase offsets are adjusted to 0° at $f = 0\ \frac{\text{cyc}}{\text{samp}}$. Given that the reference waveform spans 90% of the Nyquist zone, phase behavior outside the reference bandwidth of $-0.45 \leq f \leq 0.45$ is inconsequential given its low signal energy.

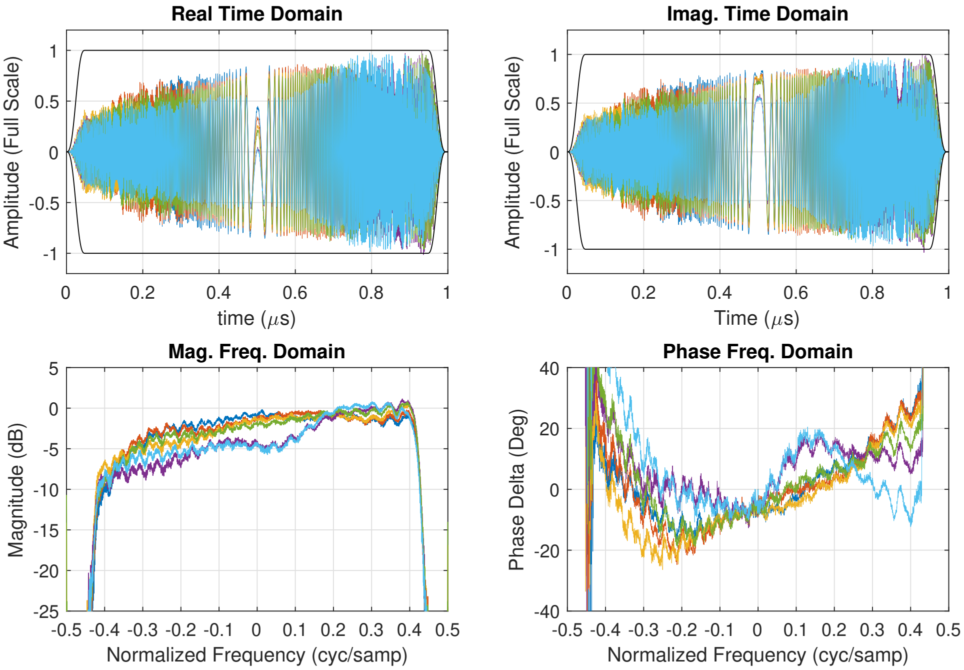

Figure 4.20: Channel 1–6 Characterization Data - Top Left: Real Time Domain, Top Right: Imaginary Time Domain, Bottom Left: Magnitude Frequency Domain, Bottom Right: Wrapped Phase Frequency Domain

The appreciable frequency dependence in both magnitude and phase of each channel is readily apparent. Notably, dispersive effects can be seen in the phase plots necessitating higher fidelity compensation as compared to narrowband adjustments. Also apparent is a frequency-dependent ripple of about 1–2 dB and 10° due to mismatched interfaces of the commercial-off-the-shelf (COTS) hardware. As will be shown, equalizer fidelity is not sufficient to correct for this ripple behavior, at least not as constrained by the RFSoC resources.

4.2.2.2 Channel Equalization

Channel characterization data shown in Figure 4.20 was processed using both frequency and time domain filter synthesis equations discussed in Section 4.2.1. RFSoC FPGA resources used in this demonstration enabled time closure for a 15-tap filter bank with $\frac{1}{8}$-sample step size without much optimization. Resource utilization is discussed further in Section 4.2.2.3.

Equalizer frequency response data for all eight channels are provided in Figure 4.21. Although only channels 1 to 6 are utilized in the measurements provided in Section 4.2.3, data for channels 7 and 8 are provided here for reference. As both these channels show appreciably degraded performance in both magnitude and phase, there were not utilized in the measured results. In this figure, magnitude is shown in the left and relative unwrapped phase data is shown on the right. Dark blue traces correspond to the characterization data given in Figure 4.20, corresponding to $\mathbf{H}_{\mathrm{x}}$ in Eq. (4.9) and $\mathbf{X}$ in Eq. (4.19). The red-orange traces show the desired equalizer frequency response $\mathbf{H}_{\mathrm{des}}$. The yellow and green traces show the equalizer frequency response from frequency and time domain synthesis respectively. Note the strong agreement between the two synthesis techniques, as well as with the desired frequency response $\mathbf{H}_{\mathrm{des}}$.

The purple and cyan traces show the result of filtering channel characterization data with each equalizer. A flat magnitude and phase response shows good agreement

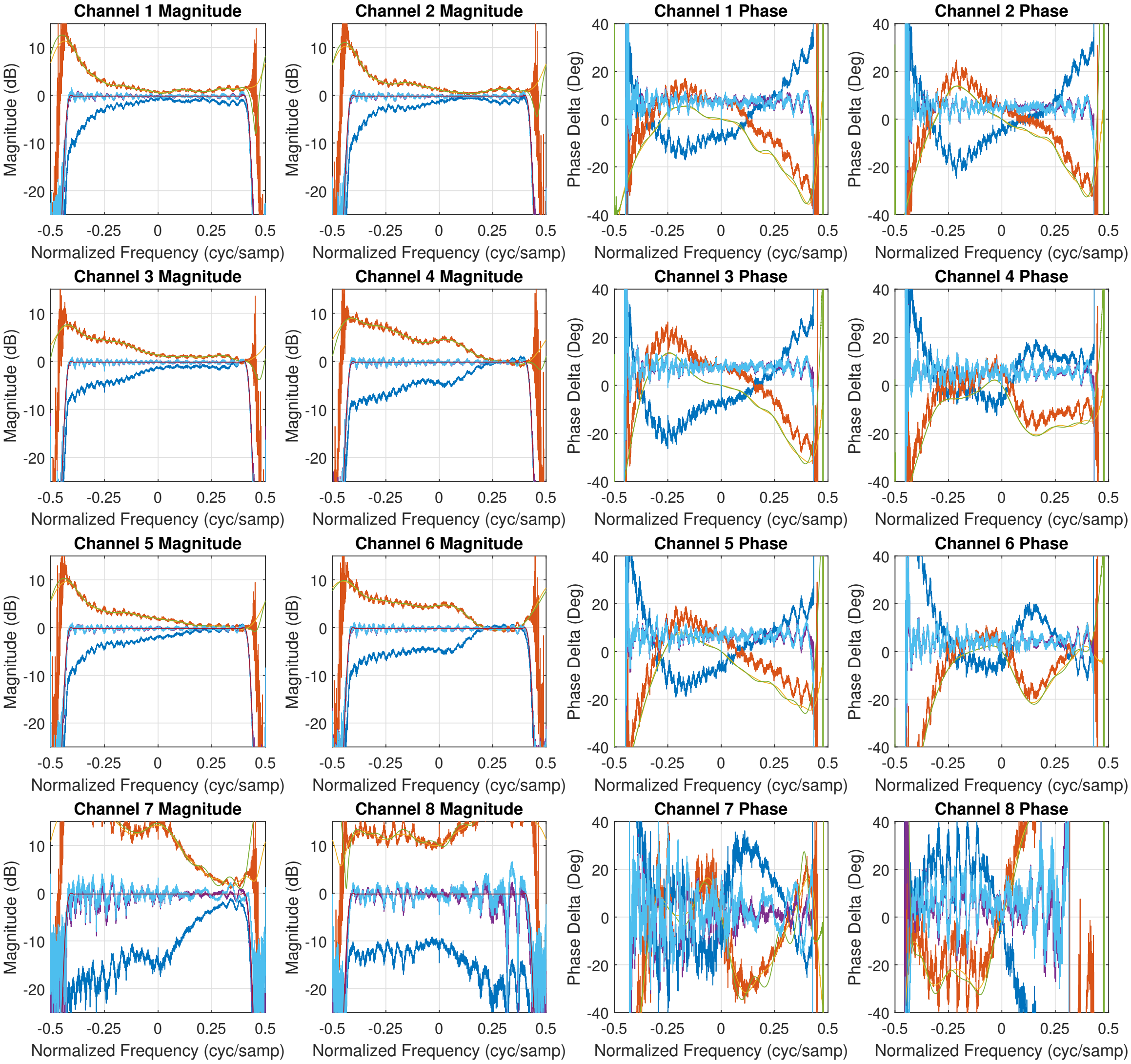

Figure 4.21: Channel Equalizer Magnitude and Unwrapped Phase Delta: Dark Blue - Channel Characterization Data, Red-Orange: Desired Filter Response, Yellow: Frequency Domain Synthesis Filter Response, Green: Time Domain Synthesis Filter Response, Purple: Frequency Domain Synthesis Filter Output, Cyan: Time Domain Synthesis Filter Output, Black: Window Function for Frequency Domain Synthesis

between the equalizer outputs and the reference waveform, although the frequency ripple present within the channel characterization data is also present at the equalizer outputs.

Equalizer Latency Optimization As mentioned in Section 4.2.1.3, equalizer effectiveness is a function of its latency, which can be adjusted by delaying the reference waveform prior to filter synthesis. Figure 4.22 shows the CPCRs and residual tracking errors, on the top and bottom, respectively, as a function of prescribed group delay Δm_{g}. Equalizers were synthesized using Eqs. (4.25) and (4.26) for a series of prescribed group delays ranging from 0 samples to the instantiated filter length $L = 15$ at a step size of $\frac{1}{8}$-sample. Frequency domain synthesis results are given in the left column while time domain synthesis results are given in the right column. The optimum delay appears around $\Delta m_{\mathrm{g}} \approx 8$ samples, but varies from channel to channel and between each synthesis technique. A delay of 0 samples results in particularly poor performance.

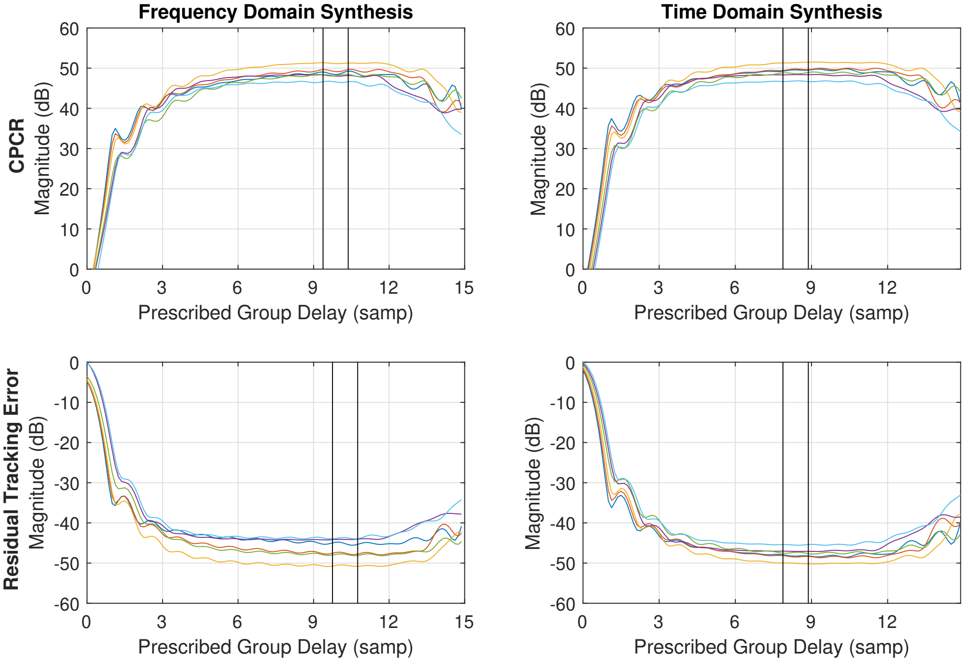

Figure 4.22: Equalizer Performance: Top Row - Frequency Domain Synthesis, Bottom Row - Time Domain Synthesis, Left Column - CPCR, Right Column - Residual Tracking Error

When incorporating the fractional-sample delay filter bank with $\frac{1}{8}$-sample step size, one desires the optimum block of 8 sets of coefficients, providing the necessary fractional-sample delay range. Because all six channels were fairly well-behaved, the block with the highest total CPCR, or the lowest residual tracking error, was selected. Namely, CPCR for a each channel was averaged over each potential block of 8 delays and the block for which the cumulative mean CPCR, summed over all channels, was greatest was selected. The resulting delay blocks, denoted by the vertical bands in Figure 4.22, different depending on synthesis technique. Also of note is the slight variation between frequency domain synthesis CPCR and residual tracking error. As this difference is minute, the CPCR block was selected for implementation.

The channel 6 filter bank frequency response for both synthesis techniques is shown in Figure 4.23, along with the equalizer output for each delay. Phase data shows the increased group delay for each delay step while magnitude data shows little variation within the band of interest. Agreement between the two synthesis techniques can be seen most easily in the pairs of phase data for each step. Additionally, there is very strong agreement between equalized output data for each filter bank as all 16 outputs virtually eclipse each other.

