Theory and Application of Broadband Frequency Invariant Beamforming

Darren Brett Ward
B.E. (Hons), B.App.Sc. (QUT)

A thesis submitted for the degree of Doctor of Philosophy
of the Australian National University

Department of Engineering
Faculty of Engineering and Information Technology
The Australian National University

July 1996

Declaration

The contents of this thesis are the result of original research and have not been submitted for a higher degree to any other university or institution.

Much of the work presented in this thesis has been published or will be submitted for publication as journal or conference papers. Following is a list of these papers. In some cases the conference papers contain material overlapping with the journal papers.

Journal Papers

- D.B. Ward, R.A. Kennedy and R.C. Williamson, "Theory and design of broadband sensor arrays with frequency invariant far-field beam patterns", *J. Acoustical Society of America*, vol. 97, no. 2, pp. 1023-1034, Feb. 1995.
- D.B. Ward, R.A. Kennedy, and R.C. Williamson, "FIR filter design for frequency invariant beamformers", *IEEE Signal Processing Letters*, vol. 3, no. 3, pp. 69-71, Mar. 1996.
- D.B. Ward, Z. Ding, and R.A. Kennedy, "Direction of arrival estimation for wideband signals using frequency invariant beamspace processing", *IEEE Trans. Signal Processing*, (to be submitted).
- R.A. Kennedy, T. Abhayapala, and D.B. Ward, "Broadband nearfield beamforming using a radial beampattern transformation", *IEEE Trans. Signal Processing*, (to be submitted).
- P.J. Kootsookos, D.B. Ward and R.C. Williamson, "Imposing pattern nulls in broadband array responses", *J. Acoustical Society of America*, (to be submitted).

Conference Papers

- D.B. Ward, R.A. Kennedy and R.C. Williamson, "Design of frequency-invariant broadband far-field sensor arrays", in *1994 IEEE Antennas and Propagation Society Int. Symp. Digest*, vol. 2, pp. 1274-1277, Seattle, USA, June 1994.

- D.B. Ward, R.A. Kennedy, and R.C. Williamson, "Broadband beamforming with a single set of filter coefficients", in *Proc. 1995 IEEE Singapore Int. Conf. on Signal Processing, Circuits and Systems*, pp. 88-93, Singapore, July 1995.

- D.B. Ward, Z. Ding, and R.A. Kennedy, "Broadband direction of arrival estimation using frequency-invariant beam-space processing", in *Proc. IEEE Int. Conf. on Acoust., Speech, and Signal Processing, (ICASSP'96)*, pp. 2892-2895, Atlanta, USA, May 1996.

- R.A. Kennedy, T. Abhayapala, D.B. Ward, and R.C. Williamson, "Near-field broadband frequency invariant beamforming", in *Proc. IEEE Int. Conf. on Acoust., Speech, and Signal Processing, (ICASSP'96)*, pp. 905-908, Atlanta, USA, May 1996.

- P.J. Kootsookos, D.B. Ward, and R.C. Williamson, "Frequency invariant beamforming with exact null design", in *Proc. IEEE Workshop on Statistical Signal and Array Processing, (SSAP'96)*, pp. 105–108, Corfu, Greece, June 1996.

The research presented in this thesis has been performed jointly with Dr Rodney A. Kennedy, Dr Robert C. Williamson, Dr Peter J. Kootsookos, Prof. Zhi Ding (Auburn University, USA) and Mr Thushara Abhayapala. The majority, approximately 70%, of this work was my own.

Darren Brett Ward

Australian National University
July 1996

Abstract

In many engineering applications, including radar, sonar, communications and seismology, the direction of impinging signal wavefronts can be used to discriminate between competing sources. Often these source signals cover a wide bandwidth and conventional narrowband beamforming techniques are ineffective, since spatial resolution varies significantly across the band. In this thesis we consider the problem of beamforming for broadband signals, primarily when the spatial response remains constant as a function of frequency. This is called a *frequency invariant beamformer* (FIB).

Rather than applying the numerical technique of multi-parameter optimisation to solve for the beamformer parameters, we attempt to address the fundamental nature of the FIB problem. The general philosophy is to use a theoretical continuous sensor to derive relationships between a desired FI beampattern and the required signal processing structure. Beamforming using an array of discrete sensors can then be formulated as an approximation problem. This approach reveals a natural structure to the FIB which is otherwise buried in a numerical optimisation procedure.

Measured results from a microphone array are presented to verify that the simple FIB structure can be successfully implemented. We then consider imposing broadband pattern nulls in the FI beampattern, and show that (i) it is possible to impose an exact null which is present over all frequencies, and (ii) it is possible to calculate *a priori* how many constraints are required to achieve a null of a given depth in a FIB. We also show that the FIB can be applied to the problem of broadband direction of arrival (DOA) estimation and provides computational advantages over other broadband DOA estimators.

Through the theoretical continuous sensor approach, we show that the FIB theory can be generalised to the problem of designing a *general broadband beamformer* (GBB) which realizes a broadband angle-versus-frequency beampattern specification. Coupled with a technique for radial beampattern transformation, the GBB can be applied to a wide class of problems covering both *nearfield* beamforming (in which the shape of the impinging wavefront must be considered) and *farfield* beamforming (which is simplified by the assumption of planar wavefronts) for a broadband beampattern specified over both angle and frequency.

Acknowledgements

The work presented in this thesis would not have been possible without the support of the following people and organisations, and they are gratefully acknowledged below:

My supervisors Dr Rod Kennedy and Dr Bob Williamson, for their guidance, insight, and enthusiasm, and individually for their unique wit.

Dr Peter Kootsookos, for his many suggestions on this work, and for helping me make annoying noises in public places in order to test the microphone array.

My advisor Dr Iven Mareels, for his encouragement and advice, and for his valuable comments at my mid-term review.

The Cooperative Research Centre for Robust and Adaptive Systems for funding a Graduate Student Assistantship; the Commonwealth Government for an Australian Postgraduate Award; and the Australian Research Council for financial support for laboratory equipment.

The Head of Department, Prof. Darrell Williamson, for the use of departmental facilities in the production of this thesis.

My fellow students in the Graduate Program in Engineering, for their friendship and for providing me with a library.

My family, for all they have given me in terms of education, encouragement, and moral and financial support.

And finally Belle, for her companionship, her unwaivering support, for understanding when I had to spend so much time working and so little time with her, and for her cute smile.

Contents

Glossary of Definitions

Notation

$\mathbb{C}$	complex plain
$\mathbb{R}$	real numbers
$\mathbb{R}^+$	non-zero real numbers
$\mathbb{Z}$	integers
$\mathbb{N}$	natural numbers
a^*	complex conjugate of scalar a
$\mathbf{a}^T$	transpose of matrix or vector $\mathbf{a}$
$\mathbf{a}^H$	conjugate transpose of matrix or vector $\mathbf{a}$
$\mathbf{A}^\dagger$	matrix pseudo-inverse: $\mathbf{A}^\dagger \triangleq [\mathbf{A}^H\mathbf{A}]^{-1}\mathbf{A}^H$
$\otimes$	Kronecker product: $\mathbf{a}\otimes\mathbf{b} \triangleq [a_1\mathbf{b}\cdots a_N\mathbf{b}]$, where N is length of $\mathbf{a}$
$\star$	convolution: if $z[k] = x[k]\star y[k]$, then $z[k] \triangleq \sum_{l=0}^{k} x[k-l]y[l]$
$\delta[\cdot]$	Kronecker delta: $\delta[k] \triangleq 1, k=0$, and $\delta[k] \triangleq 0, k \neq 0$
$E\{\cdot\}$	expectation operator
$\mathcal{F}\{\cdot\}$	Fourier transform operator
$\mathrm{Re}\{\cdot\}$	real part
$\mathrm{Im}\{\cdot\}$	imaginary part

Abbreviations

CFVB	Controlled Frequency Variant Beamformer
CSS	Coherent Signal Subspace
DOA	Direction of Arrival
FI	Frequency Invariant
FIB	Frequency Invariant Beamformer
GBB	General Broadband Beamformer
LCMV	Linear Constrained Minimum Variance

Chapter 1

Introduction

ARRAY signal processing uses a group of sensors (an *array*) to extract information from our environment, specifically from signals propagating as waves [50]. Common examples of such signals are radio and television transmissions and human speech.

In some cases these signals may be separated through their frequency components alone. For example, in commercial radio transmission each station is assigned a different frequency band. Different stations may then be received by tuning to different frequencies. However, in many situations competing signals have overlapping frequency bands and cannot be separated by their frequency components alone. Consider the situation in which you are able to pick out a single voice amongst many in the “cocktail-party” environment in which several conversations occur simultaneously. These speech signals will largely occupy the same frequency band (except for slight differences between speakers, for example, between male and female voices). Despite this, our ears (a two-sensor acoustic array)—together with the processing done in the brain—are very effective at accurately determining the direction of sound waves. This allows us to discriminate between several sources based on their directions of origin.

A beamformer is a processor used in conjunction with an array of sensors to provide spatial filtering [97]. This means that signals from a given spatial region are passed and signals from other regions are attenuated, with the usual objective of estimating a desired signal in the presence of noise and interfering signals. An example beamformer response is shown in Fig. 1.1. This shows the beamformer output power versus the direction of arrival of the impinging wavefront. Signals arriving from a direction of 0° will be passed by the beamformer, whereas signals from other directions are attenuated.

Typical beamforming applications include exploration seismology (in which an array of geophones receives signals reflected from a region inside the earth with the objective of

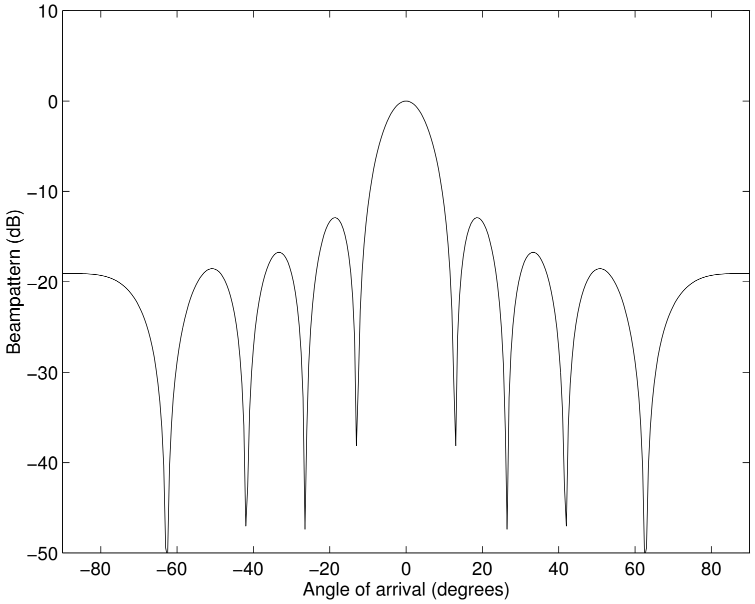


Figure 1.1: Typical response of a beamformer.

detecting minerals), sonar (in which an array of hydrophones is used to passively detect sources), radar (in which a transmitting array is used to illuminate an area surrounding the radar site and a receiving array looks for reflections from targets), and medical imaging (in which arrays are used to form cross-sectional images of objects from either transmitted or reflected data) [42].

In many cases the signals impinging on the array cover a wide bandwidth. Speech and sonar signals, for example, typically cover several octaves (approximately 200–3400 Hz for intelligible speech and 50–1000 Hz for the signals received by a towed sonar array). When using a beamformer to discriminate between broadband signals, it is desirable to have constant spatial resolution over the bandwidth of interest. A beamformer which has a constant spatial response over a specified wide frequency band will be referred to as a *broadband frequency invariant beamformer*. The response of an example frequency invariant beamformer design is shown in Fig. 1.2.[1] This figure illustrates the desired constant spatial response as a function of frequency. Such a beamformer is useful in environments in which source and noise signals are predominantly broadband in nature. An example application is speech acquisition with a microphone array, in which several speech signals may be received by the array—either from different speakers or reverberations from surrounding objects—but only one should be passed by the beamformer.

[1] This beamformer was designed using the methods outlined in Chapter 3.

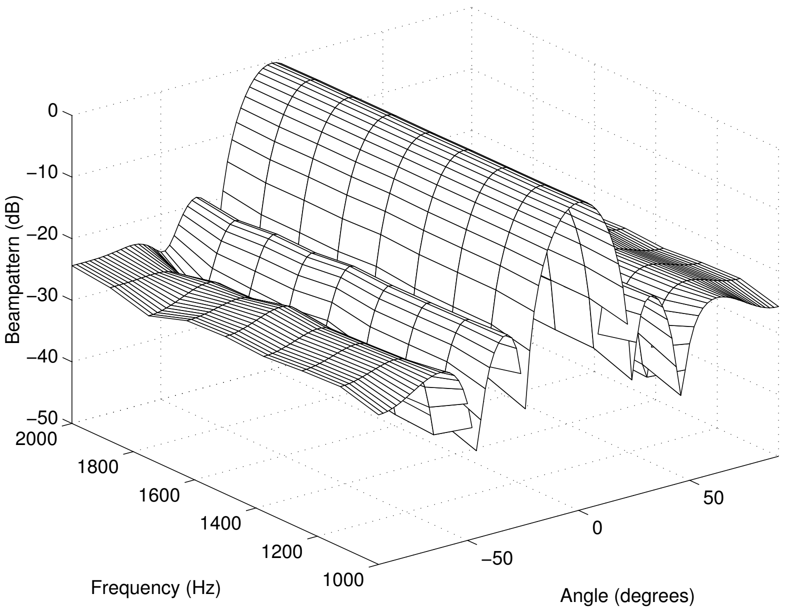


Figure 1.2: Typical response of a frequency invariant beamformer.

Several methods have been suggested to solve the problem of frequency invariant beamforming (these methods are reviewed in Chapter 2). However, most of these methods are limited in their generality, relying either on specific array geometries (usually uniformly spaced linear arrays) or specific beampatterns.

Hence, the primary motivation for this thesis is to develop a general theory of frequency invariant beamforming which is applicable to a wide class of array geometries, allows arbitrary beam shapes, and can be used over an arbitrarily wide bandwidth. In developing this theory, we formulate solutions to a wider class of problems. These are specified in §1.2.

The general philosophy of our approach is that, rather than dealing directly with a discrete set of sensors, we formulate the broadband beamforming problem in terms of a continuously distributed sensor. The practical problem of beamforming with a finite sensor array is then reduced to that of approximating the response of the theoretical continuous sensor. For the frequency invariant beamformer, this formulation is important from several perspectives. First, it allows an exact solution to the frequency invariant problem to be derived—it is impossible to produce an exactly frequency invariant beampattern using a finite number of discrete sensors. (This is discussed further in Chapter 2 and is illustrated by Fig. 1.2 which shows the slight frequency variation displayed by a typical frequency invariant beamformer realization.) Second, it provides insight into the intrinsic structure of the frequency invariant beamformer, thereby simplifying its implementation. Finally, it allows mathematical relationships to be derived between the desired frequency invariant beampattern and the design parameters, namely the filter coefficients. We consider this philosophy in further detail in §1.2 when we outline the contributions of the thesis.

The thesis is loosely grouped into two sections: the first part (up to and including Chapter 4) develops the frequency invariant beamforming theory; the remaining chapters examine some applications of this theory. A summary of the thesis is as follows.

- The remainder of this chapter reviews background beamforming theory and lists the contributions of the thesis.
- Chapter 2 presents the theory of frequency invariant beamforming.
- Chapter 3 describes implementation of the frequency invariant beamformer using discrete-time processing. Experimental results obtained from a microphone array are also presented.
- Chapter 4 investigates imposing broadband pattern nulls in the frequency invariant beamformer response.
- Chapter 5 generalises the frequency invariant beamforming theory to the problem of producing a broadband beampattern which is specified over both angle and frequency. A technique for radially transforming a beampattern is developed which, when combined with the general broadband beamforming theory, can be used to design a *nearfield* broadband beamformer.
- Chapter 6 applies the frequency invariant beamformer to the problem of estimating the directions of arrival of multiple broadband signals.
- Chapter 7 concludes the thesis and presents some topics for further research.

1.1 Background Beamforming Theory

Having outlined the problem statement in general terms in the previous section, it is now necessary to give some background beamforming theory. This will set the context for the contributions of the thesis, which are listed at the end of this chapter.

1.1.1 Beamformer Classification

A beamformer is labelled as either narrowband or broadband, depending on the bandwidth of the signal environment in which it is designed to operate. In this thesis we consider a broadband beamformer to be one in which the signal bandwidth is a *significant fraction* (say 0.5) of the mid-band frequency. Although this is a somewhat arbitrary definition, it serves to exclude some so-called wideband beamformers in the radio antenna literature which have fractional bandwidths of a few percent of the mid-band frequency. Most of the beamforming literature is concerned with narrowband beamforming. Dealing with broadband beamforming is significantly more complicated, because of the number of variables involved.[2]

Beamformers may also be classified as either data independent or statistically optimum. A data independent beamformer is designed to produce a predetermined response regardless of the signal environment, whereas the goal of statistically optimum beamforming is to optimise the beamformer response based on the statistics of the received array data. Although the scope of this thesis generally covers broadband data independent beamforming, an attempt has been made to allow for adaptive implementation in the formulations presented. Hence, the majority of the methods presented may be readily adopted for adaptive use. By only considering fixed beamforming we attempt to gain a better understanding of the performance which is actually possible using adaptive algorithms. A simple example adaptive algorithm is included in Chapter 3 for illustration.

The vast majority of beamformer literature deals only with the situation in which the impinging wavefronts are planar in nature. This is referred to as the *farfield* condition. A wave propagating from a point source radiates spherically outwards from the source location. Thus, for an array suitably close to the point source, the impinging wave will have a spherical wavefront. The assumption of planar wavefronts is only valid for an array located at a significant distance from a point source. A typical rule of thumb is that farfield operation can be assumed at a distance of $2L^2/\lambda$, where L is the total array length and λ is the wavelength [61]. Chapter 5 presents a new design methodology for a broadband beamformer with a desired nearfield beampattern; further mention of nearfield beamforming is deferred until then. Unless specifically stated, farfield operation should be assumed in the remainder of the thesis.

1.1.2 Narrowband Beamforming

A block diagram of a conventional narrowband linear beamformer is shown in Fig. 1.3. The output of the beamformer at time t is given by a linear combination of the weighted

[2] The broadband beamforming problem is multidimensional, covering the sensor locations, direction of impinging wavefronts, time, and frequency. Since the operating frequency is fixed in narrowband beamforming, the frequency (and often time) dimension can be excluded.

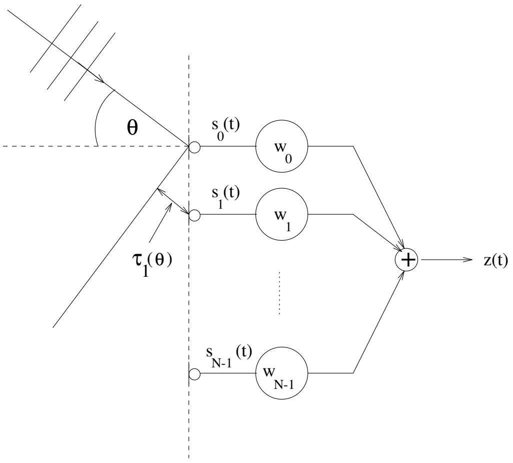


Figure 1.3: Block diagram of a conventional narrowband linear beamformer.

data at each of the N sensors, i.e.,

$$z(t) = \sum_{n=0}^{N-1} w_n \, s_n(t),$$

where w_n is the complex weight applied to the nth sensor, and $s_n(t)$ is the signal received at the nth sensor at time t. Assume that the signal impinging on the sensor array is a complex plane wave with direction of arrival θ (measured relative to array *broadside*, i.e., normal to the array axis) and frequency f. Also assume that the phase of the received signal is zero at the zeroth sensor, giving, $s_0(t) = \exp(\jmath 2\pi f t)$ and $s_n(t) = \exp[\jmath 2\pi f(t + \tau_n(\theta))]$, where $\tau_n(\theta)$ represents the time delay due to plane wave propagation from the zeroth to the nth sensor. For a farfield point source, $\tau_n(\theta) = (x_n - x_0)c^{-1} \sin\theta$, where x_n is the location of the nth sensor and c is the speed of wave propagation.

The beamformer output is then given by

$$z(t) = \exp(\jmath 2\pi f t) \sum_{n=0}^{N-1} w_n \exp[\jmath 2\pi f \tau_n(\theta)].$$

Conventionally, the time dependence is ignored, and we are only concerned with the *beamformer response* defined as

$$r(\theta) = \sum_{n=0}^{N-1} w_n \exp[\jmath 2\pi f \tau_n(\theta)]. \tag{1.1}$$

The *beampattern* is defined as the squared magnitude of the beamformer response, $|r(\theta)|^2$, and is usually expressed in decibels.

To simplify notation, the response is often expressed in vector form as

$$r(\theta) = \mathbf{w}^H \mathbf{a}(\theta) \tag{1.2}$$

where $\mathbf{w} = [w_0, \ldots, w_{N-1}]^H$ is the N dimensional weight vector, and

$$\mathbf{a}(\theta) = [\exp(\jmath 2\pi f \tau_0(\theta)), \ldots, \exp(\jmath 2\pi f \tau_{N-1}(\theta))]^T \tag{1.3}$$

is the N dimensional narrowband *array response vector* (also referred to as the steering vector, direction vector, and array manifold vector in the literature).

The beampattern is often expressed as a function of $u = \sin\theta$. In this case one is usually only concerned with the response at values of u corresponding to real angles θ, i.e., $u \in [-1, 1]$. This region is referred to as the *visible region* [50, p.91].

Narrowband Beam Shaping Techniques

Figure 1.1 shows the features of a typical beampattern. A single main beam (the *mainlobe*) pointed in the direction of the desired source passes signals from that direction (broadside in the figure), while signals arriving from other direction are attenuated by the beamformer. The smaller beams are referred to as *sidelobes*. The main beam may be steered to directions other than broadside by introducing a progressive phase delay across the array. Specifically, if the main beam is to be pointed to an angle ϕ then a phase delay of $2\pi f_0 \tau_n(\phi)$ should be applied to the nth sensor (where f_0 is the frequency of operation).

Several classical techniques exist in the antenna literature for shaping the beampattern. The major methods are briefly outlined below (see [41, 60, 61] for a full review of these methods).

The simplest beam shaping method is the Fourier transform method [61, p.112]. For an equally spaced array, (1.1) reduces to a Fourier series, and the sensor weights may be found by Fourier transforming the desired pattern. This method provides the least mean square error approximation of the desired pattern.

The method of Schelkunoff [79] again relies on equal spacings, and expresses (1.1) as a polynomial of degree $(N-1)$:

$$r(\theta) = \sum_{n=0}^{N-1} w_n \, z^n, \tag{1.4}$$

where $z = \exp(\jmath 2\pi f d c^{-1} \sin\theta)$ and d is the inter-sensor spacing. The pattern is shaped by changing the zero positions of this polynomial.

Based on this identification of the array response as a polynomial, the Dolph-Chebyshev method [21] equates the array polynomial with a Chebyshev polynomial to produce a pattern with the narrowest mainlobe width for a given constant sidelobe level.

Taylor [92] noted that the Chebyshev pattern is inefficient in that the requirement for a constant sidelobe height requires that for large arrays increasingly more of the energy is in the sidelobes. Instead, Taylor proposed a beampattern function with zeros far from the mainlobe at locations corresponding to those of a uniformly weighted array, and the zeros close to the mainlobe chosen similar to those of the Chebyshev pattern. This method was based on a continuous source function rather than an array, although it was later extended to arrays by Villeneuve [98]. Several modifications to the basic Taylor pattern have been made which allow asymmetric sidelobe levels [22], individually specified sidelobe levels [23], or specified pattern nulls [94].

Woodward pattern synthesis [103, 104] combines a set of orthogonal beams (formed either from an array or continuous aperture) to produce an arbitrary desired pattern. These orthogonal beams are linearly independent at a discrete set of points in $\sin\theta$ space. That is, at each of these points in $\sin\theta$ space, one of the beams will have a peak and all other beams will have a zero. For an array of N equally spaced elements, there are N such beams. The pattern is formed by summing the orthogonal beams, with each beam weighted according to the value of the desired pattern at its peak location.

The goal of an iterative power pattern synthesis technique due to Orchard *et al.* [72] is to best approximate a shaped pattern within a given spatial region, while maintaining control of the sidelobe level in the remainder of the pattern. This method is based on an equally spaced array geometry.

As indicated above, the majority of classical beam shaping techniques rely on uniform sensor spacings. With this assumption the beamformer response (1.1) can be expressed as a polynomial (1.4), and many methods exploit this property to simplify the beam shaping problem. Introducing nonuniform sensor spacings significantly complicates the beam shaping problem. Most methods which are applicable to nonuniform sensor spacings are either iterative [69, 76, 105] or use some form of numerical optimisation [67]. These tend to be mechanical computational processes, which provide no insight into the structure of the problem. A more fundamental approach was taken by Ishimaru [47] who used Poisson's summation formula to formulate an analytical solution to the problem of designing a beamformer with a nonuniformly spaced array geometry. In a similar vein this thesis presents an intuitive and more structured method of dealing with nonuniformly spaced arrays, and all of the techniques presented are applicable to general array geometries (with mild constraints as described in later chapters).

Narrowband Pattern Nulling

To suppress interference from unwanted sources, nulls can be imposed in the beampattern at the appropriate directions. Most interference cancelling methods use adaptive techniques (as described in the following section) to automatically place nulls in the directions of the dominant jammers. Another approach is to design the beamformer such that the beampattern is close to some desired beampattern, and additionally displays nulls at the required directions. For the case of $M < N$ interferers (where N is the number of sensors), M degrees of freedom are used to cancel the interferers with the remaining $N-M$ degrees of freedom used to approximate the desired beampattern. The problem can be stated mathematically as:

$$\min \int \varphi(\theta)\,|r_d(\theta) - r(\theta)|^2 \;d\theta,$$
$$\text{subject to } r(\theta_m) = 0, \quad m = 1, \ldots M,$$

where $r_d(\theta)$ is the initial quiescent beampattern with no nulling, $\theta_m, m = 1 \ldots M$ are the interference directions, and $\varphi(\theta)$ is a weighting function.

It has been shown [84] that for the case of a linear equally spaced array (and a uniform weighting function), the optimum solution for the constrained beampattern is the superposition of the quiescent beampattern and a set of M weighted $(\sin Nx/\sin x)$ beams, with each beam centred exactly at the corresponding pattern null.

Narrowband Adaptive Array Techniques

The conventional narrowband beamformer described above may not provide the amount of interference suppression required in many situations. Better noise suppression is afforded through statistically optimum techniques. These techniques attempt to choose the array weights such that the desired signal is enhanced and the interfering noise sources are suppressed, based on the statistics (usually second order) of the received array data [63]. Since the statistics of the array data are usually not known, the optimum solution is estimated from the available data in a time-dependent (or adaptive) fashion. Hence, these beamformers are referred to as *adaptive arrays*. In what follows we assume that the second order statistics of the array data are used.

Many criteria have been proposed for choosing statistically optimum beamformer weights. Three of these will be briefly described to give a flavour of the available methods (see [33, 36, 45, 63, 97] for a complete review).

To set notation, consider an array of N sensors. Let $\mathbf{y}$ be the N vector of received signals at the array inputs. To simplify notation the time dependence of $\mathbf{y}$ is ignored. The beamformer output is

$$z = \mathbf{w}^H \mathbf{y},$$

where $\mathbf{w}$ is the N vector of array weights. The average output power of the beamformer is given by

$$\begin{aligned} \mathbf{R}_z &\triangleq E\{zz^*\} \\ &= \mathbf{w}^H \mathbf{R}_y \mathbf{w} \end{aligned}$$

where $\mathbf{R}_y = E\{\mathbf{y}\mathbf{y}^H\}$ and $E\{\cdot\}$ denotes expectation. The goal of adaptive arrays is to use the statistics of the received array data to estimate the desired signal.

1. Reference Signal: If the desired signal were known, then the mean square error between the actual output $z = \mathbf{w}^H \mathbf{y}$ and the desired signal could be minimised. In practice, enough may be known about the desired signal that it may be represented by a reference signal z_d. For example, if the desired signal is an amplitude modulated signal, the carrier could be used as the reference signal. This approach was considered by Widrow *et al.* [101]. The criterion used is

$$\min_{\mathbf{w}} E\{|z_d - \mathbf{w}^H \mathbf{y}|^2\},$$

and the optimum weights are

$$\mathbf{w} = \left[E\{\mathbf{y}\mathbf{y}^H\}\right]^{-1} E\{\mathbf{y} z_d^*\}.$$

2. Multiple Sidelobe Canceller: The multiple sidelobe canceller [2] consists of a main, high gain channel (which may be either a single directive antenna or one of the conventional beamformers described above) pointed at the desired source and N auxiliary channels which are beamformed to form a single auxiliary output. This single auxiliary output is subtracted from the main channel output with the aim of removing the interference in the main channel. The auxiliary channels receive noise which is correlated to the noise present in the main channel, and the goal is to choose the auxiliary channel weights to cancel the main channel interference component. The criterion used is to minimise the expected value of the total output power, i.e.,

$$\min_{\mathbf{w}_a} E\{|z_m - {\mathbf{w}_a}^H \mathbf{y}_a|^2\},$$

where z_m is the output of the main channel, and $\mathbf{w}_a$ and $\mathbf{y}_a$ denote the weights and data of the auxiliary channels respectively. The optimum weights are

$$\mathbf{w}_a = \left[E\{\mathbf{y}_a {\mathbf{y}_a}^H\}\right]^{-1} E\{\mathbf{y}_a z_m^*\}.$$

Because the total output power is minimised, cancellation of the desired signal may occur since it also contributes to the total output power. Hence the multiple sidelobe canceller is only effective in situations where the desired signal is either very weak, or is known to be absent during certain time periods (within these periods the auxiliary beamformer weights are adapted).

3. Application of Linear Constraints: A more flexible approach to adaptive beamforming is provided by the linear constrained minimum variance (LCMV) beamformer. The idea here is to constrain the beamformer response so that signals from a desired direction are passed while the total output power of the system is minimised. This has the effect of suppressing unwanted noise and interference while passing the desired signal with specified gain and phase. The constraint is written in general form as $\mathbf{C}^H\mathbf{w} = \mathbf{f}$. To constrain the response to have a gain of g in the direction θ_d, the constraint reduces to $\mathbf{a}(\theta_d)^H\mathbf{w} = g^*$. Recalling that the expected value of the output power is $E\{|\mathbf{w}^H\mathbf{y}|^2\}$, the LCMV criteria is

$$\min_{\mathbf{w}} \mathbf{w}^H\mathbf{R}_y\mathbf{w} \quad \text{subject to} \quad \mathbf{C}^H\mathbf{w} = \mathbf{f},$$

where $\mathbf{R}_y = E\{\mathbf{y}\mathbf{y}^H\}$. This may be solved using Lagrange multipliers to give

$$\mathbf{w} = {\mathbf{R}_y}^{-1}\mathbf{C}\left[\mathbf{C}^H{\mathbf{R}_y}^{-1}\mathbf{C}\right]\mathbf{f}. \tag{1.5}$$

The single linear constraint is easily extended to L multiple constraints by making $\mathbf{C}$ an $N \times L$ matrix and $\mathbf{f}$ an L dimensional vector. For example, if it is desired to pass signals from direction θ_1 with gain g_1 and signals from direction θ_2 with gain g_2, then the constraints are written

$$\begin{bmatrix} \mathbf{a}(\theta_1)^H \\ \mathbf{a}(\theta_2)^H \end{bmatrix} \mathbf{w} = \begin{bmatrix} g_1^* \\ g_2^* \end{bmatrix}.$$

There are several different types of constraints which may be applied including point (as in the above example), derivative [26], and eigenvector [9]. The design of adaptive antennas which are robust to errors in the assumed model (such as errors in the assumed look direction, array weight errors, errors in the assumed signal bandwidth, etc.) is presented in [1, 27].

An alternate formulation of the LCMV problem is provided by Griffiths and Jim [37]. This formulation, the generalised sidelobe canceller, essentially transforms the constrained LCMV problem into an unconstrained form, and provides insight and simplifies implementation.

Second Order Statistics: All of these optimum solutions assume knowledge of the second order statistics of the array data. In practice this knowledge is rarely available, but with the assumption of ergodicity it may be estimated from the available data. There are two basic approaches: (i) estimate the data covariance matrix by time averaging the array data, i.e.,

$$\widehat{\mathbf{R}}_y = \frac{1}{T}\sum_{t=1}^{T}\mathbf{y}(t)\mathbf{y}(t)^H,$$

or (ii) use a recursive technique, such as the well known least mean squares (LMS) algorithm, to adapt the weights such that they converge towards the optimal solution. Refer to [45, 63] for a description of adaptive beamforming algorithms.

Narrowband Direction of Arrival Estimation

Another important problem in narrowband beamforming is the estimation of the directions of arrival (DOAs) of multiple source signals. For example, in passive sonar, the received signals from an array of hydrophones are processed to estimate the direction (and usually range) of sources.

Consider an array of N sensors, with $D < N$ farfield source signals arriving from directions $\Theta = [\theta_1, \dots, \theta_D]$, where θ_d is the direction to the dth source. The N vector of array outputs at time t is

$$\mathbf{y}(t) = \mathbf{A}(\Theta)\,\mathbf{s}(t) + \mathbf{n}(t), \tag{1.6}$$

where $\mathbf{A}(\Theta) = [\mathbf{a}(\theta_1), \dots, \mathbf{a}(\theta_D)]$ is the $N \times D$ source direction matrix, $\mathbf{s}(t)$ is the D vector of source signals received at the first sensor, and $\mathbf{n}(t)$ is the N vector of additive white noise terms, assumed to be Gaussian distributed. The problem is to determine the source directions Θ from the array data vector $\mathbf{y}(t)$.

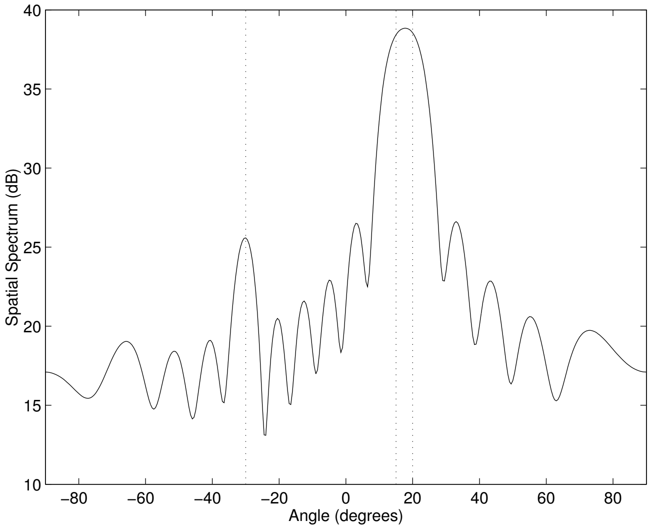


Figure 1.4: Beamformer-based spatial spectrum for 3 sources impinging from $\Theta = [-30, 15, 20]$ with powers of -3 dB, 6 dB, and 6dB respectively.

The simplest technique for determining the source directions is through the beamformer-based spatial spectrum, defined as

$$B(\theta) = \mathbf{a}(\theta)^H \mathbf{R}_y \mathbf{a}(\theta),$$

where

$$\mathbf{R}_y = E\{\mathbf{y}(t)\mathbf{y}(t)^H\}$$

is the array data covariance matrix. The estimated source directions are given by the D values of θ for which $B(\theta)$ is maximised. This method has the advantage of computational efficiency and ease of implementation. However, its main drawback is that the resolution is completely determined by the array size.

An example spatial spectrum is shown in Fig. 1.4 for three sources impinging on a 15 element narrowband array. This figure clearly shows the poor resolution of this method; sources closer together than a beamwidth cannot be resolved. The resolution capability of the beamformer-based method for two closely spaced sources has recently been presented [110].

To overcome the lack of spatial resolution, the so-called *high resolution methods* were developed. These methods are based on exploiting the eigen-structure of the data covariance matrix. An example is the multiple signal classification (MUSIC) technique [80]

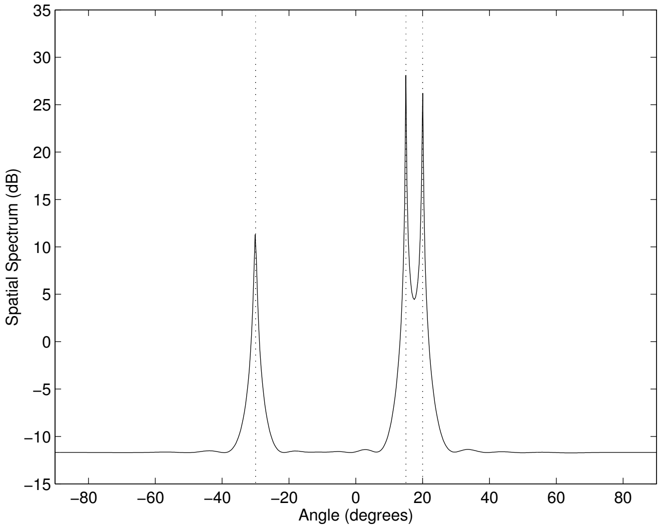


Figure 1.5: MUSIC spatial spectrum for 3 sources impinging from $\Theta = [-30, 15, 20]$ with powers of -3 dB, 6 dB, and 6dB respectively.

which is discussed in detail in Chapter 6. Figure 1.5 shows the corresponding MUSIC spatial spectrum for the example used in Fig. 1.4. Clearly, MUSIC is able to discriminate between closely clustered sources. The statistical properties of the MUSIC estimator have been analysed, including the estimate variance [87], bias [107], and the probability of resolution of two closely spaced sources [109].

1.1.3 Broadband Beamforming

Narrowband beamforming methods assume the signal bandwidth is sufficiently narrow that it may be considered to consist of only a single frequency. In practice, all signals of interest have a non-zero bandwidth, and in many cases the fractional bandwidth is very wide (a significant fraction of the central frequency). For example, sonar signals and speech signals both cover several octaves. In these situations, the assumption of a single frequency produces poor results. For example, Fig. 1.6 shows the response of a narrowband beamformer designed for operation at 1000 Hz, operated over a bandwidth of 500–3000 Hz. At frequencies below 1000 Hz the main beam spreads out and spatial resolution is lost. At frequencies above 1000 Hz the main beam becomes narrower until, at approximately 1800 Hz grating lobes (i.e., periodic repetitions of the mainlobe) begin to appear in the beampattern. The appearance of grating lobes is referred to as *spatial aliasing* and is caused by undersampling the spatial waveform, in the same way as temporal aliasing is introduced in time-sampled signals if the sampling rate does not satisfy the

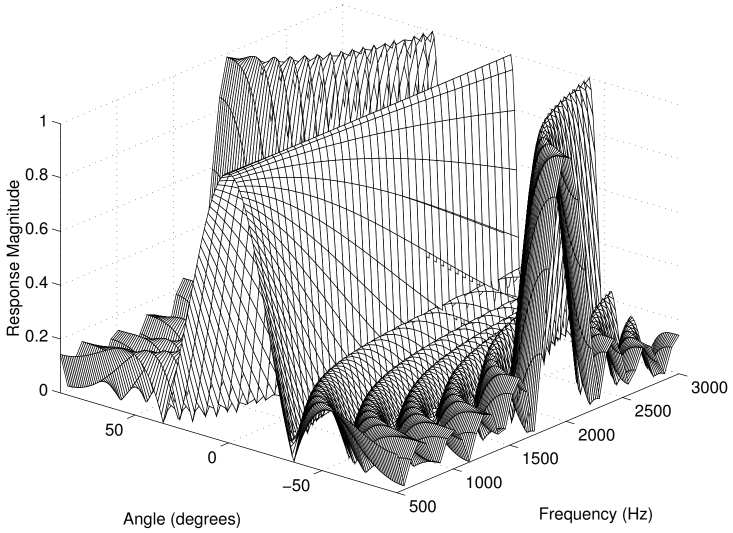


Figure 1.6: Response of a narrowband beamformer operated over a broad frequency range.

Nyquist criterion. Clearly, the spatial resolution varies significantly with frequency; this is unacceptable for broadband applications.

In RF applications, a broadband antenna generally contains a self-similar geometry which allows its pattern and impedance characteristics to be essentially frequency invariant [13]. Examples of such antennas are the spiral antenna and log-periodic dipole array, shown in Fig. 1.7. The feature of these antennas is that the size of the effective radiating

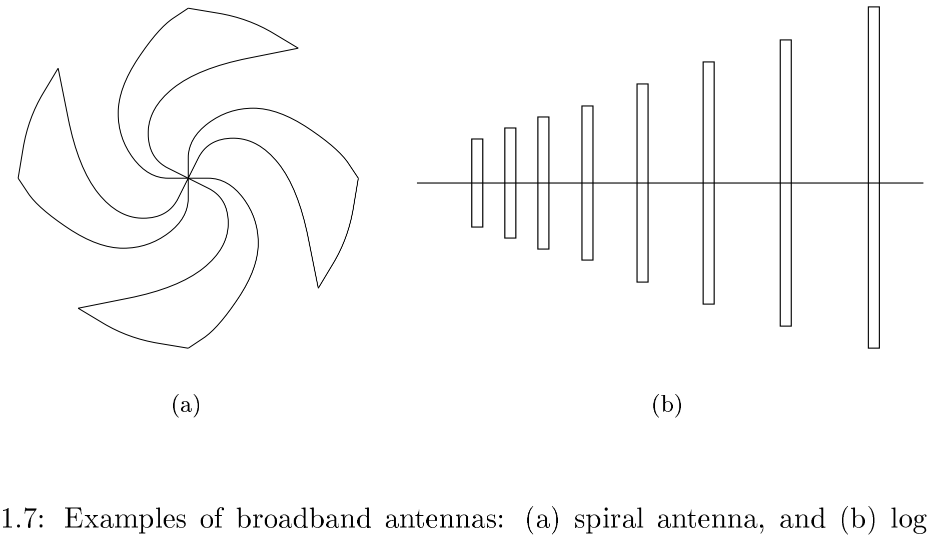


Figure 1.7: Examples of broadband antennas: (a) spiral antenna, and (b) log-periodic dipole array.

aperture in terms of wavelength does not change. For example, the log-periodic dipole array consists of dipoles of unequal lengths and nonuniform spacings (with the lengths and spacings satisfying a geometric relationship). At the operating frequency the active part of the array consists mainly of those dipoles whose lengths are approximately a half-wavelength. Because of the self-similarity of the array geometry, the antenna response will be approximately frequency independent over a wide frequency band.

Broadband beamforming concerns the problem of providing spatial filtering in an environment in which the signals of interest cover a wide frequency band. In such an environment the narrowband beamformer shown in Fig. 1.3 is not particularly useful, since the beampattern varies with frequency in an uncontrolled (yet predictable) way, as demonstrated by Fig. 1.6. It is common in broadband beamforming to replace the sensor weights in Fig. 1.3 by sensor filters. The broadband beamformer response is given by

$$r(\theta, f) = \sum_{n=0}^{N-1} H_n(f) \exp[\jmath 2\pi f \tau_n(\theta)], \tag{1.7}$$

where $H_n(f)$ is the filter on the nth sensor. In a digital system, the sensor filters are implemented by L-tap FIR filters as

$$H_n(f) = \sum_{k=0}^{L-1} h_n[k] \exp[-\jmath 2\pi f T k],$$

where T is the sampling period, and $h_n[\cdot]$ are the filter coefficients. Such a structure is referred to as a tapped delay line beamformer or simply a *broadband beamformer.*

The response of a broadband beamformer may be written in vector form as

$$r(\theta, f) = \mathbf{h}^H \mathbf{d}(\theta, f),$$

where

$$\mathbf{h} = \Big[h_0[0], \dots, h_{N-1}[0], \dots, h_0[L-1], \dots, h_{N-1}[L-1]\Big]^H$$

is the NL vector of filter weights,

$$\mathbf{d}(\theta, f) = \mathbf{e}(f) \otimes \mathbf{a}(\theta, f)$$

is the NL delay vector, $\otimes$ denotes the Kronecker product,

$$\mathbf{e}(f) = \Big[1, e^{-\jmath 2\pi f T}, \dots, e^{-\jmath 2\pi f (L-1) T}\Big]^T$$

is the L dimensional Fourier transform vector, and

$$\mathbf{a}(\theta, f) = \Big[e^{\jmath 2\pi f \tau_0(\theta)}, \dots, e^{\jmath 2\pi f \tau_{N-1}(\theta)}\Big]^T$$

is the N dimensional broadband array response vector.

Broadband Beam Shaping Techniques

A significant difference between narrowband and broadband beamformers is in the beam steering techniques employed. Whereas beam steering can be achieved in a narrowband beamformer using phase delays, true time delays are required in a broadband beamformer to implement beam steering. In RF applications this requires switching in added lengths of transmission line to effect the time delay. Fractional delay FIR filters [56] can be used in digital beamforming to produce the required time delay.

Most of the techniques described in §1.1.2 for shaping a narrowband beampattern are not applicable to broadband beamforming with the tapped delay line structure described above. One approach is to carry out an FFT at each sensor, divide the data into separate narrow frequency bins, and perform narrowband beamforming (using the structure of Fig. 1.3) in each narrow frequency band. This is referred to as *frequency decomposition.* The major disadvantage with this method as opposed to the tapped delay line beamformer is the additional computation required by the FFT operation.

It has been noted in [43] that, for the case of an equally spaced array, there is a two dimensional Fourier transform relationship between the filter coefficients and the beampattern of a broadband beamformer. This provides a simple means of designing the sensor filters for a desired broadband beampattern. However, as explained in Chapter 2, we believe it is more appropriate to use a nonuniformly spaced array for broadband beamforming, and thus seek a more generally applicable solution.

For a general array geometry, the following L_2 optimisation method may be used. Let $r_d(\theta, f)$ be a desired broadband response. If it is sampled at $P > NL$ points in (θ, f) space, then the following well known over-determined least squares minimisation problem is obtained:

$$\min_{\mathbf{h}} \left|\mathbf{D}^H \mathbf{h} - \mathbf{r}_d\right|^2,$$

where $\mathbf{D} = [\mathbf{d}(\theta_1, f_1) \cdots \mathbf{d}(\theta_P, f_P)]$ and $\mathbf{r}_d = [r_d(\theta_1, f_1) \cdots r_d(\theta_P, f_P)]$. The solution to this problem is

$$\mathbf{h} = \mathbf{D}^\dagger \mathbf{r}_d$$

where $\mathbf{D}^\dagger$ is the pseudo-inverse of $\mathbf{D}$. Although this is a reasonably simple procedure, it provides no insight into the underlying structure of the broadband beamforming process. A general broadband beamforming technique which is more insightful and simplifies implementation is developed in this thesis.

Broadband Pattern Nulling

In phased arrays, where the beamformer weights are fixed as a function of frequency, wideband pattern nulling is effected by placing multiple pattern nulls in the vicinity of the wideband source direction. Because the beampattern scales with frequency, a null trough of width Δu centred at u_0 (where $u = \sin\theta$) will provide suppression over a bandwidth of

$$\frac{\Delta f}{f_0} = \frac{\Delta u}{u_0},$$

where f_0 is the centre frequency of the band. The number of pattern nulls required within an angular region to give a desired suppression has been considered by Steyskal [85].

For a broadband beamformer with NL free parameters, M degrees of freedom can be used to impose the null in the given direction, with the remaining $NL - M$ degrees used to approximate some desired broadband pattern. An obvious way to formulate the problem is:

$$\begin{gathered}\min \int\int \varphi(\theta)\left|r_d(\theta, f) - r(\theta, f)\right|^2 \, d\theta \, df, \\ \text{subject to } r(\theta_0, f_m) = 0, \quad m = 1, \ldots, M,\end{gathered}$$

where θ_0 is the direction of the null, and $\varphi(\theta)$ is a weighting function. Based on the work of Steyskal, the number of constraints required for a frequency invariant beamformer to provide a given amount of suppression over a given bandwidth is derived in Chapter 4. With the formulation above it appears that to produce an exact null (i.e., a response which is zero for all frequencies) in a given direction requires an infinite number of constraints. However, by reformulating the problem, it is also shown in Chapter 4 that an exact null can be imposed with a finite number of constraints.

Broadband Adaptive Array Techniques

For a broadband beamformer with N sensors each having a filter with L taps, there are NL free parameters. The broadband LCMV beamformer applies $P < NL$ constraints (in space and/or frequency) and uses the remaining degrees of freedom to minimise some cost function. An example is the Frost beamformer [32] in which L degrees of freedom are used to maintain a desired frequency response in the look direction, while the remaining $NL - L$ parameters are used to minimise the output power.

Through the development presented in Chapters 2 and 3, the number of free parameters for the frequency invariant beamformer is significantly reduced. In fact, it is shown that the number of parameters requiring adaptation is independent of both the number of sensors in the array and the bandwidth of operation. This reduces the number of parameters requiring adaptation, thus reducing the computational complexity of the adaptation algorithm. A simple LCMV algorithm for the FIB is given in Chapter 3.

Broadband Direction of Arrival Estimation

Several methods of broadband DOA estimation have been proposed based on the *coherent signal subspace* method of Wang and Kaveh [99] (for example [31, 55, 59]). These methods are based on frequency decomposition. The wideband array data is first separated into several non-overlapping narrow frequency bins. Focusing matrices are then calculated which transform the data in each bin into a reference frequency bin. A composite covariance matrix is formed from the transformed array data, and conventional eigen-based DOA methods (such as MUSIC) may then be applied. Again, this frequency decomposition approach adds a large computational burden to the method. Using the results of the frequency invariant beamforming theory developed in Chapter 2, an alternative approach to broadband DOA estimation is presented in Chapter 6.