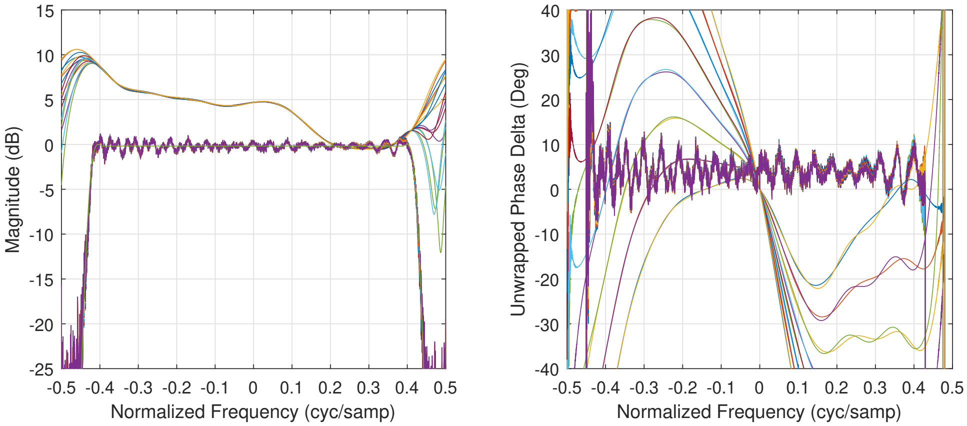

Figure 4.23: Equalizer/Fractional-Sample Delay Filter Bank

Equalizer Filter Length Equalizer performance is also dependent on filter length, which is often constrained by system resources. The CPCR for channel 3 with various filter lengths is shown in Figure 4.24. The left plot shows CPCR versus prescribed group delay Δm_{g}, similar to the top row of Figure 4.22. Each curve represents a different filter length and shows the CPCR with $\Delta m_{\mathrm{g}} \in \{0 - L\}$ where L is the current filter length. The right plot shows the maximum CPCR for each filter length. Although dependent on the given channel characteristics, a point of diminishing returns is shown around a filter length of 10–15 taps. Here, the CPCR response flattens out, plateauing near 52 dB. It is expected that a more dispersive channel will require a longer filter to achieve this point of diminishing returns.

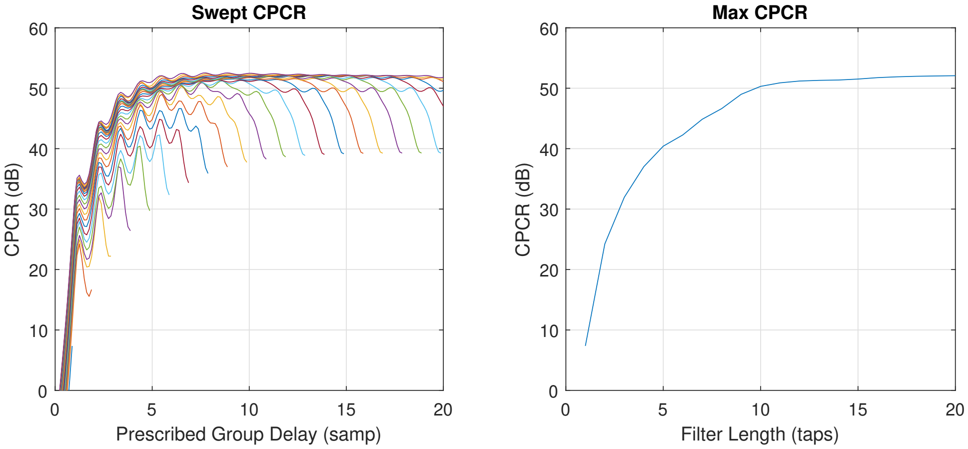

Figure 4.24: Channel 3 CPCR for Various Filter Lengths: Left - CPCR vs. Prescribed Group Delay, Right CPCR vs. Filter Length

For this specific characterization data, simulations show an improvement in CPCR of about 5 and 10 dB around 40 taps and 70 taps respectively. At these lengths, filter fidelity becomes capable of compensating the frequency ripple from system mismatches. This is not explored further for two reasons. First, the required filter lengths are impractical under our resource constraints. Second, the ripple in the channel characterization data is due to mismatches between the RFSoC DAC and ADCs under the characterization configuration outlined in Figure 4.18. For chamber measurements, the ADCs interface with the aperture elements, resulting in different mismatch characteristics.

Time/Frequency Domain Synthesis Comparison Both frequency domain synthesis and time domain synthesis yield strong results. Equalizer output results using frequency domain synthesis are shown in Figure 4.25 and output results using time domain synthesis are shown in Figure 4.26. Both figures show effective equalization of the channel characterization data given in Figure 4.20. Although there is little discernable difference the plots of Figures 4.25 and 4.26, the difference in CPCR between frequency domain synthesis and time domain synthesis is given in Figure 4.27.

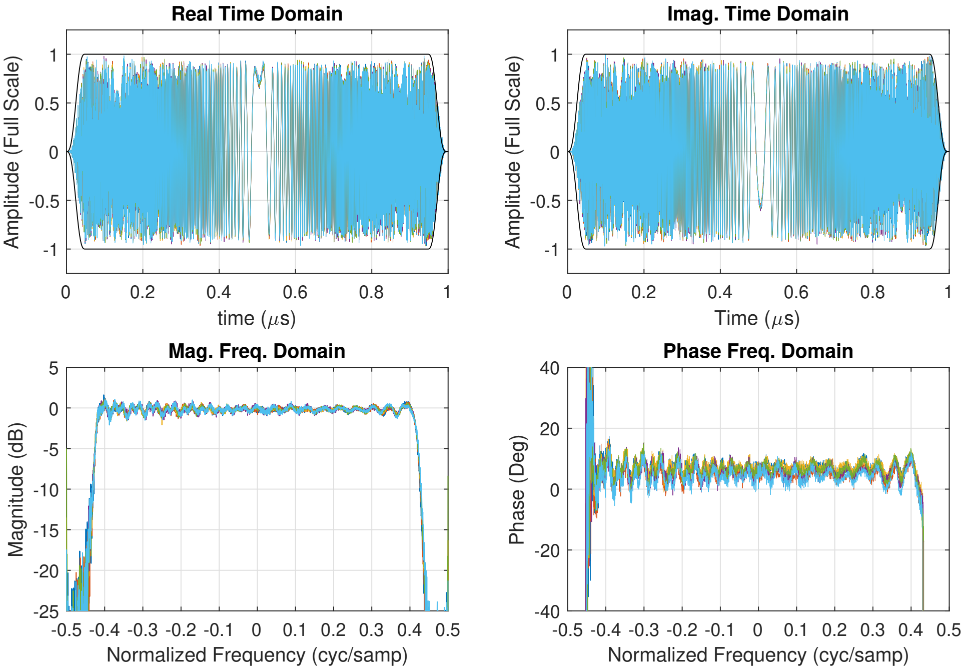

Figure 4.25: Channel 1–6 Equalizer Response Using Frequency Domain Synthesis - Top Left: Real Time Domain, Top Right: Imaginary Time Domain, Bottom Left: Magnitude Frequency Domain, Bottom Right: Wrapped Phase Frequency Domain

Negative traces indicate superior performance from time domain synthesis than from frequency domain synthesis, which is seen for any delay greater than one sample. It should be noted that this difference is quite small and almost vanishes near the optimum prescribed group delay of $\Delta m_{\mathrm{g}} \approx 9$ samples. Given that the difference between the two synthesis techniques decreases with increasing delay Δm_{g}, instances for which filter length is more limited may see a slight benefit from utilizing time domain synthesis over frequency domain synthesis.

Time Domain Synthesis Down-Sampling The computational burden of solving the time domain synthesis equation given in Eq. (4.26) can be reduced by downsampling the rows of $\mathbf{Y}_{\mathrm{des},\Delta}$ and $\mathbf{X}$. Although this may not be pertinent for the equalizer

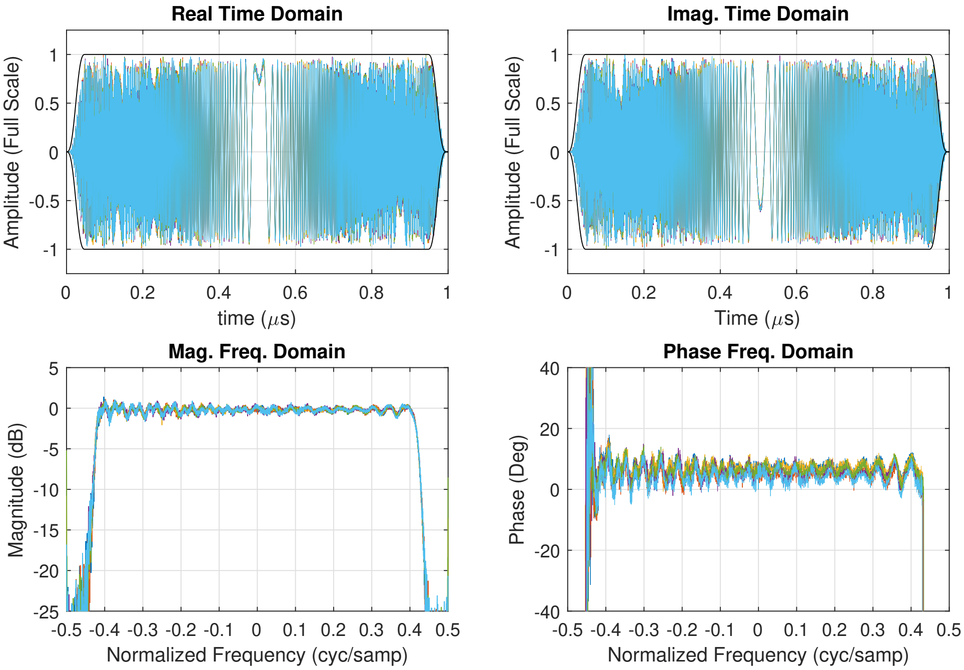

Figure 4.26: Channel 1–6 Equalizer Response Using Time Domain Synthesis - Top Left: Real Time Domain, Top Right: Imaginary Time Domain, Bottom Left: Magnitude Frequency Domain, Bottom Right: Wrapped Phase Frequency Domain

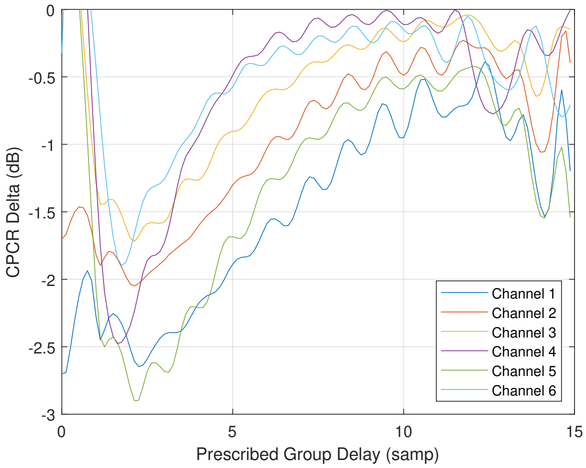

Figure 4.27: CPCR Delta between Frequency Domain and Time Domain Synthesis

design herein, it is relevant for embedded system design in which processing power is limited. This was the approach of the authors in [70] in which they demonstrate adaptive equalization through real-time implementation of time domain synthesis.

Note that for a reference matrix $\mathbf{Y}_{\text{des},\Delta}$ in which the prescribed group delay $\Delta m_{\text{g}} \in [0, 1, \cdots, L-1]$, the distribution of time domain samples in the reference matrix follows the same form as the channel characterization matrix $\mathbf{X}$. By taking the L^{th} row of each of these matrices, beginning with row L, Eq. (4.26) reduces to that described in [69] and [70], where

$$\mathbf{X} = \begin{bmatrix} x_0 & x_1 & \cdots & x_{L-1} \\ x_L & x_{L+1} & \cdots & x_{2L-1} \\ \vdots & \vdots & \ddots & \vdots \end{bmatrix} \tag{4.36}$$

and

$$\mathbf{Y}_{\text{des},\Delta} = \begin{bmatrix} y_0 & y_1 & \cdots & y_{L-1} \\ y_L & y_{L+1} & \cdots & y_{2L-1} \\ \vdots & \vdots & \ddots & \vdots \end{bmatrix} . \tag{4.37}$$

A comparison of the downsampled equalizer to the implemented equalizer reveals a small reduction in CPCR. However, as long as the number of rows M remains such $M > 5L$, equalizer performance will be nearly optimum, as known within the classic RMB algorithm [51]. In this case, downsampling the rows such that $M = 5L$ reduced CPCR by less than 1 dB. It should also be noted that the orientation of columns in $\mathbf{Y}_{\text{des},\Delta}$ places a delay of $L-1$ samples in the first column and a delay of 0 samples on the right. This is the reverse of the convention employed in Eq. (4.24) in which the smallest delay is in the first column.

4.2.2.3 Complex Coefficient Filter Implementation

In the uncalibrated beamformer, individual channel compensation relies on a pair of real-valued fractional-sample delay filter instantiations, each operating independently on the I or Q complex baseband signal samples such that

$$\mathbf{x}_{\text{out,r}} = \mathbf{h}_{\text{ttd}} * \mathbf{x}_{\text{ch,r}} \tag{4.38}$$

and

$$\mathbf{x}_{\mathrm{out,i}} = \mathbf{h}_{\mathrm{ttd}} * \mathbf{x}_{\mathrm{ch,i}} \ , \tag{4.39}$$

where $\mathbf{x}_{\mathrm{ch,r}}$ and $\mathbf{x}_{\mathrm{ch,i}}$ represent the I and Q data streams respectively. Given that a channel equalizer operating on complex baseband signal samples generally does not have conjugate symmetry, a complex-valued coefficient filter is required.

This can be implemented using four real-valued convolutions, as given by

$$\mathbf{x}_1 = \mathbf{h}_{\mathrm{r}} * \mathbf{x}_{\mathrm{ch,r}} \ , \tag{4.40}$$

$$\mathbf{x}_2 = \mathbf{h}_{\mathrm{i}} * \mathbf{x}_{\mathrm{ch,i}} \ , \tag{4.41}$$

$$\mathbf{x}_3 = \mathbf{h}_{\mathrm{r}} * \mathbf{x}_{\mathrm{ch,i}} \ , \tag{4.42}$$

and

$$\mathbf{x}_4 = \mathbf{h}_{\mathrm{i}} * \mathbf{x}_{\mathrm{ch,r}} \ . \tag{4.43}$$

The real and imaginary output components are given by

$$\mathbf{x}_{\mathrm{out,r}} = \mathbf{x}_1 - \mathbf{x}_2 \tag{4.44}$$

and

$$\mathbf{x}_{\mathrm{out,i}} = \mathbf{x}_3 + \mathbf{x}_4 \tag{4.45}$$

respectively. This doubles the number of multipliers required to implement a given filter length, often a constraining system resource. It is important to note that by computing the real and imaginary outputs after the accumulate step in the convolutions given in Eqs. (4.40) - (4.43), FIR convolution resources can be instantiated independent from the each other, simplifying FPGA routing.

By leveraging Karatsuba multiplication [85], a complex coefficient filter can be implemented using three real-valued convolutions, at the cost of some pre- and post-arithmetic, as given by

$$\mathbf{x}_1 = \mathbf{h}_{\mathrm{r}} * \mathbf{x}_{\mathrm{ch,r}} \ , \tag{4.46}$$

$$\mathbf{x}_2 = \mathbf{h}_{\mathrm{i}} * \mathbf{x}_{\mathrm{ch,i}} \ , \tag{4.47}$$

and

$$\mathbf{x}_3 = (\mathbf{h}_{\mathrm{r}} + \mathbf{h}_{\mathrm{i}}) * (\mathbf{x}_{\mathrm{ch,r}} + \mathbf{x}_{\mathrm{ch,i}}) \ . \tag{4.48}$$

The real and imaginary output components are given by

$$\mathbf{x}_{\mathrm{out,r}} = \mathbf{x}_1 - \mathbf{x}_2 \tag{4.49}$$

and

$$\mathbf{x}_{\mathrm{out,i}} = \mathbf{x}_3 - \mathbf{x}_1 - \mathbf{x}_2 \ . \tag{4.50}$$

One must ensure sufficient bit allocation to account for the bit growth associated with the summations in Eqs. (4.48) - (4.50).

Table 4.6: Calibrated Wideband Beamformer FPGA Resource Utilization

Resource	Utilization	Available	%
LUT	78237	425280	18.4
LUTRAM	3199	213600	1.5
FF	124249	850560	14.6
BRAM	62	1080	5.7
DSP	3008	4272	70.4
IO	35	347	10.1
BUFG	7	696	1.0
MMCM	1	8	12.5

Each channel compensation filter utilizes three instantiations of the interleaved filter design given in Section 4.1.2.3, corresponding to Eqs. (4.46) - (4.48). The interleaved implementation requires eight separate filter instantiations to process samples at 2 GSPS at a clock rate of 500 MHz. Thus, the RFSoC DSP requirements for

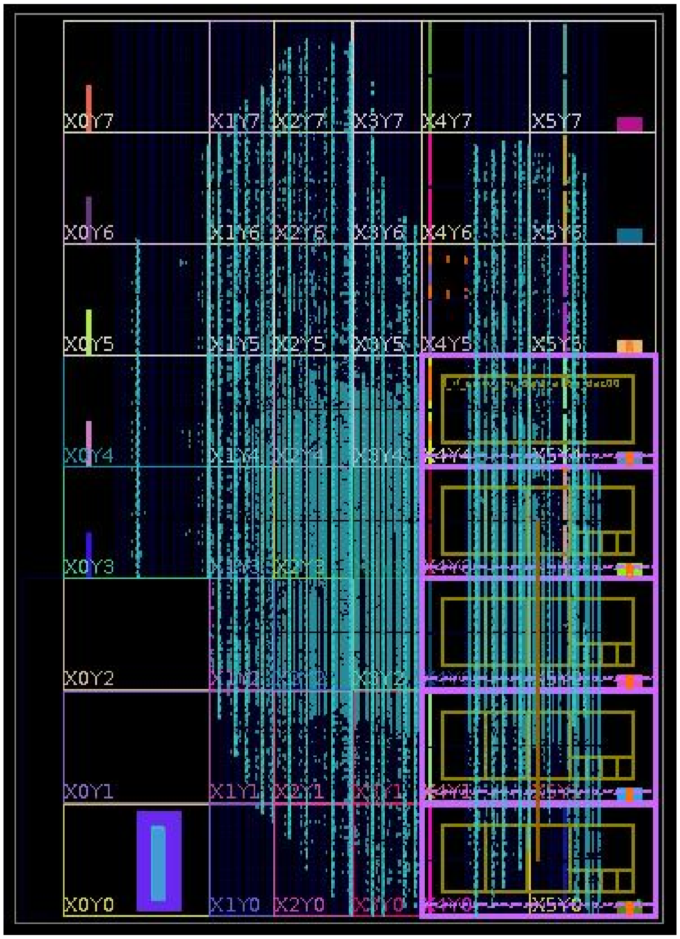

Figure 4.28: Wideband Beamformer Design Layout for the RFSoC

the implementation of a 15-tap complex coefficient FIR beamformer consists of three multipliers per tap, eight filter instantiations per channel, and eight channels totaling 2880 multipliers. The FPGA floorplan is provided in Figure 4.28, showing the layout of the routed FPGA design. Table 4.6 provides resource utilization. Of note is the DSP usage rate of 70.4%. The gain/attenuator block in each channel, shown in the block diagram given in Figure 4.6, utilizes 16 DSPs in order to simultaneously process eight IQ samples. This results in a total of 3008 DSPs for the wideband beamformer.

4.2.3 Calibrated Results

This section discusses simulated and measured data demonstrating equalizer performance. Simulated results provided a high degree of confidence prior to entering the far-field anechoic chamber. Chamber measurements demonstrate broadband performance at various steering angles for uncalibrated and calibrated configurations.

4.2.3.1 Simulation Results Calibrated

Prior to chamber measurements, equalizer performance was verified in Matlab. To generate an estimate of the uncalibrated antenna response, measured channel characterization data was provided to a bit-accurate wideband antenna model. The boresight response is shown in the left plot of Figure 4.29, which shows reasonable agreement with the measured uncalibrated boresight pattern shown in the left plot of Figure 4.31. The calibrated simulation antenna pattern was generated by passing channel characterization data through the synthesized channel equalizer filter bank. The resulting, via time domain synthesis, is shown in the right plot of Figure 4.29. The effects of the characterization data frequency ripple can be seen in this plot near the nulls.

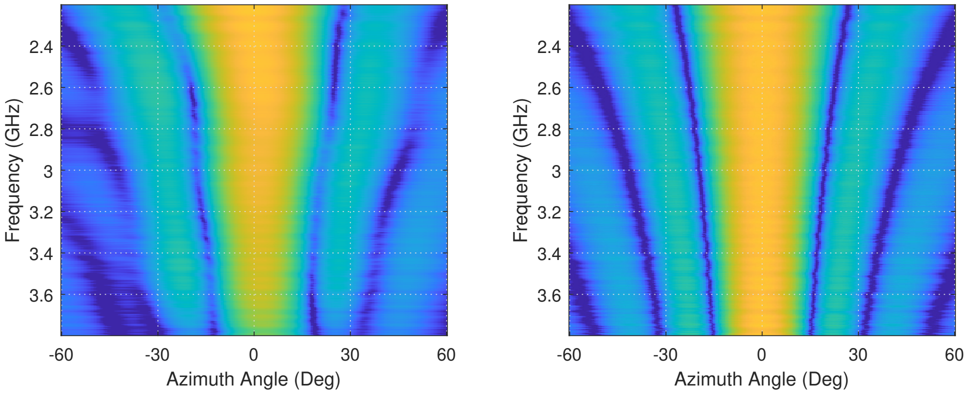

Figure 4.29: Equalizer Simulation Results: Left - Uncalibrated Data, Right - Compensated Data

4.2.3.2 Chamber Measurements

This section presents chamber measurement results for the equalized wideband digital beamformer, shown in OU's far-field anechoic chamber in Figure 4.19. Azimuth patterns consist of ±60° sweeps at 0.5° increments. At each angular step, a frequency sweep was taken from 2.2–3.8 GHz at 10 MHz steps. Patterns were captured for three system configurations: uncalibrated wideband beamforming, equalized wideband beamforming via frequency domain synthesis, equalized wideband beamforming via time domain synthesis. For each configuration, the beamformer was commanded to steer to $\theta_{\text{st}} = 0°$, -15°, -30°, and -45°. Frequency cuts are presented as well.

Amplitude Post-Processing The raw boresight pattern for the 6-element Vivaldi array is shown in the left plot of Figure 4.30. This pattern was captured with the aperture elements connected to a broadband power splitter. Main beam amplitude varies over frequency due to the chamber probe and Vivaldi aperture effects. To compensate, a frequency-dependent response was extracted along the main beam of the raw boresight pattern and its inverse applied as a correction factor across each azimuth angle in the pattern. The result is shown in the right plot of Figure 4.30. All subsequent plots incorporate this correction factor.

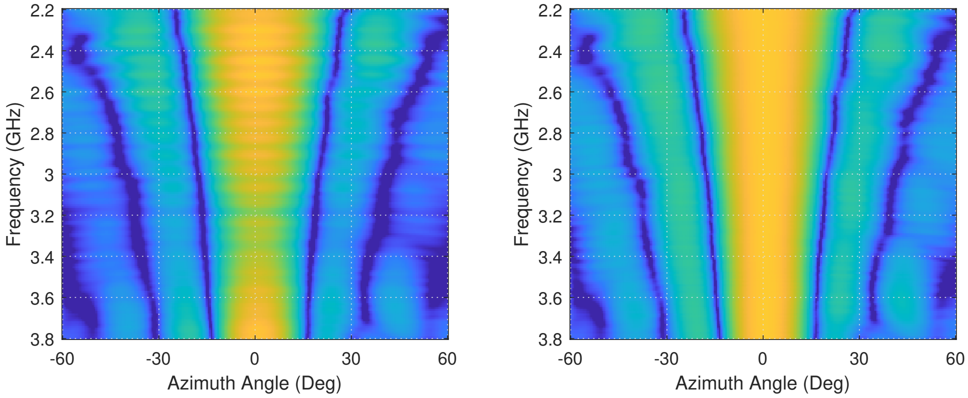

Figure 4.30: Vivaldi Aperture Boresight Pattern: Left - Raw Data, Right - Post-Processed Data

Compensated Results Frequency cuts for the uncalibrated response, equalized response via frequency domain synthesis, and equalized response via time domain synthesis for $\theta_{\text{st}} = 0°$, -15°, -30°, and -45° are shown in Figures 4.32, 4.34, 4.36, and 4.38, respectively. The blue traces show the uncalibrated response and the red-orange and yellow traces show the equalized response for frequency and time domain synthesis respectively. The corresponding two-dimensional patterns showing the antenna pattern as a function of both azimuth angle and frequency are shown in Figures 4.31, 4.33, 4.35, and 4.37. Frequency cuts span the 1.6 GHz bandwidth at 200 MHz steps. Of note is the steering error in the uncalibrated response, which is consistently off by about 3° up until the higher end of the band, when all configurations shift to the right. Sidelobe behavior is consistently improved in the calibrated patterns as compared to the uncalibrated response, indicating an improvement in channel coherency. There is a drop in gain towards the center of the band, despite the amplitude correction just discussed, likely due to mismatch between the aperture elements and RFSoC ADCs, effects which are not accounted for in the channel characterization data or amplitude post-processing.