1.2 Contributions of Thesis

The primary motivation of this thesis is the problem of broadband frequency invariant (FI) beamforming, or beamforming in which there is little variation of the spatial response with frequency over an arbitrary defined bandwidth. In solving this problem we formulate design techniques which are applicable to a wider class of broadband beamformers.

The main theme throughout the thesis is in gaining an understanding of the underlying structure of the beamforming problem, and exploiting this structure to simplify the beamformer design, rather than blindly applying optimisation procedures to solve for the required beamformer design parameters.

One important approach we have taken is that properties are derived based on a theoretically continuous sensor. By using this continuous sensor, mathematical relationships between the functional requirements of the broadband beampattern and the beamformer structure are readily derived—in much of the literature, uniformly spaced arrays are used for the same reason. However, since we believe it is more appropriate to use nonuniformly spaced arrays for broadband applications (see Chapter 2), we do not want to be restricted to uniform array geometries and have instead chosen to formulate the problem in terms of the continuous sensor. Using a finite number of discrete sensors (i.e., an array) is then considered as an approximation to the continuous sensor.

To demonstrate this philosophy, consider the response of a broadband, linear, continuous sensor to plane waves impinging from the direction θ (measured relative to broadside):

$$r(\theta, f) = \int_{\mathbb{R}} \rho(x, f)\, e^{\jmath 2\pi f x c^{-1} \sin\theta}\; dx, \tag{1.8}$$

where $\rho(x, f)$ is the sensor illumination function (or gain of the sensor at a point x for a frequency f), and c is the speed of wave propagation. As shown in Chapter 5, the sensor illumination function required to produce a given broadband response is

$$\rho(x, f) = \frac{f}{c} \int_{\pi} r(\theta, f)\, e^{-\jmath 2\pi f x c^{-1} \sin\theta}\; \cos\theta\; d\theta. \tag{1.9}$$

This formulation gives the required illumination function to achieve a desired broadband response, arbitrarily specified over both angle and frequency. A result using the two dimensional Fourier transform has been presented in [43] for an equally spaced array, but a general result for an arbitrary array geometry has not been presented. As shown in Chapter 2, we can consider a beamformer as an approximation to the theoretical continuous sensor. Thus, realizing the broadband response with a finite array of discrete sensors reduces to the problem of providing a numerical approximation to the family of integrals in (1.8).

A block diagram of the topics presented in this thesis is shown in Fig. 1.8. The contributions made for each of these topics is summarised below.

1. **Frequency Invariant Beamforming Theory**
   The theory of a general class of beamformers in which the spatial response is (approximately) constant over an arbitrarily wide bandwidth is developed in Chapter 2. A frequency invariant property is first developed for a theoretical continuous sensor, and beamforming with an array of discrete sensors is then formulated as an approximation to this continuous sensor. Some simple structural properties implicit in the frequency invariant beamforming problem are highlighted by this development, thereby simplifying the implementation and reducing the number of free variables which have to be chosen in designing the beamformer. The frequency invariant beamforming theory is then generalised to cover a parameterised class of beamformers in which the frequency dependence of the beampattern can be controlled in a continuous manner from a classical single frequency design to a frequency invariant design.

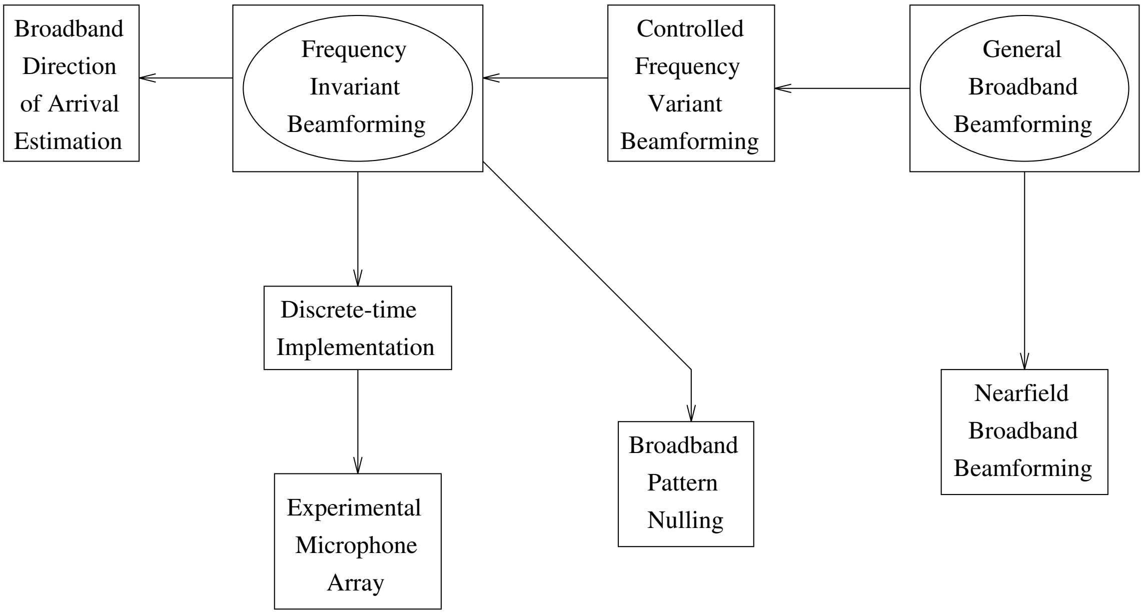

Broadband Direction of Arrival Estimation
Frequency Invariant Beamforming
Controlled Frequency Variant Beamforming
General Broadband Beamforming
Discrete-time Implementation
Experimental Microphone Array
Broadband Pattern Nulling
Nearfield Broadband Beamforming

Figure 1.8: Block diagram of topics presented in thesis.

2. **Implementation of a Frequency Invariant Beamformer**

   Discrete-time implementation of the frequency invariant beamformer is considered in Chapter 3. Using the theory developed in Chapter 2, it is shown that there is a single set of reference coefficients which defines the FI beampattern over the entire design band, regardless of the number of sensors in the array or the operating bandwidth. Furthermore, we show that these coefficients may be obtained directly from the desired FI beampattern function. Experimental results obtained from a microphone array are presented to test the validity of the frequency invariant beamforming theory in a practical setting.

3. **Broadband Pattern Nulls**

   The problem of producing a pattern null which covers a wide frequency band is considered in Chapter 4. Specifically, given a set of coefficients $\mathbf{h}$ which produces some desired broadband response $r(\theta, f)$, the aim is to find the coefficients $\widehat{\mathbf{h}}$ which produce a broadband response $\hat{r}(\theta, f)$ which is close in some respect to the original response $r(\theta, f)$, but exhibits a broadband null in a specified direction. Several formulations of this problem are considered, and two new results are presented. These are (i) producing an exact null, i.e., a null which is present over all frequencies, and (ii) determining the number of degrees of freedom which must be used to impose a null of a given depth in a frequency invariant beampattern.

4. **Nearfield Broadband Beamforming**

   A new method of nearfield broadband beamforming which allows the design of nearfield beamformers having an arbitrary (in both frequency and angle) broadband beampattern is presented in Chapter 5. The method is significantly different from existing nearfield beamforming methods which use farfield design techniques and then apply time delays to compensate for the spherical distortion caused by nearfield sources. Rather, spherical harmonics are used to transform the entire desired nearfield beampattern to an equivalent broadband farfield beampattern, and a farfield beamformer is then designed to achieve the transformed farfield beampattern. A general method of designing a farfield beamformer (with a nonuniformly spaced array geometry) having an arbitrary (over both frequency and angle) beampattern is also presented as part of the design methodology. This is referred to as *general broadband beamforming*, and is shown to describe a very large class of beamformers, including the classical single frequency beamformer and the frequency invariant beamformer developed in Chapter 2.

5. **Broadband Direction of Arrival Estimation**

   Chapter 6 presents a new method of high resolution direction of arrival (DOA) estimation for multiple farfield broadband signals. The frequency invariant beamformer is used to perform beamspace processing, such that (almost) identical source direction vectors are produced for all frequencies within the design band. This effectively focuses the beamspace data without requiring the frequency decomposition approach of most other broadband DOA methods. By avoiding frequency decomposition, the computational complexity of the DOA estimation process is reduced.

Chapter 2

Theory of Frequency Invariant Beamforming

IN MANY applications it is desirable for a broadband beamformer to have spatial resolution which is constant over the entire bandwidth of the source signals. For example, in speech acquisition with a microphone array the spatial resolution must remain constant over the entire speech bandwidth, which covers approximately four octaves. Signals with a bandwidth of several octaves are also encountered in sonar applications. A beamformer which maintains a constant spatial response over an arbitrarily wide bandwidth will be referred to as a *frequency invariant beamformer* or a FIB.[1]

2.1 Introduction

Several methods of designing a FIB have been proposed. These are briefly reviewed to motivate the solution proposed in this chapter.

Harmonic Nesting

Harmonic nesting is an approach which has often been used for designing microphone arrays for speech acquisition [28, 35, 51]. It is based on the fact that an equally spaced array with an inter-sensor spacing of d will exhibit the same beampattern at a frequency f as an equally spaced array with inter-sensor spacing $d/2$ (and the same array weights)

[1] The acronym FIB is actually very apt, since it is not possible to obtain a spatial response which is strictly frequency invariant using a finite number of discrete sensors (except in trivial cases). Claiming we can design a frequency invariant *beamformer* is truly something of a fib.

will at a frequency $2f$. The beamformer is composed of a set of nested equally spaced sub-arrays, each of which is a single frequency design. Bandpass filters are used to combine the sub-array outputs such that the appropriate sub-array is used for each octave. The idea is to reduce the frequency variation to that which would occur in a single octave.

Constrained Harmonic Nesting

Using the same nested array geometry as harmonic nesting, the sub-array outputs of two arrays “spaced” an octave apart are combined by two compensation filters, one on each sub-array output. These compensation filters allow two spatial constraints to be maintained over that octave, e.g. a unity constraint at broadside and a half-power constraint on the main beam width [44, 83].[2] Again, bandpass filters can be used to apply this technique over several octaves [57].

Frequency Sampling Method

Another approach based on the harmonic nesting array geometry is the frequency sampling method [15]. At each of K frequencies within each octave band, the required sensor weights are calculated by taking the inverse discrete Fourier transform of the sampled desired pattern. An FIR filter is then designed for each sensor to realize the sensor weight at each of the K frequencies. This method has been shown experimentally to achieve very good results [14]. However, because of the use of the discrete Fourier transform it is restricted to a uniformly spaced array geometry (within each octave band).

Multiple Beamforming

The main-beam width of a narrowband beamformer decreases as frequency increases. If several overlapping beams are simultaneously formed, the width of the resulting multi-beam may be kept constant by increasing the steering angle of the outermost beams as the beamwidth decreases [34, 95].

Optimisation Methods

The problem of determining the sensor gains and locations may be treated as a multi-dimensional optimisation problem [7, 81]. These methods do not use frequency dependent

[2] This method is sometimes referred to as the SHA technique, named after its inventors (Smith, Hixson and Au).

sensor gains, but instead attempt to find optimal sensor spacings and (fixed) gains by minimising the array power spectral density over a given frequency band. Because the sensor gains are frequency independent, the simple narrowband structure shown in Fig. 1.3 may be used. However, as demonstrated by Fig. 1.6, it is impossible to achieve a broadband frequency invariant beampattern using fixed sensor gains. In addition, these methods are very computationally intensive and provide no insight into the inherent structure of the problem.

Space Tapering

Doles and Benedict [20] have proposed a method in which the beampattern has little or no frequency dependence. The asymptotic theory of unequally spaced arrays [47, 48] is used to derive relationships between beampattern properties (such as peak response, mainlobe width, plateau sidelobe level, and clean sweep width) and beamformer design. These relationships are then used to translate beampattern requirements into functional requirements on the sensor spacings and weightings, thereby deriving a broadband design. This results in a space tapered array with frequency dependent sensor weightings; at each frequency in the design band the nonzero sensor weights identify a sub-array having total length and largest spacing which are appropriate to that frequency. Doles and Benedict minimise the number of sensors by allowing spacings within the active sub-array which are greater than half a wavelength.[3] This causes spatial aliasing, which is then controlled by placing constraints on the aperture distribution. Although this method provides a frequency invariant beampattern over a specified frequency design band, it is based on a linear array and a single-sided uniform aperture distribution with flat phase. No insight is given into the problem of designing double-sided or higher dimensional arrays, or arrays with arbitrary aperture distributions in both magnitude and phase (and thus arbitrary beampatterns). However, unlike most other methods it does allow nonuniformly spaced arrays.

Proposed Method

None of the existing methods suitably solve the general problem of designing a beamformer with an arbitrary beampattern which is frequency invariant over an arbitrary bandwidth and which may be applied to linear, planar, or three dimensional array geometries. This lack of a very general solution is addressed in this chapter. The proposed solution takes the

[3] A spacing of $\lambda/2$ is typically the maximum spacing used. This prevents grating lobes—periodic repetitions of the main beam—from appearing.

space tapering idea of [20] further by developing an FI beampattern property (applicable to one, two or three dimensional geometries) for a theoretical continuous sensor. The continuous sensor approach follows from the following observation: to obtain an identical beampattern at K discrete frequencies requires a compound array of K sub-arrays (having the self-similarity property as outlined in the harmonic nesting approach). Thus to provide an identical beampattern over a continuous range of frequencies requires an infinite number of sub-arrays, or a continuous sensor.

For a continuous sensor, the space tapering approach dictates that if an aperture function $A(x)$ (where x denotes location) is used at a frequency f_1, then an aperture function $A(xf_2/f_1)$ should be used at a frequency f_2 to obtain an *identical* response (ignoring gain). In other words, the aperture function should scale inversely with frequency to maintain a constant spatial response (ignoring gain). This is well known from antenna theory and is the basis of the spiral antenna and log-periodic dipole array described in Chapter 1.

The contributions made by this chapter are: (i) formalising the above space tapering idea for one, two and three dimensional sensors, (ii) relating this to a functional filtering requirement at any point on the continuous sensor, and (iii) demonstrating that a frequency invariant beamformer (using a nonuniformly spaced array) can be easily designed as an approximation to a theoretically continuous sensor having a frequency invariant response. An important consequence of this development is that there are specific simple structural properties that are implicit in the FIB. Such structural properties reduce the number of free variables which have to be chosen in designing the beamformer, thus simplifying the design process. The chapter concludes by showing that the FIB is a special case of a more general class of beamformers in which the frequency variation of the beampattern can be controlled.

2.2 Linear Frequency Invariant Sensor

Consider a one dimensional (linear) continuous sensor. The response of the sensor to plane waves (i.e., from a farfield point source) impinging from an angle θ measured relative to broadside with propagation speed c is

$$r(\theta, f) = \int_{\mathbb{R}} \rho(x, f)\; e^{\jmath 2\pi f c^{-1} x \sin\theta}\; dx, \tag{2.1}$$

where $\rho : \mathbb{R}\times\mathbb{R}^{+} \to \mathbb{C}$ defines the sensor *sensitivity distribution*, i.e., the gain of the sensor at a point x for a frequency f. It is assumed that the sensitivity distribution is absolutely integrable and that for a practical finite-aperture sensor, the function $\rho(x, f)$ has finite support in x.

Theorem 2.1 (Frequency Invariant Beampattern) *Let the sensitivity distribution of a one dimensional sensor, which is a function of distance x along the sensor and frequency f, be given by*

$$\rho(x, f) = f\, G(xf), \quad \forall f > 0, \tag{2.2}$$

where $G\colon \mathbb{R} \to \mathbb{C}$ is an arbitrary absolutely integrable complex function of a single real variable. Then the farfield response, $r(\theta, f)$, which is a function of the angle θ measured relative to broadside and frequency f, will be frequency invariant, i.e.,

$$r(\theta, f) = r_{\text{FI}}(\theta) = \int_{\mathbb{R}} e^{\jmath 2\pi c^{-1} \xi \sin\theta} G(\xi)\; d\xi.$$

□

Proof: Substituting $\rho(x, f) = f\, G(xf)$ into the expression for the sensor response (2.1), yields

$$\begin{aligned} r(\theta, f) &= \int_{\mathbb{R}} e^{\jmath 2\pi c^{-1} f x \sin\theta} f G(xf)\; dx, \quad f > 0 \\ &= \int_{\mathbb{R}} e^{\jmath 2\pi c^{-1} \xi \sin\theta} G(\xi)\; d\xi \triangleq r_{\text{FI}}(\theta) \end{aligned}$$

with the change of variables $\xi = xf$. ■

This theorem simply expresses the known result that the aperture illumination scales with wavelength (or inversely with frequency) to maintain the same response (ignoring gain); the multiplicative f term is used to normalise the response.

Note that the functions $G(\cdot)$ and $r_{\text{FI}}(\cdot)$ form a Fourier transform pair (modulo the $\sin\theta$ distortion); this Fourier pair relationship is exploited in Chapter 3 to simplify the implementation of the FIB. Hence it is straightforward to take any beam response specification and translate that to a specification on the aperture illumination to achieve a broadband FI response.

Theorem 2.1 specifies a sufficient condition on the aperture sensitivity distribution to produce an FI beampattern. The necessary condition is stated by the following theorem.

Theorem 2.2 (Sensitivity Distribution) *Let $r_{\text{FI}}(\theta)$ be an arbitrary continuous square-integrable frequency invariant farfield response, which is specified for $\theta \in (-\pi/2, \pi/2)$.*

Then the sensitivity distribution, $\rho(x,f)$, of a linear sensor which realizes this response must satisfy the following conditions:

1. $\rho(x,f) = fG(xf)$ *for some function* G.
2. G *has a Fourier transform* Γ *satisfying*

   (a) $\Gamma(s) = B(s) = r_{FI}\left(\sin^{-1}(sc)\right), \quad s \in (-1/c, 1/c)$

   (b) $\Gamma(s) = A(s), \quad s \notin (-1/c, 1/c)$

   where c *is the speed of wave propagation, and* $A(\cdot)$ *is an arbitrary square integrable function such that*

   $$A\left((-1)^i/c\right) = \lim_{s \to \frac{(-1)^i}{c}} B(s)$$

   for $i = 0, 1$.

□

Proof: Assume that a frequency invariant response $r_{\mathrm{FI}}(\theta), \theta \in (-\pi/2, \pi/2)$ is given. Rewrite (2.1) as

$$B(s) = \int_{\mathbb{R}} \rho\left(\frac{y}{f}, f\right) e^{j2\pi sy}\, f^{-1} dy, \quad s \in (-1/c, 1/c) \tag{2.3}$$

with the change of variables $s = c^{-1}\sin\theta$ and $y = xf$.

Since $B(s)$ is frequency invariant, the integrand must also be frequency invariant. Therefore define $G(y) = f^{-1}\rho(y/f, f)$, for some function $G(\cdot)$. Equation (2.3) can now be rewritten as

$$B(s) = \int_{\mathbb{R}} G(y) e^{j2\pi sy} dy = \mathcal{F}^{-1}\{G(y)\}$$

where $\mathcal{F}\{\cdot\}$ represents the Fourier transform. $G(y)$ has an inverse Fourier transform, $\Gamma(s)$, satisfying

$$\Gamma(s) = \mathcal{F}^{-1}\{G(y)\} = \begin{cases} B(s), & s \in (-1/c, 1/c) \\ A(s), & \text{otherwise.} \end{cases}$$

By Plancherel's Theorem [73, p.2], the function $G(\cdot)$ is uniquely determined from $\Gamma(\cdot)$ if $B(\cdot)$ and $A(\cdot)$ are both square-integrable functions, and

$$A\left(\frac{(-1)^i}{c}\right) = \lim_{s \to \frac{(-1)^i}{c}} B(s)$$

for $i = 0, 1$. ■

It is instructive to consider the following analogy.[4] Consider a bandlimited function $H(f)$ specified only for $f \in (-F, F)$. This has an inverse Fourier transform of

$$h(t) = h_1(t) + h_2(t)$$

where

$$\mathcal{F}\{h_1(t)\} = \begin{cases} H(f), & f \in (-F, F) \\ 0, & \text{otherwise} \end{cases}$$

and

$$\mathcal{F}\{h_2(t)\} = \begin{cases} 0, & f \in (-F, F) \\ A(f), & \text{otherwise} \end{cases}$$

where $A(\cdot)$ is an arbitrary function. Hence, any high frequency perturbation in the function $h(t)$ will not produce any effect on the function $H(f), f \in (-F, F)$.

Thus the only freedom in choosing $\rho(x, f)$ for a desired FI beampattern is in the sufficiently high "spatial frequency" behaviour of G. Apart from that, $r_{\text{FI}}(\theta)$ for $\theta \in (-\pi/2, \pi/2)$ determines $\rho(x, f)$ uniquely.

2.3 General Frequency Invariant Sensor

Having developed the necessary and sufficient conditions for an FI linear sensor, a more general FI theory which is applicable to an arbitrary (physical)[5] sensor geometry is now developed.

First, consider a two dimensional (planar) sensor. The response to an impinging plane wave from direction $\Theta = [\theta, \phi]$ as defined by Fig. 2.1 is

$$r(\Theta, f) = \int_{\mathbb{R}^2} \rho(\mathbf{x}, f) \, \exp\left[\jmath 2\pi c^{-1} f (x_1 \sin\theta + x_2 \cos\theta \cos\phi)\right] \, d\mathbf{x},$$

where $\mathbf{x} = [x_1, x_2]^T$ is a point on the sensor. By substitution, it is seen that if $\rho(\mathbf{x}, f) = f^2 G(x_1 f, x_2 f)$, then $r(\Theta, f) = r_{\text{FI}}(\Theta)$ is an FI response.

[4] This is not the precise analogy (which would instead consider a finite duration time signal) but is equally valid, however, due to the duality property of the Fourier transform.

[5] In the following it is assumed that the sensor dimension $D \in \{1, 2, 3\}$. However, it is mathematically possible to design a beamformer which is frequency invariant in some fractional dimension. Whether a fractal FIB is useful in practice is questionable, and has not been explored further.

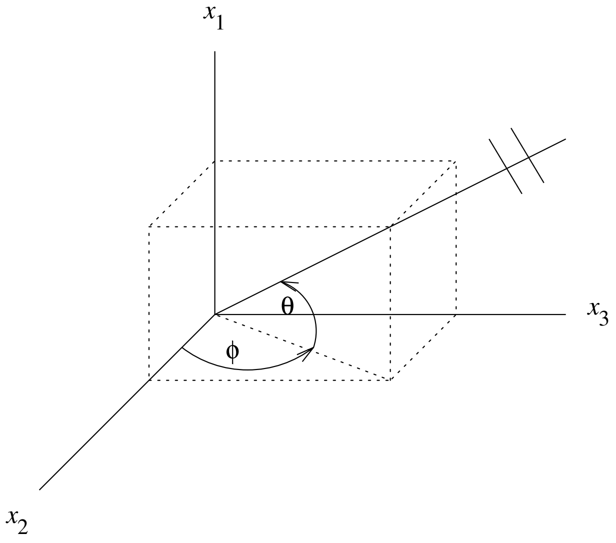


Figure 2.1: Coordinate system for a plane wave arriving from the direction $\Theta = (\theta, \phi)$. Note that this differs from the conventional right-hand coordinate system.

Similarly, the response of a three dimensional sensor to planar waves from a direction $\Theta = [\theta, \phi]$ is

$$r(\Theta, f) = \int_{\mathbb{R}^3} \rho(\mathbf{x}, f) \ \exp\left[\jmath 2\pi c^{-1} f (x_1 \sin\theta + x_2 \cos\theta \cos\phi + x_3 \cos\theta \sin\phi)\right] \ d\mathbf{x},$$

where $\mathbf{x} = [x_1, x_2, x_3]^T$ is a point on the sensor. Again, by substitution the response is FI if $\rho(\mathbf{x}, f) = f^3 G(x_1 f, x_2 f, x_3 f)$.

In summary, the sufficient condition for a general FI sensor is described by the following theorem.

Theorem 2.3 (General FI Sensor) *Let the response of a D dimensional continuous sensor to planar waves impinging from a direction Θ be given by*

$$r(\Theta, f) = \int_{\mathbb{R}^D} \rho(\mathbf{x}, f) \exp\left[\jmath 2\pi c^{-1} f \mathbf{x}^T \Upsilon(\Theta)\right] \ d\mathbf{x}$$

where $D \in \{1, 2, 3\}$, $\Upsilon(\Theta) = [\sin\theta, \cos\theta\cos\phi, \cos\theta\sin\phi]^T$, $\mathbf{x} = [x_1, x_2, x_3]^T$ is a point on the sensor, and $\rho : \mathbb{R}^D \times \mathbb{R}^+ \to \mathbb{C}$ is the sensitivity distribution. The sensor has a frequency invariant response if

$$\rho(\mathbf{x}, f) = f^D G(\mathbf{x} f), \quad \forall f > 0 \tag{2.4}$$

where $G : \mathbb{R}^D \to \mathbb{C}$ is an arbitrary absolutely integrable complex valued function. □

Proof: Substituting $\rho(\mathbf{x}, f) = f^D G(\mathbf{x}f)$ yields the following response,

$$\begin{aligned} r(\Theta, f) &= \int_{\mathbb{R}^D} f^D G(\mathbf{x}f) \exp\left[\jmath 2\pi c^{-1} f \mathbf{x}^T \Upsilon(\Theta)\right] \, d\mathbf{x} \\ &= \int_{\mathbb{R}^D} G(\xi) \exp\left[\jmath 2\pi c^{-1} \xi^T \Upsilon(\Theta)\right] \, d\xi, \end{aligned}$$

with the change of variables $\xi = \mathbf{x}f$. ■

As an aid in interpretation, the function $G(\cdot)$ which appears in (2.4) will be expressed in two equivalent representations:

$$G(\mathbf{x}f) = A_f(\mathbf{x}) = H_{\mathbf{x}}(f), \quad \forall \mathbf{x}, \forall f > 0 \tag{2.5}$$

where $A_f : \mathbb{R}^D \to \mathbb{C}$ defines the *aperture distribution* at a nominally fixed frequency, f, and $H_{\mathbf{x}} : \mathbb{R}^+ \to \mathbb{C}$ defines the *primary filter response* at a single point, $\mathbf{x}$, on the sensor.

The total filtering required at any point $\mathbf{x}$ on the sensor can now be written

$$\rho(\mathbf{x}, f) = f^D \, H_{\mathbf{x}}(f).$$

where the f^D component will be referred to as the *secondary filter response.* Separating the sensor filtering into primary and secondary filter components has practical significance, as shown in §2.4.1.

Two important properties of the FI sensor arising as a consequence of (2.5) are summarised by the following theorems.

Theorem 2.4 (Filter Shape) *If $H_{\mathbf{x}}(f)$ denotes the frequency response of the primary filter at point $\mathbf{x}$ and $A_f(\mathbf{x})$ denotes the aperture distribution for a given frequency $f > 0$, then for a frequency invariant D dimensional sensor,*

$$H_{\mathbf{x}}(f) = A_{\|\mathbf{x}\|}(f\widehat{\mathbf{x}}), \quad \mathbf{x} \in \mathbb{R}^D, \; f \in \mathbb{R}^+, \; D \in \{1, 2, 3\},$$

where

$$\widehat{\mathbf{x}} = \frac{\mathbf{x}}{\|\mathbf{x}\|}, \quad \mathbf{x} \in \mathbb{R}^D$$

defines a unit vector in the direction of $\mathbf{x}$, and $\|\cdot\|$ denotes Euclidean distance. □

Proof: The proof follows from the following straightforward manipulation:

$$\begin{aligned} H_{\mathbf{x}}(f) &= G(\mathbf{x}f) \\ &= G(f\,\widehat{\mathbf{x}}\,\|\mathbf{x}\|) \\ &= A_{\|\mathbf{x}\|}(f\,\widehat{\mathbf{x}}). \end{aligned}$$

■

This result says that the primary filter response required at point $\mathbf{x}$ can be obtained by taking a slice through the aperture distribution from the origin in the direction of $\mathbf{x}$. The aperture distribution can be determined from the desired FI response and vice versa. In the one dimensional case the result reduces to

$$H_x(f) = \begin{cases} A_x(f), & \text{if } x > 0 \\ A_{-x}(-f), & \text{if } x < 0. \end{cases}$$

Note that the subscript on the aperture function needs to be positive since it denotes the frequency of interest.

Theorem 2.5 (Filter Dilation) *All primary filter responses in a D dimensional frequency invariant sensor for a given $\widehat{\mathbf{x}}$ are identical up to a frequency dilation.* □

Proof: Let $H_{\mathbf{x}}(f)$ represent the primary filter response at an arbitrary point $\mathbf{x}$ on a frequency invariant sensor. Let $H_{\gamma\mathbf{x}}(f)$ be the primary filter response at a point $\gamma\mathbf{x}$, where $\gamma > 0$. This point lies on the radial line from the origin through $\mathbf{x}$, and implies $\widehat{(\gamma\mathbf{x})} = \widehat{\mathbf{x}}$. Then

$$\begin{aligned} H_{\gamma\mathbf{x}}(f) &= G(\gamma\mathbf{x}f) \\ &= H_{\mathbf{x}}(\gamma f), \end{aligned}$$

which is a dilation property. ■

In the one dimensional case the result is as follows: if $H_x(f)$ is the primary filter response at a point $x > 0$ on the sensor, then the primary filter response at a point $\gamma x, \gamma > 0$ is given by $H_{\gamma x}(f) = G(\gamma x f) = H_x(\gamma f)$. For $\gamma < 0$, the primary filter responses need not be related by frequency dilation; this situation is considered in §2.4.4.

2.4 Frequency Invariant Array Geometries

Having developed the FI theory for a continuous sensor, it is now necessary to consider an FI array, where an array is defined as a spatially separated set of identical, discrete, omni-directional broadband sensors. Without loss of generality, initially only single-sided linear arrays with the first element located at $x = 0$ will be considered. Issues relating to more generalised array geometries are discussed later.

2.4.1 Approximation to a Continuous Sensor

An array of sensors can only approximate the ideal FI continuous sensor. Methods of approximating a continuous sensor exist in the literature [24, 102], however, these methods are iterative and are unsuitable for the FI beamforming implementation we seek. Thus it is necessary to determine a numerical approximation to the following family of integrals describing an FI response:

$$r(\theta) = \int_{\mathbb{R}} f\, G(xf)\, \exp\left(\jmath 2\pi f c^{-1} x \sin\theta\right)\, dx, \quad f > 0. \tag{2.6}$$

To obtain an approximation, let $\{x_n\}_{n=0}^{N-1}$ denote a finite set of N (possibly nonuniformly spaced) discrete sensor locations. Further, the frequency range of interest is limited to the band $[f_L, f_U]$ where f_L and f_U are the lower and upper band edges respectively.

An approximation to the family of integrals in (2.6) can be made by the following simple class of numerical approximations:

$$\hat{r}(\theta) = \alpha f \sum_{n=0}^{N-1} g_n H_n(f) \exp\left(\jmath 2\pi f c^{-1} x_n \sin\theta\right), \quad f \in [f_L, f_U], \tag{2.7}$$

where $H_n(f) = G(x_n f)$ is the primary filter response of the nth sensor, g_n is a *spatial weighting term* to account for the possibly nonuniform sensor spacings, and α is a normalisation constant (usually chosen such that $\hat{r}(\theta_0) = 1$ where θ_0 is the direction of the main beam).

Defining the response of the FIB as in (2.7) leads to the particularly simple block diagram shown in Fig. 2.2. This diagram shows a number of important features of the FIB: (i) the primary filters are simple dilations of a single frequency response, defined in the figure as $H(f) = G(x_1 f)$; (ii) implicitly, this primary filter response shape is identical to the continuous aperture distribution in both magnitude and phase; (iii) the primary filter outputs can be combined via frequency independent spatial weights g_n which depend

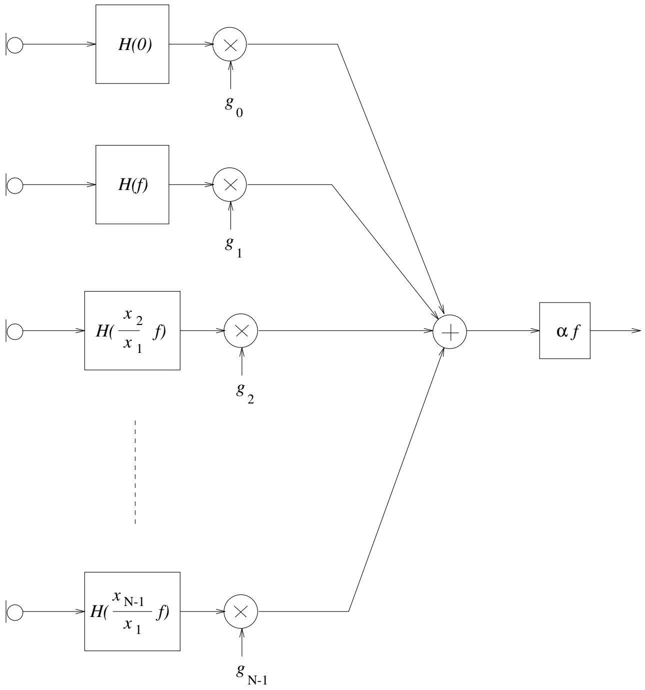


Figure 2.2: Block diagram of a general linear FIB. $H(f)$ is defined as the primary filter response at x_1.

only on the sensor locations; and (iv) all sensors share a common secondary filter response αf to generate the final FIB output.

A discrete-time implementation of the FIB is considered in Chapter 3.

2.4.2 Spatial Weighting Terms

Three methods of determining the spatial weights are now presented, and the relative performance of the methods is qualitatively compared.

I. Trapezoidal Integration

The simplest method of approximating the integral in (2.6) is through the well known trapezoidal integration method. In this case it is easy to show that the spatial weighting terms can be expressed as a linear combination of the sensor locations, i.e.,

$$\mathbf{g}^{(\mathrm{t})} = \mathbf{T}\mathbf{x}, \tag{2.8}$$

where $\mathbf{g}^{(t)} = [g_0, g_1, \ldots g_{N-1}]^T$, $\mathbf{x} = [x_0, x_1, \ldots x_{N-1}]^T$, and

$$\mathbf{T} = \frac{1}{2} \begin{bmatrix} -1 & 1 & & & \mathbf{O} \\ -1 & 0 & 1 & & \\ & & \ddots & & \\ & & -1 & 0 & 1 \\ \mathbf{O} & & & -1 & 1 \end{bmatrix}. \tag{2.9}$$

II. Poisson's Summation

Using Poisson's summation formula,[6] Ishimaru [47] has shown that the response of a narrowband array, viz.,

$$r(\theta) = \sum_{n=0}^{N-1} w_n \exp\left(\jmath 2\pi f c^{-1} x_n \sin\theta\right),$$

may be expressed as

$$\begin{aligned} r(\theta) &= \sum_{m=-\infty}^{\infty} r_m(\theta), \\ r_m(\theta) &= \int_0^L w(x) \frac{d\nu}{dx} \exp\left[\jmath 2\pi m \nu(x)\right] \exp\left(\jmath 2\pi f c^{-1} x \sin\theta\right) \, dx, \end{aligned}$$

where L is the total aperture size, $w(\cdot)$ is a continuous function defined such that $w(x_n) = w_n$, and $\nu(\cdot)$ is defined such that $\nu(x_n) = n$. Thus, the response of a narrowband array may be expressed as an infinite sum of continuous aperture responses, with the mth aperture distribution given by

$$w(x) \frac{d\nu}{dx} \exp\left[\jmath 2\pi m \nu(x)\right].$$

An important consequence of this formulation is that the mainlobe is described by $r_0(\theta)$, while $r_m(\theta)$ for $m \neq 0$ describe the grating lobes [16]. Because this is an extremely rapidly convergent series, a reasonably good approximation to the array response in the visible region is given by

$$r_0(\theta) = \int_0^L w(x) \frac{d\nu}{dx} \exp\left(\jmath 2\pi f c^{-1} x \sin\theta\right) \, dx \tag{2.10}$$

[6] Poisson's summation formula is:

$$\sum_{n=-\infty}^{\infty} f(n) = \sum_{m=-\infty}^{\infty} \int_{-\infty}^{\infty} f(\nu) e^{\jmath 2\pi m \nu} \, d\nu.$$

(with $r_1(\theta)$ and $r_{-1}(\theta)$ providing small corrections and the remaining terms having negligible effect). Hence, the above formulation is in essence the transformation of a discrete array into an equivalent continuous aperture distribution.

Theorem 2.1 shows that the response of an FI aperture may be written as

$$r_{\mathrm{FI}}(\theta) = f \int_0^L G(xf) \exp\left(\jmath 2\pi f c^{-1} x \sin\theta\right) \, dx.$$

Comparing this with (2.10) and ignoring the f term outside the integral (since it is included in the FIB after the summation), we find that

$$w(x)\frac{d\nu}{dx} = G(xf).$$

Recalling that $w_n = w(x_n)$, the weight of the nth sensor is

$$w_n = G(xf)\frac{dx}{d\nu}, \quad x = x_n.$$

Relating this back to (2.7), we see that the frequency dependent weight of each sensor is given by sampling the ideal FI aperture distribution and multiplying by a spatial weighting term defined by

$$g_n^{(\mathrm{p})} = \frac{dx}{d\nu}, \quad \text{for } x = x_n, \nu = n. \tag{2.11}$$

The spacing function (2.19) developed in the following section may be differentiated to yield

$$g_n^{(\mathrm{p})} = \begin{cases} (\lambda_U/2), & \text{for } 0 \le x_n \le P\frac{\lambda_U}{2} \\ x_n \log\left(\frac{P}{P-1}\right), & \text{for } P\frac{\lambda_U}{2} < x_n < L. \end{cases} \tag{2.12}$$

III. Least Squares Optimum

The response of a FIB (2.7) can be expressed as

$$r(\theta, f) = \mathbf{g}^T \mathbf{\Gamma}(\theta, f),$$

where

$$\mathbf{\Gamma}(\theta, f) = \alpha f \left[H_0(f) \exp\left(\jmath 2\pi f \tau_0(\theta)\right), \dots, H_{N-1}(f) \exp\left(\jmath 2\pi f \tau_{N-1}(\theta)\right)\right]^T,$$

is an N dimensional vector, and $\tau_n(\theta) = x_n c^{-1} \sin\theta$.

Define the following cost functional which measures the weighted L_2 distance between a desired FI response and the actual response over the design bandwidth:

$$\begin{aligned} J &= \int_{f_L}^{f_U} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \varphi(\theta, f) \left| \mathbf{g}^T \mathbf{\Gamma}(\theta, f) - r_d(\theta, f) \right|^2 d\theta\, df \\ &= \mathbf{g}^T \mathbf{Q} \mathbf{g} - 2\mathbf{g}^T \mathbf{q}_1 + q_0, \end{aligned} \tag{2.13}$$

where $r_d(\theta)$ is the desired FI response, $\varphi(\theta, f)$ is a general weighting function,

$$\begin{aligned} \mathbf{Q} &= \int_{f_L}^{f_U} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \varphi(\theta, f)\ \mathbf{\Lambda}(\theta, f)\, d\theta\, df, \\ \mathbf{q}_1 &= \int_{f_L}^{f_U} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \varphi(\theta, f) \Big(\mathrm{Re}\left\{ r_d(\theta) \right\} \mathrm{Re}\left\{ \mathbf{\Gamma}(\theta, f)^* \right\} - \mathrm{Im}\left\{ r_d(\theta) \right\} \mathrm{Im}\left\{ \mathbf{\Gamma}(\theta, f)^* \right\} \Big)\, d\theta\, df, \\ q_0 &= \int_{f_L}^{f_U} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \varphi(\theta, f) \left| r_d(\theta) \right|^2 d\theta\, df, \end{aligned}$$

and

$$\mathbf{\Lambda}(\theta, f) = \mathrm{Re}\left\{ \mathbf{\Gamma}(\theta, f)\, \mathbf{\Gamma}(\theta, f)^H \right\}. \tag{2.14}$$

Minimising J with respect to $\mathbf{g}$ gives the optimum spatial weights as

$$\mathbf{g}^{(\mathrm{ls})} = \mathbf{Q}^{-1} \mathbf{q}_1. \tag{2.15}$$

This method finds the real-valued spatial weights which minimise the frequency variation of the FIB response, in the weighted least squares sense.

Comparison of Methods

For a simple qualitative comparison of the methods, consider the following example for a FIB with 9 elements and an aperture size of $P = 5$ half-wavelengths designed to cover the bandwidth 1000–2000 Hz (with the filters designed using 8th order Butterworth filters as described in §2.6.1). The beampattern using each spatial weighting method is shown in Fig. 2.3, calculated at 11 frequencies within the band. Note that despite the additional computation required by the least squares method, the results are not a great improvement over the extremely simple trapezoidal integration method. This is true of all examples we have tried. It is recommended that the trapezoidal integration method be used to calculate the spatial weights, since this is by far the simplest method. Unless stated otherwise, we

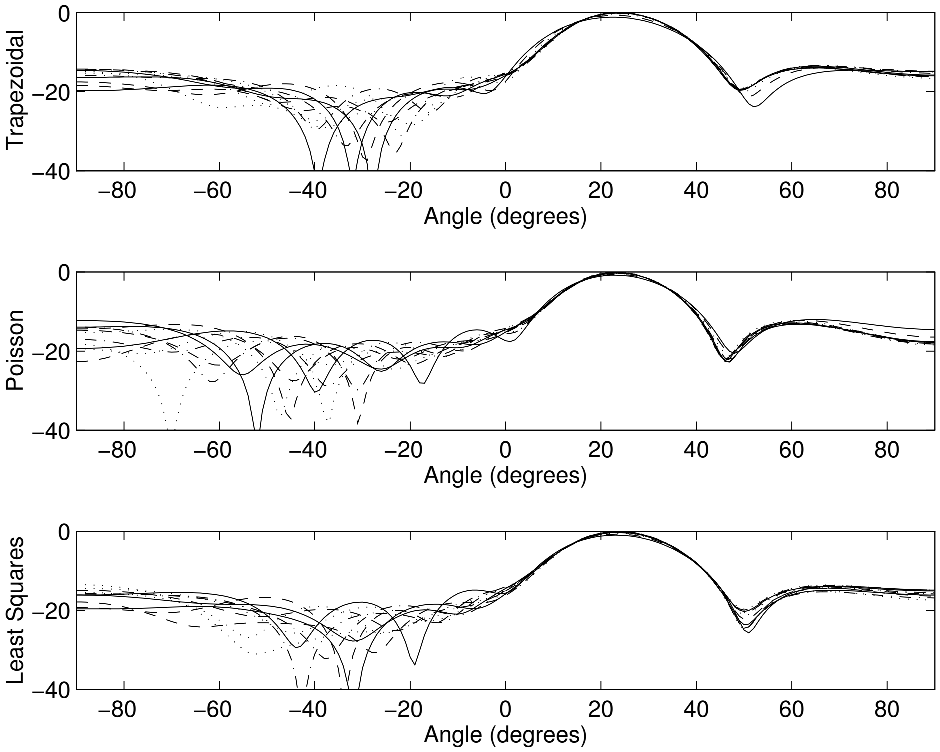


Figure 2.3: Comparison of spatial weighting methods. The beampattern magnitude is plotted in dB.

will use the trapezoidal method to calculate the spatial weights in the remainder of the thesis.

2.4.3 Sensor Locations

In determining the sensor locations for the FIB it is desirable to minimise the number of sensors required while maintaining performance over the design bandwidth. The criterion employed is to use as few sensors as possible without introducing spatial aliasing. This is the same criterion used in [20].[7]

From the theory of linear uniformly spaced arrays [4, 60] it is well known that grating lobes (i.e., periodic repetitions of the main beam) are introduced into the array response if the spacing of array elements approaches the wavelength of operation, λ. If delay beam steering is to be applied to the array the constraint reduces to a maximum spacing of $\lambda/2$.

[7] In fact, Doles and Benedict do allow a small amount of controlled spatial aliasing by imposing constraints on the aperture distribution function.

For a nonuniformly spaced array the grating lobes smear out into grating plateaus [16], but the $\lambda/2$ maximum spacing criterion still holds.

Since the FI aperture size scales with frequency to maintain the same response, the aperture size is constant if defined in terms of wavelength. Define the finite aperture size to be P half-wavelengths at all frequencies in the design band. This intuitively leads to a space tapered array, i.e., at high frequencies a small array is used, and more elements are added at lower frequencies to maintain the same aperture size of P half-wavelengths. Thus, for all frequencies except f_L some of the sensors will not be used. When a sensor is used at a particular frequency it will be referred to as being *active* at that frequency. The positions of inactive sensors, despite the potential property that they violate the $\lambda/2$ spacing requirement, are irrelevant.

The finite aperture constraint implies the first sensor positioning constraint:

$$x_n = P\frac{\lambda_n}{2}, \tag{2.16}$$

where x_n is the location of the active sensor furthest from the origin, and λ_n is the wavelength corresponding to the bandwidth of the nth primary filter, i.e., the highest frequency at which the nth sensor remains active.

The spatial aliasing constraint defines a second sensor positioning constraint:

$$x_n = x_{n-1} + \frac{\lambda_n}{2}, \quad \text{for } n > 0, \tag{2.17}$$

where all symbols have the same meaning as in (2.16).

Combining (2.16) and (2.17) gives

$$x_n = \left(\frac{P}{P-1}\right) x_{n-1}, \quad \text{for } n > 0. \tag{2.18}$$

This constraint must be maintained within the desired frequency range to avoid spatial aliasing. Since spacings less than $\lambda_U/2$ will not cause spatial aliasing at any frequency within the design band, it follows that the spacing within the densest portion of the array should be $\lambda_U/2$ to minimise the number of sensors. This densely packed portion of the array should have a total size of $P\lambda_U/2$ and will contain a minimum of $P+1$ sensors, where, to simplify notation and without a loss of generality, we restrict $P \in \mathbb{N}$. Hence, the minimum set of locations to avoid spatial aliasing is summarised as

$$x_n = \begin{cases} \frac{\lambda_U}{2} n, & \text{for } 0 \leq n \leq P, \\ P\frac{\lambda_U}{2}\left(\frac{P}{P-1}\right)^{n-P}, & \text{for } P < n < N-1, \\ P\frac{\lambda_L}{2}, & \text{for } n = N-1, \end{cases} \tag{2.19}$$

where λ_L and λ_U are the wavelengths corresponding to the lower and upper design frequencies respectively, P is the aperture size measured in half-wavelengths, and N is the number of array elements. In the sense of producing a linear array geometry which avoids spatial aliasing, this represents the optimal sensor positioning function for a FIB. Note that the last element is only used at the lowest frequency and in most cases may be left out without causing significant degradation to the FIB response.

Using this optimal sensor positioning relation, the minimum number of sensors required to implement a linear FIB over a desired frequency range is

$$N = (P+1) + \left\lceil \frac{\log\left(\frac{f_U}{f_L}\right)}{\log\left(\frac{P}{P-1}\right)} \right\rceil \tag{2.20}$$

where $\lceil \cdot \rceil$ denotes the ceiling (next higher integer) function.

The aperture size, P, is chosen as in the case of a single frequency design, i.e., the aperture should be sufficiently large to achieve the desired beam resolution properties.

2.4.4 Double Sided Linear Array

In conventional single frequency designs, the location of the origin is as much a matter of notational convenience as any consideration of implementation complexity. However, for a FIB the position of the array origin can have a dramatic effect on the implementation.

From Theorem 2.5 it follows immediately that for an asymmetric double sided linear aperture, two distinct primary filter responses are required.[8] The complexity of each of these filters is directly dependent on the position of the array origin. An example double sided aperture distribution and the corresponding real filter responses are shown in Fig. 2.4. This aperture produces the same beampattern as the single sided aperture shown in Fig. 2.5. However, by better positioning of the array origin in the double sided aperture, the required filter responses have a low pass filter shape which will be easier to implement.

[8] Clearly, for a double sided aperture distribution which is symmetric about the origin, the same primary filter response can be used for sensors located on either side of the origin.

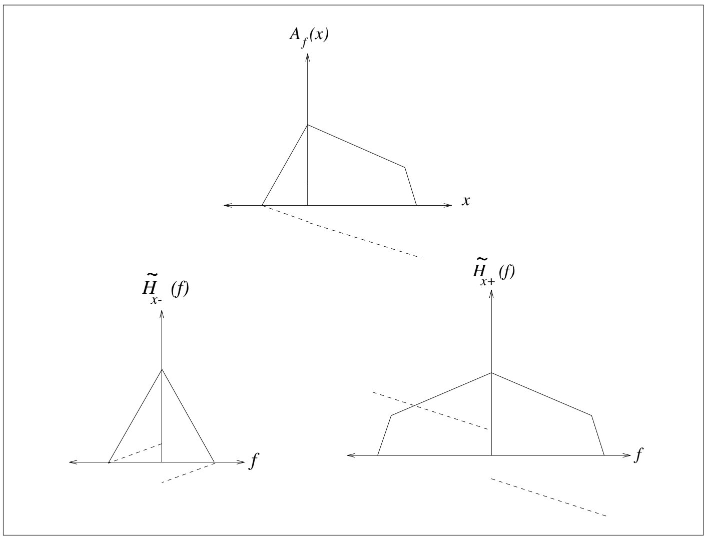


Figure 2.4: Double sided linear aperture distribution and corresponding real primary filter responses.

2.4.5 Two Dimensional Array

Theorem 2.5 gives no guarantee that the primary filter responses for two and three dimensional arrays will exhibit a dilation property. This is not a restriction on being able to build a broadband array, it simply restricts the appearance of self-similarities which may be exploited to simplify the array design. Thus, generally a two dimensional array corresponds to approximating a double integral in the spirit of (2.7) for the one dimensional case. However, there are at least two special cases which will produce primary filters which have the same frequency response at more than one position within the array. (These cases are discussed for the two dimensional case and are easily extended to the three dimensional case.) These special cases are illustrated in Fig. 2.6.