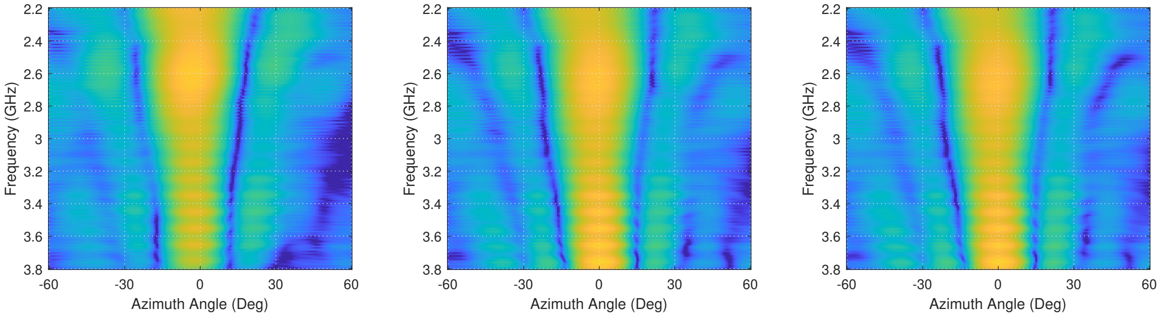

Figure 4.31: Antenna Patterns for $\theta_{\text{st}} = 0°$: Left - Uncalibrated, Center - Frequency Domain Synthesized Equalizer, Right - Time Domain Synthesized Equalizer

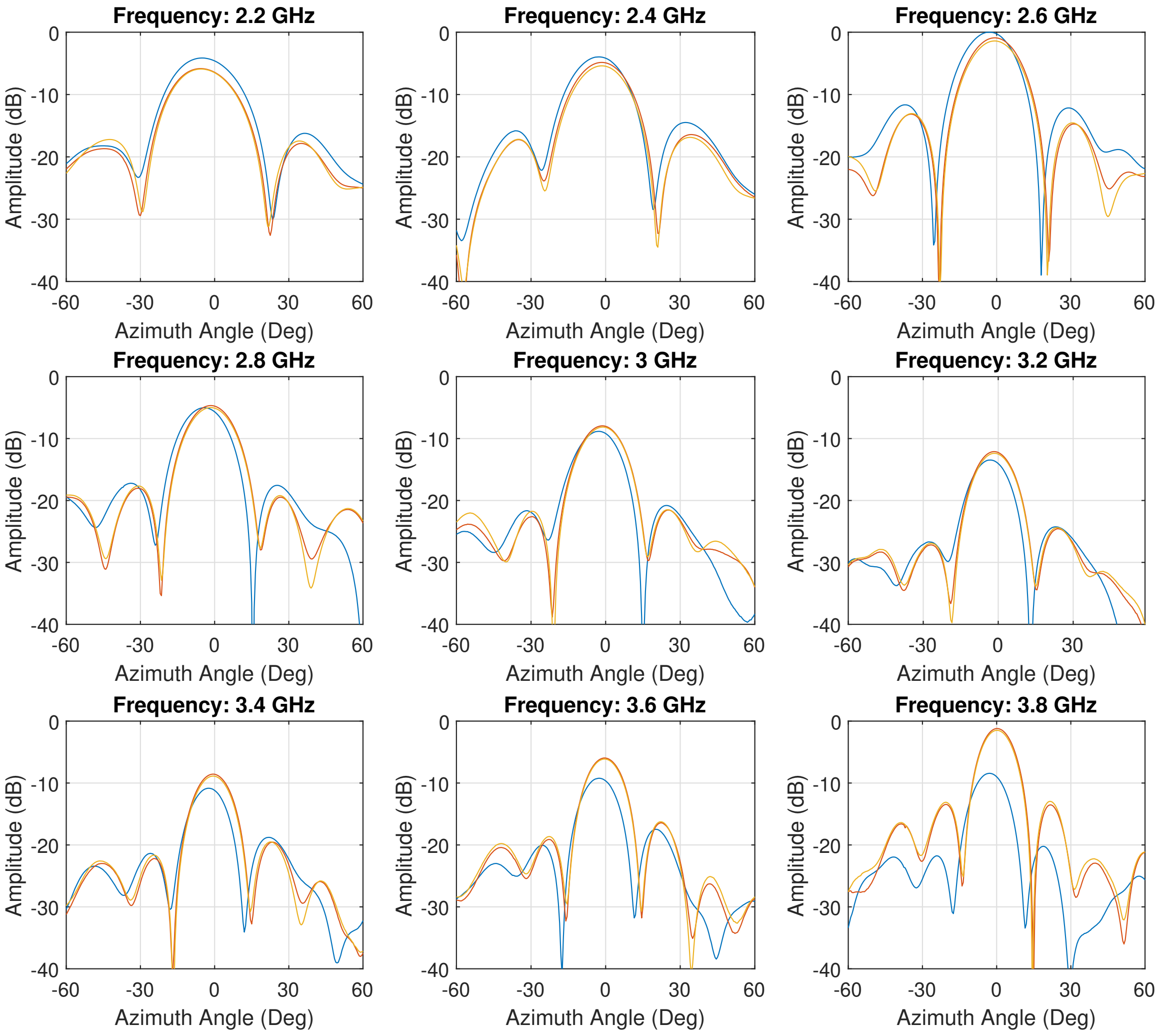

Figure 4.32: Frequency Cuts for $\theta_{\text{st}} = 0°$: Blue - Uncalibrated, Red-Orange: Frequency Domain Synthesis Equalization, Yellow - Time Domain Synthesis Equalization

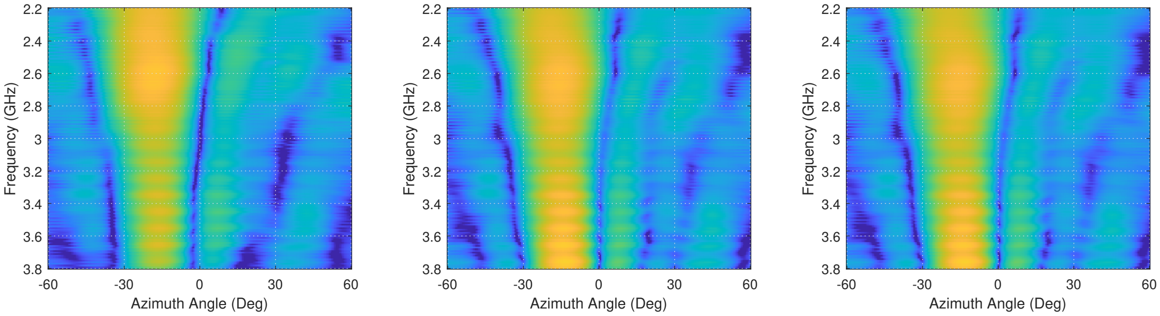

Figure 4.33: Antenna Patterns for $\theta_{\text{st}} = \text{-}15°$: Left - Uncalibrated, Center - Frequency Domain Synthesized Equalizer, Right - Time Domain Synthesized Equalizer

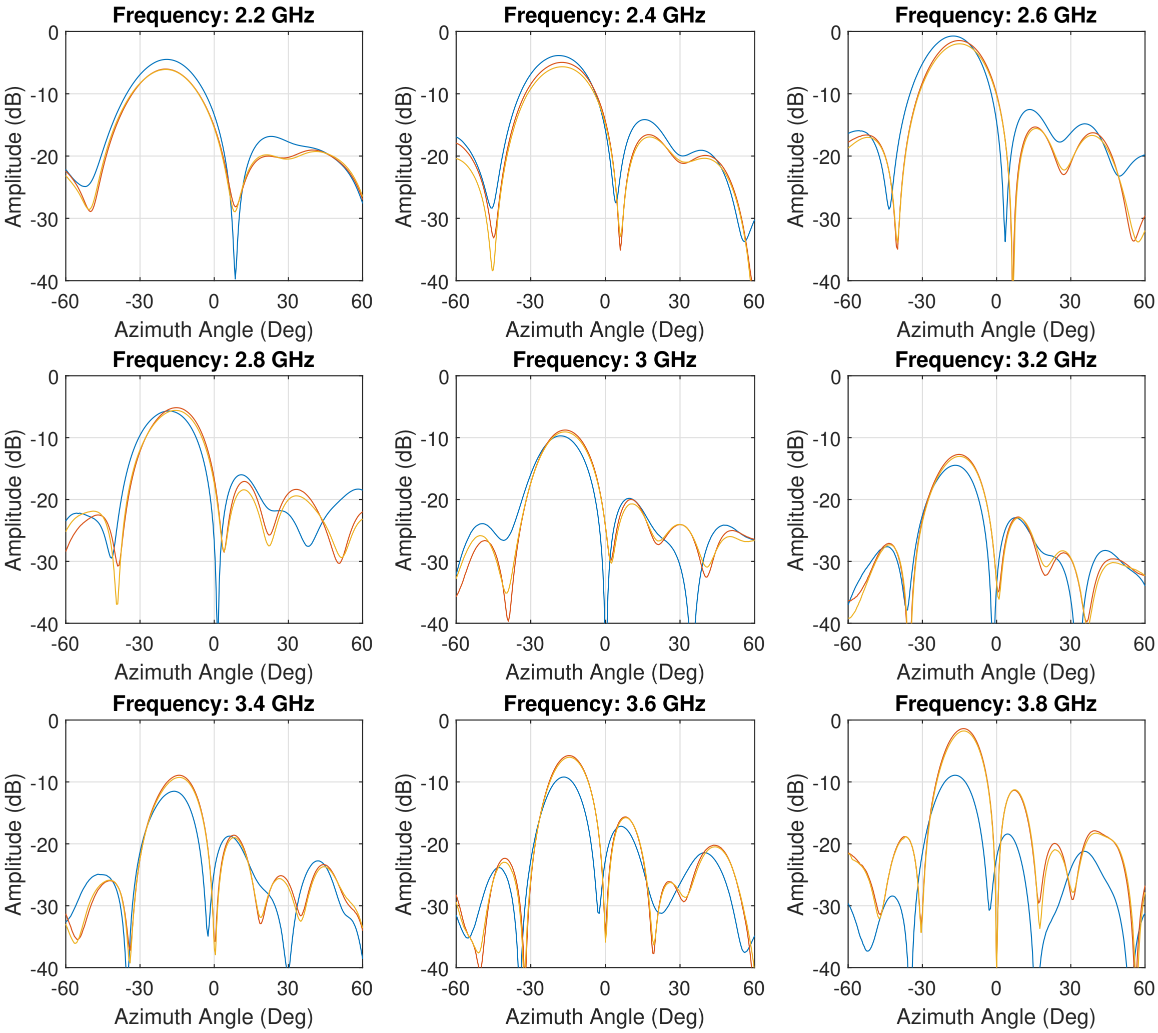

Figure 4.34: Frequency Cuts for $\theta_{\text{st}} = \text{-}15°$: Blue - Uncalibrated, Red-Orange: Frequency Domain Synthesis Equalization, Yellow - Time Domain Synthesis Equalization

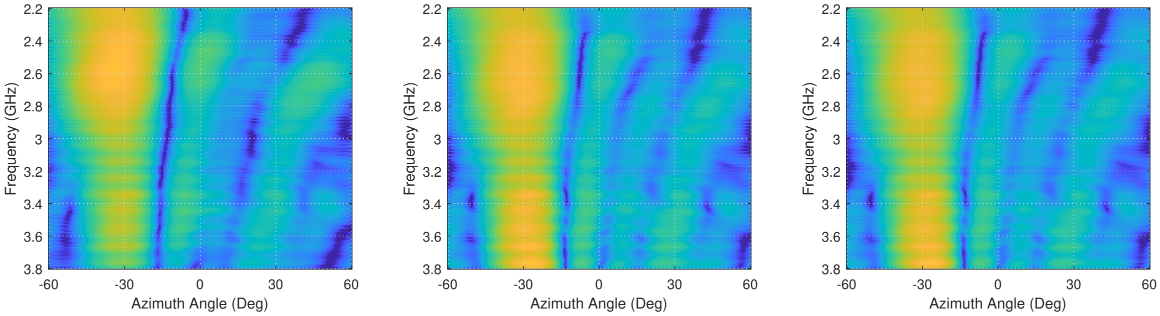

Figure 4.35: Antenna Patterns for $\theta_{\mathrm{st}} = \text{-}30°$: Left - Uncalibrated, Center - Frequency Domain Synthesized Equalizer, Right - Time Domain Synthesized Equalizer