I. Separable Aperture Distributions

If the aperture distribution is separable into the product of two one-dimensional aperture distributions as shown in Fig. 2.6(a), i.e.,

$$A_f(x_1, x_2) = A'_f(x_1)A''_f(x_2)$$

then the primary filter responses are also separable, meaning that at any point $[x_1, x_2]$,

$$H_{x_1,x_2}(f) = H'_{x_1}(f)H''_{x_2}(f).$$

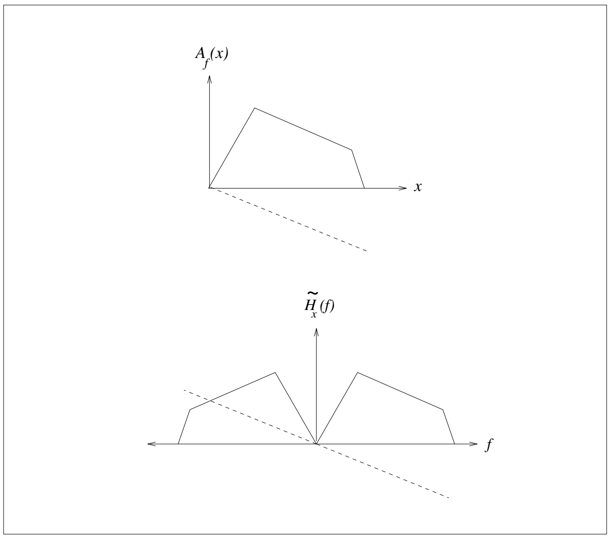


Figure 2.5: Single sided aperture distribution $A_f(x)$ and corresponding real primary filter response $\widetilde{H}_x(f)$.

Hence at least two, and at most four, different filter responses are required (depending on whether the component one dimensional arrays are one or two sided). Note that this class of aperture sensitivities requires that

$$G(x_1 f, x_2 f) = G_1(x_1 f) G_2(x_2 f).$$

II. Discrete Sensor Radial Pattern

If the array elements are arranged in radial patterns from the origin as shown in Fig. 2.6(b), then each of these radial lines is equivalent to a linear one dimensional array, and thus each of the primary filters on the radial line is given by a dilation of the same function. For an array with N elements which is arranged into $K < N$ different radial lines, there will be only K distinct filter responses, as opposed to N. This is true for any arbitrary two dimensional aperture distribution. In the sense that this does not restrict the aperture distribution (and thus the desired beampattern) then this type of sensor location pattern is recommended for any design. Naturally if the desired aperture distribution further satisfies a radially symmetric pattern the design is further simplified and only a single filter shape is required, and the discrete sensors need not be restricted to radial lines.

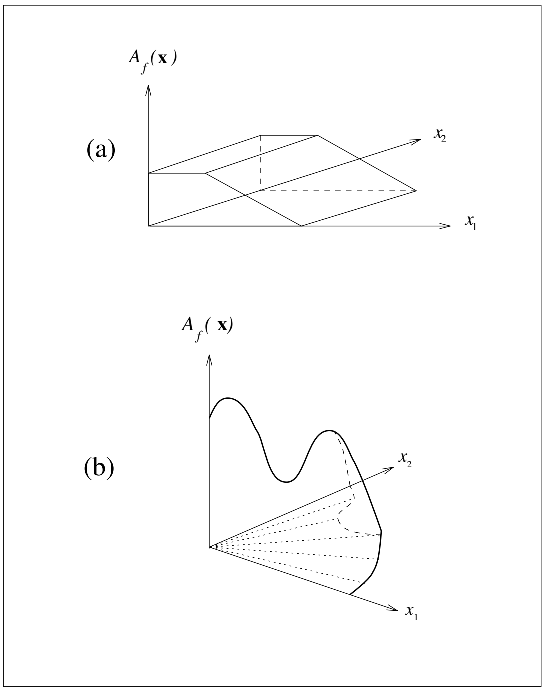


Figure 2.6: Examples of two dimensional apertures which produce primary filters having the same response at more than one location.

2.5 Controlled Frequency Variant Beamforming

Having developed the theory of frequency invariant beamforming, it is now desirable to generalise this theory to cover a parameterised class of beamformers in which the frequency dependence of the beampattern can be controlled in a continuous manner. It will be shown that both the conventional narrowband beamformer and the frequency invariant beamformer are special cases of this more general beamformer class, which will be referred to as a *controlled frequency variant beamformer* (CFVB).

The frequency variation of the CFVB beampattern is controlled as described by the following theorem. The coordinate system is as shown in Fig. 2.1.

Theorem 2.6 (Controlled Frequency Variant Sensor) *Let the response of a D dimensional continuous sensor to planar waves impinging from a direction Θ be given by*

$$r(\Theta, f) = \int_{\mathbb{R}^D} \rho(\mathbf{x}, f) \exp\left[\jmath 2\pi c^{-1} f\, \mathbf{x}^T \Upsilon(\Theta)\right]\, d\mathbf{x},$$

where $D \in \{1,2,3\}$, $\Upsilon(\Theta) = [\sin\theta, \cos\theta\cos\phi, \cos\theta\sin\phi]^T$, $\mathbf{x} = [x_1, x_2, x_3]^T$ is a point on the sensor, and $\rho : \mathbb{R}^D \times \mathbb{R}^+ \to \mathbb{C}$ is the sensitivity distribution. The sensor beampattern *is a function of* $f^{1-\eta}$ *if*

$$\rho(\mathbf{x}, f) = f^{D\eta} G(\mathbf{x} f^{\eta}), \quad \forall f > 0 \tag{2.21}$$

where $\eta \in [0, 1]$ *and* $G : \mathbb{R}^D \to \mathbb{C}$ *is an arbitrary absolutely integrable complex valued function.* □

Proof: With the substitution $\rho(\mathbf{x}, f) = f^{D\eta} G(\mathbf{x} f^{\eta})$, the response becomes

$$\begin{aligned} r(\Theta, f) &= \int_{\mathbb{R}^D} f^{D\eta} G(\mathbf{x} f^{\eta}) \exp\left[\jmath 2\pi c^{-1} f \, \mathbf{x}^T \Upsilon(\Theta)\right] \, d\mathbf{x} \\ &= \int_{\mathbb{R}^D} G(\xi) \exp\left[\jmath 2\pi c^{-1} f^{1-\eta} \xi^T \Upsilon(\Theta)\right] \, d\xi \end{aligned}$$

with the change of variables $\xi = \mathbf{x} f^{\eta}$. ■

The relationship of the CFVB to both narrowband and frequency invariant beamformers is seen by noting that: (i) for $\eta = 0$ the beampattern varies directly with frequency (corresponding to a conventional single frequency beamformer operated over a range of frequencies), and (ii) for $\eta = 1$ the beampattern is frequency invariant. Thus, the frequency variation of the CFVB beampattern can be controlled in a continuous manner from a classical narrowband beampattern to a frequency invariant beampattern by changing the parameter $\eta \in [0, 1]$.

The importance of the CFVB is that by allowing controlled frequency variation into the beampattern, less sensors are required than for a corresponding FIB. For example, in speech acquisition with a microphone array a bandwidth of several octaves is required. However, it has been shown that directivity is less important at lower frequencies [52]. Thus, it is common to design the beamformer to be FI only over the top two or three octaves of the speech band, and allow directivity to be sacrificed at lower frequencies. The CFVB could be used at the lower frequencies to allow more directivity than could be obtained by a single frequency design, without the requirement for a prohibitive aperture size.

2.5.1 Properties

Without loss of generality properties of the CFVB will only be given for a single sided linear sensor aligned with the x axis. The results could be readily extended to a three dimensional sensor.

Array Length

Let L_1 be the length of the active array at frequency f_1, and similarly for L_2 and f_2. The ratio of active array lengths is given by

$$\frac{L_1}{L_2} = \left(\frac{f_2}{f_1}\right)^{\eta}. \tag{2.22}$$

Aperture Size

Let the active aperture size be P_1 half-wavelengths at frequency f_1, and similarly for P_2 and f_2. The ratio of active aperture sizes is given by

$$\frac{P_1}{P_2} = \left(\frac{f_2}{f_1}\right)^{\eta-1}. \tag{2.23}$$

Dilation

Represent $G(xf^{\eta})$ by $A_f(x)$ and $H_x(f)$ (as in the case of the FIB). Then Theorem 2.5 becomes

$$H_{\gamma x}(f) = H_x(\gamma^{1/\eta} f), \tag{2.24}$$

and a similar relation for the aperture distribution is

$$A_{\gamma f}(x) = A_f(\gamma^{\eta} x). \tag{2.25}$$

Thus the filter responses (and aperture distributions) are related by a dilation property (as was the case for the FIB), but the dilation is not a linear function of the position of the sensor when $\eta \neq 1$.

2.5.2 Sensor Locations

The method of approximating a continuous frequency variant sensor is identical to that for an FI sensor (see §2.4.1); only the spacing function requires further comment. The sensor positioning function for a CFVB with a single sided aperture and the origin at $x_0 = 0$ is given by

$$x_n = \begin{cases} \frac{\lambda_U}{2} n, & \text{for } 0 \leq n \leq P_U \\ \left(\frac{P_n}{P_n - 1}\right) x_{n-1}, & \text{for } P_U < n \leq N-1, \end{cases} \tag{2.26}$$

where

$$P_n = P_{n-1}\left(\frac{x_n}{x_{n-1}}\right)^{1-\frac{1}{\eta}}, \tag{2.27}$$

and P_U is the aperture size at f_U. These equations cannot be solved analytically, so the following recursive procedure is suggested to determine the sensor locations:

1. Choose f_L, f_U, η, and P_U. Calculate P_L from (2.23).
2. Calculate
$$x_{N-1} = \frac{P_L c}{2f_L}.$$
3. Repeat
$$x_n = x_{n+1} - \frac{c}{2f_{n+1}}, \tag{2.28}$$
$$f_n = f_L\left(\frac{x_{N-1}}{x_n}\right)^{\frac{1}{\eta}}, \tag{2.29}$$
until
$$x_n \leq \frac{P_U c}{2f_U}.$$
4. Divide the remainder of the array into P_U equally spaced sections (with spacing of approximately $\lambda_U/2$).

2.6 Design Simulations

2.6.1 FIB Example

To demonstrate the FI theory, consider the design of a beamformer with a single sided uniform aperture and an aperture size of $P = 5$ half-wavelengths. The beamformer is intended to have an FI beampattern over a 10:1 frequency range.

From (2.20) it follows that a minimum of $N = 17$ sensors are required to avoid spatial aliasing. The sensor location relation (2.19) yields the sensor locations shown in Fig. 2.7. These sensor locations have been made dimensionless by expressing them in terms of $\lambda_U/2$. (Using a bandwidth suitable for speech with f_L=300 Hz, f_U=3000 Hz, and $c = 342$ m/s results in an array that is approximately 2.7 metres long.) Note that in this case the last sensor (with a relative sensor location of 25), which is theoretically only active at f_L, is very close to the adjacent one and could be left out without causing severe degradation to the FI response.

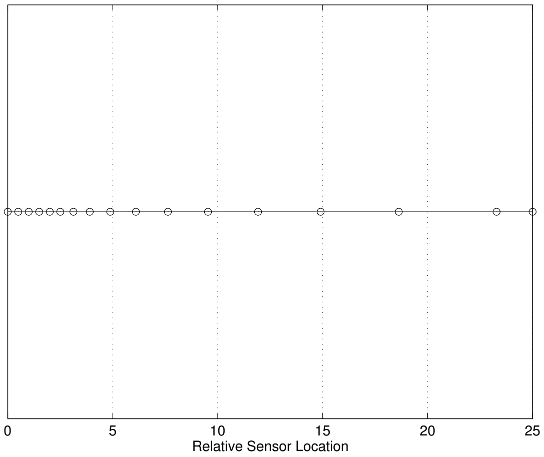


Figure 2.7: Sensor locations for the example FIB, expressed in terms of $\lambda_U/2$.

For a uniform aperture distribution, Theorem 2.4 implies the use of primary filters having ideal lowpass filter characteristics. To demonstrate a practical design, consider the analog implementation of the primary filters with causal 8th order Butterworth lowpass filters (possessing both magnitude and phase components). This will result in an aperture distribution having the same Butterworth shape. The magnitude and phase of the practical aperture distribution are shown in Fig. 2.8 (solid curves) along with the ideal zero-phase uniform aperture distribution (dashed curves); the spatial variable is expressed in terms of half-wavelength. Since a Butterworth lowpass filter is not strictly bandlimited, it follows from Theorem 2.4 that the resultant aperture distribution will not have strictly finite support; the significance of this statement is made apparent later.

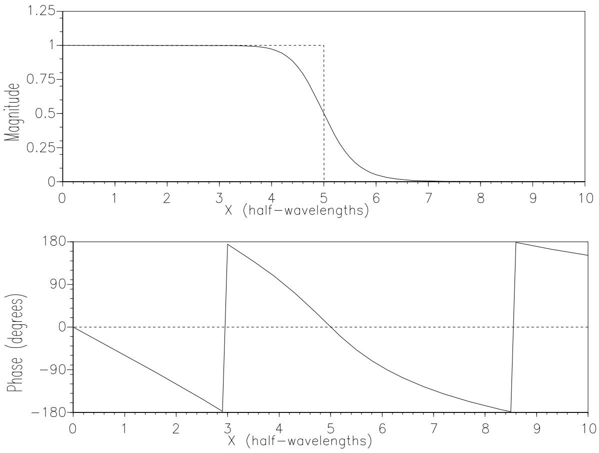


Figure 2.8: Aperture distribution used in the example FIB (solid) and an ideal uniform aperture distribution (dashed).

The beampattern produced by the given aperture distribution is shown in Fig. 2.9 along with the pattern that would be produced by an ideal uniform aperture. The effect of the nonzero phase component of the aperture distribution is apparent in this diagram. The negative slope of the phase is approximately equivalent to delay steering, thus resulting in the main beam being offset from $\theta = 0°$; this effect could be reduced by use of appropriate delays across the array. The asymmetric sidelobes are due to the phase non-linearity.

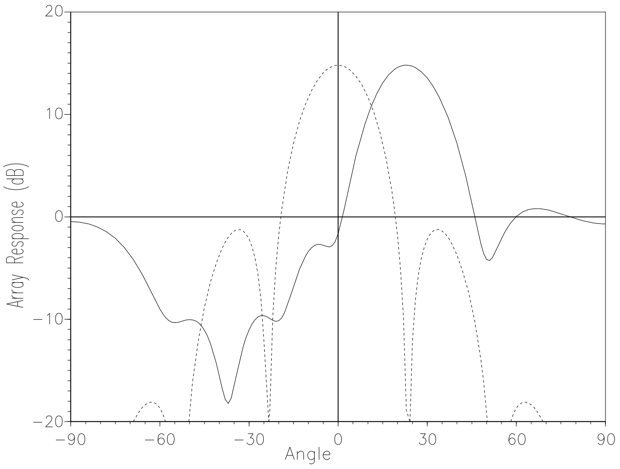


Figure 2.9: Beampattern produced by the example FIB (solid) and an ideal uniform aperture distribution (dashed). The patterns are calculated at f_U.

Applying the trapezoidal approximation method described in §2.4.2 results in the frequency invariant beampattern shown in Fig. 2.10 in which the array spatial response is displayed as a function of frequency over the entire design frequency band. Frequency has been expressed as multiples of f_L. The beampattern is remarkably close to being

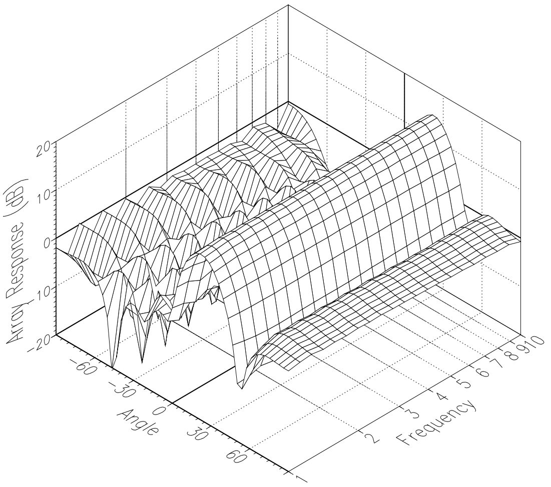


Figure 2.10: Beampattern of example FIB.

frequency independent with negligible variation in main beam magnitude or beamwidth. Slight ripple is evident in the sidelobes. The peaks of the sidelobe ripple correspond to the cutoff frequencies of the sensors given by

$$f_n = \frac{Pc}{2x_n}, \quad n = 0, 1, \ldots N-1.$$

The peak response of the beamformer as a function of frequency is shown in Fig. 2.11. The variation in peak response at frequencies close to f_L is due to the primary filters not being strictly bandlimited, and thus not placing a finite support constraint on the aperture distribution. However, because of the finite size of the array, a portion of the aperture distribution is not realized. This effect is most pronounced at frequencies close to f_L where a significant portion of the aperture distribution is not realized, resulting in a slight difference in beampattern in the lowest portion of the design frequency band. There are several methods which could be used to alleviate this inconsistency in the beampattern at low frequencies:

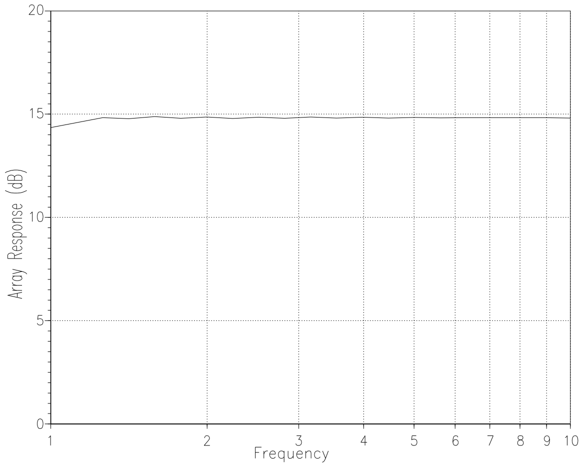


Figure 2.11: Peak response of example FIB.

1. The primary filters could be made strictly bandlimited, thus producing an aperture distribution which has finite support. (This is not physically realizable).
2. The cutoff frequencies of the primary filters could be reduced so that a negligible portion of the aperture distribution was discarded at frequencies close to f_L. This is equivalent to lengthening the array to produce the same result.
3. The secondary filter, which depends only on frequency, could be modified such that the peak main beam level was equalised. This method attempts to compensate for the loss of a portion of the aperture distribution at low frequencies by weighting the

remainder of the aperture more strongly. This demonstrates an important practical consideration of the proposed design method: a simple filter can be used for each of the N primary filters, and any ripple on the main beam level can then be removed by modification of the single secondary filter response.

2.6.2 CFVB Example

To demonstrate the use of the CFVB, a simple design example is presented. The design is for $\eta = 0.75$, covers a frequency range of 10:1, and has an aperture size of $P_U = 5$ half-wavelengths at the upper design frequency. Again causal 8th order Butterworth filters are used to approximate an ideal uniform aperture distribution. The beampattern of this design is shown in Fig. 2.12. This figure should be compared with Fig. 2.10 which shows the beam pattern of a FIB (i.e. $\eta = 1$) with $P = 5$, designed for the same frequency range. The CFVB with $\eta = 0.75$ has a total length of $14.1\lambda_U$ and uses 12 elements, compared with the FIB (with $\eta = 1$) which has a total length of $25\lambda_U$ and uses 17 elements.

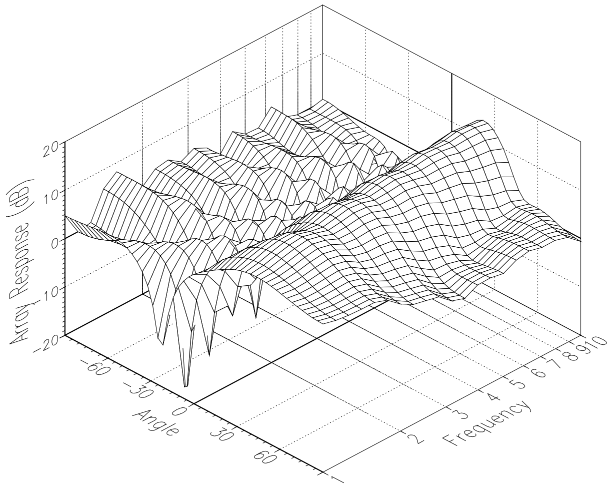


Figure 2.12: Beampattern of example CFVB.

2.7 Conclusions

The general theory of a broadband beamformer having a frequency invariant beampattern over an arbitrarily wide bandwidth has been presented in this chapter. Rather than the conventional signal processing approach of formulating a multi-dimensional optimisation problem to solve for the beamforming parameters, the fundamental nature of the frequency invariant beamforming operation has been addressed. Taking this approach has revealed implicit relationships between the functional requirements of the frequency invariant beampattern and the beamforming structure; these relationships would otherwise be needlessly buried within an optimisation problem.

It was shown that the frequency response of the filter applied to the output of each sensor can be factored into two components: (i) a primary filter response which is related to a slice of the required aperture distribution function, and (ii) a secondary filter which is independent of the sensor and depends only on the dimensions of the array geometry. Furthermore, in the case of a linear array (and for suitable geometries in two and three dimensional arrays) the primary filters are related to each other by a frequency dilation property. This property is exploited in the following chapter to simplify the discrete-time implementation of the FIB.

Chapter 3

Discrete-Time Implementation of a Frequency Invariant Beamformer

3.1 Introduction

THE general theory of the frequency invariant beamformer was derived in the previous chapter. An analog implementation is possible by direct application of the equations. However, as almost all signal processing algorithms are implemented using digital computers, further development of the theory is required to allow discrete-time implementation. Specifically, the implementation of the beamforming filters using discrete-time techniques is presented in this chapter. To simplify the development, and without loss of generality, only linear single-sided array geometries are considered.

In conventional broadband beamforming techniques, for a beamformer with N sensors each feeding an FIR filter with L coefficients, it is necessary to find NL filter coefficients. These coefficients are usually obtained by solving a multi-dimensional optimisation problem. Rather than taking this somewhat mechanical approach, a more insightful design technique is sought in this chapter.

Of prime importance is the dilation property of the primary filters which states that if $H_x(f)$ is the primary filter response at a point $|x| > 0$ on the array, then the primary filter response at a point $\gamma x, \gamma > 0$, is $H_{\gamma x}(f) = H_x(\gamma f)$. As outlined in this chapter, by exploiting this property all primary filter coefficients may be derived from a single set of reference coefficients, regardless of the number of sensors in the array or the bandwidth of operation.

Another way in which the implementation proposed in this chapter differs from conventional broadband beamforming techniques is in the method of steering the beampattern. Conventionally, time delays are inserted in each channel of a broadband beamformer to effect beam steering. In a digital beamformer, an additional fractional delay FIR filter [56] is usually inserted in each channel to produce the required time delay. However, in §3.4 it is shown that beam steering can be performed implicitly by modification of the single set of reference coefficients.

Although this chapter primarily presents the straightforward application of standard signal processing techniques to the design of the FIB, several important design features are highlighted by this presentation. These include the use of a single underlying set of filter coefficients for the primary filters, an example of adaptation applied to the FIB, and the possibility of beam steering without designing additional delay filters for each channel.

To demonstrate that the techniques developed in this chapter may be successfully applied in practice, measured results from a broadband microphone array are presented in §3.6.

3.2 Design of Primary Filters

In this section the design of the primary filters of a FIB is considered. Two related methods are presented: one based on multiple sampling rates and the other on a single sampling rate.

To establish notation, let $H_{\mathrm{ref}}(f)$ be the primary filter response required at some reference location x_{ref} to produce some desired FI response. By the dilation property of the primary filters (see Theorem 2.5), the primary filter response required at the nth sensor location is

$$H_n(f) = H_{\mathrm{ref}}\left(\gamma_n f\right),$$

where

$$\gamma_n = \frac{x_n}{x_{\mathrm{ref}}}, \tag{3.1}$$

is the nth primary filter *dilation factor*. Note that we assume $x_{\mathrm{ref}} > 0$, ensuring that for a linear single-sided array geometry, $\gamma_n \geq 0, \forall n$.

Using discrete-time processing, the nth primary filter response can be written as

$$H_n(f) = \sum_k h_{\mathrm{ref}}[k]\, e^{-j2\pi f \gamma_n T k}, \tag{3.2}$$

where $h_{\mathrm{ref}}[\cdot]$ are the FIR filter coefficients of the reference primary filter and $T = 1/f_s$ is the sampling period for a sampling rate of f_s.

Equation (3.2) demonstrates the existence of a single set of filter coefficients which defines the response of all primary filters. To highlight the practical significance of this fact, we now consider two implementations of (3.2). Methods of obtaining the reference primary filter coefficients are presented in §3.2.4.

3.2.1 Multiple Sampling Rate Method

Let $h_{\mathrm{ref}}[k]$ be a set of filter coefficients which produces a desired primary filter response $H_{\mathrm{ref}}(f)$ at some reference location x_{ref} with a sampling period T, i.e.,

$$H_{\mathrm{ref}}(f) = \sum_k h_{\mathrm{ref}}[k] e^{-\jmath 2\pi f T k}.$$

The primary filters will have the required dilation property if the nth primary filter response is given by

$$H_n(f) = \sum_k h_{\mathrm{ref}}[k] e^{-\jmath 2\pi f T_n k}, \tag{3.3}$$

where

$$T_n = \gamma_n T \tag{3.4}$$

is the sampling period of the nth sensor. The importance of this formulation is it demonstrates that the same set of filter coefficients may be used for each primary filter, with only the sampling rate varying across the array. The sampling rate is a function of the array geometry only.

A common method of multirate sampling is to sample every sensor at the highest rate and then use decimation to achieve the desired sampling rate [96]. In this case it is conventional to express the filter response in terms of the discrete-time frequency variable $\omega = 2\pi f T$. The nth primary filter response can now be written

$$H_n(\omega) = \sum_k h_{\mathrm{ref}}[k]\, e^{-\jmath \omega \gamma_n k}.$$

Using this method, the filtering structure for each primary filter is shown in Fig. 3.1, in which $D_n(\omega)$ and $F_n(\omega)$ are the nth decimation and interpolation filters respectively; these filters have ideal lowpass filter characteristics with cutoff frequencies of $\omega = \pi/\gamma_n$.

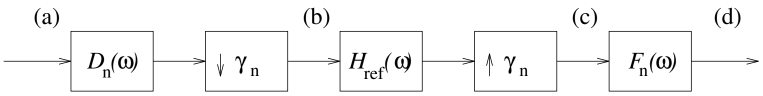


Figure 3.1: Block diagram of the multirate primary filter implementation. Labelled points correspond to the plots of Fig. 3.2.

The symbol $\boxed{\downarrow \gamma}$ denotes downsampling (or decimation) by a factor of γ (i.e., only every γth sample is kept, the rest are discarded), and the symbol $\boxed{\uparrow \gamma}$ denotes upsampling (or expanding) by a factor of γ (i.e., $(\gamma-1)$ zeros are inserted between each sample). Note that the dilation factors γ_n must be integers, requiring modification of the sensor positioning function (2.19) such that $x_n/x_{\mathrm{ref}} \in \mathbb{Z}$. As we will see when we consider the location of x_{ref}, this does not have an adverse effect on the array geometry (such as requiring a prohibitive number of sensors).

The operations performed by this structure are shown in Fig. 3.2. The decimation factors, decimation filters and interpolation filters are only dependent on the array geometry; they do not depend on the beampattern in any way. Thus, modification of the beampattern only requires modification of the reference primary filter coefficients $h_{\mathrm{ref}}[\cdot]$. The beamforming structure constrains the resulting broadband beampattern to be FI over the design band regardless of the actual reference primary filter coefficients used. This important property means that in the case of an adaptive FIB only a single set of coefficients require adaptation—all other parts of the FIB structure are non-adaptive. One simple adaptive FIB algorithm is considered in §3.2.4.

The required sampling rate is now considered. Recall that the aperture length is defined to be P half-wavelengths at all frequencies within the design band. Thus the nth primary filter is (ideally) bandlimited, with

$$H_n(f) = 0, \quad |f| > \frac{Pc}{2x_n}. \tag{3.5}$$

Ignoring the zeroth primary filter (which has a constant response), the primary filter with the widest bandwidth is located at $x_1 = c/(2f_U)$, assuming the sensors are positioned according to (2.19). By substitution into (3.5) the effective bandwidth of this sensor is Pf_U, requiring a sampling rate of

$$f_s \geq 2Pf_U. \tag{3.6}$$

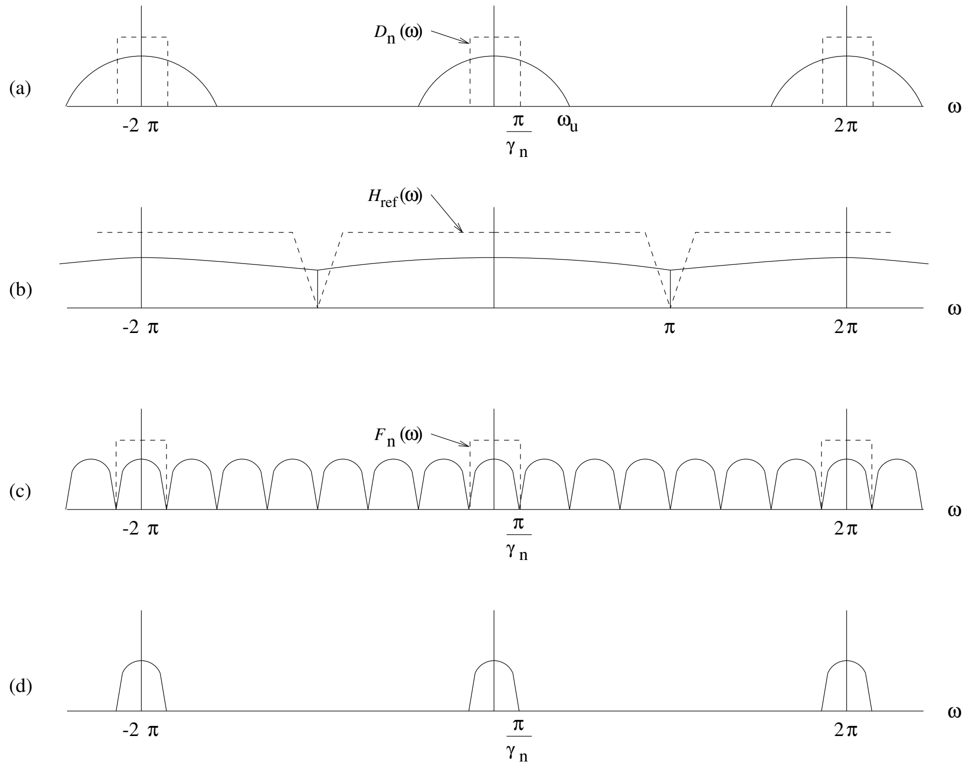


Figure 3.2: Operations performed by the multirate primary filter: (a) the bandlimited input signal spectrum (with the response of the ideal decimation filter shown dotted), (b) the decimated signal spectrum (with the response of the reference primary filter shown dotted), (c) the filtered and upsampled signal spectrum (with the response of the ideal interpolation filter shown dotted), and (d) the output signal spectrum.

The reference sensor in this case is located at

$$x_{\mathrm{ref}} = \frac{c}{2f_U}. \tag{3.7}$$

The requirement that $x_n/x_{\mathrm{ref}} \in \mathbb{Z}$ means the array geometry is essentially uniformly spaced with a spacing of $c/(2f_U)$. For wide bandwidths, the geometry will become thinned (although still with an underlying uniform spacing) for sensors far from the origin.

Note that for a symmetric double sided array geometry the sampling rate can be reduced by half, i.e., $f_s \geq P f_U$.

3.2.2 Single Sampling Rate Method

An alternative method is now presented which utilises the same sampling rate across the array. As in the multiple sampling rate case, assume there is a set of reference coefficients

$h_{\mathrm{ref}}[k]$ having some desired reference primary filter response $H_{\mathrm{ref}}(f)$ at some reference location x_{ref}.

For $n > 0$, the response of the nth primary filter may be written

$$H_n(\omega) = \sum_k h_n[k] e^{-\jmath\omega k}.$$

Hence, a different set of coefficients is used for each sensor. However, these coefficients are a linear combination of the reference coefficients. This is shown as follows. Reconstruct the continuous-time equivalent of the reference coefficients [70, pp. 87–91]:

$$h_{\mathrm{ref}}(t) = \sum_k h_{\mathrm{ref}}[k]\ \mathrm{sinc}[(t - kT)/T],$$

where $\mathrm{sinc}(x) = \sin(\pi x)/(\pi x)$. The frequency scaling property of the Fourier transform is [71, pp. 207–208]:

$$\begin{aligned} \text{If} \quad & x(t) \overset{\mathcal{F}}{\longleftrightarrow} X(\omega) \\ \text{then} \quad & x(at) \overset{\mathcal{F}}{\longleftrightarrow} \frac{1}{|a|} X\left(\frac{\omega}{a}\right), \end{aligned}$$

where a is a real constant and $\mathcal{F}$ denotes the Fourier transform. Hence, by the duality property of the Fourier transform [71, pp. 208–210],

$$H_n(\omega) = H_{\mathrm{ref}}(\gamma_n \omega) \overset{\mathcal{F}}{\longleftrightarrow} \frac{1}{\gamma_n} h_{\mathrm{ref}}\left(\frac{t}{\gamma_n}\right),$$

and,

$$h_n(t) = \frac{1}{\gamma_n} \sum_k h_{\mathrm{ref}}[k]\ \mathrm{sinc}\left[\left(\frac{t}{\gamma_n} - kT\right)/T\right],$$

where we have dropped the absolute value notation since we assume $\gamma_n > 0$. Resampling this frequency scaled impulse response gives

$$h_n[m] = \frac{1}{\gamma_n} \sum_k h_{\mathrm{ref}}[k]\ \mathrm{sinc}\left(\frac{m}{\gamma_n} - k\right). \tag{3.8}$$

Note that even if $h_{\mathrm{ref}}[k]$ is of finite length, there is no requirement that $h_n[m]$ is of finite length. There are three different ranges of γ_n which must be considered:

1. For $\gamma_n \geq 1$, equation (3.8) may be directly applied. The length of the nth primary filter should be $\lceil L\gamma_n \rceil$, where L is the length of the reference primary filter. An example is shown in Fig. 3.3.

2. For $0 < \gamma_n < 1$, temporal aliasing will occur if the reference primary filter has a non-zero frequency response for $|f| > \gamma_n f_s/2$ (or $|\omega| > \gamma_n \pi$). To prevent this aliasing, form a set of modified reference primary filter coefficients

$$\tilde{h}_{\text{ref}}[k] = h_{\text{ref}}[k] \star h_{\text{lpf}}[k],$$

where $h_{\text{lpf}}[\cdot]$ is a set of lowpass filter coefficients having a response with a cutoff frequency of $|\omega| = \gamma_n \pi$, and $\star$ denotes convolution. Equation (3.8) may now be applied using $\tilde{h}_{\text{ref}}[k]$ as the reference primary filter coefficients. In this case the length of the nth primary filter should be $\lceil \tilde{L}\gamma_n \rceil$, where $\tilde{L}$ is the length of $\tilde{h}_{\text{ref}}[\cdot]$. An example is shown in Fig. 3.3.

3. For $\gamma_n = 0$, the nth primary filter coefficients are simply an impulse,

$$h_0[m] = \delta[m] \sum_k h_{\text{ref}}[k],$$

where $\delta[\cdot]$ is the Kronecker delta.

In determining where to locate the reference primary filter, there is a fundamental tradeoff between sampling rate and computational complexity. Recall that the primary filters are ideally bandlimited with $H_n(f) = 0$ for $|f| > (Pc)/(2x_n)$. The reference primary filter should be located such that its non-zero frequency response lies below the fold-over frequency of $|f| = fs/2$ (or $|\omega| = \pi$). Choosing the widest bandwidth possible for the reference primary filter gives

$$x_{\text{ref}} = \frac{Pc}{f_s}. \tag{3.9}$$

Assuming the input signals are bandlimited to f_U, the minimum sampling rate is $f_s = 2f_U$, resulting in $x_{\text{ref}} = Pc/(2f_U)$. From the sensor positioning function (2.19) developed in Chapter 2 it is clear that there will be sensors located closer to the origin than x_{ref}, requiring $0 < \gamma_n < 1$. As demonstrated above, this will require modification of the reference primary filter coefficients for these sensors. Alternatively, the location of the reference primary filter can be set such that there are no sensors in the array for which $0 < \gamma_n < 1$. This results in $x_{\text{ref}} = c/(2f_U)$. But by (3.9) this requires a sampling rate of $f_s = 2Pf_U$ (as in the multirate method). In summary, the obvious options for the location of the reference primary filter are

$$x_{\text{ref}} = \frac{c}{2f_U} \quad \text{and} \quad f_s = 2Pf_U, \tag{3.10}$$

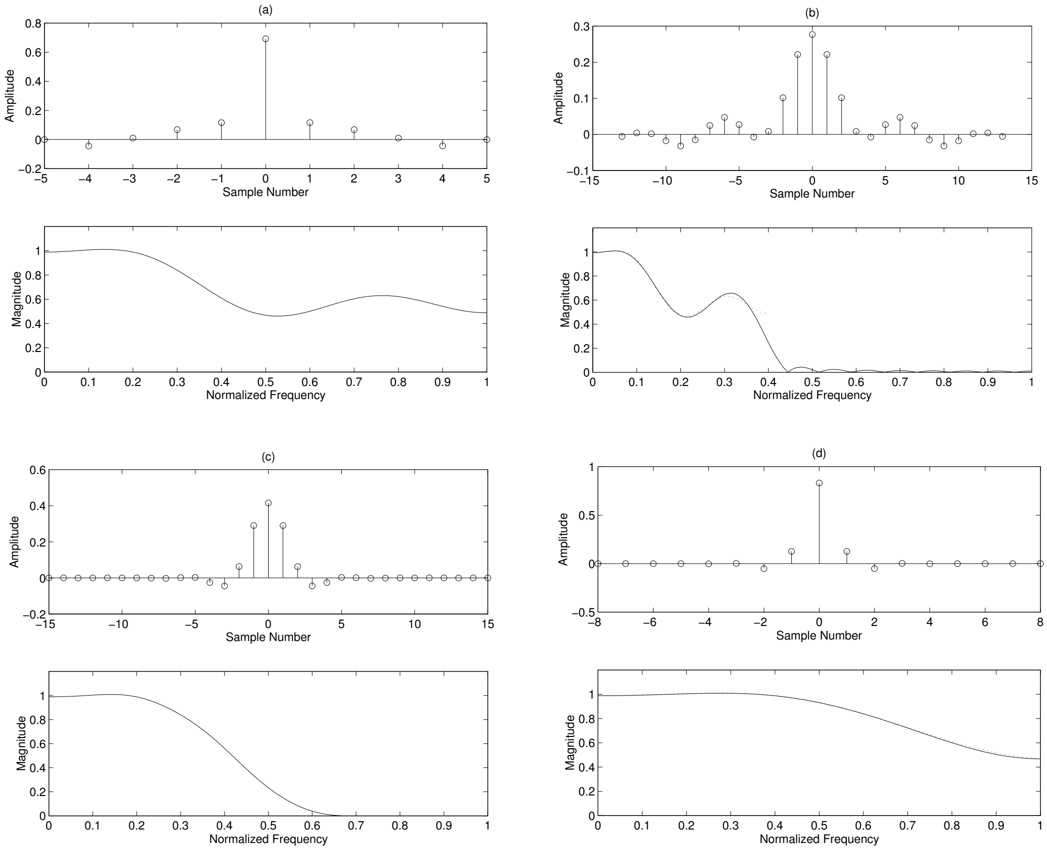


Figure 3.3: Example primary filters. (a) Reference primary filter coefficients ($L = 11$) and corresponding frequency response. (b) Primary filter coefficients for $\gamma_n = 2.5$ calculated from (3.8) and corresponding frequency response (the dotted plot is the desired scaled frequency response). (c) Modified reference primary filter coefficients for $\gamma_n = 0.5$ (convolved with a 21 coefficient Hamming low-pass filter) and corresponding frequency response. (d) Primary filter coefficients for $\gamma_n = 0.5$ using the modified reference primary filter coefficients of (c), and corresponding frequency response (the dotted plot is the desired scaled frequency response).

which allows the use of the same set of reference primary filter coefficients for all sensors but requires a higher sampling rate, or

$$x_{\text{ref}} = \frac{Pc}{2f_U} \quad \text{and} \quad f_s = 2f_U, \tag{3.11}$$

which uses the minimum sampling rate but requires modification of the reference primary filter coefficients for some sensors.

Since effectively each of the primary filters is of a different length, there will be a differing group delay across the array. The effect of this varying group delay is to introduce phase distortion which destroys the FI property of the response. To eliminate this distortion it is necessary to remove the phase component from the reference primary filter response. Assuming the reference primary filter is a linear phase FIR filter, the phase component may be removed by centring $h_{\mathrm{ref}}[k]$ about $k=0$. Although this results in non-causal primary filters when (3.8) is applied, causality can easily be restored by inserting the appropriate number of zeros to the front of each primary filter such that all primary filters have the same group delay.

3.2.3 Comparison of Methods

In this section we compare the number of operations (specifically multiplications) required to implement each of the methods outlined above.

First consider the multirate method. Let the decimation and interpolation filters on each channel be FIR filters with A coefficients each. Similarly, let $H_{\mathrm{ref}}(f)$ be an FIR filter of L coefficients. Hence, for each channel there are $(2A+L)$ computations required to implement this structure. Recall that the primary filter on the zeroth channel is simply an impulse (requiring one multiplication) and no decimation/interpolation is required on the first channel (since it is at x_{ref}). Thus, the total number of operations per sample is

$$\mathrm{NOP}_{\mathrm{MR}} = 1 + L + (N-2)(L+2A). \tag{3.12}$$

Consider the single rate method with the reference primary filter located at $x_{\mathrm{ref}} = c/(2f_U)$. We will refer to this as SR_H (to denote the high sampling rate). In this case no sensors (apart from the zeroth) are located closer to the origin than the reference sensor, so no modification of the reference primary filter coefficients is required for any sensor. The length of the nth filter is $\lceil L\gamma_n \rceil$, (recall $\gamma_n = x_n/x_{\mathrm{ref}}$). With the sensor spacings as defined in Chapter 2, we have

$$\gamma_n = \begin{cases} n, & 0 \le n \le P \\ P\left(\frac{P}{P-1}\right)^{n-P}, & P < n \le N-1. \end{cases}$$

The total number of operations per sample is

$$\mathrm{NOP}_{\mathrm{SR}_H} = 1 + L\sum_{k=1}^{P} k + \sum_{k=1}^{N-P-1} \left\lceil LP\left(\frac{P}{P-1}\right)^k \right\rceil . \tag{3.13}$$

Finally, consider the single rate method with the reference primary filter located at $x_{\mathrm{ref}} = Pc/(2f_U)$. We will refer to this as SR_L (to denote the low sampling rate). In this case there are sensors located closer to the origin than the reference sensor, requiring modification of the reference primary filter coefficients as outlined above. We must treat these channels differently than those located further from the origin than x_{ref}. For $0 < x_n < x_{\text{ref}}$, the filter length is $\lceil \tilde{L}\gamma_n \rceil$ where $\tilde{L} = L + A - 1$, and A is the length of the lowpass FIR filter[1] which is convolved with the original reference filter coefficients to form the modified reference filter coefficients (see §3.2.2). For $x_n > x_{\text{ref}}$ the filter length is $\lceil L\gamma_n \rceil$. It can be shown that the length of the filter on the nth channel is

$$L_n = \begin{cases} 1, & n = 0 \\ \lceil (L+A)n/P \rceil, & 0 < n < P \\ L, & n = P \\ \left\lceil L\left(\frac{P}{P-1}\right)^{n-P} \right\rceil, & P < n \leq N-1, \end{cases}$$

giving the total number of operations per sample as

$$\text{NOP}_{\text{SR}_L} = 1 + \sum_{k=1}^{P-1} \left\lceil (L+A)\frac{k}{P} \right\rceil + \sum_{k=0}^{N-P-1} \left\lceil L\left(\frac{P}{P-1}\right)^{k} \right\rceil . \tag{3.14}$$

The methods are summarised in Table 3.1. It is difficult to make any general conclusions about the complexity of each method from the equations derived above. However, a specific example is considered in §3.5 to compare the number of operations required per second for each method.

Method	**Multirate**	**Single Rate (H)**	**Single Rate (L)**
Definitions	N, no. of sensors P, aperture size L, length of primary filter A, length of decimation and interpolation filters	N, no. of sensors P, aperture size L, length of primary filter	N, no. of sensors P, aperture size L, length of primary filter A, length of LPF which modifies reference primary filter
x_{ref}	$c/(2f_U)$	$c/(2f_U)$	$Pc/(2f_U)$
f_s	$2Pf_U$	$2Pf_U$	$2f_U$
Number of operations per sample	(3.12)	(3.13)	(3.14)
Notes	Uses same set of reference coefficients for all channels	Uses same set of reference coefficients for all channels	Requires modification of the reference coefficients for some channels

Table 3.1: Comparison of methods for implementing a discrete-time FIB.

[1] For comparison purposes, this is the same length as the decimation and interpolation filters in the multirate method.

3.2.4 Reference Filter Coefficients

We now consider three alternate methods of forming the reference primary filter coefficients.

I. Weighted L_2 Optimum

First we consider a design criterion based on weighted least squares optimisation. Specifically, our aim is to find the set of reference coefficients which provides the best least squares fit to a desired FI beampattern over the design frequency band.

Since in both the multiple and single sampling rate methods the nth primary filter response is a linear combination of the reference filter coefficients, it may be written

$$H_n(f) = {\mathbf{h}_{\mathrm{ref}}}^H \boldsymbol{\Psi}_n(f),$$

where

$$\mathbf{h}_{\mathrm{ref}} = [h_{\mathrm{ref}}[-K], \ldots, h_{\mathrm{ref}}[K]]^H$$

is the $L = (2K + 1)$ vector of non-causal reference primary filter coefficients. For the multirate method $\boldsymbol{\Psi}_n(f)$ is given by

$$\boldsymbol{\Psi}_n^{(\mathrm{MR})}(f) = \left[e^{-\jmath 2\pi f T_n(-K)}, \ldots, e^{-\jmath 2\pi f T_n K}\right]^T, \tag{3.15}$$

where, recall, $T_n = \gamma_n T$. For the single rate method where each of the primary filters has $2M + 1$ coefficients,

$$\boldsymbol{\Psi}_n^{(\mathrm{SR})}(f) = \frac{1}{\gamma_n}\mathbf{S}_n\mathbf{e}(f), \tag{3.16}$$

where

$$\mathbf{e}(f) = \left[e^{-\jmath 2\pi f T(-M)}, \ldots, e^{-\jmath 2\pi f T M}\right]^T$$

is the Fourier transform vector,

$$\mathbf{S}_n = [\mathbf{s}_n(-M), \ldots, \mathbf{s}_n(M)]$$

is an $L \times (2M + 1)$ matrix, and

$$\mathbf{s}_n(m) = \left[\operatorname{sinc}\left(\frac{m}{\gamma_n} - (-K)\right), \operatorname{sinc}\left(\frac{m}{\gamma_n} - (-K + 1)\right), \ldots, \operatorname{sinc}\left(\frac{m}{\gamma_n} - K\right)\right]^T.$$

The broadband FIB response is then

$$\begin{aligned}\hat{r}(\theta,f) &= \alpha f \sum_n g_n H_n(f)\, e^{\jmath 2\pi f \tau_n(\theta)} \\ &= \alpha f \sum_n g_n \mathbf{h}_{\text{ref}}{}^H \boldsymbol{\Psi}_n(f)\, e^{\jmath 2\pi f \tau_n(\theta)} \\ &= \mathbf{h}_{\text{ref}}{}^H \boldsymbol{\Gamma}(\theta,f), \end{aligned} \tag{3.17}$$

where the L dimensional vector $\boldsymbol{\Gamma}(\theta,f)$ is given by

$$\boldsymbol{\Gamma}(\theta,f) = \boldsymbol{\beta}(f)\mathbf{a}(\theta,f),$$

$\mathbf{a}(\theta,f)$ is the N dimensional array response vector, and

$$\boldsymbol{\beta}(f) = \alpha f \left[g_1 \boldsymbol{\Psi}_1(f), \dots, g_N \boldsymbol{\Psi}_N(f) \right],$$

is an $L \times N$ matrix.

Define the following cost function which measures the weighted L_2 distance between the desired FI response $r_d(\theta)$ and the actual response over the design frequency band:

$$\begin{aligned} J &= \int_{f_L}^{f_U} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \varphi(\theta,f) \left| \mathbf{h}_{\text{ref}}{}^H \boldsymbol{\Gamma}(\theta,f) - r_d(\theta) \right|^2 \, d\theta \, df \\ &= \mathbf{h}_{\text{ref}}{}^H \mathbf{Q} \mathbf{h}_{\text{ref}} - 2\mathbf{h}_{\text{ref}}{}^H \mathbf{q}_1 + q_0, \end{aligned}$$

where $\varphi(\theta,f)$ is a general weighting function and

$$\begin{aligned} \mathbf{Q} &= \int_{f_L}^{f_U} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \varphi(\theta,f) \boldsymbol{\Gamma}(\theta,f) \boldsymbol{\Gamma}(\theta,f)^H \, d\theta \, df, \\ \mathbf{q}_1 &= \int_{f_L}^{f_U} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \varphi(\theta,f) \boldsymbol{\Gamma}(\theta,f) r_d(\theta)^* \, d\theta \, df, \\ q_0 &= \int_{f_L}^{f_U} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \varphi(\theta,f) |r_d(\theta)|^2 \, d\theta \, df. \end{aligned}$$

Minimising J with respect to $\mathbf{h}_{\text{ref}}$ gives the optimum reference coefficients as

$$\mathbf{h}_{\text{ref}}^{(L_2)} = \mathbf{Q}^{-1} \mathbf{q}_1. \tag{3.18}$$

It is straightforward to show that for the multirate method, the lth element of $\mathbf{\Gamma}(\Theta, f)$ is

$$\mathbf{\Gamma}(\Theta, f)_l = \alpha f \sum_{n=0}^{N-1} g_n e^{\jmath 2\pi f[\tau_n(\theta) - T_n(l-1)]},$$

and for the single rate method the lth element of $\mathbf{\Gamma}(\Theta, f)$ is

$$\mathbf{\Gamma}(\Theta, f)_l = \alpha f \sum_{n=0}^{N-1} \sum_{m} \frac{g_n}{\gamma_n} \mathrm{sinc}\left[\frac{m}{\gamma_n} - (l-1)\right] e^{\jmath 2\pi f[\tau_n(\theta) - Tm]}.$$

In both cases the integrands of $\mathbf{Q}$ and $\mathbf{q}_1$ are well-defined (for a well-defined $r_d(\theta)$), and the double integrals may be evaluated explicitly. However, this involves a reasonable amount of computation. Furthermore, this is a relatively mechanical approach—it provides no indication of the relationship between the reference coefficients and the desired FI beampattern.