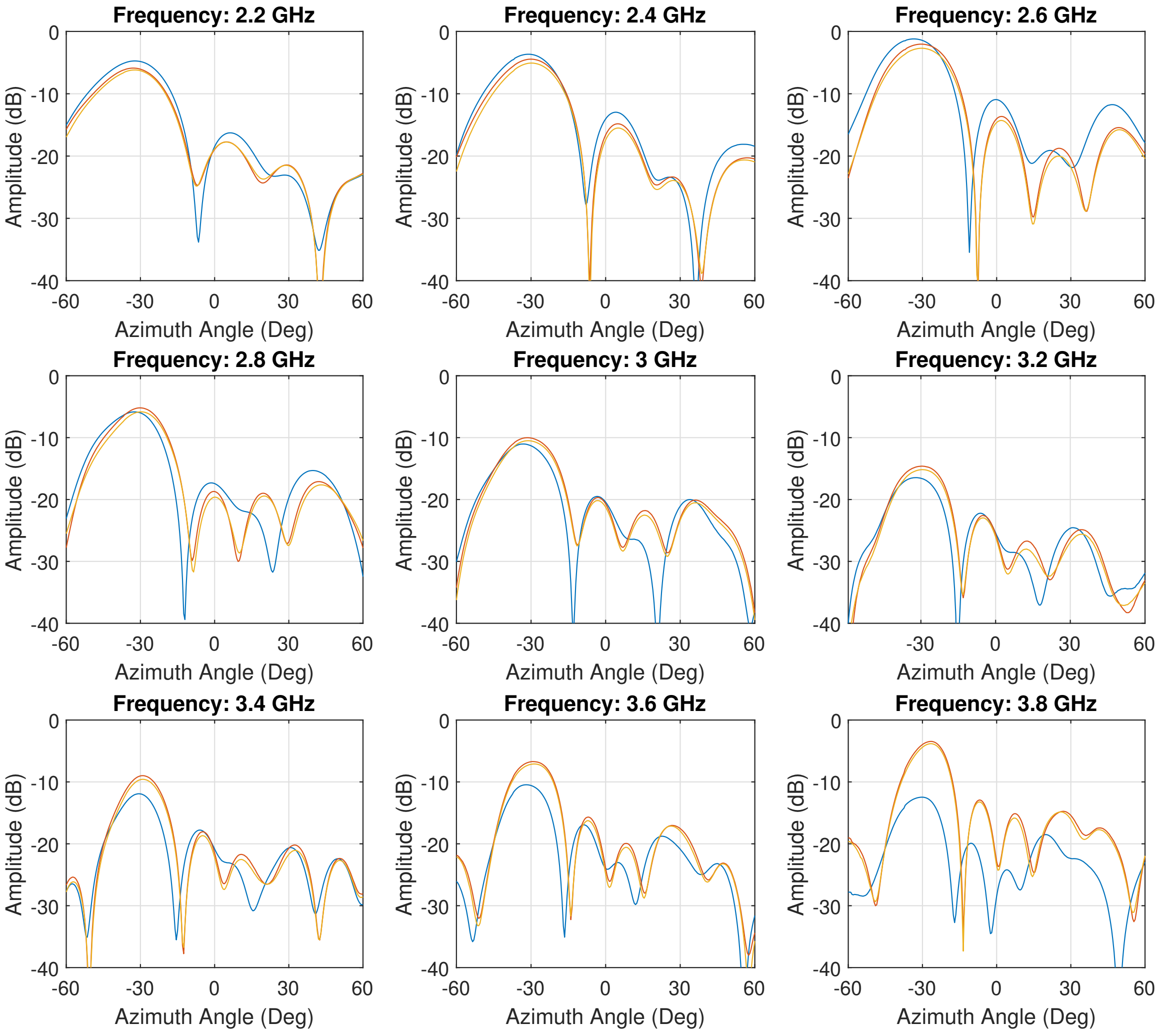

Figure 4.36: Frequency Cuts for $\theta_{\mathrm{st}} = \text{-}30°$: Blue - Uncalibrated, Red-Orange: Frequency Domain Synthesis Equalization, Yellow - Time Domain Synthesis Equalization

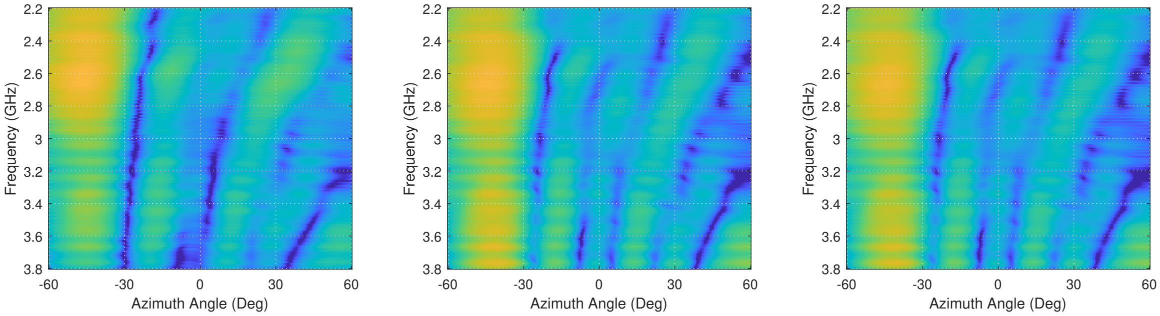

Figure 4.37: Antenna Patterns for $\theta_{\mathrm{st}} =$ -45°: Left - Uncalibrated, Center - Frequency Domain Synthesized Equalizer, Right - Time Domain Synthesized Equalizer

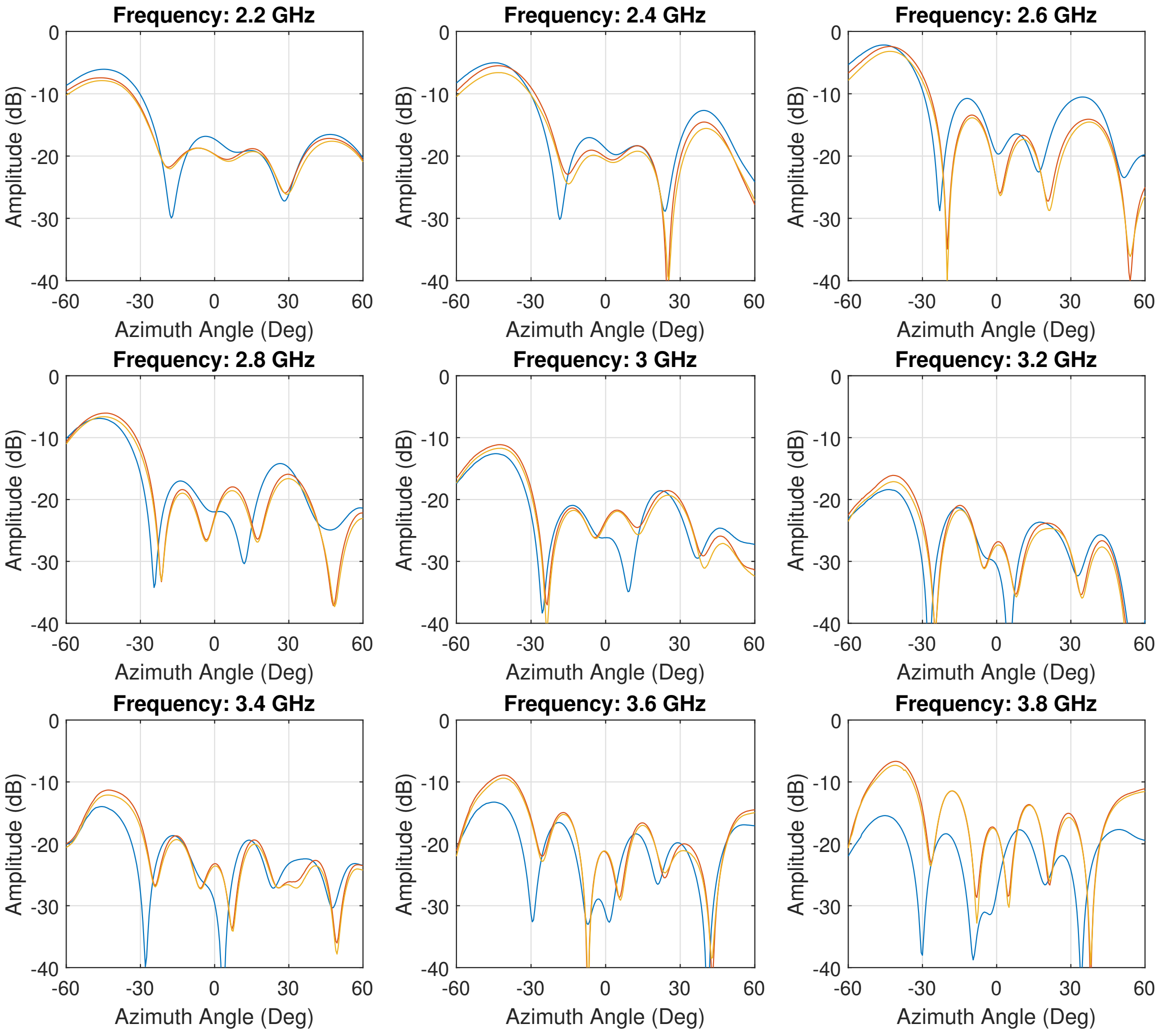

Figure 4.38: Frequency Cuts for $\theta_{\mathrm{st}} =$ -45°: Blue - Uncalibrated, Red-Orange: Frequency Domain Synthesis Equalization, Yellow - Time Domain Synthesis Equalization

4.2.3.3 Future Efforts

Notable factors which contribute to sub-optimal behavior include the following. First, mismatch between the antenna elements and ADCs results in ripple across the system bandwidth which is not captured in the channel characterization data and can not be compensated at the practical equalizer lengths of this RFSoC. If this mismatch is significant enough to impact the lower order phase response, which is compensated by the equalizer, there will disagreement between the synthesis results and the realized channel response. Second, system characterization disregarded the effects of mutual coupling, which is particularly influential in the Vivaldi aperture due to its reliance on mutual coupling to meet performance metrics. The authors in [68] explore this further by fitting their measurements to an array model in which they include coupling terms between adjacent elements. By solving for the coupling terms using their measured data, they successfully modeled a reasonable approximation to their measurements. Sidelobe behavior and null depth in their measurements show similarities to these captured herein.

Chapter 5

Conclusion

In conclusion, despite their many advantages, there are many challenges associated with the design, fabrication, and operation of phased array systems. The continued improvement of ADC and DAC sample rates and their integration with FPGA fabrics into a compact package is revolutionizing the phased array design space. By instantiating channel equalization with fractional-sample delays, digital wideband beamforming mitigates bandwidth limitations due to beam squint and pulse distortion. This enables phased array support for novel wideband communication and radar waveforms, improving communication data rates and radar operating characteristics. This dissertation contributes the development and demonstration of a wideband digital beamformer on an RFSoC and provides an in-depth examination of beamforming and equalization theory.

Chapter 2 revisits phased array processing, providing an overview of phased array systems and revisiting spatial and time domain Fourier processing. Classical beamforming for narrowband and wideband systems is examined along with a comparison between the two. A method for null formation and the physical limitations of null effectiveness are discussed. Arbitrary beamshaping enables the synthesis of arbitrary beam patterns, including frequency-invariant antenna responses. Lastly, the justifications for and implications of sub-Nyquist-sampled beamforming are discussed in detail. Included are illustrative simulation results for various system architectures.

Chapter 3 revisits adaptive array processing. Using the given mathematical system representation, derivations for several adaptive filtering techniques and their applications to phased array processing are provided. These include the LMS and RLS iterative algorithms as well as statistically optimum MMSE, ML, MVDR, and LCMV algorithms. As the statistically optimum methods rely on the system statistics for operation, a method for acquiring an estimate of the covariance matrix is given. The RMB rule provides a method to determine how much data is required for an accurate representation. For the statistically optimum algorithms, distinctions are made between whether or not an estimate of the noise environment is available without the signal of interest. Lastly, comparisons of complexity and performance are provided. Simulated results as well as details on algorithm computational complexity are given.

Chapter 4 presents demonstrations of wideband digital beamforming on an RFSoC. Uncalibrated TTD beamforming is implemented digitally through a fractional-sample delay filter bank. A derivation for a fractional-sample delay filter and trade-offs between performance and system resources are provided. Details are given on the RFSoC testbed including the antenna array as well system hardware, firmware, and software. An interleaved firmware architecture for FIR filter implementation, which maintains full digital bandwidth, is also provided. Measured results showing wideband and narrowband RFSoC measurements demonstrate the effectiveness of the TTD implementation. Calibrated measurements are given as well. Least-squares equalization techniques are explored in detail for both frequency domain synthesis and time domain synthesis. Details for reconfigurable complex-coefficient FIR filter required for the equalizer implementation are provided along with calibrated measurement results.

Additional work would explore characterization methods which capture aperture effects, both for inter-element variations and mutual coupling characteristics. Incorporating mutual coupling, especially for wideband apertures which commonly rely on high mutual coupling to achieve performance metrics, would improve system performance. A lack of channel amplification reduced the ENOB considerably, degrading the manifestation of nulls in the measured pattern. In addition to improving SNR, incorporating amplification may also provide the opportunity to explore NLEQ on the testbed. The complex-coefficient channel compensation filters can be updated to demonstrate many different applications such as the arbitrary beamshaping previously discussed.

The wideband beamformer presented herein serves as a stepping stone towards the holy grail of fully digital RFSoC-based phased array systems. Relevant performance metrics and capabilities demonstrated in this body of work match or exceed those available in the current literature, as it is the first RFSoC-based TTD beamformer at full digital bandwidth, the first example of RFSoC-based sub-Nyquist beamforming, and the first wideband digital equalizer implemented on an RFSoC. These results serve as an indicator of future capabilities of digital phased array systems in the next several years. The continued improvement of data converter sample rates will enable increased system bandwidths and direct sampling of waveforms at higher carrier frequencies, simplifying the analog hardware design. This will also result in increased challenges associated with the real-time processing of the resulting quantities of digital data, synchronization across multiple devices, thermal management of highly concentrated processing power, and effective system calibration. The next decade will provide the solutions to many of the challenges associated with all-digital phased array systems, reducing system costs and increasing performance metrics.

Bibliography

[1] J. Capon, R. Greenfield, and R. J. Kolker, "Multidimensional maximum-likelihood processing of a large aperture seismic array," *Proceedings of the IEEE*, vol. 55, no. 2, pp. 192–211, 1967.

[2] K. Jorgensen, B. Grose, and F. Crandall, "Doppler sonar applied to precision underwater navigation," in *Proceedings of OCEANS '93*, pp. II469–II474 vol.2, 1993.

[3] C. S. Allen, W. K. Blake, R. P. Dougherty, D. Lynch, P. T. Soderman, and J. R. Underbrink, *Aeroacoustic Measurements.* Springer Science & Business Media, 2002.

[4] R. J. Mailloux, *Phased Array Antenna Handbook.* Artech house, 2017.