II. Beampattern Sampling

Having considered a typical numerical technique above, we now attempt to formulate a more insightful method of determining the reference primary filter coefficients from a desired FI response.

From Chapter 2, the FI response of a continuous finite-support aperture is

$$r(\theta) = \int_0^{x_{\max}} f G(xf)\, e^{\jmath 2\pi f x c^{-1} \sin\theta}\, dx,$$

where $f \in [f_L, f_U]$, and the aperture has finite support for $x \in [0, x_{\max}]$. Let $s = c^{-1}\sin\theta$ and $y = xf$, giving

$$r(s) = \int_0^{y_{\max}} G(y)\, e^{\jmath 2\pi y s}\, dy, \tag{3.19}$$

which is just the inverse Fourier transform of $G(\cdot)$. Hence

$$G(y) = \int_{-\infty}^{\infty} r(s)\, e^{-\jmath 2\pi y s}\, ds. \tag{3.20}$$

Equations (3.19) and (3.20) define the Fourier transform relationship which exists between the aperture distribution and FI response.

Recall from Chapter 2 that the function $G(xf)$ can be interpreted as a filter function $H_x(f)$ at a fixed location x. Hence, (3.20) can be written

$$H_x(f) = \int_{-\infty}^{\infty} r(s)\, e^{-\jmath 2\pi f x s}\, ds.$$

At $x = 1$ this becomes

$$H_x(f)\Big|_{x=1} = \int_{-\infty}^{\infty} r(s)\, e^{-\jmath 2\pi f s}\, ds. \tag{3.21}$$

In terms of its impulse response, this filter response is

$$H_x(f)\Big|_{x=1} = \int_{-\infty}^{\infty} h(t)\, e^{-\jmath 2\pi f t}\, dt. \tag{3.22}$$

Equating (3.21) and (3.22) it is clear that the impulse response of the primary filter required at $x = 1$ is identical to the desired FI response.

Since the simplest method of forming a discrete-time filter from a continuous-time filter is to simply sample the continuous-time impulse response, the set of coefficients for the primary filter at $x = 1$ is given by directly sampling the desired FI response function. By the scaling property of the Fourier transform, the coefficients of the reference primary filter are given by

$$h_{\text{ref}}^{(\text{bs})}[k] = \frac{1}{x_{\text{ref}}}\, r_d\left(\frac{kT}{x_{\text{ref}}}\right), \tag{3.23}$$

where $T = 1/f_s$ is the sampling period, and $r_d(\cdot)$ is the desired FI response. Typically, $r_d(s)$ is specified only within the visible range $s \in [-1/c, 1/c]$, corresponding to real angles. Thus, the number of reference coefficients will be limited to

$$k \in \{-\lfloor x_{\text{ref}}/(Tc)\rfloor, \ldots, \lfloor x_{\text{ref}}/(Tc)\rfloor\},$$

where $\lfloor\cdot\rfloor$ denotes the floor (next lower integer) function.

Note that a real response is only produced by a Hermitian symmetric aperture (in the same way as a real frequency response is only produced by a Hermitian symmetric impulse response). Thus, for a single-sided aperture (or any other aperture which is not Hermitian symmetric) the desired response $r_d(\cdot)$ which is sampled in (3.23) must be an appropriate complex-valued function.

The beampattern sampling technique is a very simple and insightful method of obtaining the reference primary filter coefficients from a desired FI response. As shown in §3.5, the response obtained from the beampattern sampling technique provides a good approximation to the desired FI response, and because of its simplicity, is generally to be preferred over the weighted least squares optimisation method. In the remainder of the thesis, the beampattern sampling technique has been used exclusively in any simulation in which the reference primary filter coefficients are obtained from a desired FI response.

III. Linear Constrained Minimum Variance FI Beamforming

As a third example, we consider determining the reference primary filter coefficients in an adaptive environment where no desired FI beampattern is available. Specifically, a linear constrained minimum variance (LCMV) formulation is considered.

It was shown in (3.17) above that the response of a FIB may be written as

$$\hat{r}(\theta, f) = \mathbf{h}_{\text{ref}}{}^{H} \boldsymbol{\beta}(f) \mathbf{a}(\theta, f),$$

where $\mathbf{h}_{\text{ref}}$ is an L vector of reference primary filter coefficients which define the FI beampattern, $\boldsymbol{\beta}(f)$ is an $L \times N$ matrix, and $\mathbf{a}(\theta, f)$ is the N dimensional array response vector.

Let $\mathbf{s}(t; f)$ be the N vector of received signals at the sensor array at time t with frequency component f. The beamformer output at this time is then

$$z(t) = \mathbf{h}_{\text{ref}}{}^{H} \boldsymbol{\beta}(f) \mathbf{s}(t; f).$$

The expected beamformer output power is

$$\begin{aligned} E\{|z(t)|^2\} &= E\{\mathbf{h}_{\text{ref}}{}^{H} \boldsymbol{\beta}(f) \mathbf{s}(t; f) \mathbf{s}(t; f)^{H} \boldsymbol{\beta}(f)^{H} \mathbf{h}_{\text{ref}}\} \\ &= \mathbf{h}_{\text{ref}}{}^{H} \boldsymbol{\beta}(f) \mathbf{R}_s(f) \boldsymbol{\beta}^{H}(f) \mathbf{h}_{\text{ref}}, \end{aligned} \tag{3.24}$$

where $\mathbf{R}_s(f) = E\{\mathbf{s}(t; f) \mathbf{s}(t; f)^{H}\}$ is the signal covariance matrix for frequency f.

Divide the design frequency band into J sub-bands centred on $f_j, j = 1, \dots, J$ and let

$$\mathbf{D} = \sum_{j=1}^{J} \boldsymbol{\beta}(f_j) \mathbf{R}_s(f_j) \boldsymbol{\beta}^{H}(f_j).$$

(This frequency decomposition may be performed either via a bank of J bandpass filters, or by data segmentation and discrete Fourier transform.) Consider the LCMV beamforming problem:

$$\min_{\mathbf{h}_{\text{ref}}} \mathbf{h}_{\text{ref}}{}^{H}\, \mathbf{D}\, \mathbf{h}_{\text{ref}}, \tag{3.25a}$$

$$\text{subject to } \mathbf{C}^{H}\mathbf{h}_{\text{ref}} = 1, \tag{3.25b}$$

where $\mathbf{C} = \boldsymbol{\Gamma}(\theta_0, f_0)$ is a vector which constrains the response to be unity in the source direction θ_0 at some frequency $f_0 \in [f_L, f_U]$. Solution of this optimisation problem finds the reference primary filter coefficients which minimise the beamformer output power subject to a constraint that the response is unity in a given source direction θ_0. Note that because the beamformer is structurally constrained to be frequency invariant, a broadband unity response is imposed in the source direction by enforcing a linear response constraint at a single frequency. The solution of the optimisation problem (3.25) may be found by the method of Lagrange multipliers as

$$\mathbf{h}_{\text{ref}} = \mathbf{D}^{-1}\mathbf{C}\left[\mathbf{C}^{H}\mathbf{D}^{-1}\mathbf{C}\right]^{-1}. \tag{3.26}$$

This problem is identical to that considered by Frost [32]. Frost developed a least-mean squares algorithm to minimise the output power of a broadband array while maintaining a chosen frequency characteristic in the look direction. Applying Frost's method, the adaptive algorithm which converges to the optimum solution (3.26) may be written as:

$$\mathbf{h}_{\text{ref}}(0) = \mathbf{q} \tag{3.27}$$

$$\mathbf{h}_{\text{ref}}(t+1) = \mathbf{Q}\left[\mathbf{h}_{\text{ref}}(t) - \mu\widehat{\mathbf{D}}(t)\mathbf{h}_{\text{ref}}(t)\right] + \mathbf{q}, \tag{3.28}$$

where $\mathbf{h}_{\text{ref}}(t)$ is the set of reference primary filter coefficients to use at time t, μ is the adaptation step size, $\mathbf{q}$ is the L vector

$$\mathbf{q} = \mathbf{C}\left[\mathbf{C}^{H}\mathbf{C}\right]^{-1},$$

and $\mathbf{Q}$ is the $L \times L$ matrix

$$\mathbf{Q} = \mathbf{I} - \mathbf{C}\left[\mathbf{C}^{H}\mathbf{C}\right]^{-1}\mathbf{C}^{H}.$$

The $L \times L$ matrix $\widehat{\mathbf{D}}(t)$ is given by

$$\widehat{\mathbf{D}}(t) = \sum_{j=1}^{J} \boldsymbol{\beta}(f_j)\mathbf{s}(t; f_j)\mathbf{s}(t; f_j)^{H}\boldsymbol{\beta}^{H}(f_j), \tag{3.29}$$

where the outer product of $\mathbf{s}(t; f)$ is used as a simple approximation to $\mathbf{R}_s(f)$ at the t-th time instant.

The LCMV-FIB algorithm is summarised in Table 3.2.

LCMV-FIB Algorithm:

(i) Initially set $\mathbf{h}_{\text{ref}}(0) = \mathbf{q}$. (Clearly this satisfies the constraint (3.25b).)

(ii) Store the received signal vector $\mathbf{s}(t)$ at time t. (This is usually referred to as a *snapshot*.)

(iii) Perform frequency decomposition on this snapshot to give the narrowband signal vectors $\mathbf{s}(t; f_j), j = 1, \ldots, J$.

(iv) Calculate $\widehat{\mathbf{D}}(t)$ from (3.29), and update the reference coefficients using (3.28).

(v) Repeat steps (ii) to (iv) for each new snapshot.

Table 3.2: Example linear constrained minimum variance algorithm for the FIB.

Comparison of Methods

A summary of the three different methods for determining the reference primary filter coefficients is given in Table 3.3

Method	**Weighted L_2 Optimum**	**Beampattern Sampling**	**LCMV Algorithm**
Given	Desired FI pattern $r_d(\theta)$ Weighting function, $\varphi(\theta, f)$	Desired FI pattern, $r_d(\theta)$	Look direction, θ_0
Criterion	Minimise weighted least squares variation from $r_d(\theta)$ over design bandwidth.	Exploit Fourier transform relationships of FIB.	Minimise received signal power subject to a linear constraint on the look direction response.
Solutions	(3.18)	(3.23)	(3.27), (3.28)
Notes	Requires computation of double integrals.	Number of filter coefficients is limited.	Requires frequency decomposition.

Table 3.3: Comparison of methods for designing reference primary filter coefficients.

3.3 Design of Secondary Filter

Because the secondary filter is effectively a differentiator, its design is straightforward. It should either be a type III FIR filter (i.e., odd length with odd symmetric impulse response) or type IV FIR filter (i.e., even length with odd symmetric impulse response) [74]. The type III filter is constrained to have zero response at $\omega = 0$ and $\omega = \pi$, whereas the type IV filter only constrains the response to be zero at $\omega = 0$. Hence, the type IV filter provides a wider bandwidth than the type III filter. A simple example is presented to illustrate the use of both types of FIR filter as a secondary filter.

Consider the design of a secondary filter for operation in the frequency range 1–2 kHz with a sampling rate of $f_s = 16$ kHz. Figure 3.4 shows the response of a type III FIR filter with $L = 7$ coefficients and Fig. 3.5 shows the response of a type IV FIR filter with $L = 6$ coefficients.

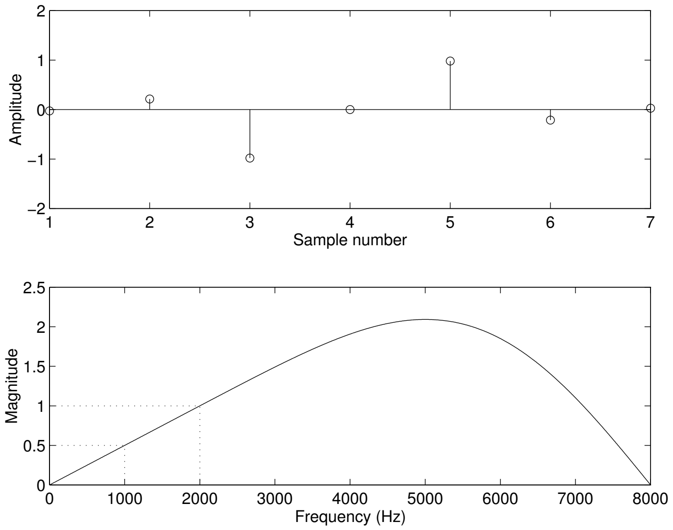


Figure 3.4: Response of type III secondary filter. The top plot shows the filter coefficients ($L = 7$) and the bottom plot shows the magnitude of the frequency response. The dotted lines indicate the design frequency band.

3.4 Beam Steering

So far it has been assumed that the peak of the FIB response is at broadside. Conventionally, beam steering in a broadband beamformer is achieved by using appropriate delay elements on each sensor. In a digital broadband beamformer the required delays are produced by fractional delay filters [56]. In this section it is demonstrated that beam steering can be achieved without explicitly requiring additional delay filters on each sensor.

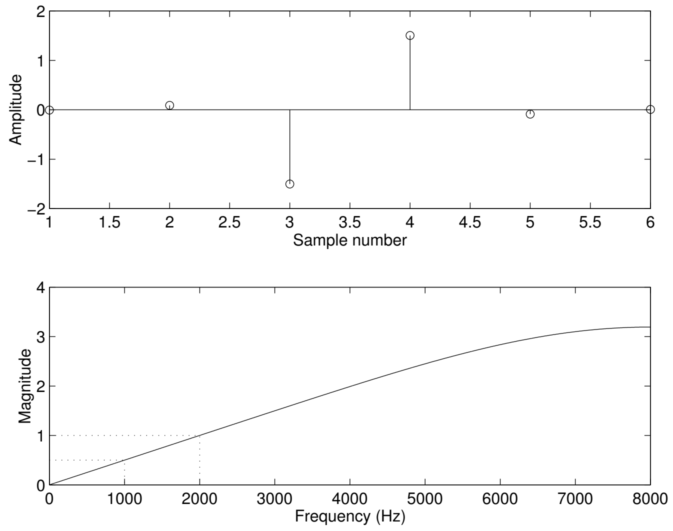


Figure 3.5: Response of type IV secondary filter. The top plot shows the filter coefficients ($L = 6$) and the bottom plot shows the magnitude of the frequency response. The dotted lines indicate the design frequency band.

When an array is used to approximate the ideal FI aperture (as detailed in Chapter 2), the resulting response function $r(s)$ becomes periodic[2] with a period of $2/c$, i.e., corresponding to the visible range of $s = c^{-1} \sin\theta$. Because of this periodicity, beam steering results in a circular shift of the response function. Equation (3.23) demonstrated that one method of obtaining the reference primary filter coefficients is by direct sampling of the desired response function. It follows that for a discrete set of steering angles, beam steering can be achieved by circular rotation of the unsteered primary filter coefficients. This set of angles can be calculated for a given aperture size and sampling rate as follows.

Let θ_s denote the steering angle from broadside. Substituting $s = c^{-1} \sin\theta_s$ into (3.23), with x_{ref} and f_s as given in §3.2, gives

$$\sin\theta_s = \frac{ac}{x_{\text{ref}} f_s} = \frac{a}{P}, \tag{3.30}$$

where $a \in \mathbb{Z}$ is the number of coefficients by which the reference primary filter coefficients have been rotated. This is analogous to the case where steering can be achieved using integer delay filters (rather than fractional delay filters). However, rather than explicitly requiring the design of an integer delay filter on each channel, these delay filters are implicitly included by the circular rotation of the reference primary filter coefficients.

[2] Strictly speaking, the period of $r(s)$ is $2/c$ only for a uniformly spaced array, which only occurs in the FIB at the upper design frequency.

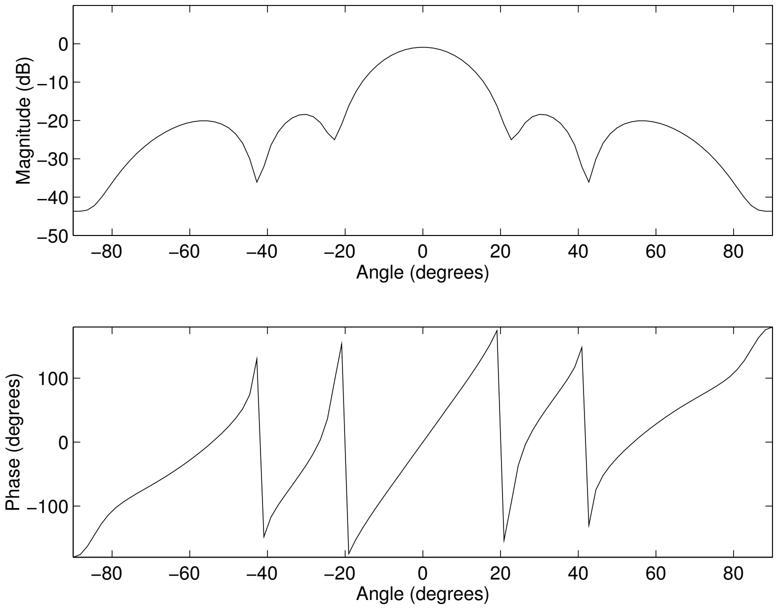


Figure 3.6: Desired beampattern for the example FIB.

To steer to angles other than those given by (3.30), the beampattern which is sampled in (3.23) can be rotated by the appropriate amount (or equivalently, the reference primary coefficients may be interpolated and shifted). This case corresponds to that of using fractional delay filters, but again there is no requirement for the explicit design of a fractional delay filter on each channel—these delay filters are implicitly included.

Beam steering by circular rotation of the reference primary filter coefficients is demonstrated in the following section.

3.5 Design Simulations

To illustrate the methods developed in this chapter, consider the implementation of a FIB with an aperture size of $P = 6$ half-wavelengths, designed to cover the band 1–2 kHz. The speed of wave propagation is 342 metres/second (i.e., sound waves propagating in air). The desired beampattern is shown in Fig. 3.6.

The reference primary filter coefficients were found by direct sampling of the desired beampattern through (3.23). This resulted in the set of $L = 13$ reference coefficients given in Table 3.4. The secondary filter had $L_s = 6$ coefficients.

Three examples were considered. The first was a multirate implementation using an array of $N = 13$ sensors with a total array size of 1.105 metres. The sampling rate was

k	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
$h_{\text{ref}}[k]$	0	0.1-j0.02	-0.02	0.1-j0.07	-0.14	0.1-j0.63	0.9	0.1+j0.63	-0.14	0.1+j0.07	-0.02	0.1+j0.02	0

Table 3.4: Reference primary filter coefficients used in the example FIB.

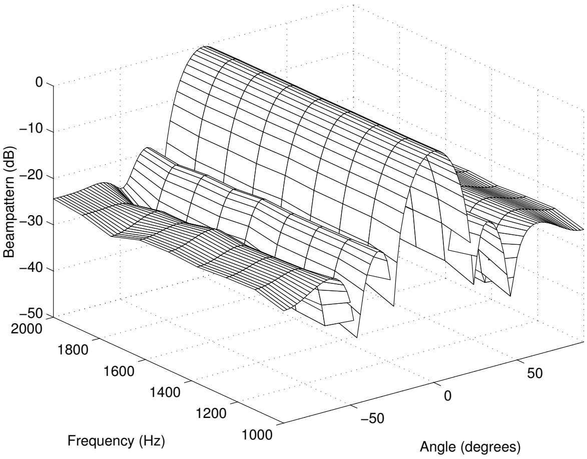


Figure 3.7: Beampattern of the example multirate FIB.

given by (3.6) as $f_s = 24$ kHz and the location of the reference filter was as given by (3.7). The response of the multirate FIB at 9 frequencies within the design band is shown in Fig. 3.7.

The second example was a single rate implementation using an array of $N = 11$ sensors with a total array size of 1.02 metres. (The array geometry was different to the multirate implementation since the sensor locations must be quantised for the multirate case; see §3.2.1 for details.) The sampling rate and reference sensor location were as given by (3.10), again requiring a sampling rate of $f_s = 24$ kHz. The response of the single rate FIB at 9 frequencies within the design band is shown in Fig. 3.8. The single rate design was repeated with the sampling rate and reference sensor location as given by (3.11), requiring a sampling rate of $f_s = 4$ kHz. The results were virtually identical to the other two designs, and are not shown.

The final example was again a single rate implementation, and demonstrates the beam steering method outlined in §3.4. The array geometry and sampling rate were as for Fig. 3.8, but the reference primary filter coefficients were a circularly rotated copy of the coefficients in Table 3.4, with the coefficients rotated by 2 positions. From (3.30) this will result in the main beam being steered to 19.5°. The angles to which the main beam could

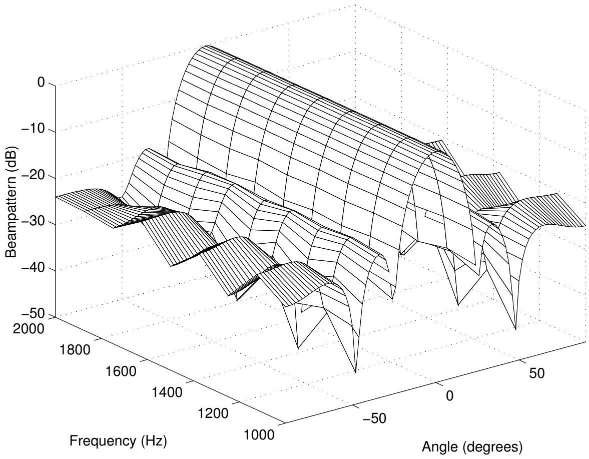


Figure 3.8: Beampattern of the example single rate FIB.

be steered by circular rotation of the reference coefficients are

$$\theta_s = \{0, \pm 9.6, \pm 19.5, \pm 30, \pm 41.8, \pm 56.4, \pm 90\} .$$

The response of the FIB at 9 frequencies within the design band is shown in Fig. 3.9 in which it can be seen that the main beam is steered to the calculated angle.

It is now instructive to calculate the number of operations required by this example for each of the three methods. These were calculated using (3.12), (3.13) and (3.14). The results are as follows:

$$\mathrm{NOP}_{\mathrm{MR}} = 1031$$
$$\mathrm{NOP}_{\mathrm{SR}_H} = 778$$
$$\mathrm{NOP}_{\mathrm{SR}_L} = 256$$

where SR_H is the single rate method with $x_{\mathrm{ref}} = c/(2f_U)$ and $f_s = 2Pf_U$, and SR_L is the single rate method with $x_{\mathrm{ref}} = Pc/(2f_U)$ and $f_s = 2f_U$. Multiplying each of these by the appropriate sampling rate, we obtain the following number of operations per second for each method.

$$\mathrm{NOPS}_{\mathrm{MR}} = 24.7 \times 10^6$$
$$\mathrm{NOPS}_{\mathrm{SR}_H} = 18.7 \times 10^6$$
$$\mathrm{NOPS}_{\mathrm{SR}_L} = 1.0 \times 10^6$$

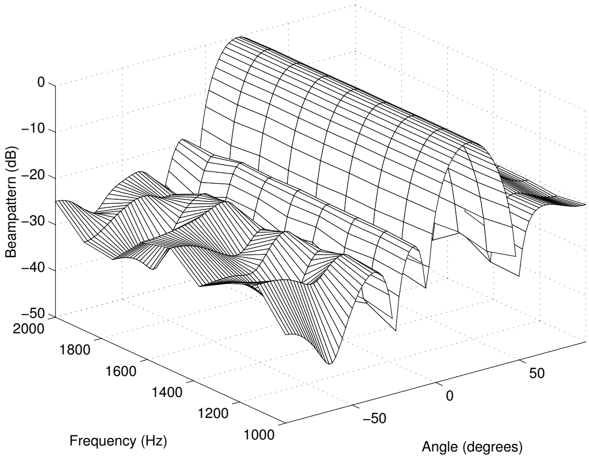


Figure 3.9: Beampattern of the example steered single rate FIB. Steering is achieved by circular rotation of the reference primary filter coefficients used in Fig. 3.8.

Note that methods MR and SR$_H$ are of similar magnitude (because they have the same high sampling rate), but the lower sampling rate of method SR$_L$ affords an order of magnitude less operations per second.

3.6 Microphone Array Test Results

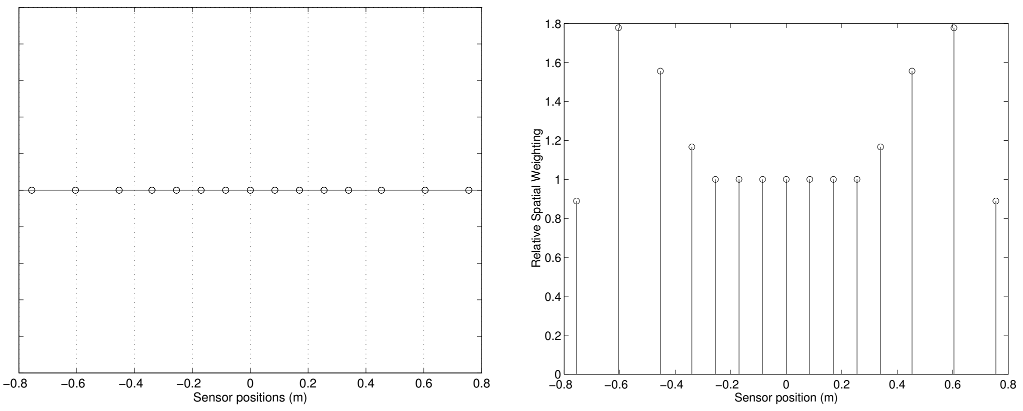
To confirm that the frequency invariant beamformer can be successfully implemented in practice, experimental results were obtained from a microphone array. Full details of the microphone array testing system are given in Appendix A.

3.6.1 Beamformer Design

A single rate FIB was designed to have a Chebyshev 25 dB beampattern over the octave band 1–2 kHz, with an aperture size of $P = 8$ half-wavelengths and a sampling rate of $f_s = 8$ kHz. A symmetric double-sided linear array geometry[3] requiring 15 sensors was used. The sensor positions (2.19) and spatial weighting terms (2.8) are shown in Fig. 3.10.

The reference primary filter coefficients were obtained by sampling the desired beampattern through (3.23), using $x_{\rm ref} = (P/2\,c)/(2 f_U)$. Filter coefficients for each primary filter were obtained as outlined in §3.2.2; a 21 coefficient Hamming low pass filter was used

[3] Although they are not explicitly covered, double-sided linear array geometries are a straightforward extension of the single-sided linear array geometry considered in this chapter.



(a)

(b)

Figure 3.10: (a) Sensor positions, and (b) spatial weighting terms for the experimental microphone array.

to modify the reference coefficients when required (see §3.2.2 for details). Because of the symmetry of the array geometry, only 7 different primary filter response were required. The primary filter coefficients and resulting filter responses are shown in Figs 3.11 and 3.12 respectively. These figures clearly show the dilation property of the primary filters.

A secondary filter with four coefficients was designed according to §3.3. The secondary filter coefficients and filter response are shown in Fig. 3.13.

Using computer simulations, the resulting beampattern was calculated at five frequencies uniformly spaced within the design frequency band. A comparison of the simulated beampatterns with the desired Chebyshev 25 dB beampattern is shown in Fig. 3.14, in which the solid curve is the desired beampattern and the dotted curves are the simulated beampatterns. The simulated beampatterns match the desired beampattern closely in the main beam region. However, the simulated beampatterns do not in general achieve the desired 25 dB sidelobe suppression. This is consistent with the fact that the reference filter coefficients were obtained through the simplest method of beampattern sampling (see §3.2.4).

3.6.2 Experimental Results

The microphone array was horizontally mounted on a rotating shaft 1.2 metres above the ground. Its output power was measured as it was rotated through 180° in the presence of a sinusoidal signal emitted from a loudspeaker. The source was located approximately

15 metres from the array centre to give farfield results.[4] Since an anechoic chamber large enough to perform farfield measurements was not available, the experiments were performed in an open area away from any nearby reflecting objects.[5] Although these test conditions were far from ideal, they were considered adequate for the preliminary experimental verification that was sought.

The beampattern of the experimental microphone array was measured at five frequencies within the design bandwidth. The results are shown in Fig. 3.15, in which the solid curve is the desired Chebyshev 25 dB beampattern, the dotted curves are the simulated beampatterns (from Fig. 3.14), and the circles indicate the measured values. Each plot has been normalised such that the peak is unity.

The results shown in Fig. 3.15 indicate that at each frequency the measured beampattern sufficiently closely matches the desired and simulated beampatterns in the main beam region. The sidelobes are generally about 5 dB higher than the simulated sidelobes.

There are three factors which contributed to the difference between measured and simulated beampatterns in Fig. 3.15. First, there was a reasonable amount of background noise which meant that it was virtually impossible to achieve an output power as low as -25 dB. Second, the computer simulations assume identical sensors, whereas the microphones used were not. Accurate calibration of the microphones is required to equalise the gains of the channels. However, because of the non-ideal test conditions, this calibration could not be accurately performed. The third factor causing pattern degradation was a consequence of the nature of the data acquisition card used in the testing system. Specifically, the microphone signals were sampled sequentially, rather than simultaneously as assumed in the computer simulations. For an equally spaced array this would result in a conventional steered main beam. However, because the array used was nonuniformly spaced, the staggered sampling also resulted in some beampattern degradation.[6] Because of this staggered sampling, the measured beampatterns are steered to approximately $-5°$. (In Fig. 3.15, the desired and simulated beampatterns are also steered to $-5°$ to facilitate comparison with the measured patterns.)

[4] The general rule-of-thumb is that the farfield approximation becomes valid at a distance of $2L^2/\lambda$, where L is the largest array dimension at the operating wavelength λ [61]. For an active aperture size of P half-wavelengths, this distance can be expressed as $P^2\lambda/2$. For a 1–2 kHz bandwidth with a propagation speed of $c = 342$ m/s, the required farfield distance is 10.94 metres at 1 kHz and 5.47 metres at 2 kHz.

[5] Multipath signals would no doubt have been received by the array via reflection from the ground. However, because the tests were performed on a grassed surface, it was assumed that these reflections were randomly scattered. Thus, no significantly correlated reflections should have been received at the array.

[6] This degradation could have been removed by inserting appropriate fractional delay filters in each channel. However, because of the computational constraints imposed by the DSP hardware, this could not be achieved in real-time.

To compare the measured beampatterns at different frequencies, the normalised results are plotted in Figs 3.16(a) and (b) using two different views of the same data. Despite the non-ideal test conditions, Fig. 3.16 indicates that the response of the experimental microphone array was approximately frequency invariant over the octave design bandwidth. This provides a preliminary verification that the frequency invariant beamforming theory can be successfully applied in practice. More controlled conditions would naturally be required for a detailed evaluation of the usefulness of the method to specific applications.

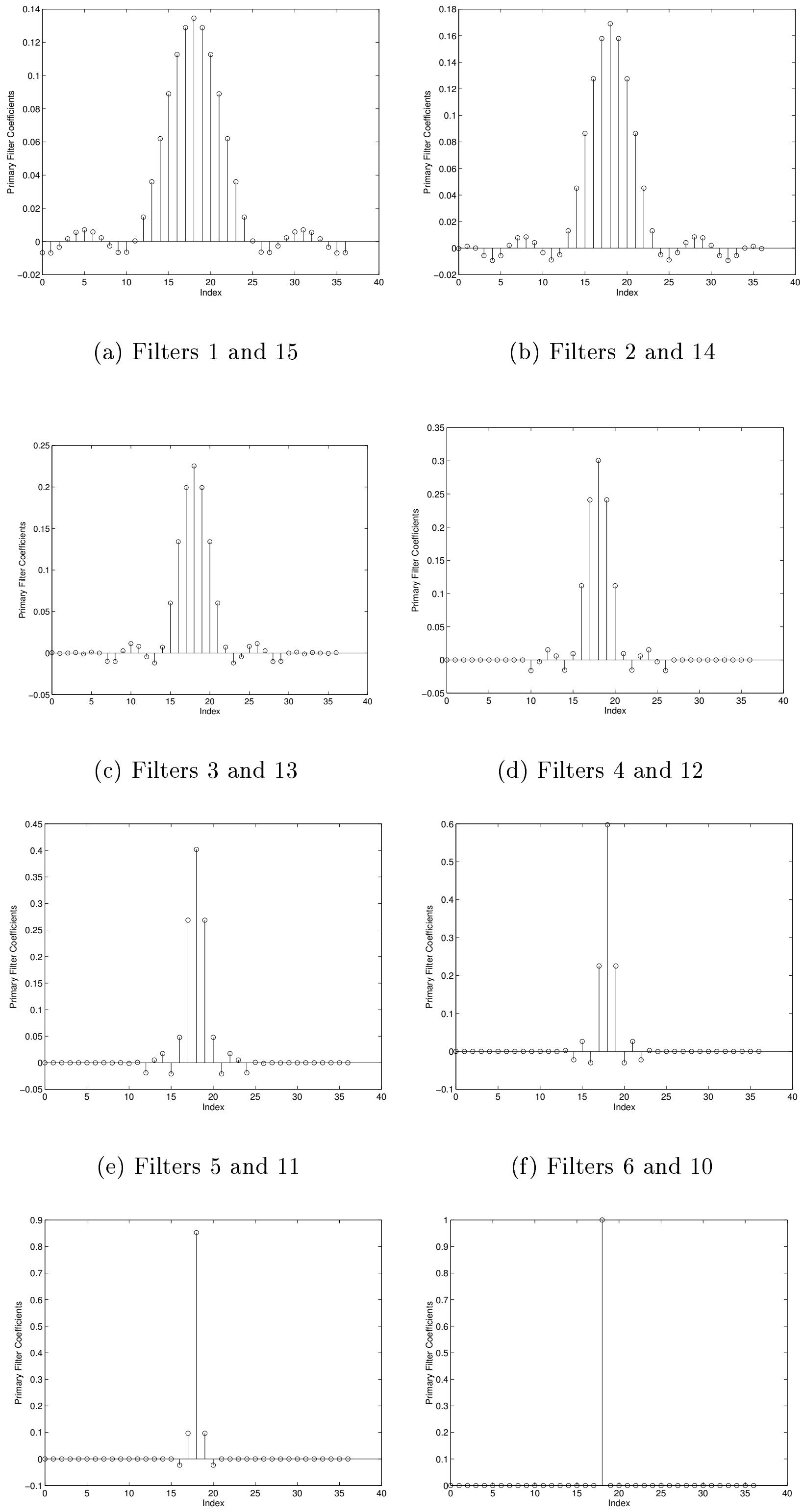
3.7 Conclusions

Implementation of the frequency invariant beamformer using discrete-time processing was considered in this chapter. Two related implementations were presented for determining the coefficients of the primary filter on each sensor. One was based on multirate processing and the other on a single sampling rate.

The feature of both methods is that there is an underlying set of reference coefficients from which all primary filter responses are derived. This set of coefficients defines the frequency invariant beampattern over the entire design band. The actual number of reference coefficients is independent of the number of sensors in the array and the bandwidth of operation.

Three methods of obtaining the reference coefficients were considered. Of these, the most important (from both a practical and theoretical standpoint) was a beampattern sampling technique, whereby the reference primary filter coefficients can be obtained immediately from the desired frequency invariant response, without the need to consider the required aperture distribution or primary filter frequency responses.

Finally, experimental results were obtained from a microphone array which verified that the frequency invariant beamforming theory could be successfully applied in a practical setting.



(a) Filters 1 and 15

(b) Filters 2 and 14

(c) Filters 3 and 13

(d) Filters 4 and 12

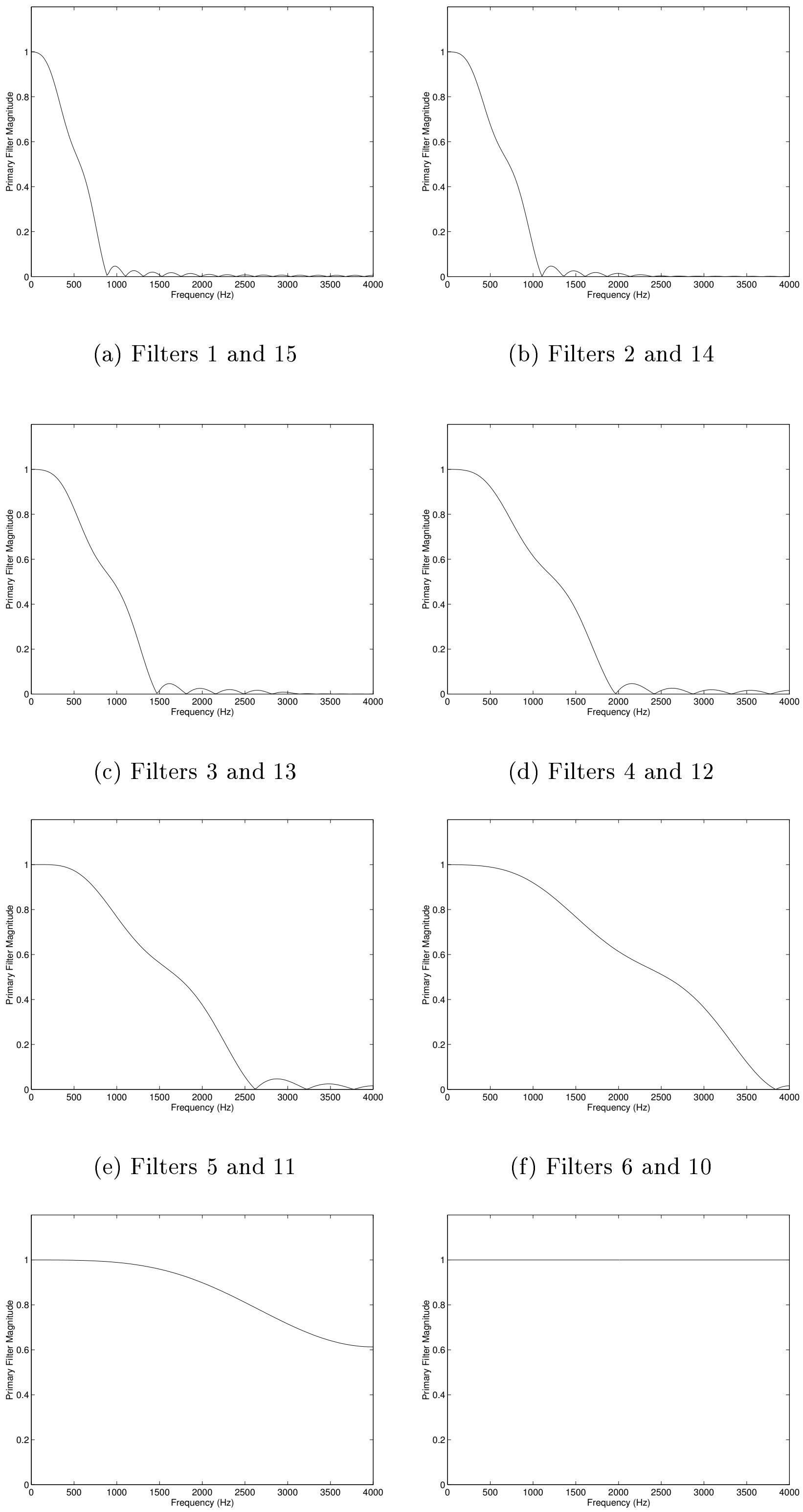
(e) Filters 5 and 11

(f) Filters 6 and 10

(g) Filters 7 and 9

(h) Filter 8

Figure 3.11: Primary filter coefficients used in the experimental microphone array.



(a) Filters 1 and 15

(b) Filters 2 and 14

(c) Filters 3 and 13

(d) Filters 4 and 12

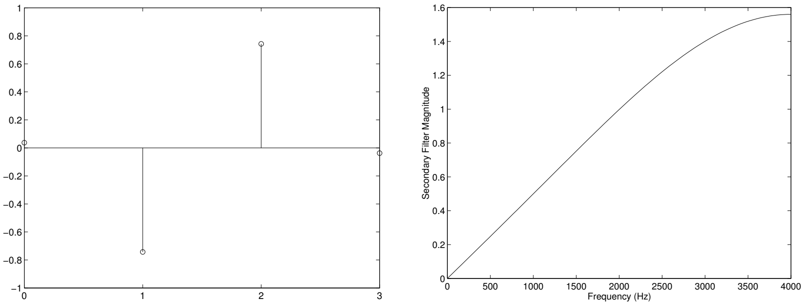
(e) Filters 5 and 11

(f) Filters 6 and 10

(g) Filters 7 and 9

(h) Filter 8

Figure 3.12: Primary filter responses used in the experimental microphone array.



(a)

(b)

Figure 3.13: (a) Secondary filter coefficients, and (b) secondary filter response for the experimental microphone array.

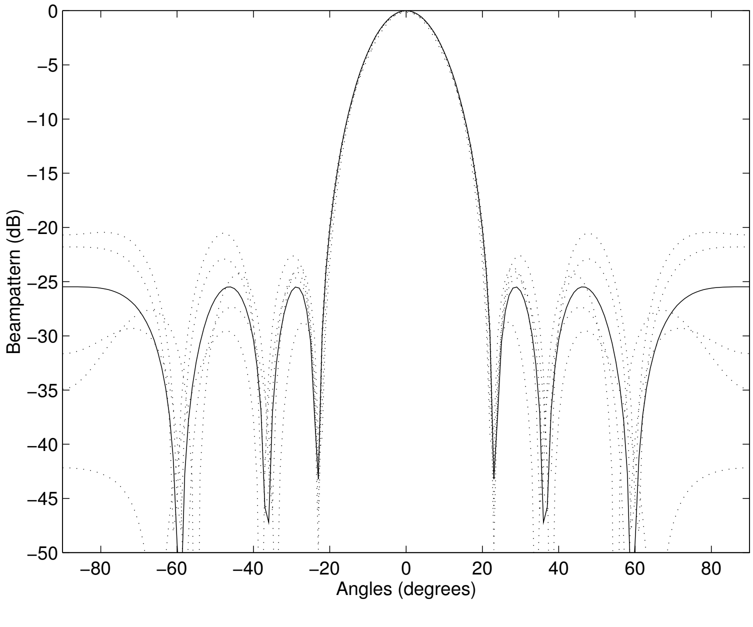
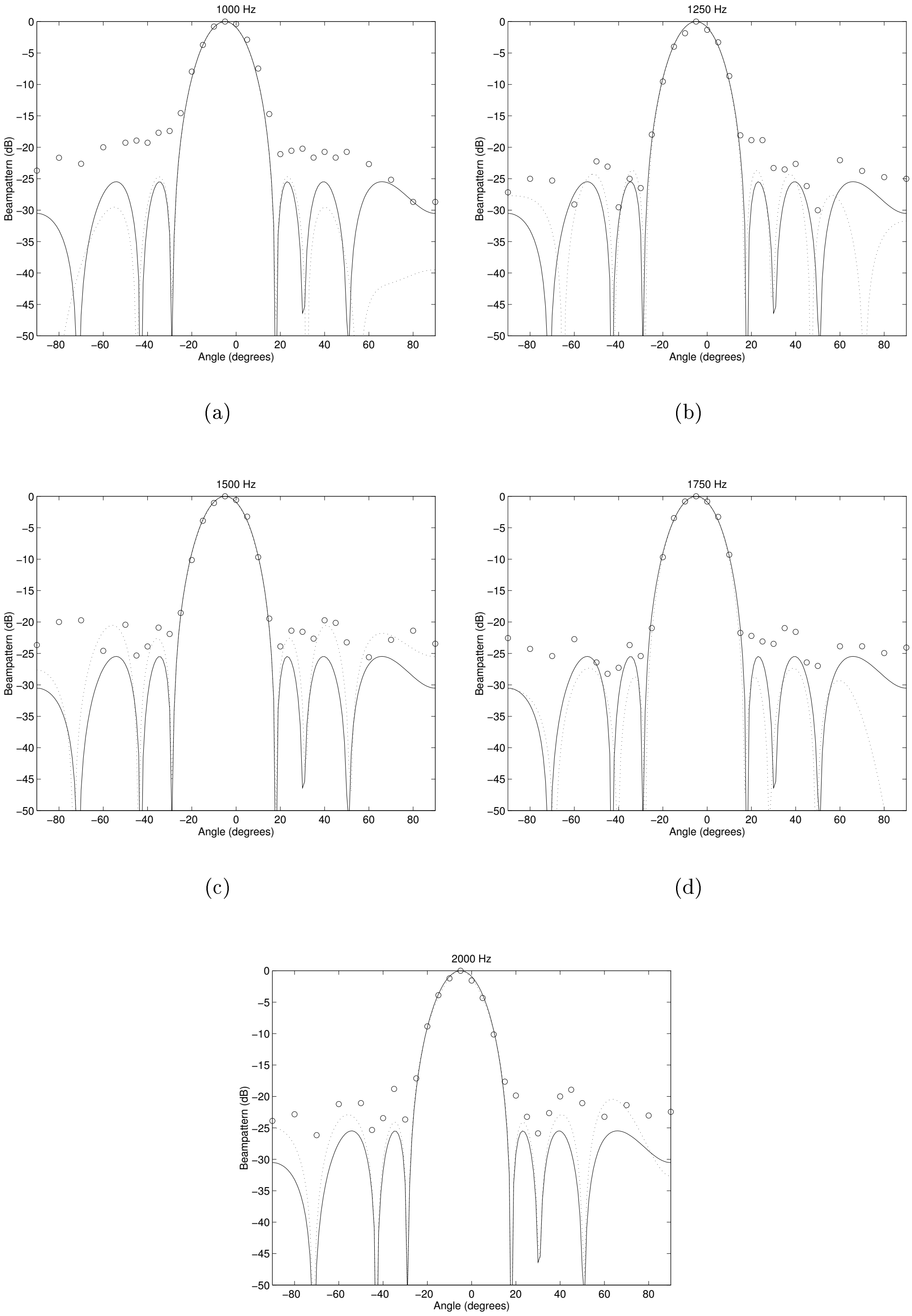


Figure 3.14: Comparison of desired beampattern (solid) and simulated beampatterns (dotted) at five frequencies within the design bandwidth.



(a)

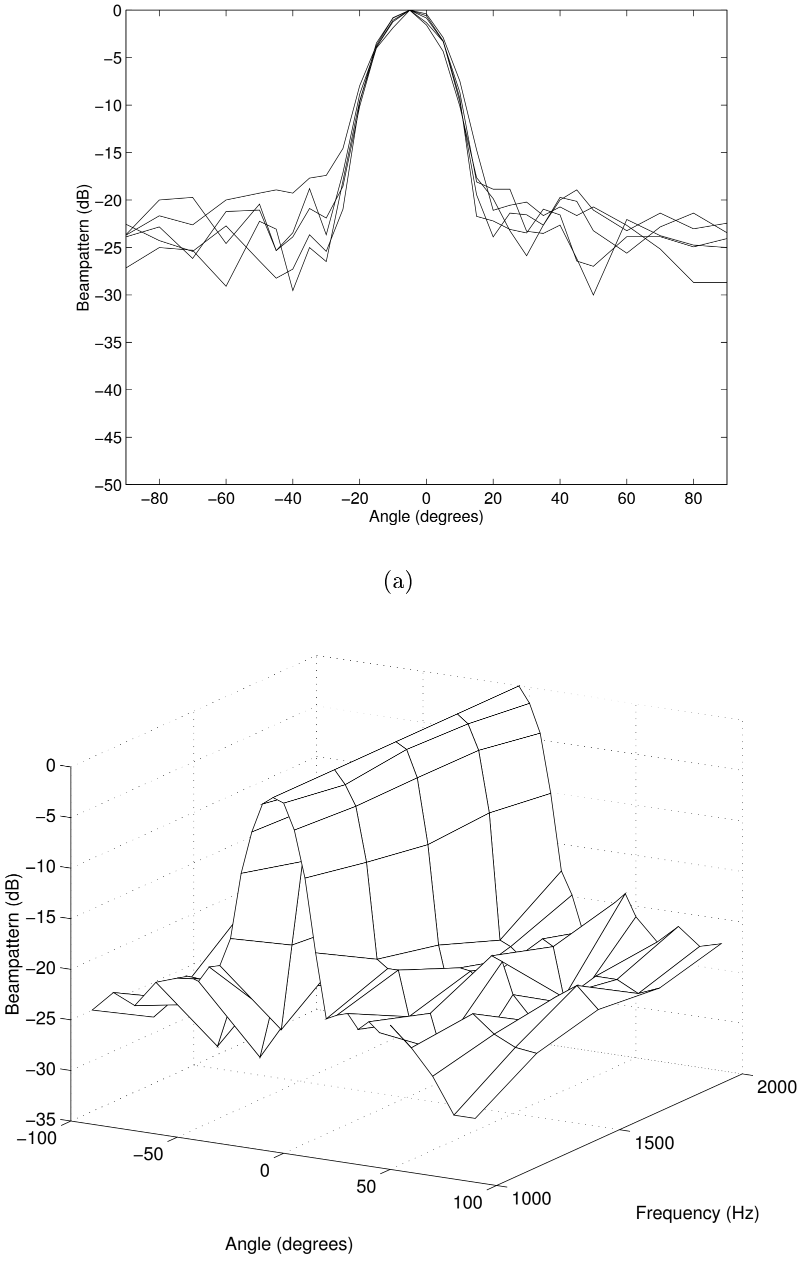
(b)

(c)

(d)

(e)

Figure 3.15: Comparison of results obtained from the experimental microphone array (circles) with the desired (solid) and simulated (dotted) beampatterns.



(a)

(b)

Figure 3.16: Comparison of normalised beampatterns measured at different frequencies: (a) superposition of measured beampatterns, and (b) variation of measured beampattern with frequency.

Chapter 4

Frequency Invariant Beamforming with Broadband Pattern Nulls

4.1 Introduction

THE frequency invariant beamforming (FIB) theory of Chapter 2 was based on approximating a theoretical continuous sensor, having a frequency invariant response, by a finite array of discrete sensors. Because of this approximation, the beampattern of a FIB will exhibit a small amount of frequency variation especially in the sidelobes and pattern nulls. To produce a broadband pattern null whose position is frequency invariant over a wide bandwidth requires additional constraints on the FIB. In this chapter we consider how these constraints may be imposed.

The problem of designing narrowband beamformers with controlled nulls has been extensively studied in the literature. In general the requirement is to place a pattern null in the direction of a strong interferer with the aim of reducing the noise due to the interference source. Several methods of controlling the positions of the nulls have been employed, including modification of the sensor amplitude and phase [12, 86], modification of the sensor phase only [86, 89], and sensor position perturbation [49, 68, 93].

In phased arrays, where the beamformer weights are fixed as a function of frequency, broadband pattern nulling is effected by imposing a null over a wide spatial region centred on the required broadband null direction. Because the beampattern scales directly with frequency, a null trough of width Δu centred at u_0 (where $u = \sin\theta$) will provide suppression over a bandwidth of

$$\frac{\Delta f}{f_0} = \frac{\Delta u}{u_0},$$

where f_0 is the centre frequency of the band. Methods of producing a null trough over a wide angular region include imposing multiple pattern nulls in the vicinity of u_0 [62, 84, 85], imposing derivative constraints at u_0 [54, 84, 91], and constraining the average power over the angular region Δu [25, 38].

In this chapter we instead consider the problem of producing a null in a single direction, and forcing the null to cover some desired bandwidth. We consider a broadband beamformer with N sensors, each containing an FIR filter with L coefficients. Thus there is a maximum of NL free parameters which may be used to impose the broadband null. In general we will assume that we are given some quiescent broadband beampattern and we are required to impose a broadband null in a given direction. Hence, we will use $M < NL$ parameters to impose the broadband null, and the remaining free parameters to approximate the original beampattern. Primarily we are concerned with the case in which the quiescent broadband beampattern is frequency invariant, although the methods we outline are suitable for more general broadband beampatterns.

We will consider a number of formulations of the problem and present some new results concerning both approximate and exact broadband null placement. In considering an exact broadband null, our aim is to demonstrate that through proper design it is possible to produce a pattern zero which is present over all frequencies. In the approximate null formulation our aim is to determine the number of degrees of freedom which must be used to impose a null of a given depth in a frequency invariant beampattern, regardless of the specific shape of the original frequency invariant beampattern. The style of this chapter is to present a collection of design methods, rather than trying to find a single "right" solution to the broadband pattern nulling problem.

4.2 Problem Statement

Consider a linear array of N omni-directional sensors. Each sensor signal is filtered using an FIR filter with L coefficients, and the filtered signals combined.[1] The spatial response for a plane wave impinging on the array from a direction θ measured relative to broadside is

$$r(u, \omega) = \sum_{n=1}^{N} H_n(\omega) \exp\left[\jmath \omega \tau_n(u)\right], \tag{4.1}$$

[1] Note that in this chapter we are not explicitly considering the FIB structure developed in Chapter 2, viz., a primary filter on each sensor and a common secondary filter. Instead, we only consider a structure with a filter on each sensor. For the case of a FIB, the spatial weighting terms and the secondary filter are implicitly included in each sensor filter.

where

$$H_n(\omega) = \sum_{k=0}^{L-1} h_n[k] \exp\left(-\jmath\omega k\right) \tag{4.2}$$

is the frequency response of the nth sensor filter, $\omega = 2\pi f/f_s$ is the discrete-time frequency variable with a sampling rate of f_s, $\tau_n(u) = f_s c^{-1}(x_n - x_1)u$ is the relative propagation delay to the nth sensor with a propagation speed of c, x_n is the location of the nth sensor, and $u = \sin\theta$.

The response may be written in vector form as

$$r(u,\omega) = \mathbf{h}^H\,\mathbf{d}(u,\omega),$$

where

$$\mathbf{h} = \Big[h_1[0]\dots h_N[0]\dots h_1[L-1]\dots h_N[L-1]\Big]^H$$

is the NL vector of complex FIR filter coefficients,

$$\mathbf{d}(u,\omega) = \mathbf{e}(\omega) \otimes \mathbf{a}(u,\omega),$$

is the NL delay vector, $\otimes$ denotes the Kronecker product,

$$\mathbf{e}(\omega) = \Big[1, e^{-\jmath\omega}, \dots, e^{-\jmath\omega(L-1)}\Big]^T$$

is the L dimensional Fourier transform vector, and

$$\mathbf{a}(u,\omega) = \Big[e^{\jmath\omega\tau_1(u)}, \dots, e^{\jmath\omega\tau_N(u)}\Big]^T$$

is the N dimensional array response vector.

The problem considered here may be stated in general terms as follows. Given a set of coefficients $\mathbf{h}$ which produces some desired broadband response $r(u,\omega)$, find the coefficients $\widehat{\mathbf{h}}$ which produce a broadband response $\hat{r}(u,\omega)$ which has a broadband null in a specified direction and is close in some respect to the quiescent response $r(u,\omega)$.

4.3 Problem Formulation

There are several ways in which this problem may be formulated. Assume the coefficients $\widehat{\mathbf{h}}$ are formed by adding a set of perturbing coefficients to $\mathbf{h}$, i.e., $\widehat{\mathbf{h}} = \mathbf{h} + \mathbf{b}$. The resulting response $\hat{r}(u,\omega)$ is equivalent to the quiescent response $r(u,\omega)$ plus the response of a *nulling beamformer* whose coefficients are $\mathbf{b}$, i.e.,

$$\begin{aligned}\hat{r}(u,\omega) &= \widehat{\mathbf{h}}^H\mathbf{d}(u,\omega)\\ &= \mathbf{h}^H\mathbf{d}(u,\omega) + \mathbf{b}^H\mathbf{d}(u,\omega)\\ &= r(u,\omega) + \mathbf{b}^H\mathbf{d}(u,\omega).\end{aligned}$$

The general broadband nulling problem can be formulated in general terms as

$$\min_{\mathbf{b}} \; J(\mathbf{b}) \tag{4.3a}$$

$$\text{subject to } \; |\hat{r}(u_0,\omega)| \le \epsilon, \quad \forall\omega \in [\omega_L, \omega_U], \tag{4.3b}$$

where $J(\mathbf{b})$ is some suitably defined cost functional which measures the distance between $r(u,\omega)$ and $\hat{r}(u,\omega)$, ϵ is the desired null depth in the nulling direction u_0, and $[\omega_L, \omega_U]$ is the bandwidth of interest (i.e., the bandwidth of the source and interfering signals).

4.3.1 Cost Functionals

Several candidate cost functionals will now be considered.

I. L_∞ Error

Minimise the maximum distance between the quiescent response and the perturbed response over a spatial sector $\mathcal{U}$ and frequency sector Ω:

$$\begin{aligned}J_{L_\infty}(\mathbf{b}) &= \max_{u\in\mathcal{U},\omega\in\Omega} |\hat{r}(u,\omega) - r(u,\omega)|\\ &= \max_{u\in\mathcal{U},\omega\in\Omega} \left|\mathbf{b}^H\mathbf{d}(u,\omega)\right|.\end{aligned}$$

II. Weighted L_2 Error

Minimise the weighted least square distance between the quiescent response and the perturbed response over a spatial sector $\mathcal{U}$ and frequency sector Ω:

$$\begin{aligned}J_{L_2}(\mathbf{b}) &= \int_\Omega \int_\mathcal{U} \varphi(u,\omega)\, |\hat{r}(u,\omega) - r(u,\omega)|^2 \; du\, d\omega\\ &= \mathbf{b}^H\mathbf{Q}\mathbf{b},\end{aligned}$$

where

$$\mathbf{Q} = \int\limits_{\Omega} \int\limits_{\mathcal{U}} \varphi(u,\omega)\mathbf{d}(u,\omega)\mathbf{d}(u,\omega)^H \; du\, d\omega,$$

and $\varphi(u,\omega)$ is a weighting function.

The calculation of $\mathbf{Q}$ deserves some comment. The (a,b)th element of $\mathbf{d}(u,\omega)\mathbf{d}(u,\omega)^H$ is

$$\mathbf{q}_{a,b} = e^{\jmath\omega(\tau_n - \tau_m)u} e^{-\jmath\omega(k-l)},$$

where $a = (k-1)+n$ and $b = (l-1)+m$.[2] Without loss of generality let $\varphi(u,\omega) = 1$. It can be shown that

$$\begin{aligned}\mathbf{Q}_{a,b} &= \int_{\omega_1}^{\omega_2} \int_{u_1}^{u_2} \mathbf{q}_{a,b} \; du\, d\omega \\ &= \frac{\jmath}{t}\Big\{ \mathrm{Ei}[-\jmath\omega_2(tu_2 - \kappa)] - \mathrm{Ei}[-\jmath\omega_1(tu_2 - \kappa)] \\ &\quad -\mathrm{Ei}[-\jmath\omega_2(tu_1 - \kappa)] + \mathrm{Ei}[-\jmath\omega_1(tu_1 - k)]\Big\},\end{aligned}$$

where $t = (\tau_n - \tau_m)$, $\kappa = (k-l)$, and

$$\mathrm{Ei}(x) = \int_x^{\infty} \frac{e^{-t}}{t} \; dt$$

is the exponential integral. For the specific case of $\omega_1 = -\pi$, $\omega_2 = \pi$, $u_1 = -1$, $u_2 = 1$, this reduces to

$$\mathbf{Q}_{a,b} = \frac{2}{t}\Big\{ \mathrm{Si}[\pi(t-\kappa)] + \mathrm{Si}[\pi(t+\kappa)] \Big\},$$

where

$$\mathrm{Si}(x) = \int_0^{x} \frac{\sin t}{t} \; dt$$

is the sine integral.

III. Nulling Gain

Minimise the gain of the nulling beamformer, i.e., the ℓ_2 norm between the quiescent coefficients and the perturbed coefficients:

$$\begin{aligned}J_{\mathrm{NG}}(\mathbf{b}) &= \|\widehat{\mathbf{h}} - \mathbf{h}\|^2 \\ &= \|\mathbf{b}\|^2\end{aligned}$$

[2] This rather obscure indexing is used for notational simplicity.

where $\|\cdot\|$ denotes the vector 2-norm.

IV. Nulling Power

Minimise the output power of the nulling beamformer:

$$\begin{aligned} J_{\mathrm{NP}}(\mathbf{b}) &= E\{\mathbf{b}^H \mathbf{s}(t)\mathbf{s}(t)^H \mathbf{b}\} \\ &= \mathbf{b}^H \mathbf{R}_s \mathbf{b}, \end{aligned}$$

where $\mathbf{s}(t)$ is the vector of received signals at the sensor array at time t, and $\mathbf{R}_s$ is the received signal covariance matrix. Clearly, for white signals $J_{\mathrm{NP}}(\mathbf{b})$ reduces to $J_{\mathrm{NG}}(\mathbf{b})$.

The similarity between $J_{\mathrm{NP}}(\mathbf{b})$ and the cost functional considered by Frost [32] should be noted. Frost considered minimising the array output power subject to a linear constraint on the broadband array response at broadside. We consider minimising the output power of the *nulling* beamformer subject to a constraint on the *combined* broadband array response. We could therefore use a modified version of Frost's adaptive algorithm to control the nulling weights $\mathbf{b}$.

It is also instructive to note the similarity of $J_{\mathrm{NP}}(\mathbf{b})$ with the multiple sidelobe canceller (MSC) of Applebaum [2]. The MSC consists of a main channel (which may be a single high gain directive antenna or a fixed beamformer) and several auxiliary channels. Interfering signals are assumed to be in the sidelobes of the main channel, and are received by the auxiliary channels. The auxiliary channels are linearly combined via a weighting vector, and this combined output is subtracted from the main channel output with the goal of choosing the weighting matrix to cancel the main channel interferences. In our case, the main channel is the quiescent beamformer $r(u, \omega)$ and the auxiliary channels correspond to the nulling beamformer, which in our case must also be constrained to impose the broadband null.

Of these cost functionals, $J_{\mathrm{NG}}(\mathbf{b})$ is by far the easiest to compute, and it will be used exclusively in the remainder of this chapter.

4.3.2 Constraints

Having determined a practical cost functional, the practicality of the L_∞ constraint (4.3b) must be assessed. Using any of the quadratic cost functions results in a mixed L_2-L_∞ problem, i.e., minimise the L_2 error subject to an L_∞ constraint. This is a problem which exists in the control literature [64]. We have instead chosen to focus on the following more tractable constraints.

I. Mean-Square Null Depth

Consider the following problem:

$$\min_{\mathbf{b}} \mathbf{b}^H\mathbf{b} \tag{4.4a}$$

$$\text{subject to } \textstyle\int_{\omega_L}^{\omega_U} |\hat{r}(u_0,\omega)|^2\,d\omega \leq \epsilon. \tag{4.4b}$$

The constraint in this case corresponds to restricting the average power over the design band in the null direction to be less than some desired value. Er [25] considered a similar constraint for synthesising a pattern null which covers a wide spatial region in a phased array. A similar constraint has also been considered by Guella and Davis [38]. However, they formulated the problem as an unconstrained minimisation problem by including the constraint as part of the cost function, i.e.,

$$\min_{\mathbf{b}} \quad \mathbf{b}^H\mathbf{b} + \alpha \int_{\omega_L}^{\omega_U} |\hat{r}(u_0,\omega)|^2\,d\omega,$$

where α is a weighting constant which is used to trade off null depth for beampattern degradation.

The constraint (4.4b) can be rewritten as

$$(\mathbf{h}+\mathbf{b})^H\mathbf{Q}(\mathbf{h}+\mathbf{b}) \leq \epsilon, \tag{4.5}$$

where

$$\mathbf{Q} = \int_{\omega_L}^{\omega_U} \mathbf{d}(u_0,\omega)\mathbf{d}(u_0,\omega)^H\,d\omega.$$

Using the Lagrange multiplier method, define an augmented cost function as

$$J' = \mathbf{b}^H\mathbf{b} + \alpha\left[(\mathbf{h}+\mathbf{b})^H\mathbf{Q}(\mathbf{h}+\mathbf{b}) - \epsilon\right],$$

where $\alpha \geq 0$ is the Lagrange multiplier for the inequality constraint (4.5). Taking the gradient of J' with respect to $\mathbf{b}$ gives

$$\nabla_{\mathbf{b}} J' = 2\mathbf{b} + 2\alpha\mathbf{Q}(\mathbf{h}+\mathbf{b}).$$

Equating $\nabla_{\mathbf{b}} J'$ to zero gives

$$\mathbf{b} = -\alpha \frac{\mathbf{Q}}{\mathbf{I} + \alpha \mathbf{Q}} \mathbf{h}. \tag{4.6}$$

The Lagrange multiplier is given by the solution of

$$\left([\mathbf{I} + \alpha \mathbf{Q}]^{-1} \mathbf{h} \right)^H \mathbf{Q} \left([\mathbf{I} + \alpha \mathbf{Q}]^{-1} \mathbf{h} \right) = \epsilon$$

A root finding method can be used to solve for α and the result substituted into (4.6), or α can be incremented from 0 until the constraint (4.5) is satisfied.

An alternative to this Lagrange multiplier solution can be formulated as follows. Since $\mathbf{Q}$ is Hermitian, it can be factored as

$$\mathbf{Q} = \mathbf{\Gamma} \mathbf{\Lambda} \mathbf{\Gamma}^H,$$

where

$$\mathbf{\Lambda} = \text{diag}[\lambda_1, \ldots, \lambda_{NL}],$$

$\lambda_1 \geq \ldots \geq \lambda_{NL} \geq 0$ are the ordered eigenvalues of $\mathbf{Q}$, and

$$\mathbf{\Gamma} = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_{NL}]$$

is an $NL \times NL$ unitary matrix where $\mathbf{e}_i, i = 1, \ldots, NL$ are the corresponding eigenvectors. The constraint (4.5) is equivalent to imposing n_0 eigenvector constraints of the form

$$(\mathbf{h} + \mathbf{b})^H \mathbf{e}_i = 0, \quad i = 1, \ldots, n_0,$$

where n_0 is the smallest integer such that the fractional trace of the $\mathbf{Q}$ matrix, viz.,

$$\%\text{trQ} = \sum_{i=1}^{n_0} \lambda_i \Big/ \sum_{i=1}^{NL} \lambda_i,$$

is greater than or equal to some threshold value. Choosing different values of %trQ is equivalent to specifying different values of ϵ for the inequality constraint (4.5).

The previous formulation is a reasonably straightforward modification of the method presented by Er [25]. Rather than further analysing this method, we have instead chosen to focus our attention on the following alternate problems.

II. Exact Nulling

Consider the following problem:

$$\min_{\mathbf{b}} \mathbf{b}^H\mathbf{b} \tag{4.7a}$$

$$\text{subject to } \hat{r}(u_0,\omega) = 0, \quad \forall\omega. \tag{4.7b}$$

At first the requirement to obtain a zero response in the null direction for all frequencies (i.e., an *exact broadband null*) would appear impossible with a finite number of filter coefficients. However, as shown in §4.4, this corresponds to a constraint on the time-domain impulse responses of the sensor filters. By proper selection of the sensor positions and sampling rate a solution may be found.

III. Multiple Frequency Nulling

An alternative to an exact broadband null is to impose multiple zeros in the frequency domain response of the beamformer at the null angle. In this case the problem is formulated as:

$$\min_{\mathbf{b}} \mathbf{b}^H\mathbf{b} \tag{4.8a}$$

$$\text{subject to } \hat{r}(u_0,\omega) = 0, \quad \omega = \omega_1, \dots, \omega_M. \tag{4.8b}$$

The challenge now is in choosing the set of M frequencies $\{\omega_m\}_{m=1}^M$ at which to impose the frequency domain zeros, such that the broadband null is below a specified depth for all frequencies in the design band. This problem is considered in §4.5.

4.4 Time Domain Nulling

In this section we demonstrate that through proper array design it is possible to produce an exact broadband null, i.e., a pattern zero which is present over all frequencies. However, we show that stringent requirements are imposed on the array geometry and sampling rate to achieve this. In §4.4.4 we evaluate the degradation of an exact null which would occur in a practical setting with sensor location errors.

4.4.1 Exact Broadband Nulling

An exact broadband null in the direction u_0 requires

$$r(u_0, \omega) = \sum_{n=1}^{N} H_n(\omega) e^{\jmath\omega\tau_n(u_0)} = 0, \quad \forall\omega. \tag{4.9}$$

The response in the null direction may be rewritten using (4.2) as

$$r(u_0, \omega) = \sum_{n=1}^{N} \sum_{k=0}^{L-1} h_n[k] e^{-\jmath\omega k} e^{\jmath\omega\tau_n(u_0)}.$$

Taking the inverse Fourier transform of $e^{\jmath\omega\tau_n(u_0)}$, and changing the order of summation gives

$$r(u_0, \omega) = \sum_{k=0}^{L-1} \left(\sum_{n=1}^{N} h_n[k] \star \mathrm{sinc}[k + \tau_n(u_0)] \right) e^{-\jmath\omega k},$$

where $\mathrm{sinc}(x) = \sin(\pi x)/(\pi x)$, and $\star$ denotes convolution. Thus, (4.9) is equivalent to

$$\sum_{n=1}^{N} h_n[k] \star \mathrm{sinc}[k + \tau_n(u_0)] = 0, \quad \forall k. \tag{4.10}$$

It is not immediately clear how this may be easily enforced. However, if we let

$$\tau_n(u_0) \in \mathbb{Z}, \quad \forall n, \tag{4.11}$$

then (4.10) becomes

$$\sum_{n=1}^{N} h_n[k] \star \delta[k + \tau_n(u_0)] = 0, \quad \forall k,$$

where $\delta[\cdot]$ is the Kronecker delta. Equivalently this may be written

$$\sum_{n=1}^{N} h_n[k + \tau_n(u_0)] = 0, \quad \forall k. \tag{4.12}$$

Thus, there are two conditions for an exact null in the direction u_0: (i) the sensors must be located such that the relative propagation delay to the nth sensor is an integer for all n (4.11)—this will be referred to as the *integer delay property*; and (ii) the sum of the delayed filter impulse responses must be exactly zero (4.12).

Note that it is always possible to place a null at $u_0 = 0$ because in this case

$$\tau_n(u_0) = \frac{f_s}{c}(x_n - x_1)u_0 = 0, \ \forall n$$

and (4.12) reduces to requiring

$$\sum_{n=1}^{N} h_n[k] = 0,\ \forall k.$$

Using this idea, the condition for an exact null at broadside may be written as

$$\mathbf{C}_0^H \mathbf{h} = 0$$

where

$$\mathbf{C}_0 = \mathbf{I}_L \otimes \mathbf{1}_N$$

where $\mathbf{I}_L$ is the $L \times L$ identity matrix and $\mathbf{1}_N$ is the N vector of ones.

The null constraint (4.12) is illustrated in Fig. 4.1. Specifically, when $\tau_n(u_0) \in \mathbb{Z}, \forall n$, the sum of the delayed filter impulse responses can be made exactly zero. In this case, if $\mathbf{h}$ is a vector of filter coefficients which produces some quiescent response, it is always possible to find a set of coefficients $\mathbf{b}_0$ which satisfy:

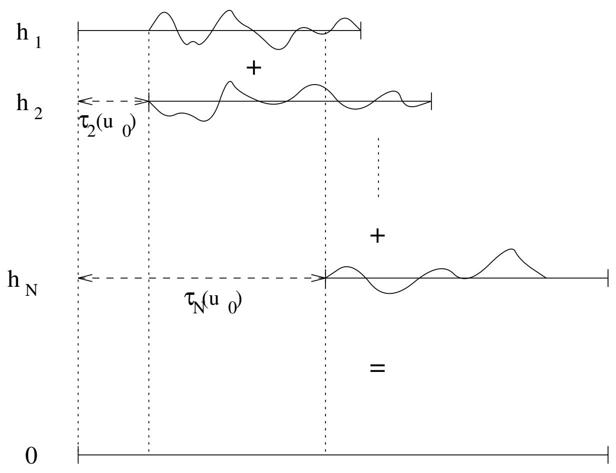


Figure 4.1: Time domain nulling constraint.

$$b_0[k] + \sum_{n=1}^{N} h_n[k + \tau_n(u_0)] = 0, \quad \forall k.$$

The $\mathbf{b}_0$ are referred to as the *nulling filter coefficients*. Obviously, one could simply set $b_0[k] = -\sum_{n=1}^{N} h_n[k + \tau_n(u_0)]$, and add this to the quiescent primary filter coefficients on the first sensor. This corresponds to putting all the nulling effort into the first sensor filter. In the next section we will consider how to best spread the $\mathbf{b}_0$ among the N sensors so that the quiescent pattern suffers the least degradation relative to $J_{\text{NG}}(\mathbf{b})$.

For any array satisfying the integer delay property, the constraint (4.12) can be written as

$$\sum_{n=1}^{N} \tilde{h}_n[k] = 0,$$

where $\tilde{h}_n[k] = h_n[k + \tau_n(u_0)]$. The $\tilde{h}_n[k]$ are effectively the filter coefficients which steer the null direction to broadside, i.e., if the coefficients $\tilde{h}_n[k]$ were used as the beamformer filter coefficients the resulting response would be steered to $-u_0$ and the null would appear at broadside. This null steering is only exact for $\tau_n(u_0) \in \mathbb{Z}, \forall n$. The null constraint can now be written as

$$\mathbf{C}_0^H \tilde{\mathbf{h}} = 0.$$

Define a linear transformation matrix $\mathbf{T}_u$ which satisfies $\tilde{\mathbf{h}} = \mathbf{T}_u \mathbf{h}$. The null constraint now becomes

$$\mathbf{C}_u^H \mathbf{h} = 0, \tag{4.13}$$

where $\mathbf{C}_u = \mathbf{T}_u^H \mathbf{C}_0$ is the transformed constraint matrix.

At this point it is instructive to consider what the integer delay property implies for a practical design. The integer delay property (4.11) requires

$$\tau_n(u_0) = f_s c^{-1}(x_n - x_1)u_0 \in \mathbb{Z}, \quad \forall n.$$

Let $x_1 = 0$ and $x_n > 0, n > 1$. The first constraint on the array geometry is

$$\frac{x_n}{x_2} \in \mathbb{Z}, \quad n > 2, \tag{4.14}$$

regardless of the desired null angle. Second, a null at u_0 requires

$$x_2 = \frac{mc}{f_s u_0}, \quad m \in \mathbb{Z}. \tag{4.15}$$

Assume that the sensors are positioned as described in Chapter 2, i.e., the sensor spacings are logarithmically increasing, and the part of the array closest to the origin is used only at the highest frequency, with more and more elements becoming active at lower frequencies. Assuming we want to use the minimum number of sensors to avoid spatial aliasing, then

$$x_2 = \frac{c}{2f_U},$$

where f_U is the upper frequency of interest. The directions in which we may form a broadband null are then

$$u_0 = \frac{m 2 f_U}{f_s}, \quad m \in \mathbb{Z}.$$

Clearly, for a minimum sampling rate of $f_s = 2f_U$ it is only possible to produce an exact broadband null at $u_0 \in \{-1, 0, 1\}$, or equivalently $\theta_0 \in \{-\pi/2, 0, \pi/2\}$. Hence, the minimum sampling rate required for a null at some angle θ_0 is

$$f_s = \frac{2f_U}{\sin\theta_0}. \tag{4.16}$$

This demonstrates the major disadvantages of time domain nulling: oversampling is required to produce a null which is away from broadside or endfire, and the sensor locations must be quantised to multiples of the first sensor spacing. However, time domain nulling does allow an exact broadband null to be formed in situations where the required null direction is known and the sensors can be accurately placed.

4.4.2 Producing an Exact Null in a Quiescent Pattern

Having demonstrated that it is possible to form an exact broadband null, we now return to the original problem of producing a broadband null in a quiescent broadband response while perturbing that pattern the least with regard to $J_{\mathrm{NG}}(\mathbf{b})$. Specifically, if $r(u, \omega) = \mathbf{h}^H \mathbf{d}(u, \omega)$ is the quiescent beamformer response, find the coefficients $\mathbf{b}$ such that

$$\hat{r}(u, \omega) = (\mathbf{h}^H + \mathbf{b}^H)\mathbf{d}(u, \omega) \tag{4.17}$$

is close to $r(u, \omega)$ in some respect, and $\hat{r}(u_0, \omega) = 0, \forall \omega$.

The under-determined constraints to place an exact null can be written as

$$\sum_{n=1}^{N} h_n[k + \tau_n(u_0)] + b_n[k + \tau_n(u_0)] = 0, \quad \forall k,$$

or, using (4.13), as

$$\mathbf{C}_u^H (\mathbf{h} + \mathbf{b}) = 0.$$

The optimum $\mathbf{b}$ is then found as the solution to the constrained optimisation problem:

$$\min_{\mathbf{b}} \mathbf{b}^H \mathbf{b} \tag{4.18a}$$

$$\text{subject to } \mathbf{C}_u^H (\mathbf{h} + \mathbf{b}) = 0. \tag{4.18b}$$

The solution to this constrained optimisation problem may be found using Lagrange multipliers as follows. Define an unconstrained cost function

$$H(\mathbf{b}) = \frac{1}{2}\mathbf{b}^H\mathbf{b} + \alpha\mathbf{C}_u^H(\mathbf{h}+\mathbf{b}),$$

where α is the Lagrange multiplier for the constraint (4.18b). The minimum is found by setting the derivative of H to zero, giving

$$\mathbf{b}^H = -\alpha\mathbf{C}_u^H.$$

The Lagrange multiplier must also satisfy the constraint (4.18b), giving

$$\alpha = \mathbf{h}^H\mathbf{C}_u\left[\mathbf{C}_u^H\mathbf{C}_u\right]^{-1}.$$

Hence, the optimum weights are

$$\mathbf{b}_{\text{opt}} = -\mathbf{C}_u\left[\mathbf{C}_u^H\mathbf{C}_u\right]^{-1}\mathbf{C}_u^H\mathbf{h}. \tag{4.19}$$

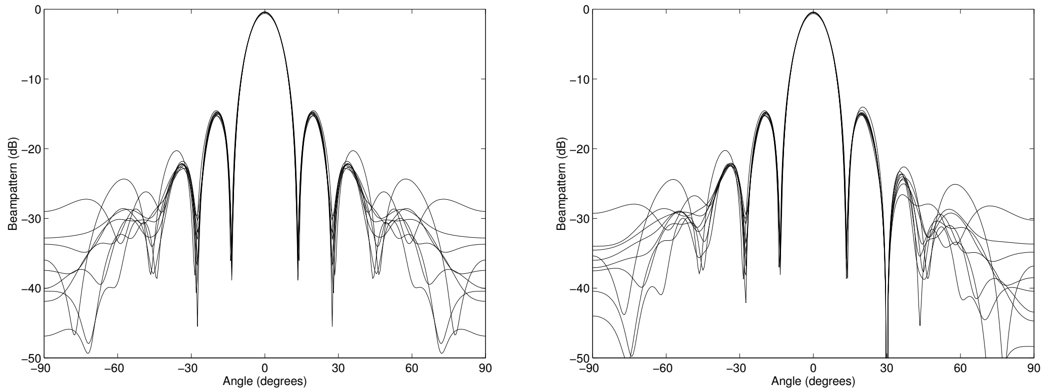
4.4.3 Design Example

To illustrate the time domain nulling method, consider the quiescent frequency invariant beampattern shown in Fig. 4.2(a), calculated at 9 frequencies within the design band. This was designed using the single rate method of Chapter 3 for an array with 18 sensors and 35 filter taps per sensor to cover the octave band 1–2 kHz, with a sampling rate of 8 kHz. The total array size was 1.53 metres (for sound waves propagating in air). Nulling coefficients were designed using (4.19) to produce an exact null at $\theta_0 = 30°$. Figure 4.2(b) shows the resulting beampattern, calculated at 9 frequencies within the design band. Note that the beampattern exhibits an exact null at 30°, and the remainder of the pattern approximates the quiescent beampattern.

4.4.4 Sensor Position Errors

Given that the optimum nulling coefficients have been determined to produce an exact null in a given direction, we now consider the expected null depth taking into account sensor positioning errors. We follow the standard method in the array literature [82].

Assume the actual position of the nth sensor is $\hat{x}_n = x_n + \chi_n$, where x_n is the ideal sensor location and the χ_n are independent zero-mean Gaussian random variables with variance σ_x^2.



(a)

(b)

Figure 4.2: (a) Quiescent beampattern at 9 frequencies within the design band, and (b) beampattern with an exact broadband null at 30°.

The actual broadband response is

$$\hat{r}_a(u,\omega) = \sum_{n=1}^{N} \widehat{H}_n(\omega) e^{\jmath\omega\widehat{\tau}_n(u)} \tag{4.20}$$

where

$$\widehat{H}_n(\omega) = \sum_{k=0}^{L-1} \left(h_n[k] + b_n[k]\right) e^{-\jmath\omega k},$$

with the $b_n[k]$ calculated from (4.19) assuming ideal sensor locations. The actual propagation delay to the nth sensor is

$$\widehat{\tau}_n(u) = f_s c^{-1}\left(x_n + \chi_n\right) u = \tau_n(u) + \Delta_n(u). \tag{4.21}$$

The actual broadband beampattern is

$$|\hat{r}_a(u,\omega)|^2 = \sum_{n=1}^{N}\sum_{m=1}^{N} \widehat{H}_n(\omega)\widehat{H}_m^*(\omega) e^{\jmath\omega[\tau_n(u)-\tau_m(u)]} e^{\jmath\omega[\Delta_n(u)-\Delta_m(u)]}.$$

This can be separated into two parts as

$$|\hat{r}_a(u,\omega)|^2 = \sum_{\substack{n=1 \\ n\neq m}}^{N}\sum_{m=1}^{N} \widehat{H}_n(\omega)\widehat{H}_m^*(\omega) e^{\jmath\omega[\tau_n(u)-\tau_m(u)]} e^{\jmath\omega[\Delta_n(u)-\Delta_m(u)]} + \sum_{n=1}^{N} \left|\widehat{H}_n(\omega)\right|^2.$$

The expected value of the beampattern is

$$E\{|\hat{r}_a(u,\omega)|^2\} = \sum_{n=1}^{N}\sum_{\substack{m=1\\ n\neq m}}^{N} \widehat{H}_n(\omega)\widehat{H}_m^*(\omega)e^{\jmath\omega[\tau_n(u)-\tau_m(u)]}\,|E_\Delta(u,\omega)|^2 + \sum_{n=1}^{N}\left|\widehat{H}_n(\omega)\right|^2 \tag{4.22}$$

where

$$E_\Delta(u,\omega) = E\{e^{\jmath\omega f_s c^{-1}\chi u}\}. \tag{4.23}$$

This is just the characteristic function of the Gaussian random variable χ, which is [17, p.138]

$$E_\Delta(u,\omega) = e^{-\sigma_x{}^2(\omega f_s c^{-1}u)^2/2}. \tag{4.24}$$

Recall that the ideal beampattern is

$$|\hat{r}(u,\omega)|^2 = \sum_{n=1}^{N}\sum_{\substack{m=1\\ n\neq m}}^{N} \widehat{H}_n(\omega)\widehat{H}_m^*(\omega)e^{\jmath\omega[\tau_n(u)-\tau_m(u)]} + \sum_{n=1}^{N}\left|\widehat{H}_n(\omega)\right|^2.$$

Hence, the actual beampattern can be expressed as

$$E\{|\hat{r}_a(u,\omega)|^2\} = |\hat{r}(u,\omega)|^2\,|E_\Delta(u,\omega)|^2 + \sum_{n=1}^{N}\left|\widehat{H}_n(\omega)\right|^2\left(1-|E_\Delta(u,\omega)|^2\right).$$

Assuming that the ideal pattern has an exact null at u_0, then $\hat{r}(u_0,\omega) = 0, \forall\omega$, and the expected null direction response is

$$E\{|\hat{r}_a(u_0,\omega)|^2\} = \sum_{n=1}^{N}\left|\widehat{H}_n(\omega)\right|^2\left(1-\left|e^{-\sigma_x{}^2(\omega f_s c^{-1}u_0)^2/2}\right|^2\right). \tag{4.25}$$

An example is now presented to illustrate the result obtained above. The design is the same as that used in the previous section. Figure 4.3 shows the actual null depth compared with the expected null depth calculated from (4.25) for 5 different trials with $\sigma_x = 1$ millimetre (mm). It is clear that the actual null depth is close to the expected null depth in all cases.

The expected null depth for several different positioning errors is shown in Fig. 4.4. As expected, the depth of the null decreases as the sensor position errors increase.

Finally, the expected null depth for several different null directions is shown in Fig. 4.5, with a fixed positioning error of $\sigma_x = 1$ mm. The bottom line is for $u_0 = 0.1$ and the top

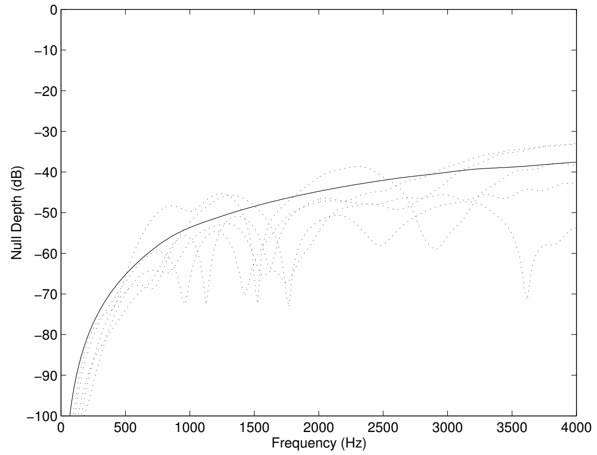


Figure 4.3: Actual null depth (dotted) for five trails, and expected null depth (solid) versus frequency. The standard deviation of the sensor positioning errors was $\sigma_x = 1$ mm.

is for $u_0 = 1$, (i.e., endfire) with u_0 increasing in steps of 0.1. As expected, positioning errors have the most pronounced effect on the exact null for null angles close to endfire.

4.5 Frequency Domain Nulling

In this section we consider a, perhaps more practical, alternative to exact broadband nulling. Specifically, we consider the problem of placing multiple zeros in the null direction frequency response such that a given null depth is achieved over the design band while minimising the disturbance to the quiescent broadband response. There are a number of ways in which this problem may be formulated. However, we will only consider one formulation of the problem which results in a simple analytical solution.

Let $r(u, \omega) = \mathbf{h}^H \mathbf{d}(u, \omega)$ be the quiescent broadband response. The problem may be stated as

$$\min_{\mathbf{b}} \mathbf{b}^H \mathbf{b} \tag{4.26a}$$

$$\text{subject to} \quad \mathbf{d}(u_0, \omega_m)^H (\mathbf{h} + \mathbf{b}) = 0, \quad m = 1, \ldots, M. \tag{4.26b}$$

This problem is identical to (4.18) with $\mathbf{C}_u = [\mathbf{d}(u_0, \omega_1), \ldots, \mathbf{d}(u_0, \omega_M)]$. The solution is again given by (4.19).

It is now necessary to determine the zero locations $\omega_m, m = 1, \ldots, M$ in order to achieve the required null depth. As a simplification we will assume the M zeros are equally spaced

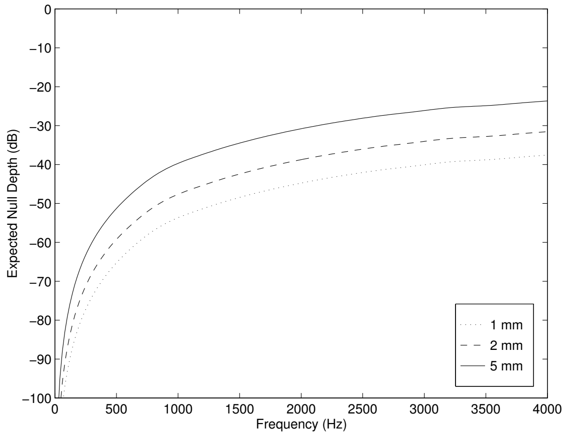


Figure 4.4: Expected null depth for several different values of σ_x.

within the design frequency band, so the problem now is only to choose the number of zeros.

4.5.1 Multiple Zeros in Null Direction Frequency Response

Consider the problem of determining the number M of equally spaced zeros to impose in the null direction frequency response, in order to achieve a given null depth within the design frequency band $\Omega = [\omega_L, \omega_U]$. An analogous problem was considered by Steyskal [85] in which he determined the number of constraints required for a phased array to achieve a specified null depth over a spatial sector. Steyskal's method can be modified as follows.

The optimisation problem (4.26) may be rewritten as

$$\min_{\widehat{\mathbf{h}}} \, (\widehat{\mathbf{h}} - \mathbf{h})^H (\widehat{\mathbf{h}} - \mathbf{h}) \tag{4.27a}$$

$$\text{subject to } \mathbf{d}_m^H \widehat{\mathbf{h}} = 0, \quad m = 1, \dots, M, \tag{4.27b}$$

where $\widehat{\mathbf{h}} = (\mathbf{h} + \mathbf{b})$ and $\mathbf{d}_m = \mathbf{d}(u_0, \omega_m)$. Hence $\widehat{\mathbf{h}}$ is orthogonal to the constraint vectors $\mathbf{d}_m$.

In [84] Steyskal shows that the weights which solve the constrained optimisation are given by

$$\widehat{\mathbf{h}} = \mathbf{h} - \sum_{m=1}^{M} \alpha_m \mathbf{d}_m.$$

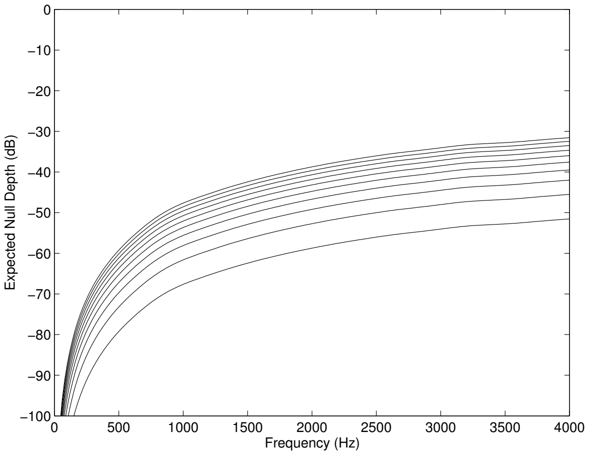


Figure 4.5: Expected null depth for several different null directions with $\sigma_x = 1$ mm. The bottom line is for $u_0 = 0.1$ and the top line is for $u_0 = 1$, with u_0 increasing in steps of 0.1.

Equivalently, the optimum response is given by

$$\hat{r}(u,\omega) = r(u,\omega) - \sum_{m=1}^{M} \alpha_m q_m(u,\omega), \tag{4.28}$$

where

$$q_m(u,\omega) = \mathbf{d}_m^H \mathbf{d}(u,\omega). \tag{4.29}$$

The parameters α_m are obtained by solving the set of M simultaneous equations

$$\begin{bmatrix} r(u_0,\omega_1) \\ \vdots \\ r(u_0,\omega_M) \end{bmatrix} = \begin{bmatrix} q_1(u_0,\omega_1) & \cdots & q_M(u_0,\omega_1) \\ \vdots & & \vdots \\ q_1(u_0,\omega_M) & \cdots & q_M(u_0,\omega_M) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_M \end{bmatrix}. \tag{4.30}$$

Define the null depth as

$$\epsilon = \max_{\omega \in \Omega} \hat{r}(u_0,\omega)^2, \tag{4.31}$$

where $\hat{r}(u,\omega)$ is given by (4.28).

Thus, given a quiescent broadband beampattern (or equivalently, the quiescent coefficients $\mathbf{h}$), the null depth achieved for a given M can be calculated. Note that the null depth is implicitly dependent on the array geometry, number of coefficients, sampling rate, design bandwidth, and null direction.

4.5.2 FIB with Multiple Frequency Nulls

In this section our aim is to determine M for the specific case where the quiescent beamformer is a FIB. In particular, our goal is to determine an approximate relation for the null depth which is valid for any FI beampattern. This means that if the quiescent FIB coefficients are found by adaptation, we can determine *a priori* the number of frequency zero constraints required to produce a given null depth.

Consider the following assumptions.

1. The M nulls are equally spaced over the region Ω.
2. The quiescent FIB response $r(u, \omega)$ is real.
3. The quiescent FIB response in the nulling direction is approximately constant over the nulling frequency band, i.e., $r(u_0, \omega) \approx A, \omega \in \Omega$, where $A = r(u_0, \omega_L)$.
4. $\hat{r}(u_0, \omega)$ is oscillatory and crosses the ω axis at $\omega = \omega_1, \dots, \omega_M$.
5. $\hat{r}(u_0, \omega)$ attains its maximum value at $\omega_{\max} = (\omega_1 + \omega_2)/2$.

With these assumptions, the null depth can be approximated as

$$\epsilon \approx \hat{r}\left(u_0, \frac{\omega_1 + \omega_2}{2}\right)^2, \tag{4.32}$$

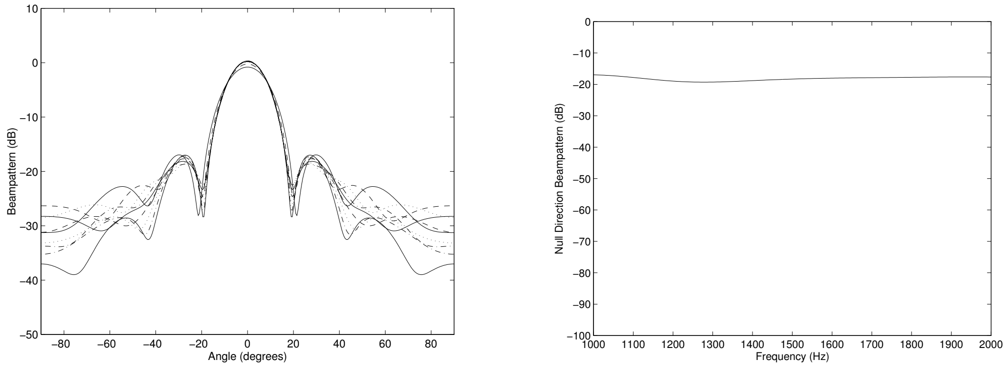
where

$$\hat{r}(u_0, \omega) = A - \sum_{m=1}^{M} \alpha_m q_m(u_0, \omega), \tag{4.33}$$

with $q_m(u_0, \omega)$ given by (4.29), and α_m given by the solution of (4.30) with $r(u_0, \omega_m) = A, m = 1, \dots, M$. Note that (4.32) does not depend on the quiescent coefficients. Hence, given the array geometry, number of coefficients, sampling rate, design bandwidth, and null direction, it is possible to determine the expected null depth for any FIB, regardless of the actual quiescent FIB beampattern. In practice, for a given beamformer all of these variables will be fixed, and one can *a priori* determine the number of frequency constraints required to achieve a given null depth in a given null direction. Then, even if the FIB filter coefficients are determined from an adaptive algorithm (such as the linear constrained minimum variance algorithm proposed in Chapter 3) the null depth in the null direction will be approximately as determined by (4.32).

4.5.3 Design Example

In order to illustrate the method derived above, consider the following example. A FIB was designed to cover the band 1–2 kHz, with an aperture size of $P = 5$ half-wavelengths and a sampling rate of $f_s = 8$ kHz. The primary filters were each of length $L = 21$ coefficients, and the secondary filter had 4 coefficients. The null was to be at $\theta = 30°$. The quiescent beampattern, and the response in the null direction are shown in Fig. 4.6.



(a)

(b)

Figure 4.6: (a) Quiescent beampattern at 9 frequencies within the design band, and (b) quiescent null direction response over the entire design band.

Using the method outlined above, the nulling coefficients $\mathbf{b}$ were calculated for various values of M, and the resulting responses in the nulling direction are shown in Fig. 4.7. Also shown is the expected null depth calculated from (4.32). In all cases the expected null depth gives a good approximation to the maximum value of the null direction output power over the design band.

4.6 Conclusions

The broadband nulling problem considered in this chapter is stated as follows. Given an NL vector of filter coefficients which produces some desired broadband response $r(u, \omega)$ (for a beamformer with N sensors and L filter taps per sensor), find the coefficients which produce a broadband response $\hat{r}(u, \omega)$ which has a broadband null in a specified direction and is close in some respect to the original response $r(u, \omega)$. The problem was formulated in terms of a constrained minimisation problem, i.e., minimise the distance between $r(u, \omega)$ and $\hat{r}(u, \omega)$ subject to the constraint that $\hat{r}(u, \omega)$ exhibits a broadband null in the direction

$u = u_0$. Several candidate cost functions and constraints were considered. These are summarised in Table 4.1, with boldface items indicating those that were studied in detail.

Cost Function
L_∞ error: $\max\limits_{u\in\mathcal{U},\omega\in\Omega} \lvert\hat{r}(u,\omega) - r(u,\omega)\rvert$
Weighted L_2 error: $\int_\Omega \int_\mathcal{U} \varphi(u,\omega)\lvert\hat{r}(u,\omega) - r(u,\omega)\rvert^2 \, du \, d\omega$
Nulling gain: $\mathbf{b}^H\mathbf{b}$
Nulling power: $\mathbf{b}^H E\{s(t)s(t)^H\}\mathbf{b}$

Constraint
Mean-square null depth: $\int_{\omega_L}^{\omega_U} \lvert\hat{r}(u_0,\omega)\rvert^2 d\omega \leq \epsilon$
Exact nulling: $\hat{r}(u_0,\omega) = 0, \quad \forall\omega$
Multiple frequency nulling: $\hat{r}(u_0,\omega) = 0, \quad \omega = \omega_1, \ldots, \omega_M$

Table 4.1: Summary of cost functions and constraints. Boldface items have been considered in detail in this chapter.

For the exact null constraint, we demonstrated that it was possible to produce a pattern zero over all frequencies by formulating the constraint as a time domain constraint. This required oversampling to produce a null at directions other than broadside or endfire, and placed stringent constraints on the sensor locations. The degradation of the exact null which would occur in a practical setting with sensor positioning errors was then considered, and it was shown that a reasonably deep null was still achieved for small sensor position perturbations.

For the multiple frequency nulling constraint, we derived a relationship for the number of multiple zeros to impose in the quiescent beampattern to achieve a broadband null of a given depth over the design bandwidth. Specifically, for a frequency invariant beampattern we showed that (for a given array geometry, number of filter coefficients, sampling rate, design bandwidth and null direction) the number of zeros for a given null depth was independent of the actual quiescent beampattern. Thus it is possible to determine *a priori* the number of frequency zeros required to impose the null, even in the case of an adaptive beamformer.