[5] S. D. Silverstein, "Application of orthogonal codes to the calibration of active phased array antennas for communication satellites," *IEEE Transactions on Signal Processing*, vol. 45, no. 1, pp. 206–218, 1997.

[6] S. S. Blackman, "Multiple-target tracking with radar applications," *ah*, 1986.

[7] M. A. Richards, *Fundamentals of Radar Signal Processing.* McGraw-Hill Education, 2005.

[8] M. A. Richards, J. Scheer, W. A. Holm, and W. L. Melvin, *Principles of Modern Radar.* SciTech Publishing, Inc., 2010.

[9] R. G. Wiley, *ELINT: The Interception and Analysis of Radar Signals.* Artech House Inc., 2006.

[10] R. Rotman, M. Tur, and L. Yaron, "True time delay in phased arrays," *Proceedings of the IEEE*, vol. 104, no. 3, pp. 504–518, 2016.

[11] J. Vivekanandan, A. Karboski, and E. Loew, "Airborne polarimetric doppler weather radar: Antenna aperture and beam forming architecture," in *2019 IEEE International Symposium on Phased Array System Technology (PAST)*, pp. 1–6, 2019.

[12] I. Norheim-Naess and E. Finden, "DVB-T passive radar experimental comparisons of a custom made passive radar receiver, RFSoC and a Software Defined Radio," in *2021 21st International Radar Symposium (IRS)*, pp. 1–10, 2021.

[13] R. Fagan, F. C. Robey, and L. Miller, "Phased array radar cost reduction through the use of commercial RF systems on a chip," in *2018 IEEE Radar Conference (RadarConf18)*, pp. 0935–0939, IEEE, 2018.

[14] B. Schweizer, A. Grathwohl, G. Rossi, P. Hinz, C. Knill, S. Stephany, H. J. Ng, and C. Waldschmidt, "The fairy tale of simple all-digital radars: How to deal with 100 Gbit/s of a digital millimeter-wave MIMO radar on an FPGA [application notes]," *IEEE Microwave Magazine*, vol. 22, no. 7, pp. 66–76, 2021.

[15] T. G. Williamson, J. Whelan, W. Disharoon, P. Simmons, J. Houck, B. Holman, J. Alward, K. McDonald, S. Kim, D. Andreasen, and B. Faust, "Techniques for digital array radar planar near-field calibration by retrofit of an analog system," in *2021 IEEE Radar Conference (RadarConf21)*, pp. 1–6, 2021.

[16] M. Ispir and A. Yildirim, "Real-time signal generator for noise radar," *IEEE Aerospace and Electronic Systems Magazine*, vol. 35, no. 9, pp. 42–49, 2020.

[17] M. Harger, M. D. Conway, H. Thomas, M. Weber, J. Bendickson, A. Morris, T. Hoffmann, and N. Van Schaick, "Fully digital phased array development for next generation weather radar," in *2021 IEEE Radar Conference (RadarConf21)*, pp. 1–6, 2021.

[18] S. Jang, R. Lu, J. Jeong, and M. P. Flynn, "A 1-GHz 16-element four-beam true-time-delay digital beamformer," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 5, pp. 1304–1314, 2019.

[19] S. Pulipati, V. Ariyarathna, U. De Silva, N. Akram, E. Alwan, A. Madanayake, S. Mandal, and T. S. Rappaport, "A direct-conversion digital beamforming array receiver with 800 MHz channel bandwidth at 28 GHz using xilinx RF SoC," in *2019 IEEE International Conference on Microwaves, Antennas, Communications and Electronic Systems (COMCAS)*, pp. 1–5, IEEE, 2019.

[20] T. Hoffmann, C. Fulton, M. Yeary, A. Saunders, D. Thompson, B. Murmann, B. Chen, and A. Guo, "IMPACT — a common building block to enable next generation radar arrays," in *2016 IEEE Radar Conference (RadarConf)*, pp. 1–4, 2016.

[21] D. Brandwood, "A complex gradient operator and its application in adaptive array theory," in *IEE Proceedings H-Microwaves, Optics and Antennas*, vol. 130, pp. 11–16, IET, 1983.

[22] R. A. Mozingo, *Adaptive Arrays.* John Wiley & Sons, 1980.

[23] C. Cowan and P. Grant, *Adaptive Filters.* Prentice Hall, 1985.

[24] B. Widrow and S. D. Stearns, *Adaptive Signal Processing.* Prentice Hall, Inc., 1985.

[25] F. B. Gross and J. Volakis, *Smart Antennas.* McGraw-hill, 2005.

[26] M. H. Hayes, *Statistical Digital Signal Processing and Modeling.* John Wiley & Sons, 2009.

[27] J. O. Coleman, "Signals and systems II part V: Frequency conversion," *IEEE Potentials*, vol. 29, no. 5, pp. 41–44, 2010.

[28] S. C. Cripps, *RF power amplifiers for wireless communications.* Artech house Norwood, MA, 2006.

[29] C. Fulton, M. Yeary, D. Thompson, J. Lake, and A. Mitchell, "Digital phased arrays: Challenges and opportunities," *Proceedings of the IEEE*, vol. 104, no. 3, pp. 487–503, 2016.

[30] D. M. Pozar, *Microwave Engineering.* John Wiley and Sons, Inc., 2012.

[31] M. Löhning, T. Hentschel, and G. Fettweis, "Digital down conversion in software radio terminals," in *2000 10th European Signal Processing Conference*, pp. 1–4, IEEE, 2000.

[32] R. C. Hansen, *Phased Array Antennas*, vol. 213. John Wiley & Sons, 2009.

[33] D. S. Zrnic and R. J. Doviak, "System requirements for phased array weather radar," *NOAA/NSSL Rep*, 2005.

[34] J. D. Díaz, J. L. Salazar, J. A. Ortiz, C. Fulton, N. Aboserwal, R. Kelley, and R. Palmer, "A dual-polarized cross-stacked patch antenna with wide-angle and low cross-polarization for fully digital multifunction phased array radars," in *2016 IEEE International Symposium on Phased Array Systems and Technology (PAST)*, pp. 1–4, IEEE, 2016.

[35] A. Papoulis, *The Fourier Integral and its Applications.* McGraw-Hill Book Company, Inc, 1962.

[36] S. K. Mitra and Y. Kuo, *Digital Signal Processing: A Computer-Based Approach*, vol. 2. McGraw-Hill New York, 2006.

[37] B. D. Van Veen and K. M. Buckley, "Beamforming: A versatile approach to spatial filtering," *IEEE ASSP magazine*, vol. 5, no. 2, pp. 4–24, 1988.

[38] C. A. Balanis, *Antenna Theory: Analysis and Design.* John Wiley & Sons, 2016.

[39] S. Haykin, *Adaptive Filter Theory.* Prentice-Hall, Inc, 1996.

[40] R. L. Haupt, “Phased array bandwidth defined by beam squint and pulse dispersion,” in *2019 IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting*, pp. 387–388, IEEE, 2019.

[41] M. Longbrake, “True time-delay beamsteering for radar,” in *2012 IEEE National Aerospace and Electronics Conference (NAECON)*, pp. 246–249, IEEE, 2012.

[42] F. Uysal, Z. Dunn, M. Yeary, and R. Rincon, “Application of waveform weighting for a frequency-invariant transmit beampattern,” *IEEE Aerospace and Electronic Systems Magazine*, vol. 31, no. 12, pp. 4–12, 2016.

[43] T. Clark and E. Jaska, “Million element ISIS array,” in *2010 IEEE International Symposium on Phased Array Systems and Technology*, pp. 29–36, IEEE, 2010.

[44] H. L. Van Trees, *Optimum Array Processing: Part IV of Detection, Estimation, and Modulation Theory.* John Wiley & Sons, 2004.

[45] L. C. Godara, “Application of antenna arrays to mobile communications. II. beam-forming and direction-of-arrival considerations,” *Proceedings of the IEEE*, vol. 85, no. 8, pp. 1195–1245, 1997.

[46] J. C. A. Barata and M. S. Hussein, “The Moore–Penrose pseudoinverse: A tutorial review of the theory,” *Brazilian Journal of Physics*, vol. 42, no. 1-2, pp. 146–165, 2012.

[47] R. E. A. C. Paley and N. Wiener, *Fourier Transforms in the Complex Domain*, vol. 19. American Mathematical Soc., 1934.

[48] S. Kak, “New criterion of realisability,” *Electronics Letters*, vol. 4, no. 24, pp. 537–538, 1968.

[49] P. B. Kenington and L. Astier, “Power consumption of A/D converters for software radio applications,” *IEEE Transactions on Vehicular Technology*, vol. 49, no. 2, pp. 643–650, 2000.

[50] N. Morrison, *Introduction to Fourier analysis.* Wiley New York, 1994.

[51] I. S. Reed, J. D. Mallett, and L. E. Brennan, “Rapid convergence rate in adaptive arrays,” *IEEE Transactions on Aerospace and Electronic Systems*, no. 6, pp. 853–863, 1974.

[52] A. H. Sayed, *Fundamentals of Adaptive Filtering.* John Wiley & Sons, 2003.

[53] S. Applebaum, “Adaptive arrays,” *IEEE Transactions on Antennas and Propagation*, vol. 24, no. 5, pp. 585–598, 1976.

[54] A. Krishnamoorthy and D. Menon, “Matrix inversion using Cholesky decomposition,” in *2013 signal processing: Algorithms, architectures, arrangements, and applications (SPA)*, pp. 70–72, IEEE, 2013.

[55] G. H. Golub and C. F. Van Loan, *Matrix Computations*, vol. 3. JHU press, 2013.

[56] R. F. Harrington, *Field Computation by Moment Methods.* Wiley-IEEE Press, 1993.

[57] F. Nai, S. M. Torres, and R. D. Palmer, “Adaptive beamspace processing for phased-array weather radars,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 10, pp. 5688–5698, 2016.

[58] H. Cohen, *Mathematics for Scientists and Engineers.* Prentice Hall, 1992.

[59] E. Yoshikawa, T. Ushio, Z. Kawasaki, S. Yoshida, T. Morimoto, F. Mizutani, and M. Wada, “MMSE beam forming on fast-scanning phased array weather radar,” *IEEE Transactions on Geoscience and remote sensing*, vol. 51, no. 5, pp. 3077–3088, 2012.

[60] F. Qian and B. D. Van Veen, "Quadratically constrained adaptive beamforming for coherent signals and interference," *IEEE transactions on signal processing*, vol. 43, no. 8, pp. 1890–1900, 1995.

[61] P. Stoica, Z. Wang, and J. Li, "Robust capon beamforming," in *Conference Record of the Thirty-Sixth Asilomar Conference on Signals, Systems and Computers, 2002.*, vol. 1, pp. 876–880, IEEE, 2002.

[62] L. Scott and B. Mulgrew, "Sparse LCMV beamformer design for suppression of ground clutter in airborne radar," *IEEE transactions on signal processing*, vol. 43, no. 12, pp. 2843–2851, 1995.

[63] L. Miller, "Adaptive beamforming for radar: Floating-point QRD+ WBS in an FPGA," *Xilinx White Paper WP452, June 2014*, 2014.

[64] S. S. Skiena, *The Algorithm Design Manual: Text*, vol. 1. Springer Science & Business Media, 1998.

[65] T. Hoffmann, D. Jensen, J. Moran, J. Lake, J. Dixon, T. Goodwin, N. Haglof, and M. Valayil, "Rapid integration of a flexible, wideband beamformer with wideband antenna technology," in *2019 IEEE International Symposium on Phased Array System & Technology (PAST)*, pp. 1–5, IEEE, 2019.

[66] T. Hoffmann, C. Fulton, M. Yeary, A. Saunders, D. Thompson, B. Murmann, B. Chen, and A. Guo, "Measured performance of the IMPACT common module — a building block for next generation phase arrays," in *2016 IEEE International Symposium on Phased Array Systems and Technology (PAST)*, pp. 1–7, 2016.

[67] J. Corbin and R. L. Howard, "TDU quantization error impact on wideband phased-array performance," in *Proceedings 2000 IEEE International Conference on Phased Array Systems and Technology (Cat. No. 00TH8510)*, pp. 457–460, IEEE, 2000.

[68] M. Longbrake, L. L. Liou, D. M. Lin, P. Buxa, J. McCann, T. Pemberton, T. Dalrymple, and S. Hary, "Wideband phased array calibration method for digital beamforming," in *2012 IEEE National Aerospace and Electronics Conference (NAECON)*, pp. 11–17, 2012.

[69] J. Lake, M. Yeary, and R. Palmer, "Real-time digital equalization to enhance element-level digital beamforming," in *2019 IEEE Radar Conference (RadarConf)*, pp. 1–6, 2019.

[70] J. Johnson, A. Fenn, H. Aumann, and F. Willwerth, "An experimental adaptive nulling receiver utilizing the sample matrix inversion algorithm with channel equalization," *IEEE Transactions on Microwave Theory and Techniques*, vol. 39, no. 5, pp. 798–808, 1991.

[71] D. Thompson, M. Yeary, and C. Fulton, "RF array system equalization and true time delay with FPGA hardware-in-the-loop," in *2016 IEEE International Symposium on Phased Array Systems and Technology (PAST)*, pp. 1–5, 2016.