Finally, we conclude by noting that there is no single “right” answer to the broadband nulling problem considered in this chapter. We have attempted to give a brief review of some possible formulations of the problem, and have obtained some new results concerning both approximate and exact broadband pattern nulls. These new results are specifically aimed at the frequency invariant beamformer, although they have wider application to more general broadband beamformers.

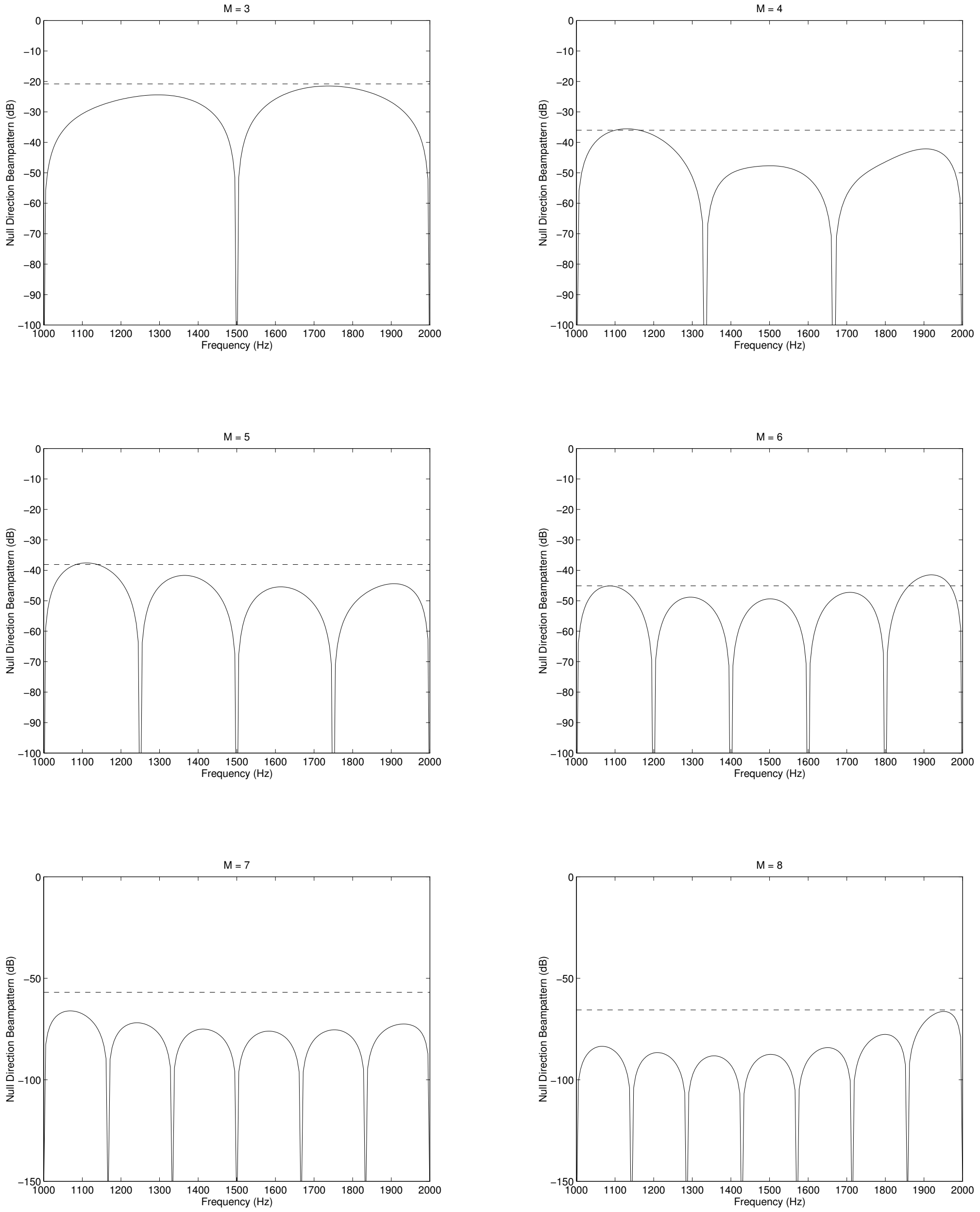


Figure 4.7: Expected null depth (dashed) and actual null depth (solid) for various values of M.

Chapter 5

Nearfield Broadband Frequency Invariant Beamforming

5.1 Introduction

THE vast majority of array literature deals with the case in which the source is assumed to be in the farfield of the array. That is, the source is assumed to be at an infinite distance from the array and hence the received waveform from a single point source is planar. This significantly simplifies the beamforming problem. In many cases the approximate distance at which the farfield approximation begins to be valid is $r = 2L^2/\lambda$, where r is the distance from an arbitrary array origin, L is the largest array dimension, and λ is the operating wavelength [61]. However, for complicated beampatterns with either low sidelobes or deep nulls, a distance of $10L^2/\lambda$ or more may be required [39, 40]. In many practical situations the source is well within this distance, and using the farfield assumption results in severe degradation in the designed beampattern. Furthermore, when broadband operation of the beamformer is required the problem becomes more acute: low frequency components may appear in the nearfield whereas high frequency components may appear in the farfield of the array.

The problem of designing a broadband beamformer for use in the nearfield is important in many areas. One such field is speech acquisition using a microphone array, which finds application in teleconferencing, hands-free telephones, and voice-only data entry. Effective reception of the speech signal in these situations is traditionally hampered by reverberation and external noise sources.

Up until now the methods of nearfield broadband beamforming have been relatively limited in their generality, and based on applying time delays[1] to compensate for the differing propagation delays due to spherical propagation (see [52] for example). We will examine this method in the following section.

In this chapter a new method of nearfield beamforming is proposed in which a desired arbitrary (in both frequency and angle) broadband beampattern may be produced. Specifically we are concerned with the case in which this beampattern is frequency invariant, however the method is applicable to a more general class of nearfield broadband beampatterns.

The methodology is based on the solution to the wave equation and uses spherical harmonics to transform the desired nearfield beampattern to an equivalent farfield pattern. The general character of the radial transformation which is important, as far as the frequency invariant beamforming application is concerned, is that a nearfield *frequency invariant* beampattern can be transformed to a farfield *frequency varying* beampattern. Conversely, a farfield *frequency invariant* beampattern in general transforms to a nearfield *frequency varying* beampattern. This latter fact, which is illustrated in this chapter, highlights the importance of addressing the nearfield issue in the sense that conventional farfield design methods fail.

As part of the nearfield design technique, a broadband beamforming method is developed to realize a general angle-versus-frequency *farfield* beampattern specification. This is referred to as the *general broadband beamformer* (GBB). The development of the GBB theory is based on the philosophy of the FIB theory, namely, derive relationships based on a theoretical continuous sensor and then approximate the response of this continuous sensor using a sensor array. It is shown that the FIB is in fact a special case of the GBB. Coupled with the radial beampattern transformation technique, the GBB can be applied to a wide class of beamforming problems, as indicated in this chapter.

The notation used in this chapter differs slightly from that used in the balance of the thesis. The angle of wave arrival θ for a linear array is measured relative to endfire (rather than broadside), r is used to denote radial distance (rather than a beamformer response), and b is used to denote an aperture response or beamformer response.

[1] In the case of narrowband nearfield beamformers only a phase delay is required. However, for broadband beamformers a true time delay is required.

5.2 Nearfield Compensation

In this section the commonly used method for designing nearfield beamformers is outlined, and the inherent limitations of this method are highlighted.

To illustrate the method, consider a narrowband linear array of N sensors with a complex weight on each sensor. The response of the array to a signal at a distance r and angle θ (measured relative to endfire) to the zeroth sensor is

$$b_{\mathrm{N}}(r,\theta) = \sum_{n=0}^{N-1} w_n \frac{d_0(r,\theta)}{d_n(r,\theta)} \exp\left(\jmath 2\pi f c^{-1}[d_n(r,\theta) - d_0(r,\theta)]\right), \tag{5.1}$$

where w_n is the complex weight on the nth sensor, f is the frequency of operation, c is the speed of wave propagation,

$$d_n(r,\theta) = \left[r^2 + 2r(x_n - x_0)\cos\theta + (x_n - x_0)^2\right]^{\frac{1}{2}} \tag{5.2}$$

is the distance from the source to the nth sensor, and x_n is the location of the nth sensor.

Compare this with the response to a farfield source at an angle θ:

$$b_{\mathrm{F}}(\theta) = \sum_{n=0}^{N-1} w_n \exp\left(\jmath 2\pi f c^{-1} x_n \cos\theta\right). \tag{5.3}$$

The common method of nearfield beamforming is to apply *nearfield compensation*, by which the nearfield response is effectively transformed to the farfield and standard farfield techniques are used to design the array weights. The compensated nearfield response is

$$b_{\mathrm{C}}(r,\theta) = \sum_{n=0}^{N-1} w_n \psi_n \frac{d_0(r,\theta)}{d_n(r,\theta)} \exp\left(\jmath 2\pi f c^{-1}[d_n(r,\theta) - d_0(r,\theta)]\right),$$

where

$$\psi_n = \frac{d_n(r,\phi)}{d_0(r,\phi)} \exp\left(\jmath 2\pi f c^{-1}[d_0(r,\phi) - d_n(r,\phi) + x_n\cos\phi]\right),$$

is the nth compensation term. It is clear that the compensated nearfield response is identical to the farfield response only at $\theta = \phi$, and is approximately equal for angles close to ϕ. To design a nearfield beamformer with a desired response, the weights w_n are obtained using standard narrowband farfield design techniques. The resulting compensated nearfield response is then approximately equal to the designed farfield response.

An example is shown in Fig. 5.1 to demonstrate a typical compensated nearfield response versus an uncompensated response at a radius of 6 half-wavelengths. Clearly

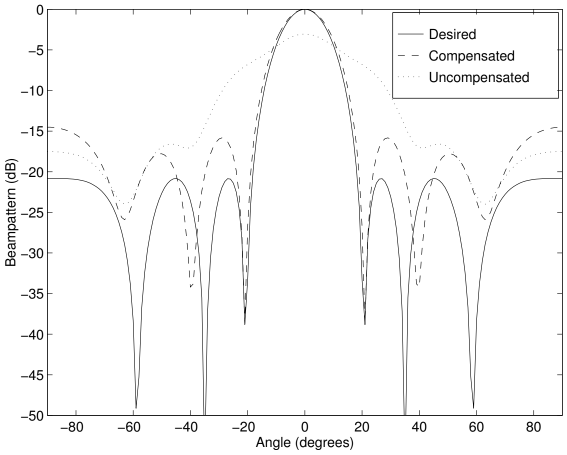


Figure 5.1: Comparison of compensated and uncompensated nearfield beampatterns.

nearfield compensation provides a vast improvement over an uncompensated array, although the compensated response still does not achieve the desired response.

5.3 Radial Beampattern Transformation

Although the nearfield compensation method is relatively straightforward to implement, the results achieved are not very satisfying (as demonstrated by Fig. 5.1). With this is mind, we have sought an alternative broadband nearfield beamforming method. The nearfield beamforming method proposed here is to transform a desired nearfield pattern to a physically equivalent farfield pattern, and design a farfield beamformer to realize this transformed farfield pattern. For a broadband nearfield beamformer we would strictly want to do this over a continuum of frequencies. However, in the following we will develop a transformation for any particular frequency. We will first develop a general transformation using a spherical coordinate system which is applicable to any three dimensional sensor array and any physically realizable beampattern.

5.3.1 Solution to the Spherical Wave Equation

The beampattern transformation is obtained by solving the physical problem governed by the wave equation. Let r denote radial distance, ϕ and θ the azimuth and elevation angles as shown in Fig. 5.2. Note that this is the conventional right-hand coordinate system and is different to that used in Chapter 2. Using this coordinate system simplifies notation.

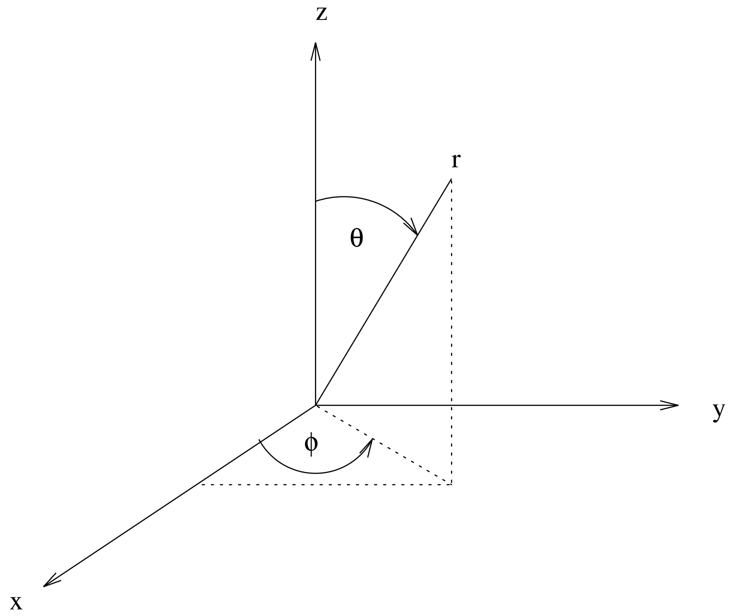


Figure 5.2: Spherical coordinate system.

A general valid beampattern, b, will satisfy the wave equation expressed in spherical coordinates:

$$\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial b}{\partial r}\right)+\frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial b}{\partial\theta}\right)+\frac{1}{r^2\sin^2\theta}\frac{\partial^2 b}{\partial\phi^2}=\frac{1}{c^2}\frac{\partial^2 b}{\partial t^2}. \tag{5.4}$$

Note that by considering the spherical wave equation we are assuming that the beampattern is measured on the surface of a sphere. This may be thought of as a restriction but makes the calculations more tractable. Alternately, a more appropriate coordinate system may be employed.

One group of solutions has a separation property, and the solution may be written:

$$b = R(r)\,\Theta(\theta)\,\Phi(\phi)\,T(t). \tag{5.5}$$

For notational convenience, in most of the remainder of this chapter the explicit functional dependence of R, Θ, Φ, and T on r, θ, ϕ, and t respectively will be dropped.

Although partial solutions to the spherical wave equation may be found in standard texts (e.g. [5, 18]), a complete solution as presented here is lacking. For this reason a full derivation of the solution is given.

Synthesis Equation

Substituting the solution (5.5) into the wave equation (5.4) gives

$$\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\Theta\Phi T\frac{\partial R}{\partial r}\right)+\frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta R\Phi T\frac{\partial\Theta}{\partial\theta}\right)+\frac{1}{r^2\sin^2\theta}R\Theta T\frac{\partial^2 b}{\partial\phi^2}=\frac{1}{c^2}R\Theta\Phi\frac{\partial^2 b}{\partial t^2}.$$

Dividing through by $R\Theta\Phi T$ gives

$$\frac{1}{R}\left(\frac{\partial^2 R}{\partial r^2}+\frac{2}{r}\frac{\partial R}{\partial r}\right)+\frac{1}{r^2\Theta\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial\Theta}{\partial\theta}\right)+\frac{1}{r^2\Phi\sin^2\theta}\frac{\partial^2\Phi}{\partial\phi^2}=\frac{1}{c^2T}\frac{\partial^2 T}{\partial^2 t}. \tag{5.6}$$

The expression on the left hand side is a function of r, θ, and ϕ only, whereas the function on the right hand side is a function of t only. Hence, for the equality to hold, both sides must be equal to a constant. Let this constant be $-k^2$. The right hand side then becomes

$$\frac{1}{c^2T}\frac{\partial^2 T}{\partial t^2}=-k^2,$$

which has a solution

$$T(t)=Me^{\jmath kct}+Ne^{-\jmath kct}, \tag{5.7}$$

where M and N are constants. Equating this with a conventional modulation at the frequency of interest, it follows that

$$k=\frac{2\pi f}{c}, \tag{5.8}$$

which is the wave number. Since we assume that the propagation speed c is fixed, k is simply a constant multiple of frequency f. For this reason, throughout this chapter we will often refer to k as “frequency”.

Setting the left hand side of (5.6) to $-k^2$ and multiplying through by $r^2\sin^2\theta$ gives

$$\frac{r^2\sin^2\theta}{R}\left(\frac{\partial^2 R}{\partial r^2}+\frac{2}{r}\frac{\partial R}{\partial r}\right)+\frac{\sin\theta}{\Theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial\Theta}{\partial\theta}\right)+k^2r^2\sin^2\theta=-\frac{1}{\Phi}\frac{\partial^2\Phi}{\partial\phi^2}. \tag{5.9}$$

The expression on the left hand side is now a function of r and θ only, and the right hand side is a function of ϕ only. Again, for the equality to hold, both sides must be equal to a constant. Let this constant be m^2. Hence, the right hand side becomes

$$\frac{1}{\Phi}\frac{\partial^2\Phi}{\partial\phi^2}=-m^2, \tag{5.10}$$

which has a solution

$$\Phi(\phi) = Ce^{\jmath m\phi} + De^{-\jmath m\phi}, \tag{5.11}$$

where C and D are constants.

Setting the left hand side of (5.9) to m^2 and rearranging gives

$$\frac{r^2}{R}\left(\frac{\partial^2 R}{\partial r^2} + \frac{2}{r}\frac{\partial R}{\partial r}\right) + k^2 r^2 = \frac{m^2}{\sin^2\theta} - \frac{1}{\Theta\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial\Theta}{\partial\theta}\right). \tag{5.12}$$

Again, both sides must be equal to a constant. Let this constant be l^2. Therefore

$$l^2 = \frac{r^2}{R}\left(\frac{\partial^2 R}{\partial r^2} + \frac{2}{r}\frac{\partial R}{\partial r}\right) + k^2 r^2. \tag{5.13}$$

Rearranging gives

$$r^2\frac{\partial^2 R}{\partial r^2} + 2r\frac{\partial R}{\partial r} + \left(k^2 r^2 - l^2\right) R = 0.$$

Let $R = r^{-\frac{1}{2}}S$. With the substitution $l^2 = n(n+1)$ we have [18, p12]

$$r^2\frac{\partial^2 S}{\partial r^2} + r\frac{\partial S}{\partial r} + \left(k^2 r^2 - \left(n + \frac{1}{2}\right)^2\right) S = 0.$$

This is Bessel's equation, which has a solution

$$S(r) = A'\, J_{n+\frac{1}{2}}(kr) + B'\, Y_{n+\frac{1}{2}}(kr),$$

where $J_{n+\frac{1}{2}}(\cdot)$ is the half integer order Bessel function of the first kind, $Y_{n+\frac{1}{2}}(\cdot)$ is the half integer order Neumann function (or Bessel function of the second kind), and A' and B' are constants. This can equivalently be written as

$$\begin{aligned} S(r) &= A\left(J_{n+\frac{1}{2}}(kr) + \jmath\, Y_{n+\frac{1}{2}}(kr)\right) + B\left(J_{n+\frac{1}{2}}(kr) - \jmath\, Y_{n+\frac{1}{2}}(kr)\right) \\ &= A\, H^{(1)}_{n+\frac{1}{2}}(kr) + B\, H^{(2)}_{n+\frac{1}{2}}(kr), \end{aligned}$$

where $H^{(1)}_{n+\frac{1}{2}}(\cdot)$ is the half integer order spherical Hankel function of the first kind, and $H^{(2)}_{n+\frac{1}{2}}(\cdot)$ is the half integer order spherical Hankel function of the second kind (complex conjugate of the spherical Hankel function of the first kind). Substituting $R = r^{-\frac{1}{2}}S$ gives

$$R(r) = r^{-\frac{1}{2}}\left(A\, H^{(1)}_{n+\frac{1}{2}}(kr) + B\, H^{(2)}_{n+\frac{1}{2}}(kr)\right). \tag{5.14}$$

Returning to (5.12), we have

$$n(n+1) = \frac{m^2}{\sin^2\theta} - \frac{1}{\Theta\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial\Theta}{\partial\theta}\right),$$

or equivalently,

$$\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial\Theta}{\partial\theta}\right) + \left(n(n+1) - \frac{m^2}{\sin^2\theta}\right)\Theta = 0.$$

This is the generalised Legendre equation [18, p11] which has a solution

$$\Theta(\theta) = P_n^m(\cos\theta), \tag{5.15}$$

where $P_n^m(\cdot)$ is the associated Legendre polynomial (which reduces to the Legendre polynomial for $m = 0$).

From (5.11), (5.14) and (5.15) the solution (5.5) can now be written as

$$b = r^{-\frac{1}{2}} A\, H^{(1)}_{n+\frac{1}{2}}(kr)\, P_n^m(\cos\theta)\left(Ce^{\jmath m\phi} + De^{-\jmath m\phi}\right). \tag{5.16}$$

Note that the complete solution includes another term containing the spherical Hankel function of the second kind. However, by excluding this term we are excluding standing wave terms, i.e., it is sufficient to consider either waves propagating generally away from the origin or waves propagating generally towards the origin, but not both. This representation is therefore valid on a manifold (typically a sphere) that encapsulates but does not penetrate the physical array.

Summing up all possible terms and rearranging we obtain the complete solution, which we will refer to as the *synthesis equation*:

$$\begin{aligned} b_r(\theta,\phi;k) = r^{-\frac{1}{2}}\Bigg(&\sum_{n=0}^{\infty}\alpha_n H^{(1)}_{n+\frac{1}{2}}(kr)\, P_n(\cos\theta) + \\ &\sum_{n=1}^{\infty}\sum_{m=1}^{n} H^{(1)}_{n+\frac{1}{2}}(kr)\, P_n^m(\cos\theta)\left(\beta_{nm}e^{\jmath m\phi} + \gamma_{nm}e^{-\jmath m\phi}\right)\Bigg). \end{aligned} \tag{5.17}$$

This is analogous to the Fourier transform synthesis equation in that a beampattern is completely characterised by the coefficients α_n, β_{nm} and γ_{nm}.

Analysis Equations

First, consider derivation of α_n. Multiplying (5.17) by $P_v(\cos\theta)\sin\theta$ and integrating with respect to θ and ϕ gives

$$\int_0^{2\pi}\int_0^{\pi} b_r(\theta,\phi;k)P_v(\cos\theta)\sin\theta\ d\theta d\phi$$
$$= 2\pi r^{-\frac{1}{2}}\sum_{n=0}^{\infty}\alpha_n H^{(1)}_{n+\frac{1}{2}}(kr)\int_0^{\pi} P_n(\cos\theta)P_v(\cos\theta)\sin\theta\ d\theta,$$

where the terms involving β_{nm} and γ_{nm} have disappeared since $\int_0^{2\pi} e^{\jmath m\phi}d\phi = 0, \forall m \in \mathbb{N}$. Let $\sin\theta\, d\theta = d(\cos\theta)$. We now make use of the orthogonality property of the Legendre function [77, p.22]:

$$\int_{-1}^{1} P_n(\cos\theta)P_v(\cos\theta)d(\cos\theta) = \begin{cases} 0, & n \neq v, \\ \frac{1}{n+\frac{1}{2}}, & n = v. \end{cases}$$

Substitution gives

$$\int_0^{2\pi}\int_0^{\pi} b_r(\theta,\phi;k)P_n(\cos\theta)\sin\theta\ d\theta d\phi = 2\pi r^{-\frac{1}{2}}\alpha_n H^{(1)}_{n+\frac{1}{2}}(kr)\frac{1}{n+\frac{1}{2}}.$$

Rearranging gives

$$\alpha_n = \frac{n+\frac{1}{2}}{2\pi r^{-\frac{1}{2}}H^{(1)}_{n+\frac{1}{2}}(kr)}\int_0^{2\pi}\int_0^{\pi} b_r(\theta,\phi;k)P_n(\cos\theta)\sin\theta\ d\theta\, d\phi. \tag{5.18}$$

Now consider derivation of β_{nm}. Multiply (5.17) through by $P_v^q(\cos\theta)\sin\theta e^{-\jmath q\phi}$ where $q \in \mathbb{N}$, and integrate with respect to θ and ϕ to give

$$\int_0^{2\pi}\int_0^{\pi} b_r(\theta,\phi;k)P_v^q(\cos\theta)\sin\theta e^{-\jmath q\phi}\ d\theta d\phi$$
$$= r^{-\frac{1}{2}}\sum_{n=0}^{\infty}\alpha_n H^{(1)}_{n+\frac{1}{2}}(kr)\int_0^{2\pi}\int_0^{\pi} P_n(\cos\theta)P_v^q(\cos\theta)\sin\theta e^{-\jmath q\phi}\ d\theta d\phi$$
$$+ r^{-\frac{1}{2}}\sum_{n=1}^{\infty}\sum_{m=1}^{n} H^{(1)}_{n+\frac{1}{2}}(kr)\int_0^{2\pi}\int_0^{\pi} P_n^m(\cos\theta)P_v^q(\cos\theta)\left(\beta_{nm}e^{\jmath(m-q)\phi}+\gamma_{nm}e^{-\jmath(m+q)\phi}\right)\sin\theta\, d\theta\, d\phi.$$

The term involving α_n disappears since $\int_0^{2\pi} e^{-\jmath q\phi} d\phi = 0, \forall q \in \mathbb{N}$. The term involving γ_{nm} disappears since $\int_0^{2\pi} e^{-\jmath(m+q)\phi} d\phi = 0, \forall m, q \in \mathbb{N}$). For the term involving β_{nm} we have

$$\int_0^{2\pi} e^{\jmath(m-q)\phi}\ d\phi = \begin{cases} 0, & m \neq q, \\ 2\pi, & m = q. \end{cases}$$

This gives

$$\int_0^{2\pi}\int_0^{\pi} b_r(\theta,\phi;k) P_v^m(\cos\theta)\sin\theta e^{-\jmath m\phi}\ d\theta d\phi$$
$$= 2\pi r^{-\frac{1}{2}} \sum_{n=1}^{\infty} H^{(1)}_{n+\frac{1}{2}}(kr)\beta_{nm} \int_0^{\pi} P_n^m(\cos\theta)P_v^m(\cos\theta)\sin\theta\ d\theta.$$

Let $\sin\theta = d(\cos\theta)$. The orthogonality property of the associated Legendre function is [77, pp.103-105]

$$\int_{-1}^{1} P_n^m(\cos\theta)P_v^m(\cos\theta)d(\cos\theta) = \begin{cases} 0, & n \neq v, \\ \frac{(n+m)!}{(n-m)!}\frac{1}{n+\frac{1}{2}}, & n = v. \end{cases}$$

Substitution gives

$$\int_0^{2\pi}\int_0^{\pi} b_r(\theta,\phi;k)P_n^m(\cos\theta)\sin\theta e^{-\jmath m\phi}\ d\theta\ d\phi = 2\pi r^{-\frac{1}{2}} H^{(1)}_{n+\frac{1}{2}}(kr)\beta_{nm}\frac{(n+m)!}{(n-m)!}\frac{1}{n+\frac{1}{2}}.$$

Rearranging gives

$$\beta_{nm} = \frac{n+\frac{1}{2}}{2\pi r^{-\frac{1}{2}} H^{(1)}_{n+\frac{1}{2}}(kr)} \frac{(n-m)!}{(n+m)!} \int_0^{2\pi}\int_0^{\pi} b_r(\theta,\phi;k)P_n^m(\cos\theta)\sin\theta e^{-\jmath m\phi}\ d\theta\ d\phi. \tag{5.19}$$

Similarly it can be shown that

$$\gamma_{nm} = \frac{n+\frac{1}{2}}{2\pi r^{-\frac{1}{2}} H^{(1)}_{n+\frac{1}{2}}(kr)} \frac{(n-m)!}{(n+m)!} \int_0^{2\pi}\int_0^{\pi} b_r(\theta,\phi;k)P_n^m(\cos\theta)\sin\theta e^{\jmath m\phi}\ d\theta\ d\phi. \tag{5.20}$$

We will refer to (5.18), (5.19) and (5.20) as the *analysis equations.* These equations are analogous to the Fourier transform analysis equations in that they give the coefficients which completely characterise a given beampattern.

Complete Solution

The general synthesis and analysis equations are repeated here for convenience.

Synthesis Equation:

$$b_r(\theta,\phi;k) = r^{-\frac{1}{2}}\Bigg(\sum_{n=0}^{\infty}\alpha_n H^{(1)}_{n+\frac{1}{2}}(kr)\,P_n(\cos\theta) + \sum_{n=1}^{\infty}\sum_{m=1}^{n} H^{(1)}_{n+\frac{1}{2}}(kr)\,P_n^m(\cos\theta)\,(\beta_{nm}e^{\jmath m\phi}+\gamma_{nm}e^{-\jmath m\phi})\Bigg) \tag{5.21}$$

Analysis Equations:

$$\alpha_n = \frac{n+\frac{1}{2}}{2\pi r^{-\frac{1}{2}}H^{(1)}_{n+\frac{1}{2}}(kr)}\int_0^{2\pi}\int_0^{\pi} b_r(\theta,\phi;k)P_n(\cos\theta)\sin\theta\,d\theta\,d\phi \tag{5.22a}$$

$$\beta_{nm} = \frac{n+\frac{1}{2}}{2\pi r^{-\frac{1}{2}}H^{(1)}_{n+\frac{1}{2}}(kr)}\frac{(n-m)!}{(n+m)!}\int_0^{2\pi}\int_0^{\pi} b_r(\theta,\phi;k)P_n^m(\cos\theta)\sin\theta e^{-\jmath m\phi}\,d\theta\,d\phi \tag{5.22b}$$

$$\gamma_{nm} = \frac{n+\frac{1}{2}}{2\pi r^{-\frac{1}{2}}H^{(1)}_{n+\frac{1}{2}}(kr)}\frac{(n-m)!}{(n+m)!}\int_0^{2\pi}\int_0^{\pi} b_r(\theta,\phi;k)P_n^m(\cos\theta)\sin\theta e^{\jmath m\phi}\,d\theta\,d\phi \tag{5.22c}$$

The utility of the radial beampattern transformation is as follows. Given a beampattern $b_{r_1}(\theta,\phi;k)$ measured at some radius r_1, calculate α_n, β_{nm} and γ_{nm} from (5.22) with $r=r_1$. The beampattern can now be reconstructed at any radius r_2 by using (5.21) with $r=r_2$.

The method we propose to design a nearfield beamformer is outlined below.

1. Calculate the beampattern coefficients for the desired nearfield beampattern $b_{r_d}(\theta,\phi;k)$ using (5.22) with $r=r_d$.

2. Calculate $b_\infty(\theta,\phi;k)$ from (5.21) at $r=\infty$.

3. Design a farfield beamformer to realize this beampattern using classical farfield array design techniques.

4. Using this farfield beamformer in the nearfield at $r=r_d$ will produce the desired nearfield beampattern $b_{r_d}(\theta,\phi;k)$.

An important feature of this formulation is that the actual array geometry is of only secondary importance. Any array geometry which can realize the resulting transformed farfield beampattern may be used. This is important in a practical situation in which the array is mounted on a complex three dimensional surface, such as a microphone array mounted on the curved dashboard of a car. The question of whether a specific array can realize a specific farfield beampattern is a separate issue and is not addressed here (some solutions are given in [103, 104]).

5.3.2 Transformation to the Farfield

The requirement to transform to $r = \infty$ exposes some potential numerical problems; these are easily resolved as follows.

The half integer order spherical Hankel function of the first kind satisfies the asymptotic form [5, p.243]

$$\begin{aligned} H^{(1)}_{n+\frac{1}{2}}(kr) &\sim \sqrt{\frac{2}{\pi kr}} \exp\left(\jmath\left[kr - \frac{\pi}{2}(n+1)\right]\right) \\ &= (-\jmath)^{n+1}\sqrt{\frac{2}{\pi kr}} \exp(\jmath kr), \quad \text{as } r \to \infty. \end{aligned}$$

Hence, in the synthesis equation (5.21) there is an attenuation with distance like r^{-1}, i.e.,

$$\left| r^{-1/2} H^{(1)}_{n+\frac{1}{2}}(kr) \right| \sim r^{-1}; \quad \text{as } r \to \infty.$$

Synthesis at $r = \infty$ gives $b_\infty(\theta, \phi; k) = 0$ which is clearly unacceptable. This can be easily compensated for by normalising the magnitude of $b_r(\theta, \phi; k)$ by multiplying by r (this works because all modes exhibit this attenuation, i.e., it is independent of m and n).

The other problem is what phase to associate with $r = \infty$. This is somewhat arbitrary, but is easily dealt with by setting the phase of the asymptotic half integer order spherical Hankel function to zero at a nominal frequency k_0.

We thus obtain the following farfield synthesis equation

$$\begin{aligned} \tilde{b}_\infty(\theta, \phi; k) \sim \sqrt{\frac{2}{\pi k}}\, e^{\jmath(k-k_0)} \Bigg(&\sum_{n=0}^{\infty} \alpha_n (-\jmath)^{n+1} P_n(\cos\theta) \\ &+ \sum_{n=1}^{\infty}\sum_{m=1}^{n} (-\jmath)^{n+1} P_n^m(\cos\theta)(\beta_{nm} e^{\jmath m\phi} + \gamma_{nm} e^{-\jmath m\phi}) \Bigg), \end{aligned} \tag{5.23}$$

where $\tilde{b}$ denotes a normalised beampattern.

5.3.3 Linear Array

Consider a linear array aligned with the z axis. In this case the beampattern is rotationally symmetric with respect to ϕ, and the beampattern can be expressed as $b_r(\theta, \phi; k) =$

$b_r(\theta; k)$. The only non-zero components are those for which $m = 0$, which leads to the following set of equations:

$$b_r(\theta; k) = \sum_{n=0}^{\infty} \alpha_n r^{-1/2} H^{(1)}_{n+\frac{1}{2}}(kr) P_n(\cos\theta) \tag{5.24}$$

and

$$\alpha_n = \frac{n + \frac{1}{2}}{r^{-\frac{1}{2}} H^{(1)}_{n+\frac{1}{2}}(kr)} \int_0^{\pi} b_r(\theta; k) P_n(\cos\theta) \sin\theta d\theta. \tag{5.25}$$

Applying the normalisations described above we obtain the following farfield synthesis equation for a linear array aligned with the z axis.

$$\tilde{b}_{\infty}(\theta; k) \sim \sqrt{\frac{2}{\pi k}} e^{\jmath(k-k_0)} \left(\sum_{n=0}^{\infty} \alpha_n (-\jmath)^{n+1} P_n(\cos\theta) \right) \tag{5.26}$$

5.3.4 Parseval Relation

The synthesis equation (5.21) requires an infinite number of terms to exactly characterise the beampattern. In this section we derive a Parseval relation for the radial beampattern transformation and use it to provide an expression for the error associated with using a finite number of terms in the synthesis equation. For simplicity, we only consider a linear array aligned with the z axis.

Rewrite (5.24) and (5.25) as

$$b_r(\theta; k) = \sum_{n=0}^{\infty} A_n \, P_n(\cos\theta) \tag{5.27}$$

$$A_n = \left(n + \frac{1}{2} \right) \int_0^{\pi} b_r(\theta; k) \, P_n(\cos\theta) \sin\theta \; d\theta, \tag{5.28}$$

where $A_n = \alpha_n r^{-\frac{1}{2}} H^{(1)}_{n+\frac{1}{2}}(kr)$. Although A_n is a function of r and k, we suppress this dependence to simplify notation.

The Parseval relationship is derived as follows. Consider the integral of the beampattern over all real angles:

$$\begin{aligned} \int_0^{\pi} |b_r(\theta; k)|^2 \sin\theta \, d\theta &= \int_{-1}^{1} |b_r(u; k)|^2 \; du \\ &= \int_{-1}^{1} b_r(u; k) \, b_r(u; k)^* \; du, \end{aligned}$$

with the change of variable $u = \cos\theta$. Replace $b_r(u;k)^*$ by its series representation from (5.27),

$$b_r(u;k)^* = \sum_{n=0}^{\infty} A_n^* \, P_n(u).$$

Interchange summation and integration to give

$$\begin{aligned}\int_0^{\pi} |b_r(\theta;k)|^2 \sin\theta \, d\theta &= \sum_{n=0}^{\infty} A_n^* \int_{-1}^{1} b_r(u;k) \, P_n(u) \; du \\ &= \sum_{n=0}^{\infty} A_n^* \, A_n \, \frac{1}{n+\frac{1}{2}}.\end{aligned}$$

The Parseval relation may now be written as

$$\int_0^{\pi} |b_r(\theta;k)|^2 \sin\theta \, d\theta = \sum_{n=0}^{\infty} \frac{1}{n+\frac{1}{2}} \left|A_n\right|^2. \tag{5.29}$$

Assume we wish to approximate $b_r(\theta;k)$ by a finite series of the form

$$\hat{b}_r(\theta;k) = \sum_{n=0}^{N} \hat{A}_n \, P_n(\cos\theta). \tag{5.30}$$

The integral squared error between $b_r(\theta;k)$ and $\hat{b}_r(\theta;k)$ is given by

$$\begin{aligned}\epsilon &= \int_0^{\pi} \left| b_r(\theta;k) - \hat{b}_r(\theta;k) \right|^2 \sin\theta \, d\theta \\ &= \sum_{n=0}^{\infty} \frac{1}{n+\frac{1}{2}} \left| A_n - \hat{A}_n \right|^2,\end{aligned}$$

using the Parseval relation (5.29). Separating the summation into two parts gives

$$\epsilon = \sum_{n=0}^{N} \frac{1}{n+\frac{1}{2}} \left| A_n - \hat{A}_n \right|^2 + \sum_{n=N+1}^{\infty} \frac{1}{n+\frac{1}{2}} \left|A_n\right|^2.$$

Clearly, to minimise ϵ, set $\hat{A}_n = A_n$, for $n = 0, \ldots, N$. The residual error is then given by

$$\begin{aligned}\epsilon_{\min} &= \sum_{n=N+1}^{\infty} \frac{1}{n+\frac{1}{2}} \left|A_n\right|^2 && (5.31) \\ &= \int_0^{\pi} |b_r(\theta;k)|^2 \sin\theta \, d\theta - \sum_{n=0}^{N} \frac{1}{n+\frac{1}{2}} \left|A_n\right|^2, && (5.32)\end{aligned}$$

where the A_n are given by (5.28).

Parseval Relation Example

To demonstrate the utility of the Parseval relation, consider a beampattern defined by

$$b_r(\theta; k) = \frac{1}{M} \sum_{m=0}^{M-1} e^{-\jmath \pi m \cos\theta}. \tag{5.33}$$

In this case it can be shown that

$$\int_{-1}^{1} |b_r(\theta; k)|^2 \sin\theta \, d\theta = \frac{2}{M}.$$

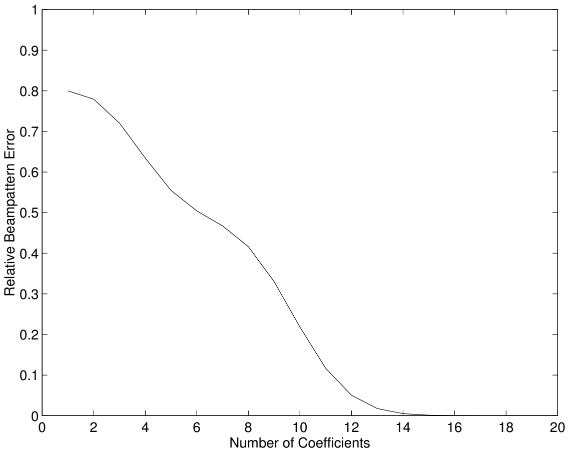


Figure 5.3: Relative beampattern error calculated from (5.32) for the example beampattern shown in Fig. 5.4.

Figure 5.3 shows the relative beampattern error versus the number of coefficients, calculated from (5.32) for the case of $M = 5$. The plot has been normalised by $2/M$.

Figure 5.4 shows the resulting approximate beampatterns calculated from (5.30) for several different values of N. Also shown is the desired asymptotic beampattern corresponding to $N = \infty$. Figure 5.3 predicts that 15 coefficients should be sufficient to obtain a good approximation to the asymptotic beampattern. This is confirmed by Fig. 5.4.

5.3.5 Radial Transformation Example

To demonstrate the utility of the radial transformation, consider the beampattern defined by (5.33) with $M = 5$, at a radius of $r = 10\lambda/2$. A set of 20 coefficients were calculated from (5.25). (From Fig. 5.3 we note that 20 coefficients should be sufficient to produce negligible error from the asymptotic beampattern.) The pattern was transformed to the

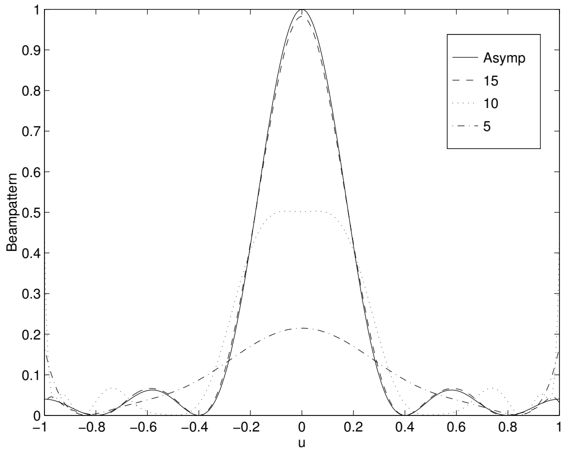
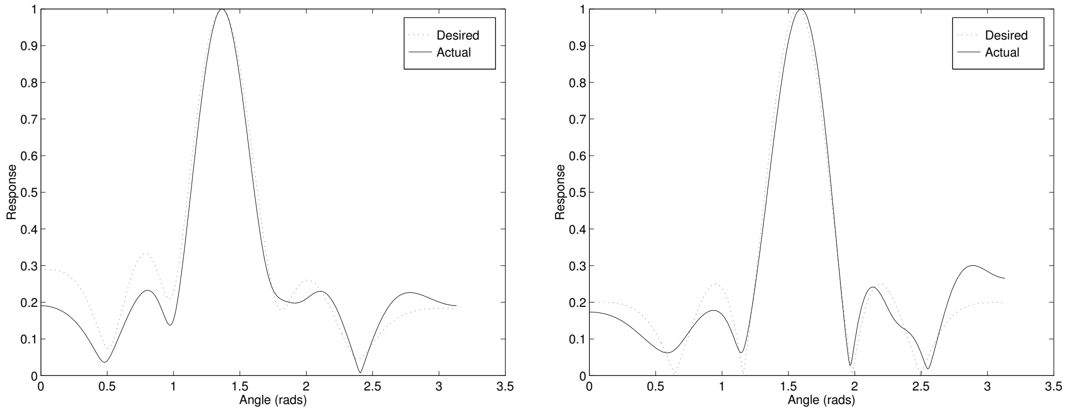


Figure 5.4: Approximate beampattern calculated from (5.30) for several different values of N. Also shown is the desired asymptotic beampattern.

farfield using (5.26), and the resulting farfield beampattern is shown dotted in Fig. 5.5(a). Note that the main beam has moved away from broadside due to the non-zero phase of the desired nearfield beampattern as defined by (5.33). A farfield beamformer with 11 half-wavelength spaced sensors (centred on the origin) was then designed using a complex-valued least squares criterion to achieve this desired farfield pattern. The resulting farfield beamformer response is shown solid in Fig. 5.5(a). This farfield beamformer was then used in the nearfield at a radius of $r = 10\lambda/2$. The response of this beamformer is shown solid in Fig. 5.5(b). Also shown is the desired nearfield response (dotted). Since the farfield beamformer does not provide a good enough approximation to the required farfield response shown in Fig. 5.5(a), the resulting nearfield response shows some divergence from the desired nearfield response.

As a second example, consider the same pattern as in the previous example, but now with zero phase. The resulting transformed farfield pattern is shown dotted in Fig. 5.6(a). This pattern was then approximated using a farfield beamformer with five half-wavelength spaced sensors (centred on the origin), resulting in the solid pattern in Fig. 5.6(a). Since this farfield beamformer provides a good approximation to the desired farfield response, the corresponding nearfield response is very close to the desired response, as shown by Fig. 5.6(b).

In comparing Fig. 5.5(b) with Fig. 5.6(b), it is clear that for our nearfield beamforming method to work effectively it is necessary to provide a good approximation to the transformed farfield pattern. We have found that nearfield patterns with zero phase generally transform to well-behaved farfield patterns which may be approximated reasonably well.



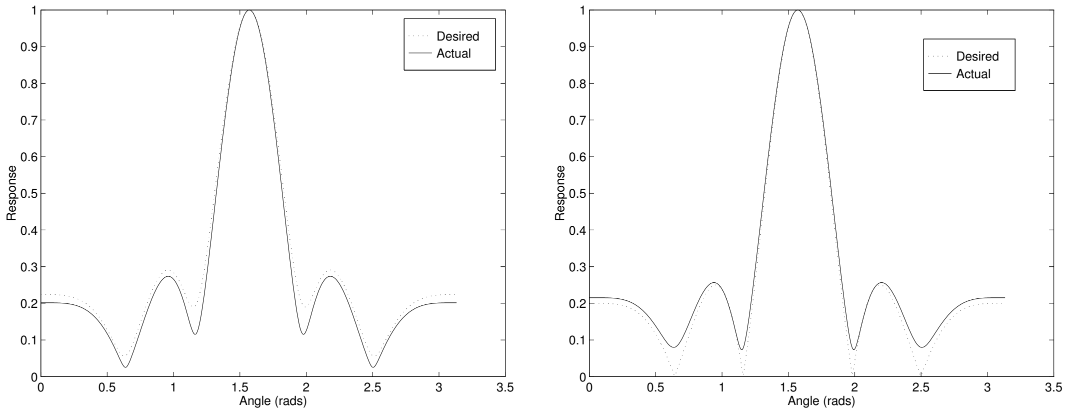
(a) (b)

Figure 5.5: (a) Transformed farfield response (dotted), and farfield response of designed beamformer (solid). (b) Resulting nearfield response from designed beamformer (solid), and desired nearfield response (dotted).

5.4 General Broadband Beamforming

So far we have only considered application of the radial transformation at a single frequency. For a broadband nearfield beamformer, the radial transformation would be applied over a range of frequencies. Assuming radial symmetry in the ϕ variable (i.e., for a linear array aligned with the z axis) this would produce a general farfield beampattern $b_\infty(\theta; k)$ specified over both angle θ and frequency k. In this section we consider how to produce a farfield beamformer which realizes a general desired broadband beampattern. Specifically, the problem is to find the filters to apply to each sensor in an array to achieve the desired general broadband beampattern. We refer to this as the *general broadband beamformer* (GBB) problem.

For a uniformly spaced array it is known that a two dimensional discrete Fourier transform relationship exists between the broadband response and the weights of the sensor filters [43]. However, we believe it is more appropriate to use nonuniformly spaced arrays for broadband applications (see Chapter 2), and hence desire a more general design method. As in Chapter 2 we will initially develop the GBB theory in terms of a linear continuous aperture (in practice, and as illustrated later, we can approximate this by a linear array of nonuniformly spaced sensors).



(a) (b)

Figure 5.6: (a) Transformed farfield response (dotted), and farfield response of designed beamformer (solid). (b) Resulting nearfield response from designed beamformer (solid), and desired nearfield response (dotted).

The response of a continuous linear aperture to planar waves arriving from an angle θ (measured relative to endfire) is

$$a(u,k) = \int_{-\infty}^{\infty} \rho(x,k) e^{jkxu}\, dx \tag{5.34}$$

where $u = \cos\theta$, $k = 2\pi f/c$ is the wave number (frequency) and $\rho(x,k)$ is the broadband aperture illumination (which is a function of both location and frequency). For a finite aperture size, $\rho(x,k)$ will have finite support in x. Compare the response of the continuous linear aperture to that of a linear array

$$\hat{a}(u,k) = \sum_{n=0}^{N-1} w_n(k) e^{jkx_n u},$$

where $w_n(k)$ is the complex weight on the nth sensor at a frequency k. Our aim is to find the $w_n(k)$ such that $\hat{a}(u,k)$ approximates $a(u,k)$.

Temporarily fix the frequency to some arbitrary value $k = k_0$, and introduce $a_0(uk_0) = a(u,k_0)$, $\rho_0(x) = \rho(x,k_0)$, and $y = uk_0$. Substituting into (5.34) gives

$$a_0(y) = \int_{-\infty}^{\infty} \rho_0(x)\, e^{jyx}\, dx = \mathcal{F}^{-1}\left\{\rho_0(x)\right\},$$

where $\mathcal{F}\{\cdot\}$ and $\mathcal{F}^{-1}\{\cdot\}$ denote the Fourier transform and its inverse respectively. Hence,

$$\rho_0(x) = \mathcal{F}\{a_0(y)\} = \frac{1}{2\pi}\int_{-\infty}^{\infty} a_0(y)\, e^{-\jmath yx}\, dy.$$

In general, the above relation holds for all values of k_0. By substitution,

$$\rho(x,k) = \frac{k}{2\pi}\int_{-\infty}^{\infty} a(u,k)\, e^{-\jmath kxu}\, du. \tag{5.35}$$

Note that $\rho(x,k)$ has the interpretation of being the filter response (function of frequency) required at displacement x (relative to a nominal origin) to achieve the desired broadband beampattern $a(u,k)$.

With regard to implementation, where discrete sensor locations are necessary, a practical design is readily obtained by approximating (5.34) by a summation in the manner introduced in Chapter 2, i.e.,

$$\hat{a}(u,k) = \frac{\eta k}{2\pi}\sum_{n=0}^{N-1} g_n H_n(k) e^{\jmath k x_n u}, \tag{5.36}$$

where $\{x_n\}_{n=0}^{N-1}$ is a set of N discrete sensor locations, $H_n(k)$ is the filter response on the nth sensor, g_n is a *spatial weighting term* which is used to account for the (possibly) nonuniformly spaced sensor locations, and η is a normalisation constant. Each of the channel filters is given by

$$H_n(k) = \int_{-\infty}^{\infty} a(u,k) e^{-\jmath k x_n u} du. \tag{5.37}$$

The corresponding filter coefficients may be found by simple inverse Fourier transform of the resulting filter response.[2] In comparing (5.37) with (5.35), note that we have taken out the common $k/(2\pi)$ term in (5.35) and chosen to implement it as a secondary filter (c.f. Chapter 2). We are thus led to the block diagram shown in Fig. 5.7. Note that the secondary filter may be perturbed slightly from its ideal differentiator form to normalise any scaling errors introduced by the approximation of (5.34) by the summation (5.36).

[2] Note that the beampattern sampling technique for obtaining the filter coefficients (see §3.2.4) cannot be used unless $a(u,k)$ is frequency invariant.

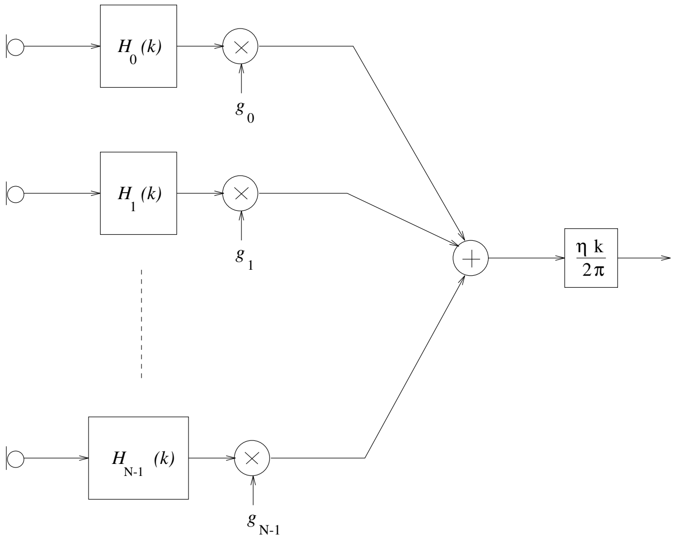


Figure 5.7: Block diagram of general broadband beamformer.

5.4.1 Application to Nearfield Beamforming

This formulation can now be directly applied to the problem of implementing the transformed farfield broadband beampattern. Specifically, if $\tilde{b}_\infty(\theta; k)$ is the normalised broadband farfield beampattern resulting from a radial transformation of some desired broadband nearfield beampattern through (5.25) and (5.26), then the implementation equations are

$$\hat{b}_\infty(\theta; k) = \frac{k}{2\pi} \sum_{n=0}^{N-1} g_n H_n(k) e^{\jmath k x_n \cos\theta} \tag{5.38}$$

where

$$H_n(k) = \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \tilde{b}_\infty(\theta; k) e^{-\jmath k x_n \cos\theta} \sin\theta d\theta. \tag{5.39}$$

5.4.2 Relationship to Frequency Invariant Beamforming

As stated in the introduction, the frequency invariant beamforming theory developed in Chapter 2 is a special case of the general broadband beamforming theory developed here. To see this, let $a(u, k) = a(u)$, i.e., the response is independent of frequency. Substituting into (5.35) gives

$$\begin{aligned}\rho(x,k) &= \frac{k}{2\pi}\int_{-\infty}^{\infty} a(u)e^{-\jmath kxu}du\\ &= k\ G(xk).\end{aligned}$$

That is, to obtain a frequency response the aperture illumination should scale directly with frequency. This is precisely the result obtained in Chapter 2.

5.4.3 Relationship to Narrowband Beamforming

A conventional narrowband beamformer is also a special case of the general broadband beamformer. In this case, let $a(u,k) = a(uk)$, i.e., the response varies directly with frequency. Substituting into (5.34) gives

$$\begin{aligned}\rho(x,k) &= \frac{k}{2\pi}\int_{-\infty}^{\infty} a(uk)e^{-\jmath kxu}du\\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty} a(z)e^{-\jmath xz}dz\\ &= G(x),\end{aligned}$$

with the change of variables $z = ku$. In other words, to obtain a response which varies directly with frequency, the aperture illumination should remain constant for all frequencies. This is consistent with the observation that the response of a narrowband aperture (in which the aperture illuminations remains constant) scales directly with frequency.

5.5 Nearfield Frequency Invariant Beamforming

In general, a nearfield frequency *invariant* pattern transforms to a farfield frequency *varying* pattern, i.e., $b_r(\theta,\phi) \Rightarrow b_\infty(\theta,\phi;k), k \in [k_L, k_U]$, where r is a nearfield radius, and $[k_L, k_U]$ is the bandwidth over which the nearfield pattern is frequency invariant.[3] This highlights the need for the general broadband beamforming theory developed in §5.4 in

[3] It is interesting to note from (5.21) that if a nearfield pattern consists only of terms with the same order n, then the half integer order Hankel function of the first kind factors out of the summation. In this case, the beampattern shape remains constant for all values of r (apart from attenuation with distance). This means that a nearfield frequency invariant pattern would transform to a farfield frequency invariant pattern. Although this is an appealing concept, we have found that no "useful" beampattern can be produced using only associated Legendre functions of the same order n.

the nearfield frequency invariant beamforming problem. From (5.22) note that for a frequency invariant beampattern, the integrands are independent of frequency, resulting in some efficiency in the computation of the analysis coefficients.

5.5.1 Design Example

The following example illustrates the proposed broadband nearfield beamforming method. We wish to design a nearfield frequency invariant beamformer having the response shown in Fig. 5.8 at a distance of 2 metres from the array phase centre.

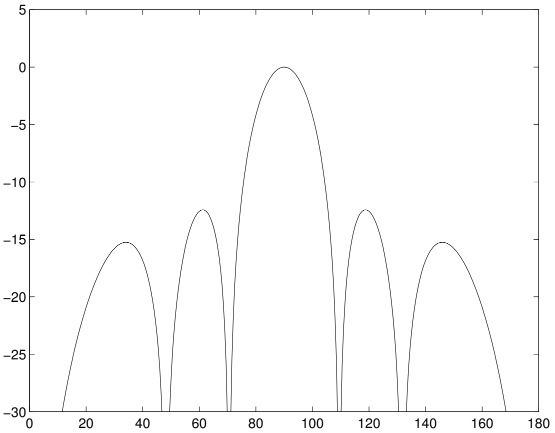


Figure 5.8: Desired nearfield beampattern.

An array of 13 non-uniformly spaced sensors was designed as described in Chapter 2 to cover the frequency range 500–1000 Hz, with a sampling rate of 2000 Hz.

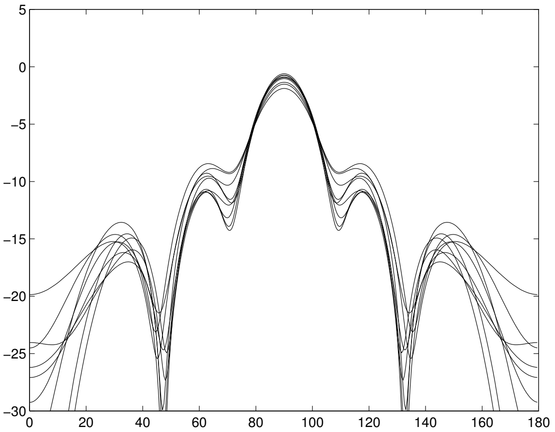


Figure 5.9: Using a farfield FI design at a radius of 2 metres.

Fig. 5.9 shows the result of using a farfield frequency invariant beamformer having the desired response, at a radius of 2 metres and for 9 different frequencies uniformly distributed in the band. Clearly there is significant variation with frequency, and the beampattern does not display the same shape as that in Fig. 5.8.

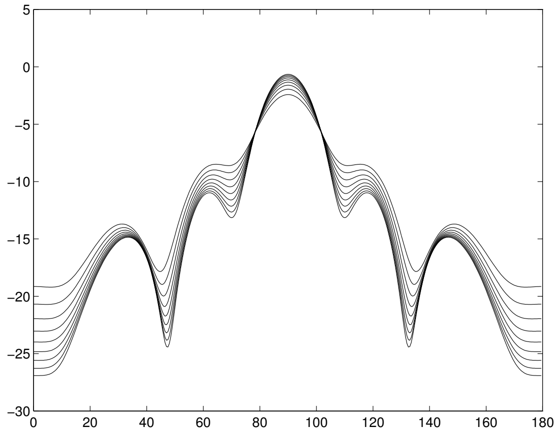


Figure 5.10: Nearfield FI beampattern transformed to farfield.

Using (5.25), a set of 25 analysis coefficients for the desired frequency invariant pattern of Fig. 5.8 was calculated at a radius of 2 metres for 9 frequencies uniformly distributed in the band. Applying (5.26), the farfield beampatterns were reconstructed for each of the 9 design frequencies, resulting in the farfield patterns shown in Fig. 5.10. Hence, we see that the frequency invariant nearfield beampattern is transformed to a frequency dependent farfield beampattern.

We then designed a general farfield beamformer of the structure shown in Fig. 5.7, with 32 complex coefficients for each sensor filter, to realize the farfield patterns shown in Fig. 5.10. The secondary filter has been slightly perturbed as outlined in section 5.4 to normalise small scaling errors introduced by the implementation. Fig. 5.11 shows the results of using this general farfield beamformer at a radius of 2 metres. Comparing this response with the desired response of Fig. 5.8 we note that the desired nearfield frequency invariant response has been achieved.

5.6 Conclusions

A new method of broadband nearfield beamforming has been presented. The methodology of the new technique can be partitioned into two steps: (i) a wave-equation based technique to radially transform an arbitrary nearfield beampattern to a corresponding

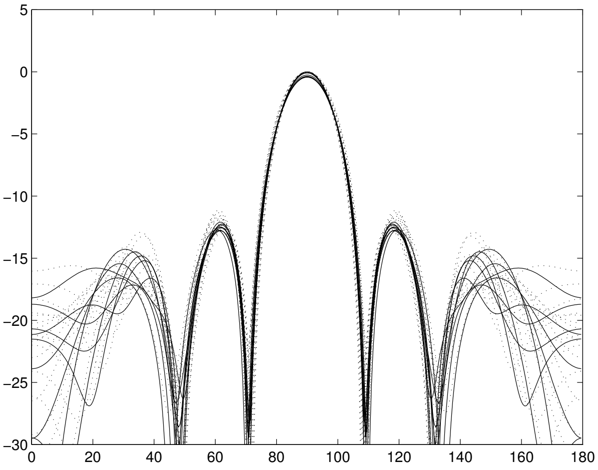


Figure 5.11: General broadband beamformer designed to give the farfield responses shown in Fig. 5.10, used in the nearfield. Solid lines are for the 9 design frequencies, dotted lines are for 16 intermediate frequencies.

equivalent farfield beampattern (or beampattern at any other radius); and (ii) a design method to achieve a desired farfield beamforming response specified over both angle and frequency. Although we were motivated by the problem of nearfield frequency invariant beamforming, the solution proposed in this chapter is applicable to a much wider class of problems. These include: (i) nearfield broadband beamforming with an arbitrary beampattern (specified over both angle and frequency) measured on a spherical manifold; and (ii) farfield broadband beamforming with an arbitrary beampattern (specified over both angle and frequency). The broadband frequency invariant beamformer and the narrowband beamformer (both nearfield and farfield) may then be considered as special cases of this more general class of beamformers.

Chapter 6

Broadband DOA Estimation using Frequency Invariant Beamforming

6.1 Introduction

ESTIMATING the locations of multiple sources is an important problem in fields such as radar, sonar, communications and seismology. A plethora of methods for narrowband direction of arrival (DOA) estimation have been proposed. One of the most promising[1] of these is the multiple signal classification (MUSIC) algorithm [80]. However, in many practical cases the signals of interest are broadband in nature. The basis of most broadband DOA estimators is to perform some kind of transformation on the received broadband array data such that narrowband estimators (such as MUSIC) may be used.

In this chapter we apply the frequency invariant beamformer (FIB) to the problem of broadband DOA estimation. Specifically, our aim is to show that through the use of the FIB many of the problems associated with other broadband DOA estimators (such as asymptotic estimate bias, added computational burden of narrowband frequency decomposition and the requirement for preliminary DOA estimates) can be eliminated.

The chapter is organised as follows. In the following section we establish notation and formally define the problem. We then give a brief review of the MUSIC algorithm and its application to the broadband DOA problem; this serves to outline some of the shortcomings of existing methods. In light of these shortcomings, the new estimator is

[1] It is stated in [78] that "in a detailed evaluation based on thousands of simulations, M.I.T.'s Lincoln Laboratory concluded that, among [the then] currently accepted high-resolution algorithms, MUSIC was the most promising and a leading candidate for further study and actual hardware implementation."

then presented. Finally, we provide simulations to compare the proposed method with one common broadband DOA estimator and draw conclusions.

6.2 Problem Statement

Consider a linear array of N sensors, not necessarily uniformly spaced. Assume $D < N$ farfield broadband signals arrive from directions $\Theta = [\theta_1, \dots, \theta_D]$ where θ_d is the direction to the dth source measured relative to broadside. The signal observed at the nth sensor at time t is

$$y_n(t; f) = \sum_{d=1}^{D} s_d(t + \tau_n(\theta_d); f) + v_n(t; f), \tag{6.1}$$

where $s_d(t; f)$ is the received signal at the first sensor from the dth source, $v_n(t; f)$ is the additive noise at the nth sensor,

$$\tau_n(\theta) = (x_n - x_1)c^{-1} \sin\theta \tag{6.2}$$

is the relative propagation delay to the nth sensor with a propagation speed of c, and x_n is the location of the nth sensor. Note that we have included f as a parameter in the above data model to denote the dependence of the array data on the frequency of the signals and noise. Without loss of generality we assume that the source and noise signals have the same finite bandwidth $[f_L, f_U]$.

The array data can be rewritten in vector form as

$$\mathbf{y}(t; f) = \mathbf{A}(\Theta, f)\,\mathbf{s}(t; f) + \mathbf{v}(t; f), \tag{6.3}$$

where

$$\mathbf{s}(t; f) = [s_1(t; f), \dots, s_D(t; f)]^T,$$
$$\mathbf{v}(t; f) = [v_1(t; f), \dots, v_N(t; f)]^T$$

are the $D \times 1$ source signal vector and $N \times 1$ additive noise vector respectively, and

$$\mathbf{A}(\Theta, f) = [\mathbf{a}(\theta_1, f), \dots, \mathbf{a}(\theta_D, f)]$$

is the $N \times D$ source location matrix. For farfield sources each column of $\mathbf{A}(\Theta, f)$ is an $N \times 1$ location vector of the form

$$\mathbf{a}(\theta, f) = \left[e^{\jmath 2\pi f \tau_1(\theta)}, \ldots, e^{\jmath 2\pi f \tau_N(\theta)}\right]^T. \tag{6.4}$$

The problem we consider is determining the source directions Θ from the observed broadband array data $\mathbf{y}(t; f)$ over a finite time period. We assume that all sources fall within a spatial sector $\Delta\theta$; the reason for this will be made apparent shortly.

6.3 Background

A brief review of the MUSIC estimator (and some of its variants) is now given to set the context for the proposed broadband DOA estimator in §6.4.

6.3.1 MUSIC

Consider the data model (6.3) for narrowband signals and noise (hence we drop the frequency dependence). The covariance matrix of this narrowband observation vector is

$$\mathbf{R} \triangleq E\{\mathbf{y}(t)\mathbf{y}(t)^H\}.$$

Assuming the signals and noise are uncorrelated, this reduces to

$$\mathbf{R} = \mathbf{A}(\Theta)\mathbf{R}_s\mathbf{A}(\Theta)^H + \sigma^2\mathbf{R}_v,$$

where $\mathbf{R}_s = E\{\mathbf{s}(t)\mathbf{s}(t)^H\}$ is the unknown $D \times D$ source covariance matrix (assumed positive definite), σ^2 is the (unknown) noise power, and $\sigma^2\mathbf{R}_v = E\{\mathbf{v}(t)\mathbf{v}(t)^H\}$ is the noise covariance matrix (where $\mathbf{R}_v$ is assumed known[2]). Assuming the noise is uncorrelated from sensor to sensor, $\mathbf{R}_v = \mathbf{I}$.

The eigen-decomposition of $\mathbf{R}$ may then be written

$$\mathbf{R}\mathbf{e}_i = \lambda_i\mathbf{e}_i, \quad i = 1, \ldots, N,$$

[2] Knowledge of the second order statistics of the noise is an integral part of eigen-based DOA methods.

where $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_N$ are the ordered eigenvalues of $\mathbf{R}$, and $\mathbf{e}_1, \ldots, \mathbf{e}_N$ are the corresponding eigenvectors. Hence

$$[\mathbf{A}(\Theta)\mathbf{R}_s\mathbf{A}(\Theta)^H + \sigma^2\mathbf{I}][\mathbf{e}_1 \cdots \mathbf{e}_N] = \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_N \end{bmatrix} [\mathbf{e}_1 \cdots \mathbf{e}_N],$$

or

$$[\mathbf{A}(\Theta)\mathbf{R}_s\mathbf{A}(\Theta)^H][\mathbf{e}_1 \cdots \mathbf{e}_N] = \begin{bmatrix} \lambda_1 - \sigma^2 & & 0 \\ & \ddots & \\ 0 & & \lambda_N - \sigma^2 \end{bmatrix} [\mathbf{e}_1 \cdots \mathbf{e}_N]. \tag{6.5}$$

Since $\mathbf{A}(\Theta)$ is of rank D (recall, there are D impinging sources), and $\mathbf{R}_s$ is positive definite, it follows that $\mathbf{A}(\Theta)\mathbf{R}_s\mathbf{A}(\Theta)^H$ is non-negative definite with rank D. For both sides of (6.5) to have equal rank, it follows that there must be $(N - D)$ eigenvalues satisfying $\lambda_i = \sigma^2$. Furthermore, since $\mathbf{A}(\Theta)\mathbf{R}_s\mathbf{A}(\Theta)^H$ is non-negative definite, they must be the smallest eigenvalues. Thus we have

$$\lambda_{D+1} = \lambda_{D+2} = \cdots = \lambda_N = \sigma^2.$$

Form a matrix of those eigenvectors corresponding to the D largest eigenvalues $\mathbf{E}_s = [\mathbf{e}_1 \cdots \mathbf{e}_D]$, and similarly for the eigenvectors corresponding to the $(N - D)$ smallest eigenvalues $\mathbf{E}_n = [\mathbf{e}_{D+1} \cdots \mathbf{e}_N]$. By definition,

$$\mathbf{A}(\Theta)\mathbf{R}_s\mathbf{A}(\Theta)^H\mathbf{E}_n = \mathbf{0}$$

or

$$\mathbf{A}(\Theta)^H\mathbf{E}_n = \mathbf{0}.$$

The second equality follows from noting that, since $\mathbf{A}(\Theta)\mathbf{R}_s$ (which is an $N \times D$ matrix) has rank D, there exists some $D \times N$ matrix $\mathbf{\Lambda}$ with rank D satisfying $\mathbf{\Lambda}\mathbf{A}(\Theta)\mathbf{R}_s = \mathbf{I}$ [99]. Thus, for the true source directions we have

$$\mathbf{a}(\theta_d)^H\mathbf{E}_n\mathbf{E}_n^H\mathbf{a}(\theta_d) = 0, \quad d = 1, \ldots, D. \tag{6.6}$$

That is, we may find the source directions by finding the D values of θ_d for which (6.6) holds. These are the MUSIC source direction estimates.

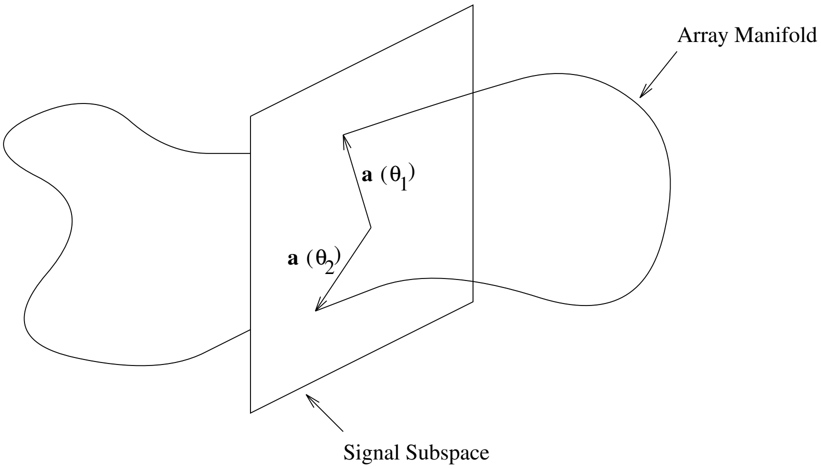


Figure 6.1: Geometrical interpretation of MUSIC for a three sensor, two source example. The source location vectors $\mathbf{a}(\theta_1)$ and $\mathbf{a}(\theta_2)$ define the signal subspace.

The geometric interpretation of MUSIC is as follows. For a fixed array geometry, the array vectors $\mathbf{a}(\theta) \in \mathbb{C}^N$ comprise the *array manifold.* This can be viewed as a rope weaving through $\mathbb{C}^N$ as θ ranges over all angles. In the absence of noise, the observed data vectors $\mathbf{y}(t) = \mathbf{A}(\Theta)\mathbf{s}(t)$ are confined to a D dimensional subspace of $\mathbb{C}^N$, called the *signal subspace.* This signal subspace is spanned by the D columns of $\mathbf{A}(\Theta)$, viz., $\mathbf{a}(\theta_d), d = 1, \ldots, D$. (The columns of $\mathbf{E}_s$ also span the signal subspace.) Thus, once we have obtained D independent data vectors, the signal subspace is known, and the source directions are given by the D points at which the array manifold intersects the signal subspace. This is shown in Fig. 6.1 for a three sensor, two source example. In the presence of noise, we have shown above that $\mathbf{E}_n$ is orthogonal to $\mathbf{A}(\Theta)$, i.e., $\mathbf{E}_n$ is orthogonal to the signal subspace. For this reason, $\mathbf{E}_n$ is referred to as the *noise subspace.* The quantity $\mathbf{a}(\theta)^H\mathbf{E}_n\mathbf{E}_n^H\mathbf{a}(\theta)$ therefore gives the ℓ_2 distance between the array vector $\mathbf{a}(\theta)$ and the signal subspace.

When only noisy measurements of $\mathbf{y}(t) = \mathbf{A}(\theta)\mathbf{s}(t) + \mathbf{v}(t)$ are available, the signal subspace (or alternatively the noise subspace) must be estimated from the measured data. In this case the data covariance matrix is not known exactly. However, it may be consistently estimated by taking measurements over several time samples as follows:

$$\widehat{\mathbf{R}} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{y}(t)\mathbf{y}(t)^H, \tag{6.7}$$

where T is the number of time samples available. Let $\widehat{\mathbf{E}}_s$ and $\widehat{\mathbf{E}}_n$ denotes the sorted eigenvectors of $\widehat{\mathbf{R}}$. Because $\widehat{\mathbf{E}}_s$ and $\widehat{\mathbf{E}}_n$ now only give estimates of the true signal and noise subspaces, it follows that there may be no values of θ for which (6.6) is satisfied exactly. Hence, define the MUSIC null spectrum as

$$\Phi(\theta) = \mathbf{a}(\theta)^H \widehat{\mathbf{E}}_n \widehat{\mathbf{E}}_n^H \mathbf{a}(\theta). \tag{6.8}$$

The MUSIC DOA estimates are given by the D values of θ for which $\Phi(\theta)$ is minimised. Minimising this quantity finds the array vectors which are closest to the estimated signal subspace in the least squares sense. This minimisation is usually done by evaluating $\Phi(\theta)$ at points on a fine grid. For the case of an equally spaced array, (6.8) can be written as a polynomial and the DOAs are then the roots of this polynomial (this is the root-MUSIC algorithm [6]).

Analytical expressions have been derived for the statistical properties of MUSIC. These include the estimate standard deviation [87], bias [107], and probability of resolution for two closely spaced sources [109].

6.3.2 Beamspace MUSIC

Using DOA estimators directly on the sensor data (as in the previous section) is referred to as *elementspace* processing. For an array with N sensors, the eigen-decomposition required by MUSIC (and other eigen-based high resolution algorithms) is $O(N^3)$. Transforming the elementspace data into a reduced dimension *beamspace* prior to applying MUSIC reduces the complexity of the eigen-decomposition to $O(J^3)$, where J is the number of beamformers employed. We will refer to this as BS-MUSIC. Several researchers [8, 30, 58, 106] have noted additional advantages in transforming the elementspace data into a reduced dimension beamspace prior to applying the high-resolution algorithm. These include “lower SNR resolution thresholds, reduced sensitivity to sensor perturbations and deviations from the assumed model, and amenability to parallel implementation” [111]. The narrowband beamspace transformation is briefly reviewed below.

Let $\mathbf{C}$ be an $N \times J, (D < J < N)$ beamspace processing matrix. The array data can be transformed to a reduced dimension beamspace by the transformation

$$\begin{aligned}\mathbf{y}_c(t) &= \mathbf{C}^H \mathbf{y}(t)\\ &= \mathbf{A}_c(\Theta)\mathbf{s}(t) + \mathbf{v}_c(t),\end{aligned}$$

where $\mathbf{A}_c(\Theta) = \mathbf{C}^H \mathbf{A}(\Theta)$ is the $N \times J$ beamspace source location matrix, and $\mathbf{v}_c(t) = \mathbf{C}^H \mathbf{v}(t)$ is the $J \times 1$ beamspace noise vector.

The beamspace data covariance matrix can be formed as

$$\begin{aligned}\mathbf{R}_c &= \mathbf{C}^H\, E\{\mathbf{y}(t)\,\mathbf{y}(t)^H\}\,\mathbf{C} \\ &= \mathbf{A}_c(\Theta)\mathbf{R}_s\mathbf{A}_c(\Theta)^H + \sigma^2\mathbf{R}_v,\end{aligned}$$

where

$$\mathbf{R}_s = E\{\mathbf{s}(t)\mathbf{s}(t)^H\}$$

is the $D \times D$ source covariance matrix, and

$$\sigma^2\mathbf{R}_v = \mathbf{C}^H E\{\mathbf{v}(t)\mathbf{v}(t)^H\}\mathbf{C}$$

is the $J \times J$ beamspace noise covariance matrix. The beamspace data covariance matrix is in a form in which conventional high resolution DOA methods can be applied through the eigen-decomposition of $\{\mathbf{R}_c, \mathbf{R}_v\}$. However, the dimension of these matrices has been reduced to J, thus reducing the order of the eigen-decomposition required by the DOA estimator. It is conventional in narrowband beamspace processing for the columns of $\mathbf{C}$ to be orthonormal. In this case, assuming the noise is uncorrelated from sensor to sensor, the noise covariance matrix reduces to $\mathbf{R}_v = \mathbf{I}$.

The beamformers comprising $\mathbf{C}$ are designed to cover a selected spatial region in which the desired sources are assumed to lie. Xu and Buckley [108] define the gain of the beamforming matrix as

$$g_c(\theta) = \frac{\mathbf{a}(\theta)^H\mathbf{C}\mathbf{C}^H\mathbf{a}(\theta)}{\mathbf{a}(\theta)^H\mathbf{a}(\theta)}.$$

If $\mathbf{C}$ is designed to cover a spatial sector $\Delta\theta$, then $g_c(\theta) \approx 1, \theta \in \Delta\theta$, and $g_c(\theta) \approx 0, \theta \notin \Delta\theta$. This suggests the use of beamformers having uniformly low sidelobes outside the selected spatial sector, for example the Chebyshev beamformer. Other methods for designing the beamforming matrix have been proposed for narrowband operation. Forster and Vezzosi [30] used prolate spheroidal sequences to design a beamforming matrix which maximises the gain over the sector $\Delta\theta$ and attenuates out of sector sources in an L_2 sense. Lee and Wengrovitz [58] derived the beamforming matrix which minimises the resolution threshold for two closely spaced sources. In this chapter we will consider another measure for the specific case of frequency invariant beamspace processing.

An expression for the BS-MUSIC estimate variance is given in [88]. It is shown that the variance of BS-MUSIC is always greater than the variance in elementspace, although in [108] it is shown that it comes close for $\mathbf{C}$ carefully chosen (e.g., using prolate spheroidal sequences [30]).

An expression for the BS-MUSIC estimate bias is given in [108] in which it is shown that for carefully chosen **C**, the bias of BS-MUSIC is less than in elementspace. However, in general, BS-MUSIC estimates have worse bias and variance than in elementspace.

6.3.3 Coherent Signal Subspace Method

The coherent signal subspace (CSS) method [99] is a broadband DOA estimator. It is based on decomposing the received broadband array data into several non-overlapping narrowband frequency bins and finding focusing matrices which transform the data in each bin to a reference frequency bin. The focused data is then combined to form a composite covariance matrix. Conventional narrowband estimators (such as MUSIC) may then be directly applied. This method was proposed as an alternative to incoherent methods (such as [100]) which find DOA estimates for each frequency bin and then statistically average these to form a broadband DOA estimate. CSS methods have been found to have lower SNR resolution and estimate variance than incoherent methods. The CSS method is applied as follows.

The elementspace data is first decomposed into K narrowband components centred at $f_k, k = 1, \dots, K$ (either via a bank of K bandpass filters, or by data segmentation and discrete Fourier transform). As shown in [99] it is possible to find a set of transformation matrices $\mathbf{T}_k$ which satisfy

$$\mathbf{T}_k \, \mathbf{A}(\Theta, f_k) = \mathbf{A}(\Theta, f_0), \quad k = 1, \dots, K,$$

where f_0 is some reference frequency. These transformation matrices effectively focus the DOA matrices of different frequency bins into the single narrowband DOA matrix corresponding to the reference frequency. They are hence referred to as *focusing matrices.*

Applying the K focusing matrices to the respective data vectors gives the following focused array data vectors:

$$\mathbf{T}_k \, \mathbf{y}(t; f_k) = \mathbf{A}(\Theta, f_0) \, \mathbf{s}(t; f_k) + \mathbf{T}_k \, \mathbf{v}(t; f_k).$$

The focused data covariance matrix is given by

$$\begin{aligned} \mathbf{R} &= \sum_{k=1}^{K} \mathbf{T}_k E\{\mathbf{y}(t; f_k) \, \mathbf{y}(t; f_k)^H\} \mathbf{T}_k^H \\ &= \mathbf{A}(\Theta, f_0) \, \mathbf{R}_s \, \mathbf{A}(\Theta, f_0)^H + \sigma^2 \mathbf{R}_v, \end{aligned} \tag{6.9}$$

where

$$\mathbf{R}_s = \sum_{k=1}^{K} E\{\mathbf{s}(t; f_k)\,\mathbf{s}(t; f_k)^H\},$$

$$\mathbf{R}_v = \sum_{k=1}^{K} \mathbf{T}_k\,\mathbf{R}_v(f_k)\,\mathbf{T}_k^H,$$

and $\sigma^2\mathbf{R}_v(f_k) = E\{\mathbf{v}(t; f_k)\mathbf{v}(t; f_k)^H\}$; it is assumed that $\mathbf{R}_v(f_k), k = 1, \ldots, K$, is known. The focused data covariance matrix (6.9) is now in a form in which conventional narrowband DOA estimators may be applied.

Several methods of forming the focusing matrices have been suggested. For the simple case where all true source directions are in the neighbourhood of a single angle β, Wang and Kaveh [99] originally proposed

$$\mathbf{T}_k = \mathrm{diag}\{a_1(\beta, f_0)/a_1(\beta, f_k), \ldots, a_N(\beta, f_0)/a_N(\beta, f_k)\},$$

where $a_n(\theta, f)$ is the nth element of $\mathbf{a}(\theta, f)$. Hung and Kaveh [46] showed that the class of unitary focusing matrices reduce SNR detection and resolution thresholds, and proposed the rotational signal subspace focusing matrix which is a member of this class. In both cases, *preliminary estimates* of the source directions are required to form the focusing matrices. It has been shown [90] that if the focusing is not accurate, the DOA estimates are asymptotically biased. This demonstrates one of the major disadvantages with coherent signal subspace methods: preliminary DOA estimates are required for effective focusing.

The other main disadvantage of CSS methods is that as source bandwidths increase, asymptotic peak bias can increase [10]. This may be explained as follows. The key idea of CSS is that by decomposing the source bandwidth into narrowband frequency bins, the data in each frequency bin may be modelled as a single frequency. For a single-frequency source, the source location vector $\mathbf{a}(\theta, f)$ is rank one, i.e, it spans a one dimensional subspace in the N dimensional observation space. However, any source with non-zero bandwidth will not be restricted to this one dimensional subspace. This means that for CSS, the broadband observations must be decomposed into very narrow frequency bins in order for the focusing to be accurate over the bandwidth of each frequency bin. As the source bandwidth increases, the number of bins becomes prohibitively large. Noting this problem with CSS, Buckley and Griffiths [10] proposed the broadband signal subspace spatial spectrum (BASS-ALE) estimator. Their method uses a low rank (rather than a rank one) source representation for broadband sources and is based on the eigen-structure of a broadband spatial/temporal covariance matrix. The BASS-ALE estimator does not suffer from the spectral content sensitivity and resulting asymptotic peak bias of CSS. However, it does require an increase in computation relative to CSS (see [10] for details).

6.3.4 Beamspace Coherent Signal Subspace Processing

One CSS method which removes the requirement for preliminary DOA estimates was recently proposed by Lee [59] (other methods include [31, 55]). Lee's approach is briefly reviewed to motivate the method proposed in this chapter.

Single Beamformer

Initially, consider a single set of beamforming coefficients, $\mathbf{w}_0$, applied at a frequency f_0. The beamformer response is

$$r(\theta, f_0) = \mathbf{w}_0^H \, \mathbf{a}(\theta, f_0).$$

Lee used least squares optimisation to find the set of beamformer weights, $\mathbf{w}_k, k = 1, \dots, K$, which produce a set of beampatterns $r(\theta, f_k), k = 1, \dots, K$, which are closest to $r(\theta, f_0)$ in the weighted least squares sense, i.e.,

$$\min_{\mathbf{w}_k} \int_0^\pi \varphi(\theta) \left| \mathbf{w}_0^H \mathbf{a}(\theta, f_0) - \mathbf{w}_k^H \mathbf{a}(\theta, f_k) \right|^2 d\theta, \quad k = 1, \dots, K, \tag{6.10}$$

where $\varphi(\theta)$ is a weighting function. The beamformers obtained by the solution of (6.10) satisfy a "frequency invariance" property, i.e., $\mathbf{w}_k^H \mathbf{a}(\theta, f_k) \approx \mathbf{w}_0^H \mathbf{a}(\theta, f_0), k = 1, \dots, K$.

Multiple Beamformers

Now, assume J beamformers are formed in each frequency bin according to this least squares optimisation. Denote the beamforming matrices as $\mathbf{W}_k = [\mathbf{w}_{k1} | \cdots | \mathbf{w}_{kJ}]$. Because of the frequency invariance property, these beamformers will perform the role of the focusing matrices in CSS, i.e.,

$$\mathbf{W}_k^H \, \mathbf{A}(\Theta, f_k) \approx \mathbf{W}_0^H \, \mathbf{A}(\Theta, f_0), \quad k = 1, \dots, K. \tag{6.11}$$

Assume the array data is transformed to a reduced dimension beamspace by applying these beamformers. The transformed beamspace data is given by

$$\begin{aligned} \mathbf{y}_b(t; f_k) &= \mathbf{W}_k^H \, \mathbf{y}(t; f_k) \\ &= \mathbf{A}_b(\Theta, f_k) \, \mathbf{s}(t; f_k) + \mathbf{v}_b(t; f_k), \end{aligned}$$

where

$$\mathbf{A}_b(\Theta, f_k) = \mathbf{W}_k^H \, \mathbf{A}(\Theta, f_k),$$

is the $J \times D$ beamspace source direction matrix, and

$$\mathbf{v}_b(t; f_k) = \mathbf{W}_k^H \, \mathbf{v}(t; f_k)$$

is the $J \times 1$ beamspace noise vector. From (6.11) we have

$$\mathbf{A}_b(\Theta, f_k) \approx \mathbf{A}_b(\Theta, f_0), \quad k = 1, \dots, K.$$

Hence, the frequency invariant beampatterns collapse the observed broadband data into the narrowband frequency bin corresponding to f_0. The focused beamspace data covariance matrix is given by

$$\begin{aligned}\mathbf{R} &= \sum_{k=1}^{K} \mathbf{W}_k^H \, E\{\mathbf{y}(t; f_k) \, \mathbf{y}(t; f_k)^H\} \, \mathbf{W}_k \\ &= \mathbf{A}_b(\Theta, f_0) \, \mathbf{R}_s \, \mathbf{A}_b(\Theta, f_0)^H + \sigma^2 \mathbf{R}_v,\end{aligned}$$

where

$$\begin{aligned}\mathbf{R}_s &= \sum_{k=1}^{K} E\{\mathbf{s}(t; f_k) \, \mathbf{s}(t; f_k)^H\}, \\ \sigma^2 \mathbf{R}_v &= \sum_{k=1}^{K} \mathbf{W}_k^H \, E\{\mathbf{v}(t; f_k) \, \mathbf{v}(t; f_k)^H\} \, \mathbf{W}_k.\end{aligned}$$

The focused beamspace data covariance matrix is in a form in which conventional narrowband DOA estimators may be applied. Furthermore, the dimension of $\mathbf{R}$ has been reduced to J, thus reducing the order of the eigen-decomposition required by high-resolution DOA estimators.

Although Lee's method does not require preliminary DOA estimates to form the focusing matrices (assuming all sources fall within the spatial sector covered by the beamformers), it still suffers from the rank one modelling problem outlined in the previous section. Another disadvantage of Lee's method is the computation required by the "unstructured" optimisation of (6.10), i.e, for J beamformers and K frequency bins, Lee's method requires solution of JK least squares optimisation problems.

6.4 Proposed Broadband DOA Estimator

The broadband DOA method proposed in this chapter is motivated by Lee's approach of using frequency invariant beamformers to focus the broadband data. However, it differs in two significant ways. First, rather than solving (6.10) for each of the J beamformers in each of the K frequency bins, we make use of the simply parameterised multirate beamforming structure proposed in Chapter 3. Second, because this beamforming structure produces beampatterns which are frequency invariant over a continuum of frequencies (not just at a set of discrete frequencies as in Lee's method), accurate focusing is achieved without resorting to the added computational burden of a narrowband frequency decomposition approach. This allows the method to be used over wide bandwidths without a corresponding increase in complexity.

Single Frequency Invariant Beamformer

Recall that the response of the frequency invariant beamformer (FIB) proposed in Chapter 2 is

$$r(\theta, f) = \alpha f \sum_{n=1}^{N} g_n H_n(f) e^{j2\pi f \tau_n(\theta)}, \tag{6.12}$$

where $H_n(f)$ is the primary filter response on the nth sensor (total of N sensors), g_n is a spatial weighting term to account for the possibly nonuniform sensor spacings, and α is a normalisation constant.

An important requirement of the FIB is that the primary filters must satisfy a dilation property

$$H_n(f) = H_{\mathrm{ref}}\left(\frac{x_n}{x_{\mathrm{ref}}} f\right),$$

where $H_{\mathrm{ref}}(f)$ is the required primary filter response at some location x_{ref}. In Chapter 3 it was shown that this dilation property could be achieved by multirate filtering. In particular, if $h_{\mathrm{ref}}[k]$ is a set of $L = (2M + 1)$ filter coefficients which produces a desired primary filter response $H_{\mathrm{ref}}(f)$ at a reference location x_{ref} with a sampling period T, then the response of the nth primary filter is

$$H_n(f) = \sum_{k=-M}^{M} h_{\mathrm{ref}}[k] e^{-j2\pi f T_n k} = \mathbf{h}^H \mathbf{d}_n(f), \tag{6.13}$$

where $T_n = Tx_n/x_{\text{ref}}$ is the sampling rate of the nth sensor, and

$$\mathbf{d}_n(f) = \left[e^{-j2\pi fT_n(-M)}, e^{-j2\pi fT_n(-M+1)}, \dots , e^{-j2\pi fT_n M}\right]^T$$

is the L vector of sampling delays. The reference filter has been made non-causal for notational convenience; causality is easily restored.

To formulate the beamspace DOA estimation problem, let

$$\mathbf{\Gamma}(f) = \alpha f[g_1\mathbf{d}_1(f), \dots , g_N\mathbf{d}_N(f)] \tag{6.14}$$

be an $L \times N$ matrix. The response of a single FIB (6.12) may now be written

$$r(\theta, f) = \mathbf{h}^H\mathbf{\Gamma}(\theta, f)\mathbf{a}(\theta, f).$$

Multiple Frequency Invariant Beamformers

Assume that J ($D < J \leq N$) independent FI beamformers are formed using J sets of reference primary filter coefficients. These beamformers are designed to cover a spatial sector $\Delta\theta$ in which the sources are assumed to lie. Typically, the J beampatterns are simply the same beampattern steered to J different angles within $\Delta\theta$.[3]

The responses of these beamformers are

$$\begin{bmatrix} r_1(\theta, f) \\ \vdots \\ r_J(\theta, f) \end{bmatrix} = \begin{bmatrix} \mathbf{h}_1^H \\ \vdots \\ \mathbf{h}_J^H \end{bmatrix} \mathbf{\Gamma}(f)\,\mathbf{a}(\theta, f)$$

or

$$\mathbf{r}(\theta, f) = \mathbf{H}^H\mathbf{\Gamma}(f)\,\mathbf{a}(\theta, f) \tag{6.15}$$

where $\mathbf{H}$ is the $L \times J$ beampattern coefficient matrix.

Let $\mathbf{C}(f) = \mathbf{\Gamma}(f)^H\mathbf{H}$, and denote the $J \times 1$ vector of stacked beamformer outputs at time t as $\mathbf{y}_c(t; f)$. This will be referred to as the *frequency invariant beamspace* (FIBS) data vector. Returning to the original problem stated in §6.2, we consider the beamformer outputs for $D < J$ farfield signals arriving from directions $\Theta = [\theta_1, \dots , \theta_D]$. We assume

[3] Recall from §3.4 that coarse beam steering of a FIB can be effected through circular rotation of a single set of reference primary filter coefficients. This method of beam steering may be applied in forming the J beamformers.

that $\Theta \in \Delta\theta$, i.e., all sources of interest are within the spatial sector covered by the beamformers, and interfering out-of-sector sources are attenuated by the beamformers such that they do not contribute to the signal subspace.[4]

The beamformer outputs are

$$\begin{aligned}\mathbf{y}_c(t;f) &= \mathbf{C}(f)^H\mathbf{A}(\Theta,f)\mathbf{s}(t;f) + \mathbf{C}(f)^H\mathbf{v}(t;f)\\ &= \mathbf{A}_c(\Theta,f)\mathbf{s}(t;f) + \mathbf{v}_c(t;f),\end{aligned}$$

where

$$\mathbf{A}_c(\Theta,f) = \mathbf{C}(f)^H\mathbf{A}(\Theta,f)$$

is the $J \times D$ FIBS source direction matrix, and

$$\mathbf{v}_c(t;f) = \mathbf{C}(f)^H\mathbf{v}(t;f)$$

is the $J \times 1$ FIBS noise vector. Relating this to conventional beamspace processing we have $\mathbf{y}_c(t;f) = \mathbf{C}(f)^H\mathbf{y}(t;f)$, where $\mathbf{y}(t;f)$ is the $N \times 1$ broadband elementspace data vector.

Because of the frequency invariant property of the beamformers, the FIBS source direction matrix is approximately constant for all frequencies within the design band, i.e., $\mathbf{A}_c(\Theta,f) \approx \mathbf{A}_c(\Theta), \forall f \in [f_L, f_U]$. Hence, the broadband source directions are completely characterised by a single beamspace DOA matrix $\mathbf{A}_c(\Theta)$. Note that although $\mathbf{C}(f)$ is necessarily a function of frequency to impose the frequency invariant beampattern property, the actual beamforming coefficients $\mathbf{H}$ are fixed as a function of frequency; there is no hidden frequency decomposition in our method.

Assuming the source signals and the noise are uncorrelated, the $J \times J$ FIBS data covariance matrix is

$$\mathbf{R}_c(f) = \mathbf{A}_c(\Theta)\mathbf{R}_s(f)\mathbf{A}_c(\Theta)^H + \sigma^2\mathbf{R}_v(f), \tag{6.16}$$

where

$$\mathbf{R}_s(f) = E\{\mathbf{s}(t;f)\mathbf{s}(t;f)^H\}$$

is the $D \times D$ source covariance matrix (assumed full rank),

$$\sigma^2\mathbf{R}_v(f) = E\{\mathbf{v}_c(t;f)\mathbf{v}_c(t;f)^H\}$$

[4] This latter assumption is somewhat idealistic but simplifies the analysis.

is the $J \times J$ FIBS noise covariance matrix, and $\mathbf{R}_v(f)$ is assumed known $\forall f \in [f_L, f_U]$.

A broadband FIBS data covariance matrix can be formed as

$$\mathbf{R}_c = \int_{f_L}^{f_U} \mathbf{R}_c(f)\ df = \mathbf{A}_c(\Theta)\mathbf{R}_s\mathbf{A}_c(\Theta)^H + \sigma^2\mathbf{R}_v, \tag{6.17}$$

where

$$\mathbf{R}_s = \int_{f_L}^{f_U} \mathbf{R}_s(f)\ df, \tag{6.18}$$

is the broadband source covariance matrix, and

$$\mathbf{R}_v = \int_{f_L}^{f_U} \mathbf{R}_v(f)\ df, \tag{6.19}$$

is the broadband FIBS noise covariance matrix.

The broadband FIBS data covariance matrix (6.17) is now in a form in which conventional narrowband MUSIC may be applied. Denote the eigen-decomposition of $\mathbf{R}_c$ as

$$\mathbf{R}_c\mathbf{E} = \mathbf{\Lambda}\mathbf{R}_v\mathbf{E},$$

where $\mathbf{\Lambda}$ is a diagonal matrix of sorted eigenvalues, $\mathbf{E} = [\mathbf{E}_s|\mathbf{E}_v]$ are the corresponding eigenvectors, $\mathbf{E}_s$ are the eigenvectors corresponding to the largest D eigenvalues, and $\mathbf{E}_v$ are the eigenvectors corresponding to the smallest $J - D$ eigenvalues. The ranges of $\mathbf{E}_s$ and $\mathbf{E}_n$ define the signal and noise subspaces respectively. The source directions are now given by the D peak positions of the following FIBS-MUSIC spatial spectrum:

$$\Phi(\theta) = \frac{\mathbf{a}_c(\theta)^H\mathbf{a}_c(\theta)}{\mathbf{a}_c(\theta)^H\mathbf{E}_v\mathbf{E}_v^H\mathbf{a}_c(\theta)} \tag{6.20}$$

where $\mathbf{a}_c(\theta)$ are the focused FIBS source location vectors, and the numerator is included for normalisation.

It is important to note the difference between the processing of our proposed method and that of [59]. For Lee's method, elementspace data is collected, split into frequency bins, and in each frequency bin the data is transformed to beamspace through multiplication by a beamspace processing matrix. In our proposed method, frequency invariant beamformers are formed, and the focused data is collected *directly in beamspace.*

6.4.1 Practical Considerations

Estimation of the Broadband Covariance Matrices

It is conventional in narrowband (or frequency decomposition) beamspace DOA methods to choose the beamformers such that the columns of $\mathbf{C}$ are orthogonal. In this way, assuming the noise is uncorrelated from sensor to sensor, the noise covariance matrix is $\sigma^2\mathbf{I}$, where σ^2 is the (unknown) noise power. However, this is not possible for FIBS-MUSIC because a single set of beamforming coefficients define the beamforming matrix over the entire frequency band.[5] This is not a problem however, since once the frequency invariant beamformers have been designed, the broadband noise covariance can be easily calculated. Specifically, for the case where the noise is spectrally white and uncorrelated from sensor to sensor, (6.19) becomes

$$\mathbf{R}_v = \mathbf{H}^H \mathbf{\Lambda} \mathbf{H}, \tag{6.21}$$

where

$$\mathbf{\Lambda} = \int_{f_L}^{f_U} \mathbf{\Gamma}(f)\mathbf{\Gamma}(f)^H \; df.$$

Note from the definition of $\mathbf{\Gamma}(f)$ (6.14) that $\mathbf{\Lambda}$ is fixed for a given frequency band and array geometry. Hence, it can be calculated off-line and does not add to the computational complexity of the FIBS method.

The integral for $\mathbf{R}_c$ (6.17) can be formed in a straightforward fashion as follows. Let

$$\mathbf{Y}_c = [\mathbf{y}_c(1), \cdots, \mathbf{y}_c(T)]$$

denote the $J \times T$ data observation matrix obtained by collecting FIBS data vectors over T time samples—the ith row of $\mathbf{Y}_c$ is the time series obtained at the output of the ith beamformer. (We have dropped the frequency dependence for notational convenience.) Form the conventional data covariance matrix (as in narrowband processing),

$$\widehat{\mathbf{R}}_c = \frac{1}{T}\sum_{k=1}^{T} \mathbf{y}_c(t)\mathbf{y}_c(t)^H = \frac{1}{T}\mathbf{Y}_c\mathbf{Y}_c^H. \tag{6.22}$$

Now, $\widehat{\mathbf{R}}_c$ contains wideband source information covering all frequency components present in $\mathbf{Y}_c$. So long as $\mathbf{Y}_c$ does not contain frequency components outside the band $[f_L, f_U]$, $\widehat{\mathbf{R}}_c$

[5] It is possible to choose the beamformers such that $\mathbf{C}(f_0)^H\mathbf{C}(f_0) = \mathbf{I}$ for a given frequency $f_0 \in [f_L, f_U]$ if desired.

will be focused by the FIBS processing and will contain the source direction information. Because of the structure of the frequency invariant beamformer, it is straightforward to include a bandpass filter in the secondary filter of each beamformer, ensuring that $\mathbf{Y}_c$ contains no frequency components outside the design band.