[72] L. L. Liou, D. M. Lin, M. Longbrake, P. Buxa, J. McCann, T. Dalrymple, J. B. Tsui, R. Qiu, Z. Hu, and N. Guo, "Digital wideband phased array calibration and beamforming using time reversal technique," in *2010 IEEE International Symposium on Phased Array Systems and Technology*, pp. 261–266, 2010.

[73] K. Steiner and M. Yeary, "A 1.6 GHz wideband sub-Nyquist sampled beamformer on an RFSoC," *IEEE Transactions on Aerospace and Electronic Systems*, vol. under review, 2021.

[74] N. Peccarelli and C. Fulton, "Array-level approach to nonlinear equalization," in *2019 IEEE International Conference on Microwaves, Antennas, Communications and Electronic Systems (COMCAS)*, pp. 1–6, IEEE, 2019.

[75] N. Peccarelli, R. Irazoqui, and C. Fulton, "Mitigation of interferers and nonlinear spurious products for digital array and mimo systems," in *2019 IEEE MTT-S International Microwave Symposium (IMS)*, pp. 1233–1236, IEEE, 2019.

[76] Z. Dunn, M. Yeary, C. Fulton, and N. Goodman, "Memory polynomial model for digital predistortion of broadband solid-state radar amplifiers," in *2015 IEEE Radar Conference (RadarCon)*, pp. 1482–1486, IEEE, 2015.

[77] Z. Dunn, M. Yeary, C. Fulton, and N. Goodman, "Wideband digital predistortion of solid-state radar amplifiers," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 5, pp. 2452–2466, 2016.

[78] M. Lang, *Algorithms for the constrained design of digital filters with arbitrary magnitude and phase responses.* Österr. Kunst-und Kulturverl., 2000.

[79] M. Yeary, *Design of a finite impulse response filter for the purpose of directional edge detection via spatial domain convolution,.* Texas A&M University., 1994.

[80] K. Lauritzen, H. Krichene, and S. Talisa, "Hardware limitations of receiver channel-pair cancellation ratio," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 1, pp. 290–303, 2012.

[81] A. S. Householder, "Unitary triangularization of a nonsymmetric matrix," *Journal of the ACM (JACM)*, vol. 5, no. 4, pp. 339–342, 1958.

[82] J. R. Rice, "Experiments on Gram-Schmidt orthogonalization," *Mathematics of Computation*, vol. 20, no. 94, pp. 325–328, 1966.

[83] W. Givens, "Computation of plain unitary rotations transforming a general matrix to triangular form," *Journal of the Society for Industrial and Applied Mathematics*, vol. 6, no. 1, pp. 26–50, 1958.

[84] R. Rotman and M. Tur, “Antenna and beamformer requirements for wideband phased array systems: A review,” in *2009 IEEE International Conference on Microwaves, Communications, Antennas and Electronics Systems*, pp. 1–1, 2009.

[85] A. Karatsuba, “Multiplication of multidigit numbers on automata,” in *Soviet Physics Doklady*, vol. 7, pp. 595–596, 1963.

Appendix A

Matlab Code - Linear Array Model

```
% ---------------------------------------------------------------------
% Uniform Linear Array True Time Delay Model
%
% Model a ULA with TTD filter for wideband beamforming
% ---------------------------------------------------------------------
clear

% Simulation Settings
% -------------------
% Angle Settings
azAngSteer = -50;  % Azimuth Steering Angle
azAngTest  = -50;  % Signal Incident Angle (if numAng == 1)

% Assert to use Measured Channel Data
useMeas = 0;
useEQ   = 0;

% Plot Settings
numAng     = 721; % Number of Azimuth Angles 721
numFreqPlt = 500; % Min Number of Frequency Points 500

% Simulation Selection
```

```
useFP      = 0;   % Assert to use fixed point
packSamp   = 0;   % Assert to process in sample packs
genHLSFile = 0;   % Assert to Create HLS Files

% System Parameters
numElem    = 6;   % Number of Elements
numFiltTTD = 8;   % Number of TTD Filters

% TTD Filter Length
numEffTaps = 17;

% Waveform Parameters
scale    = 0.125;
tPls     = .2e-6;  % Pulse Length (s)
selWave  = 1;      % Waveform Select (1. Chirp, 2. Carrier Tone)
selWin   = 2;      % Window Select (1. None, 2. Hanning)
windPerc = 0.1;    % Windowed Percentage
ampVar   = 0;      % Assert to include amplitude variation

tMsPls = sapPerDat*(numSampMeas-1);

% Frequency Parameters
% --------------------
sampFreqTx = 40e9;    % Oversampled Incident Sample Rate
sampFreqRx = 4e9;     % Receiver Sample Frequency (Hz)
sampFreqDnRx = sampFreqRx/2;     % Receiver Sample Frequency (Hz)
freqBW     = 1600e6;   % Signal Bandwidth
```

```
% --------------------------------------------------------------------
c = 3e8;      % Speed of Light (m/s)

elemSep = 0.05;

% Sample Period
sampPerRx = 1/sampFreqRx;
sampPerTx = 1/sampFreqTx;

% Data Stream Length
tData = (1+padPer)*tPls;
tMsData = (1+padPer)*tMsPls;

% Embedded System Processing
% ------------------------------
fp = fp_sett(numRxSamp);
F  = fimath(’OverflowAction’,’Saturate’);

% Filter Bank Synthesis
% ------------------------------
[ttdBank_fp,~] = ...
  fd_filt_bank_exp([numTaps,numEffTaps],numFiltTTD,fp,useEQ);

% --------------------------------------------------------------------
% Frequency Vector Constants
% --------------------------------------------------------------------
freqSprPerc = 1;
% Determine Frequencies
```

```
freqSt  = freqCent - freqSprPerc*(freqBW/2);
freqEnd = freqCent + freqSprPerc*(freqBW/2);
% Determine Frequencies
freqStAct  = freqCent - freqBW/2;
freqEndAct = freqCent + freqBW/2;
% Determine Aliased Frequencies
freqCentAl = ...
  abs(freqCent - floor(nyqZone/2)*sampFreqRx) - sampFreqRx/4;
freqStAl   = freqCentAl - freqSprPerc*(freqBW/2);
freqEndAl  = freqCentAl + freqSprPerc*(freqBW/2);

% ------------------------------------------------------------------
% Beamforming Calculations
% ------------------------------------------------------------------
sampPerRxDn = 2*sampPerRx;

% Steering Sample and Phase Shifts
timeDel = (elemSep*sin((pi/180)*azAngSteer)/c)*(0:numElem-1);

sampDel = timeDel./sampPerRxDn;
sampDel = sampDel - min(sampDel);
% Quantize
sampDelQnt = round(sampDel*numFiltTTD)/numFiltTTD;

% Integer and Fractional Sample Delays
sampDelInt = floor(sampDelQnt);
sampDelFrc = sampDelQnt - sampDelInt;
```

```
% Quantized Correction Phase (Bandpass Sampling)
phsCorrCyc = (floor(nyqZone/2)-0.25)*sampDel*2;
phsCorrRad = 2*pi*phsCorrCyc;
phsCorrDeg = 360*phsCorrCyc;

phsCorrRadWr = wrapToPi(phsCorrRad);
phsCorrQnt = wrapTo180(phsCorrDeg);

phsCorrQntDeg = wrapTo180((180/pi)*phsCorrQnt);
phsCorrCycWr  = wrapTo180(360*phsCorrCyc)/360;

% --------------------------------------------------------------------
% Impinging Plane Wave Model
%
% Model as individual channel streams with appropriate delay based on
% azimuth angle. Construct array of channel streams for each azimuth
% angle.
% --------------------------------------------------------------------
% Specify Azimuth Angle Vector
% ---------------------------
if numAng == 1
  azAng = azAngTest;
else
  azAng = -90:180/(numAng-1):90;
%   azAng = -60:120/(numAng-1):60;
end
% Angle Index
[~,indAngSteer] = min(abs(azAng-azAngSteer));
```

```
[~,indAngTest]  = min(abs(azAng-azAngTest));
% ----------------------------

% Create Transmit Waveform
% ------------------------
% Waveform 1 - Chirp
% Complex Baseband Vector (volts)
% adjust amplitude to demonstrate direction of spectrum
if ampVar
  ampVect = ones(numPlsSamp,1) + (tVectPls - tPls/2)/(4*tPls);
else
  ampVect = ones(numPlsSamp,1); %#ok<*UNRCH>
end
% For Waveform
switch selWave
  case 1
    chirpRate   = freqBW/tPls;
    phaseArg    = (-freqBW/2)*tVectPls + (chirpRate/2)*tVectPls.^2;
    chirpVectBB = 0.04*txWind.*ampVect.*exp(1i*2*pi*phaseArg);
  case 2
    chirpVectBB = 0.04*sin(2*pi*1e8*tVectPls);
  otherwise
    warning('Invalid Transmit Window - Assume no window')
end

% Upconvert to Carrier - real signal (volts)
chirpVectCarr = ...
  scale*(real(chirpVectBB).*cos(2*pi*freqCent*tVectPls) + ...
```

```
         imag(chirpVectBB).*sin(2*pi*freqCent*tVectPls));
% ---------------------------------------------------------------------


% Create Arrays of Delayed Waveforms for Azimuth Angles
% ---------------------------------------------------------------------
% Initialize Arrays
planeWaves = zeros(numTxSamp,numElem,numAng);
planeMsWaves = zeros(numMsTxSamp,numElem,numAng);
for angNum = 1:numAng
  % Waveform Delays (sec)
  % element delay
  elemDelInd = elemSep*sin((pi/180)*(-azAng(angNum)))/c;
  % delay vector
  elemDel    = (elemDelInd)*(0:numElem-1);
  if elemDelInd < 0
    elemDel = elemDel - min(elemDel);
  end

  for elemNum = 1:numElem
    planeWaves(:,elemNum,angNum) = chanEn(elemNum)*...
        wf_delay(txPlsVect,elemDel(elemNum),..
        numTxSamp,sampFreqTx);
    planeMsWaves(:,elemNum,angNum) = chanEn(elemNum)*...
        wf_delay(txMsPlsVect(:,elemNum),elemDel(elemNum),...
        numMsTxSamp,sampFreqTx);
  end
end
```

```
% -------------------------------------------------------------------
% Uniform Linear Array Model
%
% Shift channels per TTD filter and Phase Shifter Adjustments
% -------------------------------------------------------------------
% Element Pattern
% ---------------
if incPatt == 1
  elemPatt = cos((pi/180)*shiftdim(azAng,-1));
  elemPatt(elemPatt<1e-9) = 1e-9;
else
  elemPatt = ones(1,1,numAng);
end

elemPattAll = repmat(elemPatt,numTxSamp,numElem,1);
planeWavesAnt = elemPattAll.*planeWaves;

elemPattAll = repmat(elemPatt,numMsTxSamp,numElem,1);
planeMsWavesAnt = elemPattAll.*planeMsWaves;

% Analog-Digital Converter
% ------------------------
sampVectInt = (tVectRx./sampPerTx);
% Linear interpolate Rx Planewave
rxPlaneWave = interp1(sampVectTx,planeWavesAnt,sampVectInt);

sampVectMsInt = (tVectMsRx./sampPerTx);
% Linear interpolate Rx Planewave
```

```
rxMsPlaneWave = interp1(sampVectMsTx,planeMsWavesAnt,sampVectMsInt);

% True Time Delay Filter Bank
% ---------------------------
% if useEQ==1
% first EQ filter, second effective frac filter
% final length is sum(numTaps-3);

ttdBank = double(ttdBank_fp);

numTapAll = size(ttdBank,1);

fracDelVect = 0:1/(numFiltTTD):1-1/(numFiltTTD);
fltInd      = zeros(numElem,1);
ttdCoefs    = zeros(length(ttdBank(:,1)),numElem);
for elemNum = 1:numElem
  [~,fltInd(elemNum)] = ...
    find(abs(fracDelVect - sampDelFrc(elemNum)) < 0.00001);

  ttdCoefs(:,elemNum)  = ttdBank(:,fltInd(elemNum),elemNum);

  if useFP == 1
    ttdCoefs_fp(:,elemNum)  = ttdBank_fp(:,fltInd(elemNum),elemNum);
  end
end

% Print Parameters
% ------------------------------
```

```
propDel = elemSep*sin((pi/180)*azAngSteer)/c;
filtRes = sampPerRx/numFiltTTD;

phsI = real(exp(1i*phsCorrQnt));
phsQ = imag(exp(1i*phsCorrQnt));

if useFP == 1
  phsI_fp = fi(phsI,fp.phs.s,fp.phs.w,fp.phs.w-fp.phs.i);
  phsQ_fp = fi(phsQ,fp.phs.s,fp.phs.w,fp.phs.w-fp.phs.i);
end

compCoef = compCoef/(max(abs(compCoef)));

compCoef_fp  = ...
  fi(compCoef,fp.flt.s,fp.flt.w,fp.flt.w-fp.flt.i,'fimath',fp.F);

% Digital Downconversion
% --------------------------------------------------------------------
sigIdBB = zeros(numRxDnSamp,numElem,numAng);
sigMsBB = zeros(numMsRxDnSamp,numElem,numAng);
for angNum = 1:numAng
  for elemNum = 1:numElem

    % Matlab Implementation
    % ----------------------------------------------------------------
    % Frequency Shift with phase correction
    temp = rxPlaneWave(:,elemNum,angNum).*...
          exp(1i*((-pi/2)*sampVectRx + phsCorrQnt(elemNum)));
```

```
    % Image Reject
    temp = conv(imgRejCoef,temp);

    % Down-sample
    sigIdBB(:,elemNum,angNum) = temp(1:2:numRxSamp);

    % Frequency Shift with phase correction
    temp = rxMsPlaneWave(:,elemNum,angNum).*...
           exp(1i*((-pi/2)*sampVectMsRx + phsCorrQnt(elemNum)));

    % Image Reject
    temp = conv(imgRejCoef,temp);

    % Down-sample
    sigMsBB(:,elemNum,angNum) = temp(1:2:numMsRxSamp);
  end
end

if useMeas == 1
  sigBB = sigMsBB;

  numRxDnSamp = numMsRxDnSamp;
  numRxSamp = numMsRxSamp;
  sampVectRx = sampVectMsRx;
else
  sigBB = sigIdBB;
end
```

```
% Cast to Fixed-Point
% ---------------------------------------------------------------------
if useFP == 1
  sigBBI = real(sigBB);
  sigBBQ = imag(sigBB);

  % cast to unsigned fixed point with no integer bits
  sigBBI_fp = ...
    fi(sigBBI,fp.dataIn12.s,fp.dataIn12.w,fp.dataIn12.w-fp.dataIn12.i);
  sigBBQ_fp = ...
    fi(sigBBQ,fp.dataIn12.s,fp.dataIn12.w,fp.dataIn12.w-fp.dataIn12.i);
end

% Signal Compensation
% ---------------------------------------------------------------------

for angNum = 1:numAng
  % Matlab Implementation
  % -------------------------------------------------------------------
  for elemNum = 1:numElem
    % Course Delay
    sigCrsDel(:,elemNum) = [zeros(sampDelInt(elemNum),1);
    sigBB(1:end-sampDelInt(elemNum),elemNum,angNum)];
    % Fractional Delay
    temp = conv(ttdCoefs(:,elemNum),sigCrsDel(:,elemNum));
    sigTTD(:,elemNum,angNum) = temp(1:numRxDnSamp);
```

```
  sigCompI(:,elemNum,angNum) = real(sigTTD(:,elemNum,angNum));
  sigCompQ(:,elemNum,angNum) = imag(sigTTD(:,elemNum,angNum));
end

sigSumIPre(:,angNum) = squeeze(sum(sigCompI(:,:,angNum),2));
sigSumQPre(:,angNum) = squeeze(sum(sigCompQ(:,:,angNum),2));

sigSumI(:,angNum) = sigSumIPre(:,angNum);
sigSumQ(:,angNum) = sigSumQPre(:,angNum);

% Fixed-Point Implementation
% -------------------------------------------------------------------
if useFP == 1
  for elemNum = 1:numElem
    % Course Delay
    sigCrsDelI_fp(:,elemNum) = [zeros(sampDelInt(elemNum),1);
    sigBBI_fp(1:end-sampDelInt(elemNum),elemNum,angNum)];
    sigCrsDelQ_fp(:,elemNum) = [zeros(sampDelInt(elemNum),1);
    sigBBQ_fp(1:end-sampDelInt(elemNum),elemNum,angNum)];

    sigCrsDel_fp(:,elemNum) = ...
      complex(sigCrsDelI_fp(:,elemNum),sigCrsDelQ_fp(:,elemNum));

    temp = conv(ttdCoefs_fp(:,elemNum),sigCrsDel_fp(:,elemNum));
    sigTTD_fp(:,angNum) = temp(1:numRxDnSamp);

    acc_fp2(:,elemNum,angNum) = ...
      fi(sigTTD_fp(:,angNum),fp.acc.s,fp.acc.w,fp.acc.w-fp.acc.i);
```

```
accI_fp2 = real(acc_fp2);
accQ_fp2 = imag(acc_fp2);

numPack = ceil(numRxDnSamp/numInSampClk);

if packSamp
  extra   = numInSampClk*numPack - numRxDnSamp;

  chanBufPackI = sigCrsDelI_fp(:,elemNum);
  chanBufPackQ = sigCrsDelQ_fp(:,elemNum);

  chanBufPackI(end+1:end+extra,:,:) = 0;
  chanBufPackQ(end+1:end+extra,:,:) = 0;

  filtBufIPack_fp = ...
    fi(zeros(bufLen,1),fp.dataIn12.s,fp.dataIn12.w,...
    fp.dataIn12.w-fp.dataIn12.i);
  filtBufQPack_fp = ...
    fi(zeros(bufLen,1),fp.dataIn12.s,fp.dataIn12.w,...
    fp.dataIn12.w-fp.dataIn12.i);

  for packNum = 1:numPack

    filtBufIPack_fp(1:bufLen-numSampIQ) = ...
      filtBufIPack_fp(numSampIQ+1:bufLen);
    filtBufQPack_fp(1:bufLen-numSampIQ) = ...
      filtBufQPack_fp(numSampIQ+1:bufLen);
```

```
filtBufIPack_fp(bufLen-numSampIQ+1:bufLen) = ...
  chanBufPackI((packNum-1)*numInSampClk+1:...
  numInSampClk*packNum);
filtBufQPack_fp(bufLen-numSampIQ+1:bufLen) = ...
  chanBufPackQ((packNum-1)*numInSampClk+1:...
  numInSampClk*packNum);

conv1Pack_fp(:,packNum,elemNum) = ...
  conv(filtBufIPack_fp,real(ttdCoefs_fp(:,elemNum)),'valid');
conv2Pack_fp(:,packNum,elemNum) = ...
  conv(filtBufQPack_fp,imag(ttdCoefs_fp(:,elemNum)),'valid');
conv3Pack_fp(:,packNum,elemNum) = ...
  conv((filtBufIPack_fp+filtBufQPack_fp), ...
  (real(ttdCoefs_fp(:,elemNum))+ ...
  imag(ttdCoefs_fp(:,elemNum))),'valid');

accIPack_fp = conv1Pack_fp(:,packNum,elemNum) - ...
  conv2Pack_fp(:,packNum,elemNum);
accQPack_fp = conv3Pack_fp(:,packNum,elemNum) - ...
  conv1Pack_fp(:,packNum,elemNum) - ...
  conv2Pack_fp(:,packNum,elemNum);

accIPack_fp2 = ...
  fi(accIPack_fp,fp.acc.s,fp.acc.w,fp.acc.w-fp.acc.i);
accQPack_fp2 = ...
  fi(accQPack_fp,fp.acc.s,fp.acc.w,fp.acc.w-fp.acc.i);

filtBufIPacked_fp(:,packNum,elemNum) = filtBufIPack_fp;
```

```
      filtBufQPacked_fp(:,packNum,elemNum) = filtBufQPack_fp;
      accIPacked_fp2(:,packNum,elemNum)    = accIPack_fp2;
      accQPacked_fp2(:,packNum,elemNum)    = accQPack_fp2;
    end

    filtBufI_fp3(:,elemNum) = ...
      reshape(filtBufIPacked_fp(:,:,elemNum),[bufLen*numPack,1]);
    filtBufQ_fp3(:,elemNum) = ...
      reshape(filtBufQPacked_fp(:,:,elemNum),[bufLen*numPack,1]);
    accI_fp3(:,elemNum)     = ...
      reshape(accIPacked_fp2(:,:,elemNum),[numSampIQ*numPack,1]);
    accQ_fp3(:,elemNum)     = ...
      reshape(accQPacked_fp2(:,:,elemNum),[numSampIQ*numPack,1]);

    filtBufI_fp4 = filtBufI_fp3(1:numRxDnSamp,:);
    filtBufQ_fp4 = filtBufQ_fp3(1:numRxDnSamp,:);
    accI_fp4     = accI_fp3(1:numRxDnSamp,:);
    accQ_fp4     = accQ_fp3(1:numRxDnSamp,:);
  end
end

if packSamp
  accI_fp5 = accI_fp4;
  accQ_fp5 = accQ_fp4;
else
  accI_fp5 = accI_fp2(:,:,angNum);
  accQ_fp5 = accQ_fp2(:,:,angNum);
end
```

```
% Summation
sigSumIPre_fp(:,angNum) = squeeze(sum(accI_fp5,2));
sigSumQPre_fp(:,angNum) = squeeze(sum(accQ_fp5,2));

sigSumIPre_fp2(:,angNum) = ...
  fi(sigSumIPre_fp(:,angNum),fp.sout.s,...
  fp.sout.w,fp.sout.w-fp.sout.i);
sigSumQPre_fp2(:,angNum) = ...
  fi(sigSumQPre_fp(:,angNum),fp.sout.s,...
  fp.sout.w,fp.sout.w-fp.sout.i);

if packSamp
  % Buffer Length
  numTapTot = numInSampClk + length(compCoef_fp) - 1;
  bufOutLen = numTapTot;

  preSumBufPackI_fp = sigSumIPre_fp2(:,angNum);
  preSumBufPackQ_fp = sigSumQPre_fp2(:,angNum);

  preSumBufPackI_fp(end+1:end+extra,:,:) = 0;
  preSumBufPackQ_fp(end+1:end+extra,:,:) = 0;

  preBufIPack_fp = fi(zeros(bufOutLen,1),fp.dataIn12.s,...
    fp.dataIn12.w,fp.dataIn12.w-fp.dataIn12.i);
  preBufQPack_fp = fi(zeros(bufOutLen,1),fp.dataIn12.s,...
    fp.dataIn12.w,fp.dataIn12.w-fp.dataIn12.i);
```

```
for packNum = 1:numPack

  preBufIPack_fp(1:bufOutLen-numSampIQ) = ...
    preBufIPack_fp(numSampIQ+1:bufOutLen);
  preBufQPack_fp(1:bufOutLen-numSampIQ) = ...
    preBufQPack_fp(numSampIQ+1:bufOutLen);
  preBufIPack_fp(bufOutLen-numSampIQ+1:bufOutLen) = ...
    preSumBufPackI_fp((packNum-1)*numInSampClk+1:...
    numInSampClk*packNum);
  preBufQPack_fp(bufOutLen-numSampIQ+1:bufOutLen) = ...
    preSumBufPackQ_fp((packNum-1)*numInSampClk+1:...
    numInSampClk*packNum);

  conv1Pack_fp = conv(preBufIPack_fp,real(compCoef_fp),'valid');
  conv2Pack_fp = conv(preBufQPack_fp,imag(compCoef_fp),'valid');
  conv3Pack_fp = conv((preBufIPack_fp+preBufQPack_fp),...
    (real(compCoef_fp)+imag(compCoef_fp)),'valid');

  sigSumPackI_fp = conv1Pack_fp - conv2Pack_fp;
  sigSumPackQ_fp = conv3Pack_fp - conv1Pack_fp - conv2Pack_fp;

  sigSumPackI_fp2 = ...
    fi(sigSumPackI_fp,fp.acc.s,fp.acc.w,fp.acc.w-fp.acc.i);
  sigSumPackQ_fp2 = ...
    fi(sigSumPackQ_fp,fp.acc.s,fp.acc.w,fp.acc.w-fp.acc.i);

  preBufIPacked_fp(:,packNum) = preBufIPack_fp;
  preBufQPacked_fp(:,packNum) = preBufQPack_fp;
```

```
        sigSumIPacked_fp2(:,packNum) = sigSumPackI_fp2;
        sigSumQPacked_fp2(:,packNum) = sigSumPackQ_fp2;
      end

      preBufI_fp3 = reshape(preBufIPacked_fp,[bufOutLen*numPack,1]);
      preBufQ_fp3 = reshape(preBufQPacked_fp,[bufOutLen*numPack,1]);
      sigSumI_fp3 = reshape(sigSumIPacked_fp2,[numSampIQ*numPack,1]);
      sigSumQ_fp3 = reshape(sigSumQPacked_fp2,[numSampIQ*numPack,1]);

      preBufI_fp4 = preBufI_fp3(1:numRxDnSamp);
      preBufQ_fp4 = preBufQ_fp3(1:numRxDnSamp);
      sigSumI_fp4 = sigSumI_fp3(1:numRxDnSamp);
      sigSumQ_fp4 = sigSumQ_fp3(1:numRxDnSamp);

    end

    sigSumI_fp5 = sigSumIPre_fp2(:,angNum);
    sigSumQ_fp5 = sigSumQPre_fp2(:,angNum);

    sigSumI(:,angNum) = double(sigSumI_fp5);
    sigSumQ(:,angNum) = double(sigSumQ_fp5);
  end

  sigSum(:,angNum) = sigSumI(:,angNum) + 1i*sigSumQ(:,angNum);
end

% Digital Upconversion
% -------------------------------------------------------------------
```

```
for angNum = 1:numAng
  % Up-sample
  rxOutUp(:,angNum) = ...
    reshape([sigSum(:,angNum),zeros(numRxDnSamp,1)].',[numRxSamp,1]);

  % Image Reject
  temp = conv(rxOutUp(:,angNum),scOut*imgRejCoef);
  rxOutUpFilt(:,angNum) = temp(1:numRxSamp);

  % Frequency Shift
  rxOut(:,angNum) = ...
    real(rxOutUpFilt(:,angNum).*exp(1i*(pi/2)*sampVectRx));
end
% ---------------------------------------------------------------------

sigCompF.comp = fftshift(fft(rxOut,numRxSamp,1),1);
```