Hence, the FIBS data covariance matrix may be formed in the time domain; *no frequency decomposition is required.* This represents a significant computational advantage of FIBS over other broadband DOA methods (such as [59, 99]).

FIBS Source Location Vectors

In calculating the FIBS-MUSIC spatial spectrum (6.20), focused FIBS source location vectors $\mathbf{a}_c(\theta)$ are required. These location vectors are defined as $\mathbf{a}_c(\theta) \approx \mathbf{C}(f)^H \mathbf{a}(\theta, f), \forall f \in [f_L, f_U]$. In practice, these location vectors can be computed at any frequency in the design band. Hence, we define the FIBS source location vectors as

$$\mathbf{a}_c(\theta) \triangleq \mathbf{C}(f_0)^H \mathbf{a}(\theta, f_0), \tag{6.23}$$

for some $f_0 \in [f_L, f_U]$. It is important to note that this is *not* the same as the focusing frequency used in coherent subspace methods. In CSS, all the source location vectors at different frequencies are transformed to the focusing frequency and the data covariance matrix calculated from the focused data. In our method, all FIBS source location vectors are virtually identical for all frequencies, no explicit focusing calculations are performed and f_0 plays a far lesser role.[6] Thus, any frequency in the design band may be used as the frequency at which the FIBS-MUSIC spatial spectrum is calculated. Typically f_0 would be chosen to be the mid-band frequency of the design bandwidth.

6.4.2 Summary of Proposed Method

The DOA method proposed in this chapter is outlined below.

1. Design J frequency invariant beamformers (as outlined in Chapter 3) to cover the spatial region $\Delta\theta$.

2. Calculate the broadband FIBS noise covariance matrix estimate $\widehat{\mathbf{R}}_v$ (6.21). This can be done off-line (assuming the noise covariance does not change over the observation time).

[6] This is also true of Lee's method, since the beamspace source location vectors in all frequency bins are virtually identical by nature of the least squares optimisation procedure employed (see §6.3.4).

3. Collect data from each of the J beamformers over the observation time period and calculate the broadband FIBS source covariance matrix estimate $\widehat{\mathbf{R}}_c$ (6.22).

4. Find the source locations from $\widehat{\mathbf{R}}_c$ and $\widehat{\mathbf{R}}_v$ using a conventional DOA estimator. Without loss of generality, we have only considered the MUSIC estimator [80] (another candidate high-resolution estimator is ESPRIT [78]).

6.5 Simulations

We now present results of computer simulations to empirically compare FIBS with CSS, which is most often used as the "benchmark" broadband DOA estimator. Asymptotic bias, estimate bias and variance, and the ability to resolve closely spaced sources were considered.

In each case, a 21 element linear equally-spaced array, with an inter-sensor spacing of $c/(2f_U)$, was used. Two farfield sources located at $\Theta = [9°, 12°]$ were modelled. The design frequency band was 150–250 Hz, with a sampling rate of $f_s = 6$ kHz. Each broadband signal was generated by combining 100 narrowband signals, uniformly distributed across the bandwidth. For CSS, a filter bank was used to decompose the received data into 21 frequency bins within the design band (giving a frequency bin width of 5 Hz), the focusing frequency was $f_0 = 200$ Hz, and the focusing angle was 10.5°.

6.5.1 Design of FIBS processor

Using an aperture size of $P = 12$ half-wavelengths, $J = 7$ Chebyshev 35 dB frequency invariant beamformers were designed to cover the spatial region $[-30°, 30°]$, using the beampattern sampling method described in Chapter 3. The resulting beampatterns, calculated at the centre frequency $f_0 = 200$ Hz, are shown in Fig. 6.2. Note that the beampatterns are close to the desired Chebyshev 35 dB pattern. However, because we employed the simplest technique for obtaining the reference primary filter coefficients (i.e., beampattern sampling), the desired 35 dB sidelobes are not achieved exactly.

The most important consideration in FIBS-MUSIC, as far as the beamformer design is concerned, is for successful beamspace focusing. This means that the frequency variation of $\mathbf{C}(f)^H \mathbf{a}(\theta, f), f \in [f_L, f_U]$, must be minimised over the spatial region of interest; if this is not the case biasing of the DOA estimates will occur. With this in mind, we define the following normalised focusing error spectrum (similar to that given by Lee [59]) as a

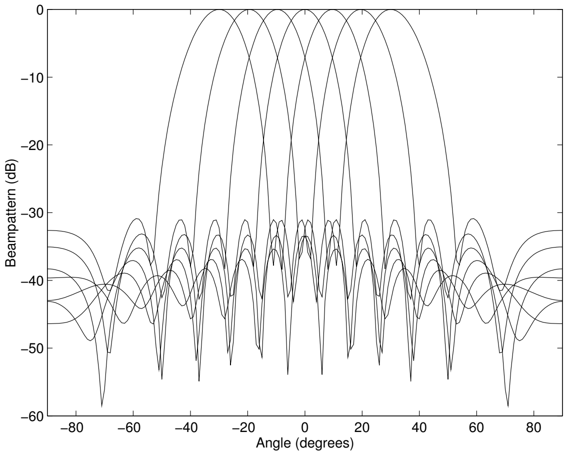


Figure 6.2: Superposition of the seven beamformers used in the simulations, calculated at the mid-band frequency $f_0 = 200$ Hz.

means of quantifying the frequency variation:

$$\epsilon(\theta)^2 = \frac{\int_{f_L}^{f_U} \|\mathbf{r}_d(\theta) - \mathbf{C}(f)^H \mathbf{a}(\theta, f)\|^2 \, df}{(f_U - f_L)\|\mathbf{r}_d(\theta)\|^2}, \tag{6.24}$$

where $\mathbf{r}_d(\theta)$ is a $J \times 1$ vector of desired FI responses, and $\|\cdot\|$ denotes the vector 2-norm. Using the beamformers shown in Fig. 6.2, the normalised focusing error spectrum shown in Fig. 6.3 was calculated from (6.24). Note the flat valley over $[-30°, 30°]$, which confirms that effective focusing is performed within the selected spatial region. To further establish the frequency invariance of the beamformers used in this example, we calculated the response of the centre beamformer at 50 frequencies over the frequency band. The resulting beampatterns are shown in Fig. 6.4. Again, this verifies that the designed beamformers are largely frequency invariant over the entire bandwidth.

6.5.2 Asymptotic Bias

It has been demonstrated in [10], and proven in [90], that CSS is asymptotically biased if either: (i) the angle of arrival is not aligned with the focusing angle; or (ii) the centre frequency of the source bandwidth does not equal the focusing frequency. By using the mean of the true source directions as the focusing angle for CSS, we have ensured virtually perfect angle focusing. In this section we consider the asymptotic performance—as the SNR and/or number of snapshots goes to infinity—of CSS and FIBS as the source bandwidth is varied. In all cases we assume that the focusing frequency for CSS is unchanged at $f_0 = 200$ Hz. We also use $f_0 = 200$ Hz to calculate the FIBS source location vectors

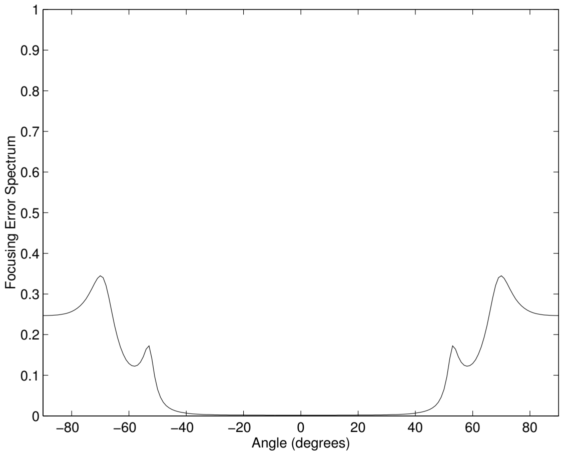
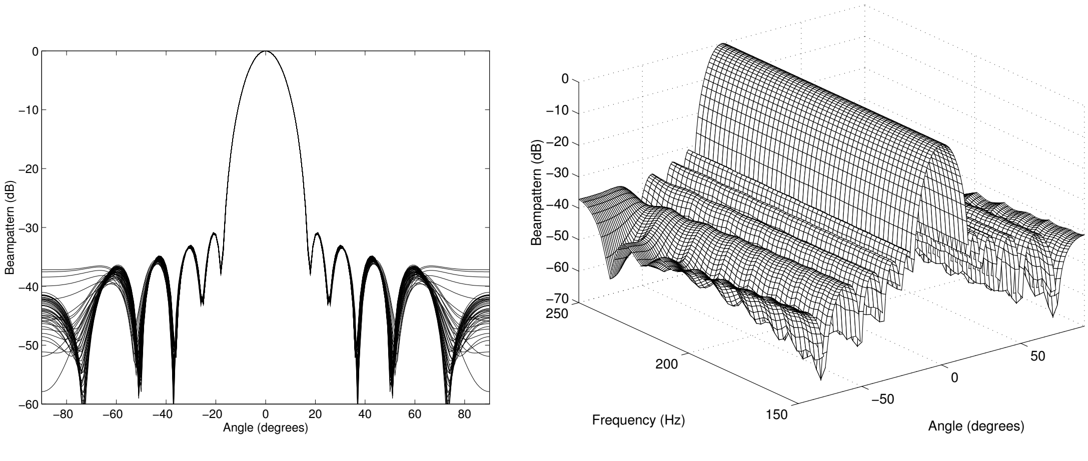


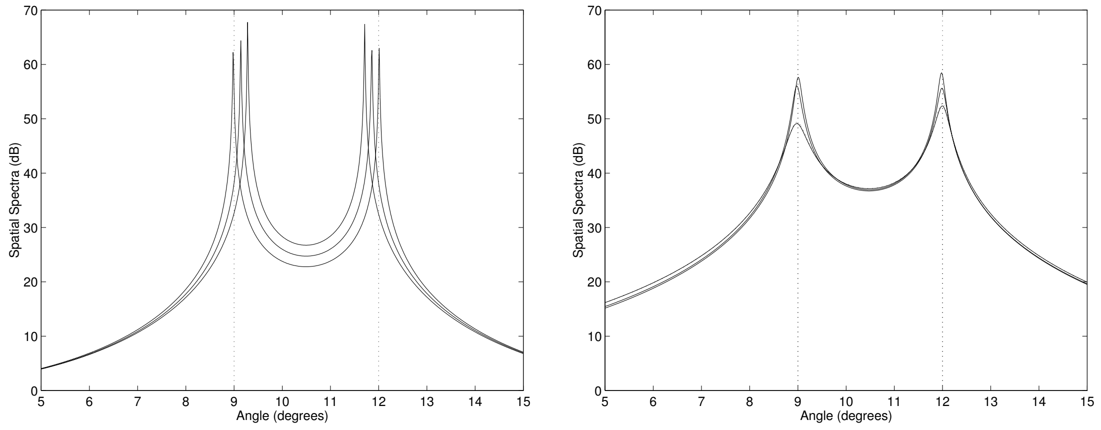
Figure 6.3: Normalised focusing error spectrum for the example frequency invariant beamformers shown in Fig. 6.2.



(a)

(b)

Figure 6.4: Beampattern of centre beamformer calculated at 50 frequencies uniformly distributed across the design band: (a) superposition of all 50 beampatterns, (b) variation of beampattern with frequency.



(a)

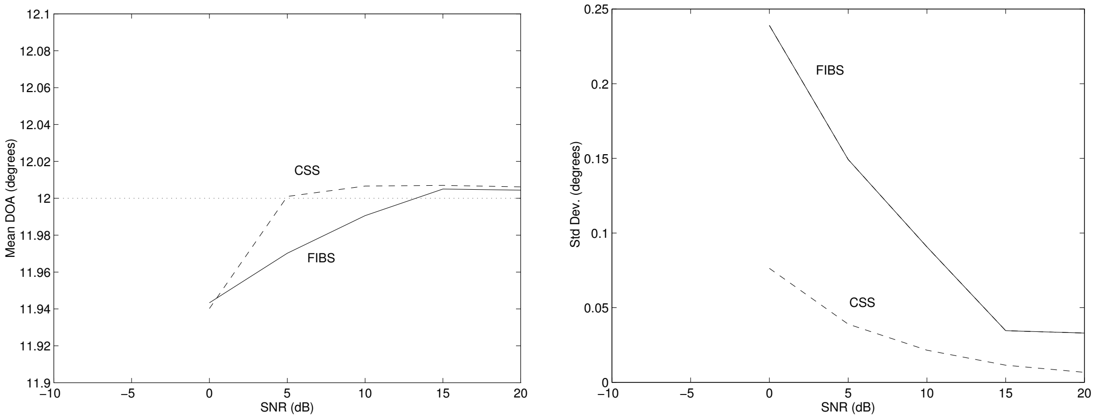
(b)

Figure 6.5: Asymptotic spatial spectra from (a) CSS and (b) FIBS, with a fractional bandwidth of 50 per cent and centre frequencies of 200, 180 and 160 Hz.

(6.23). We consider 50 per cent fractional bandwidths centred on 200, 180 and 160 Hz. Figure 6.5 compares the asymptotic bias of CSS and FIBS. These results demonstrate the asymptotic bias exhibited by CSS (as noted in [10, 90]). The results also indicate that the FIBS estimator is asymptotically unbiased, regardless of the source spectral content. Note that the low peaks in Fig. 6.5(b) correspond to the case where the centre frequency is 160 Hz (with a bandwidth of 120–200 Hz). In this case the bandwidth barely intersected the FIBS source location vector frequency (i.e., 200 Hz) used to calculate the FIBS-MUSIC spatial spectrum.

6.5.3 Mean and Variance of Estimates

Next, we performed Monte Carlo simulations to compare the mean and variance of the DOA estimates obtained by CSS and FIBS. The sources covered the entire bandwidth of 150–250 Hz. Thus, according to [90], CSS will be asymptotically unbiased (since the focusing frequency is the mid-band frequency of the source bandwidth). For each DOA estimate, 30 time samples were collected and used to calculate the covariance matrices for each method. The results for 50 independent trials for the source located at 12° are shown in Fig. 6.6. These results indicate that the accuracy of the FIBS estimator as a function of SNR is comparable to that of CSS. In comparing the standard deviations of the two estimators, it is clear that FIBS has greater variance than CSS. This is to be expected,



(a) (b)

Figure 6.6: Comparison of the performance of CSS and FIBS in estimating the 12° source: (a) estimate bias, and (b) standard deviation for several SNR values.

since FIBS is operating in beamspace and CSS is operating in elementspace: as shown in [88], beamspace methods exhibit greater estimate variance than elementspace methods.

6.5.4 Resolution Threshold

Finally, the probability of resolving two closely spaced sources was considered using the same data employed above to compare estimate mean and variance. Results are shown in Fig. 6.7. The two sources were considered resolved if the values of the MUSIC spectrum at $\theta_1 = 9^\circ$ and $\theta_2 = 12^\circ$ were both greater than the value at $\theta = 10.5^\circ$. Note that CSS performed about 5 dB better than FIBS which exhibited a resolution threshold of 5 dB.

6.6 Conclusions

In this chapter we have considered the application of frequency invariant beamforming to the problem of broadband direction of arrival estimation. In particular it was shown that the frequency invariant beamformer could perform the role of the focusing matrices in the coherent signal subspace approach, without requiring decomposition of the received data into narrowband frequency bins. The method proposed in this chapter, referred to as FIBS, was motivated by Lee's approach [59]. However, it differs in two significant ways. First, rather than solving a least-squares optimisation problem for each of the J beamformers in each of K frequency bins, we made use of the simply parameterised beamforming structure

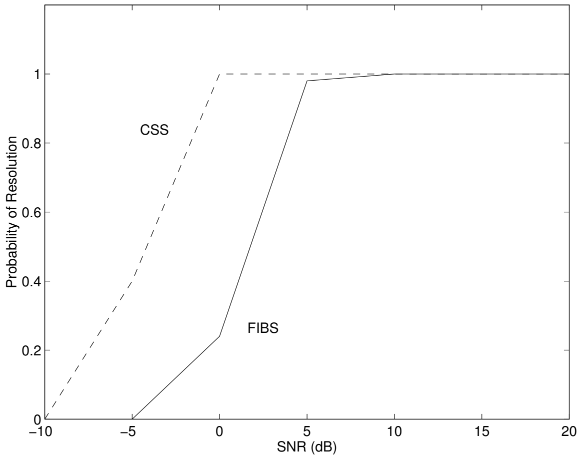


Figure 6.7: Comparison of CSS and FIBS in resolving two closely spaced sources for several SNR values.

developed in Chapter 3. Second, because the beampatterns produced by the frequency invariant beamformer proposed in Chapter 2 are frequency invariant over a continuum of frequencies (not just at a set of discrete frequencies as in [59]), accurate focusing is achieved without resorting to a narrowband frequency decomposition approach. This allows the method to be used over wide bandwidths without the added computational complexity of frequency decomposition. Simulation results indicated that the accuracy of the FIBS estimates were comparable to those of the coherent signal subspace (CSS) method, although FIBS was not able to resolve closely spaced sources as well as CSS. However, FIBS was shown to be more robust to source bandwidth modelling errors.

To summarise the performance: (i) compared to CSS, FIBS does not require preliminary DOA estimates and is not affected by source spectral modelling errors; (ii) compared to Lee's method, FIBS offers a far simpler method of designing the beamformers; and (iii) compared to CSS methods in general (including Lee's), FIBS has the advantage of avoiding the computational burden of frequency decomposition.

In closing, it should be noted that we have simply presented the straightforward application of the frequency invariant beamformer to the broadband DOA problem. We have not attempted to optimise the beamformers specifically for the DOA application. It has previously been shown [11, 58, 106] that performance can be improved by using carefully chosen beamformers. Thus, improved performance of FIBS would no doubt be gained through a more judicious choice of the frequency invariant beamformers.

Chapter 7

Conclusions and Future Research

7.1 Conclusions

THIS thesis has been motivated by the problem of *broadband frequency invariant beamforming*, that is, broadband spatial filtering in which there is little variation in spatial response over a wide bandwidth. In general, there are certain intrinsic structural properties which any frequency invariant beamformer (FIB) should possess. These properties reveal themselves more readily when one considers a continuously distributed sensor (rather than an array of spatially separated sensors). Beamforming can then be treated as the problem of approximating the response of a continuous aperture by a finite sensor array.

This philosophy of using a theoretical continuous sensor as a beamformer design tool can be readily applied to the more general problem of producing a broadband angle-versus-frequency beampattern specification. The continuous sensor approach allows simple relationships to be derived between a broadband beampattern specification and the beamformer parameters, specifically the beamformer filter bank coefficients. Such insight is rarely afforded by numerical optimisation techniques which are often used.

This summarises one of the main themes of the thesis: formulating the broadband beamforming problem in such a way that structural properties are revealed which can simplify beamformer implementation. We now discuss the specific findings of the thesis in detail.

The inherent structure of a class of frequency invariant beamformers was derived in Chapter 2. Specifically, for a linear array this structure consists of: (i) a beam shaping filter on each sensor (referred to as the *primary filter*—all primary filter frequency responses are related by a dilation property); and (ii) all sensors share a common normalisation filter (the *secondary filter*). Similar simple structures exist for certain two and three dimensional array geometries. In terms of minimising the number of sensors in a linear array while avoiding spatial aliasing, it was shown that a specific nonuniformly spaced geometry is optimal. Finally, the FIB was shown to be a member of a class of parameterised beamformers whose beampatterns can be controlled in a continuous manner from a conventional single-frequency design (in which the beampattern varies directly with frequency) to a frequency invariant design. Although this generalisation is primarily of theoretical interest, it could be used in a practical setting in which the number of sensors is limited and yet a single frequency design gives unacceptable results.

The theory of Chapter 2 is directly applicable to continuous-time implementation only. In considering a discrete-time implementation, Chapter 3 showed that there is a single set of parameters (the *reference coefficients*) which defines the FI beam shape over the entire frequency range; the size of this set is independent of the number of sensors and the operating bandwidth. The primary filters were shown to derive straight from the reference coefficients. In turn, it was shown that the reference coefficients could be obtained by sampling the desired FI beampattern function at a specific set of points. Thus, given a beampattern specification, the primary filter coefficients can be obtained immediately (without regard for the necessary aperture distribution or primary filter frequency responses). This provides further evidence of the inherent structure and simplicity of the FIB. An example adaptive FIB algorithm was developed based on the well-known Frost beamformer [32]. It was also noted that coarse beam steering can be effected by simply modifying the reference coefficients—no explicit time delay filter is required on each sensor (as is the case in conventional delay beam steering). Finally, a microphone array was used to empirically demonstrate that the FIB theory could be successfully implemented.

Noting that a FIB exhibits a small amount of frequency variation, especially in the sidelobes and pattern nulls, additional constraints must be imposed to enforce a pattern null whose position does not vary over the operating bandwidth. Thus, in Chapter 4 we considered the problem of producing a broadband null in a FIB. Formulating the requirement for a broadband null as a constraint on the primary filter impulse responses showed that it is possible to impose an exact broadband null (i.e., a pattern null which is present over *all* frequencies, not just within the design band) while simultaneously minimising the effect on some desired broadband FI beampattern. However, exact broadband nulling requires strict constraints on the sensor locations and sample rate. In the case of slight positioning errors (in the order of a fraction of a wavelength), standard analysis [82] revealed that a reasonably deep null is still produced by the method. Given the structural constraints of the exact null solution, an alternative formulation was then considered, namely, placing multiple frequency zeros in the null direction response. This is analogous to the phased array method of placing multiple spatial zeros in the region of an interferer [84, 85]. Based on this formulation we derived an expression for the number of frequency constraints required to produce a broadband null of a given depth in an FI beampattern. This expression revealed that the relative null depth is only dependent on “design” parameters (such as sensor locations, sample rate, design bandwidth and null direction) and does not depend on the “implementation” parameters, namely the filter coefficients. This means that for an adaptive FIB it is possible to determine *a priori* the number of constraints required to impose a broadband null of a given depth in a given direction.

Chapter 5 dealt with the problem of frequency invariant beamforming for nearfield point sources—for such sources the non-planar shape of the impinging wavefronts must be accounted for. There were two major outcomes from this study. First, we formulated a radial beampattern transformation based on the solution to the spherical wave equation. This transformation consisted of two equations: (i) the *analysis equation* which produces a set of coefficients which completely characterise a given beampattern for a source at any radial distance (measured relative to some reference point, usually the array centre); and (ii) the *synthesis equation* which reconstructs a beampattern at a given radial distance using the analysis coefficients. The significance of the transformation is that, given a beampattern for a source at a radial distance r_1, it is possible to determine the corresponding beampattern for a source at any radial distance r_2 (which does not intersect with the physical array). Of practical importance is the transformation to an infinite radial distance, since this represents the transformation of a nearfield beampattern (for which no satisfying broadband design method exists) to an equivalent farfield beampattern (for which a broadband design is readily derived). In general, a nearfield frequency *invariant* beampattern will transform to a farfield frequency *dependent* beampattern, and vice versa. Thus, implementation of a nearfield FIB requires the design of a farfield beamformer having an arbitrary broadband beampattern (specified over both angle and frequency). This was referred to as the *general broadband beamformer* (GBB). For a uniformly spaced array, it has been shown [43] that a two dimensional Fourier transform relationship exists between a GBB response and the sensor filter coefficients. However, a solution for a nonuniformly spaced array is lacking. This motivated the second part of the chapter. Applying the philosophy of the FIB theory—using a continuous sensor to formulate a broadband beamforming solution—allowed us to formulate an explicit expression for the filter coefficients of a nonuniformly spaced array to produce a specified GBB response. Coupled with the radial beampattern transformation, the GBB can be applied to a wide class of beamforming problems including both nearfield and farfield beamforming to achieve an arbitrary broadband beampattern, specified over both angle and frequency.

Chapter 6 considered a second application of the FIB theory, this time to the problem of estimating the directions of arrival (DOAs) of broadband signals. This was motivated by Lee's [59] beamspace formulation of the *coherent signal subspace* (CSS) method [99]. Our proposed method (referred to as the *frequency invariant beamspace* (FIBS) method) differs from Lee's in that: (i) frequency decomposition of the received data is not required; and (ii) whereas Lee's method solves a multi-parameter least squares optimisation problem to form the beamformers, we make use of the simply parameterised frequency invariant beamformer developed in Chapters 2 and 3. In both respects, FIBS offers computational advantages. Simulation results indicated that, relative to CSS (which is most often used as the benchmark broadband DOA estimator), FIBS: (i) provides DOA estimates of comparable accuracy; (ii) is less effective at resolving two closely spaced sensors; and (iii) is asymptotically unbiased when the actual signal bandwidth differs from the modelled bandwidth.

7.2 Future Research Directions

Some future research projects are now outlined. It should be noted that we have chosen a representative set of problems, constrained to be close in nature to the material found in this thesis.

1. **Error Bounds.** For the theoretical linear continuous sensor considered in Chapter 2, the resulting spatial response does not vary with frequency. This frequency invariant (FI) response is:

$$r(\theta) = f \int_{\mathbb{R}} G(xf) e^{\jmath 2\pi f x c^{-1} \sin\theta} \, dx. \tag{7.1}$$

The response of the frequency invariant beamformer (FIB) which implements this ideal response is:

$$\hat{r}(\theta) = f \sum_{n=1}^{N} g_n G(x_n f) e^{\jmath 2\pi f x_n c^{-1} \sin\theta}. \tag{7.2}$$

However, for a finite number of spatially separated sensors, the implementation (7.2) can only approximate the ideal FI response (7.1). It is natural to consider error bounds on this approximation, thereby bounding the frequency variation of the beampattern. There are two factors upon which the approximation error depends: one is the choice of spatial weighting terms g_n, and the other is the aperture distribution $G(\cdot)$ corresponding to the desired beampattern. Approximation results from numerical integration theory [19] should be applicable if the spatial weighting terms are fixed to correspond to the trapezoidal integration method.

2. **Frequency Variation versus Desired Beampattern.** From the nature of the approximation used to implement the frequency invariant beamformer, it is clear that the amount of frequency variation in the beampattern depends on the extent to which the corresponding aperture distribution may be accurately approximated over a wide bandwidth. Consider the trapezoidal integration method. The sensor locations define the subintervals over which the trapezoidal integration is performed. As frequency varies, the aperture distribution dilates and the sizes of the subintervals change. Thus, to minimise the beampattern frequency variation, the aperture distribution function should be chosen such that it is well approximated by trapezoidal integration as the sizes of the subintervals vary. This suggests the use of beampatterns having smoothly varying aperture functions. A more precise definition of what constitutes a well-behaved aperture distribution is required. This would allow us to identify those beampatterns which, when used in the FIB, would produce little variation over a wide frequency band.

3. **Adaptive Frequency Invariant Beamforming.** The performance of the FIB in an adaptive environment deserves further attention. A simple example of a linear constrained minimum variance (LCMV) FIB algorithm was presented in Chapter 3. This was a reasonably straightforward modification of the Frost beamformer [32]. A starting point for the evaluation of the adaptive FIB would be to compare the LCMV FIB (which is essentially a partially adaptive broadband beamformer) with the Frost beamformer (which uses all of the available degrees of freedom). This would rigorously establish the situations in which the FIB is to be preferred to fully adaptive broadband beamformers.

4. **Broadband Nulls.** Several questions arising from the development of the broadband exact null in Chapter 4 are briefly outlined below.

    1. What constraints are imposed to form an exact broadband null in the nearfield (e.g., for spherical wavefronts)? It should be possible to place the sensors such that they satisfy the integer delay property (see §4.4). However, avoiding spatial aliasing will further constrain the sensor locations, and it may be that it is only possible to enforce a broadband null for certain source positions. This also raises the question of what is the constraint on the sensor locations to avoid spatial aliasing for nearfield sources?

    2. Can an exact null be imposed if fractional delay FIR filters [56] are used to implement beam steering?

3. Can interpolation beamforming [75] be used to relax the constraints on the sensor locations while imposing an exact broadband null?

4. Analyse the effectiveness of the broadband null for slight source location errors, i.e., if an exact broadband null is formed at angle θ_0, how much suppression is there for a source at $\theta_0 + \epsilon$?

5. Nearfield Beamforming on a General Manifold. The beampattern transformation developed in Chapter 5 is only applicable to beampatterns measured on the surface of a sphere. This transformation could be generalised in a number of ways:

1. Other coordinate systems could be considered in which the manifold is a level surface (such as a cylindrical coordinate system). This is reasonably straightforward, requiring the solution of the wave equation in the new coordinate system in a similar manner to that outlined in Chapter 5.

2. The nearfield beampattern could be specified on a non-level surface (such as on a peanut-shaped surface). This means that the beampattern is specified for varying radial distance, and the analysis equations would involve a triple integral (over radial distance, elevation angle and azimuth angle) in the spirit of the double integral for the analysis equations in Chapter 5.

3. The nearfield beampattern could be specified in a volume rather than on a surface. This represents a substantial extension of the proposed transformation, and would have practical use in medical imaging, for example.

6. Broadband Beamformers for DOA Estimation. In formulating the *frequency invariant beamspace* (FIBS) DOA estimation method proposed in Chapter 6, there was no criterion for choosing the beampatterns of the beamformers other than they should cover a selected spatial region and attenuate sources outside this region. However, it has been shown by several authors [11, 58, 106] that the performance of beamspace DOA methods is dependent on the beampatterns used. The performance of FIBS could undoubtedly be improved by investigating the relationship between beampattern properties and DOA estimator performance. A basis for this study is the narrowband beamspace analysis of [58].

Appendix A

Microphone Array Testing System

THIS appendix contains a copy of a report which describes the design and implementation of the microphone array testing system used in Chapter 3. The primary aim of this appendix is to provide a complete engineering report describing the system in detail.

A.1 Introduction

This report summarises the design and implementation of a microphone array testing system (MATS). The primary purpose of MATS is to provide a demonstration and analysis system for algorithms developed for broadband beamforming, specifically the implementation of the frequency invariant beamformer developed in Chapter 3.

A microphone array can be used to improve the quality of acquired speech compared to a single microphone. The array is used to form a *directive* microphone which can be electronically steered towards the desired source (usually a single voice). Typical applications include speech acquisition for teleconferencing, hands-free telephones, and voice-only data entry. Effective reception of speech signals in these situations is traditionally hampered by reverberations and external noise sources.

Flexibility has been one of the major concerns in designing the system. A general broadband beamforming structure is implemented in which each of the input channels is filtered, all channels are summed, and this summed output is then filtered to provide a single output. The actual filter coefficients are loaded at run-time. This allows virtually any fixed broadband beamformer to be implemented. The system hardware could be readily applied to adaptive beamforming, however this would require substantial changes to the system software.

The report is structured as follows. Section 2 describes the system hardware, including circuit diagrams. An overview of the system software is given in Section 3, with conclusions and possible extensions to MATS outlined in Section 4. Finally, we give details of (i) system function calls for the data acquisition and DSP cards, (ii) test results of the system hardware, and (iii) source code for the system software.

The following people have contributed to the development of this system, and they are gratefully acknowledged: Robert Edwards and Marshall Shepard for their advice in the design of the Microphone Pre-amplification/Anti-aliasing Board, Martin Stonebridge for his help in designing the microphone array support structure, and Martin Buss for developing much of the system software.

A.2 Hardware

The system hardware consists of a Microphone Pre-amplification/Anti-aliasing Board (designed and built specifically for this project) which provides preamplification and anti-aliasing filtering for up to 16 microphone channels (referred to as the MPAB), a National Instruments AT-MIO-16F-5 Multifunction I/O Board which performs data acquisition (referred to as the NIDAQ), an Ariel DSP-96 Floating-Point Processor Board which performs the beamforming (referred to as the DSP), and an IBM-compatible PC which controls data transfer between the data acquisition card and the DSP. The interconnection of the various components is shown in Fig. A.1.

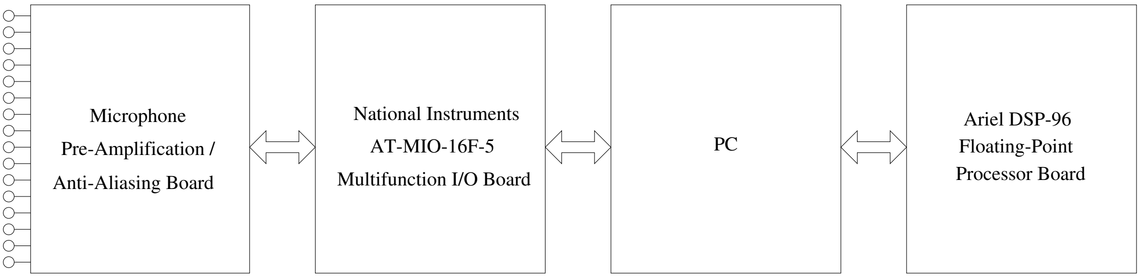


Figure A.1: Block diagram of hardware.

A.2.1 Microphone Pre-amplification/Anti-aliasing Board

The MPAB performs three specific tasks. These are:

1. amplify the microphone signal on each channel by up to 60 dB;
2. low-pass filter the amplified microphone signal on each channel to the Nyquist rate; and
3. provide reconstruction filtering for the output signal. (This is required since the National Instruments AT-MIO-16F-5 Multifunction I/O Board does not provide a reconstruction filter on its analog outputs. See §A.2.2 for more details.)

The operations performed by the MPAB for a single channel are shown in block diagram form in Fig. A.2, in which the figure numbers refer to the circuit diagrams for each of the indicated sections.

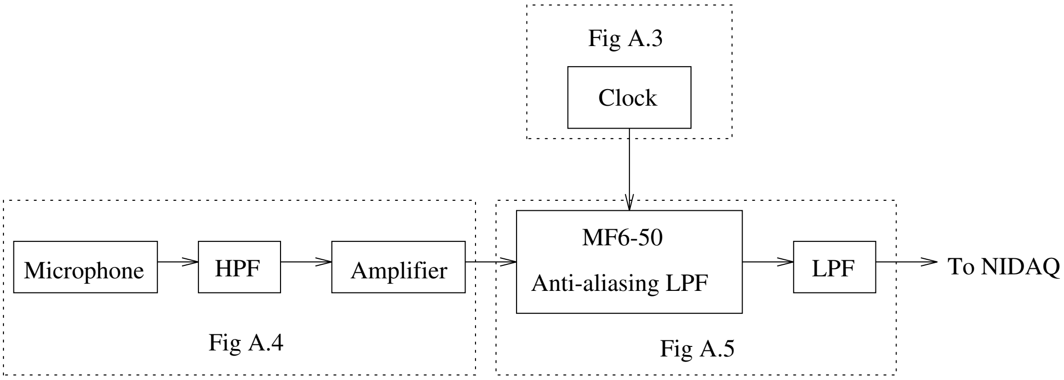


Figure A.2: Block diagram of operations performed on a single channel. The figure numbers refer to the circuit diagram of the corresponding section.

The MPAB can be divided into five distinct sections: clock, microphone biasing and amplification, anti-aliasing filters, power supplies, and reconstruction filter. Three separate boards comprise the MPAB. These are the main board (containing the clock, channel amplification, anti-aliasing filters and a ±5 volt power supply), the reconstruction board (containing a ±12 volt power supply, reconstruction filter, and a jumper header block), and the microphone power supply board (containing a 2.5 volt supply for the microphones). All three boards are housed in the MPAB box.

Clock

The clock determines the cutoff frequency of the anti-aliasing filters. Two clock frequencies are available, 400 kHz and 200 kHz, corresponding to anti-aliasing filter cutoff frequencies of 8 kHz and 4 kHz respectively. The desired cutoff frequency is selected with the toggle switch[1] on the front of the MPAB box.

The clock circuitry is shown in Fig. A.3. A crystal oscillator is used to produce a 4 MHz square wave, which is then divided by either 10 or 20 depending on the mode line of the GAL. See Section A.7 for the GAL source code.

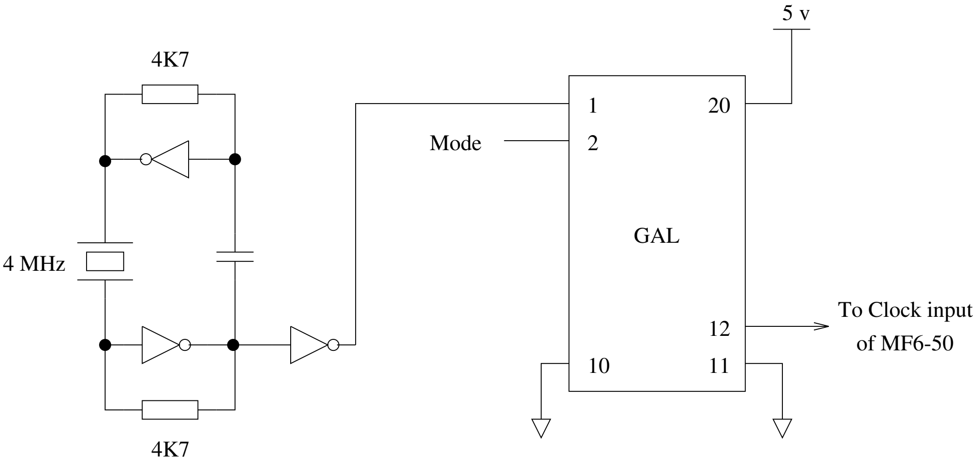


Figure A.3: Clock circuitry.

Microphone Biasing and Amplification

Each channel has a two stage amplifier based on a Texas Instruments TL054 Enhanced JFET Precision Quad Operational Amplifier, i.e., there are two channels per TL054. The first stage provides fixed amplification of 30 dB, and the second stage has tunable gain of up to 30 dB. The microphone biasing and amplification circuit (for one channel) is shown in Fig. A.4. Note that the biasing capacitor and the input resistor of the first amplifier stage form a high-pass filter with a cutoff frequency of 106 Hz.

Anti-aliasing Filters

The anti-aliasing filters are based on a National Semiconductor MF6-50 Switched Capacitor Lowpass Filter which provides a 6th order Butterworth low-pass filter as well as two general purpose op amps. An external clock is used to set the cutoff of the low-pass filter.

[1] There is a jumper on the main MPAB board which can be used to set the cutoff frequency by software rather than by the toggle switch. If the jumper is changed to its other position, the mode line of the GAL will be connected to the ADIO0 line of the NIDAQ, thus allowing the mode to be software selected.

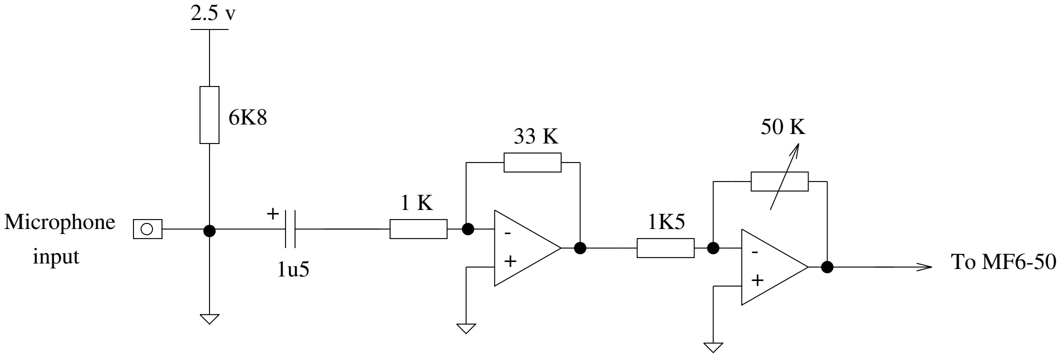


Figure A.4: Microphone biasing and amplification.

The ratio of the clock frequency to the low-pass cutoff frequency is internally set to 50 to 1. The first internal op amp is configured as an integrater and used to keep the average DC filter output level at ground (as described in the data sheets). The second op amp is configured as a two pole Butterworth active low-pass filter and used to remove the clock signal from the filter output. This active filter was designed according to [53] with a cutoff frequency of 10 kHz. The entire circuit diagram is shown in Fig. A.5.

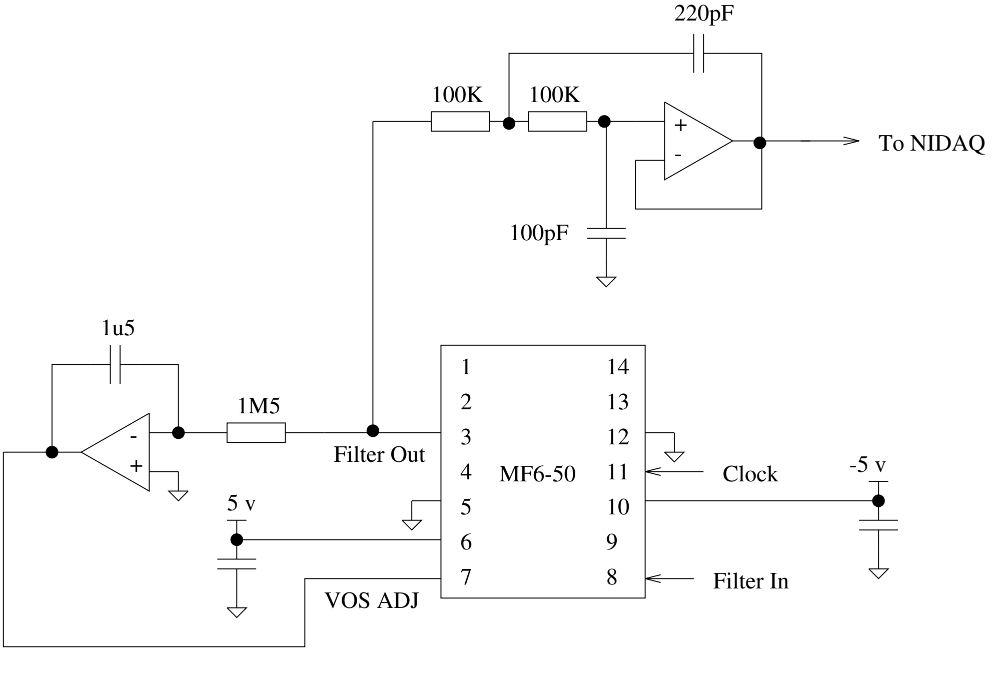


Figure A.5: Circuit diagram for anti-aliasing filter, including offset voltage adjustment and low-pass filter output stage.

Power Supplies

Three distinct power supplies are provided on the MPAB. Two ±5 volt supplies based on the National Semiconductor NMH0505 voltage converter are provided on the main board for the channel amplification and anti-aliasing filters respectively. A ±12 volt supply based on the National Semiconductor NMH0512 voltage converter is provided on the reconstruction board for the reconstruction filters. The circuits for these supplies are straightforward, and can easily be found in the respective data sheets.

A 2.5 volt supply is provided on the microphone power supply board. This is used to provide the phantom supply required for the condenser microphones, and must be low noise. It is based on the Analog Devices REF-03 Precision Voltage Supply. The circuit for this supply is shown in Fig. A.6.

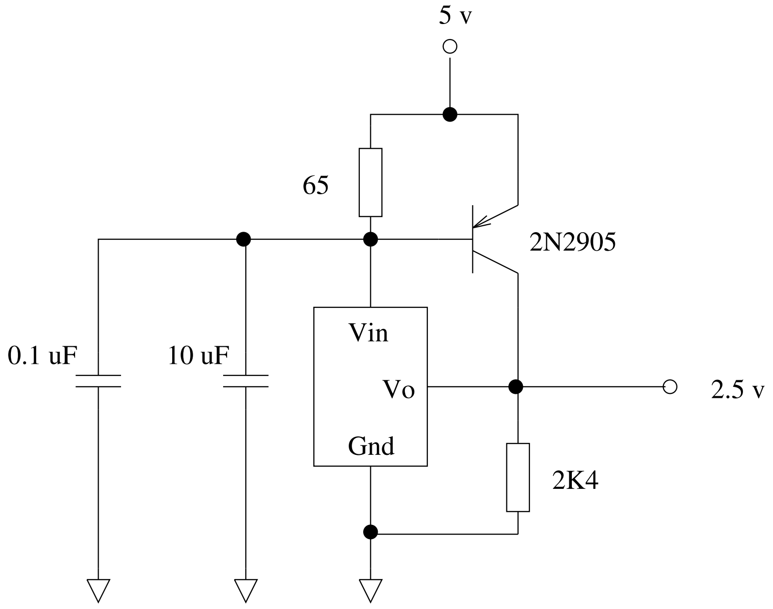


Figure A.6: Microphone power supply.

An external 5 volt supply is required to provide power for the MPAB. This should be plugged into the terminal block on the front of the MPAB box.

NOTE: *No short circuit or over-voltage protection is provided on the MPAB. Applying the wrong voltage to the terminal block may destroy the internal voltage converters.*

Reconstruction Filters

Since the NIDAQ only provides a zero order hold on the DAC outputs, a reconstruction filter is required. This reconstruction filter is a four pole Butterworth active low-pass filter based on a Texas Instruments TL054 Enhanced JFET Precision Quad Operational Amplifier.

A 4 kHz low-pass reconstruction filter[2] (designed according to [53]) is provided on the NIDAQ DAC0_OUT line.[3] The circuit diagram is shown in Fig. A.7. Currently, the beamformer output is set to DAC0_OUT. This is easily changed by modifying the function DA_setup in da.c (see §A.3.2 or Section A.7).

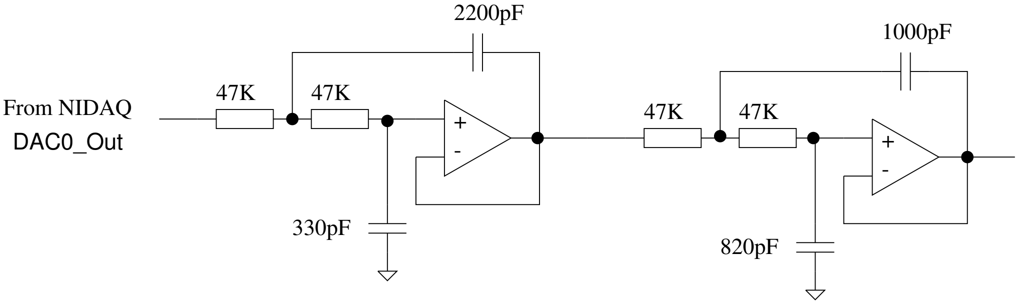


Figure A.7: Reconstruction low-pass filter with a cutoff frequency of 4 kHz.

Header Block

Included on the reconstruction board of the MPAB is a header block with six jumper positions. A top view of the header block is shown in Fig. A.8, in which the A jumpers are on the MPAB side and the B jumpers are on the NIDAQ side of the board.

For normal operation, the A jumpers of J1 to J5 should be connected to the corresponding B jumpers.

NOTE: *The A and B jumpers of J6 should never be connected.*[4]

[2] Strictly speaking, the reconstruction should not be a simple low-pass filter, but should compensate for the frequency response of the zero-order hold. See [70, p.125] for details.

[3] The reconstruction board provides space for a second reconstruction filter to be attached to the NIDAQ DAC1_OUT line.

[4] Originally it was intended that the MPAB should get its power from the NIDAQ. However, since power is now obtained from an external supply, J6 should not be connected. Connecting J6 may damage the NIDAQ or PC.

ACH0	DAC0 OUT	ACH8	DAC1 OUT	GND	5v
J1B	J2B	J3B	J4B	J5B	J6B
J1A	J2A	J3A	J4A	J5A	J6A
Ch1	Out1	Ch2	Out2	GND	5v

Figure A.8: Header block on the reconstruction board.

To test channel 1 of the MPAB without requiring the NIDAQ, J1A (channel 1) can be connected to J2A (output 1). Similarly, J3A can be connected to either J2A (output 1) or J4A (output 2). These jumper connections feed the corresponding input channels directly to the output, thus allowing testing of these two channels without requiring the NIDAQ or PC.

NOTE: *No short circuit protection is provided on the MPAB, so the J5 and J6 jumpers should not be cross-connected. This will short out either the MPAB or the NIDAQ and may cause permanent damage.*

Testing

Each component of the MPAB has been extensively tested. Full test results are given in Section A.6.

A.2.2 National Instruments AT-MIO-16F-5 Multifunction I/O Board

The National Instruments AT-MIO-16F-5 Multifunction I/O Board (hereafter referred to as the NIDAQ) performs the tasks of:

1. data acquisition;
2. ADC (analog-to-digital conversion) for the 16 input channels; and
3. DAC (digital-to-analog conversion) for the two output channels. (Note that at present only one of these output channels is used.)

Full details of the NIDAQ may be found in [65]; a brief overview is given below.

Overview

The NIDAQ has 16 analog input channels which can be software configured as either 16 single-ended channels (numbered 0 to 15) or eight differential channels (numbered 0 to 7). These channels are multiplexed through a single programmable gain stage and a 12-bit ADC. Analog inputs can be software configured as either 0 to 10 volts (unipolar mode) or -5 to 5 volts (bi-polar mode). ADC operations can be initiated through software or by an external trigger. The input mode, input range and polarity are selected through a call to AI_Configure, and the trigger is selected through a call to DAQ_Config. These functions calls are described in Section A.5.

The NIDAQ can perform either single-channel data acquisition or multiple channel scanned data acquisitions: we are only concerned with the multiple-channel scanned mode. In this mode, the board scans a set of analog input channels in a user-defined scan sequence. An ADC operation is performed once every sample interval. The channels are scanned at the beginning of a scan interval; by setting the scan interval to 0, the channels are scanned repeatedly as fast as possible. The channels to be scanned, the scan sequence, and the individual channel gains are set through a call to SCAN_Setup. The scan timing is set through a call to SCAN_Start (which also initiates the scanning operation).

By using double-buffered data acquisition, a user-specified circular buffer is continuously filled. (Double-buffered data acquisition is selected through a call to DAQ_DB_Config). The circular buffer is divided into two half-buffers: data can be transferred out of one half buffer while the other is being filled with new data. This allows continuous data acquisition to occur without overwriting previous data.[5] The double-buffering is completely controlled by the NIDAQ. The user is only required to transfer data out of one of the half-buffers when it is ready (with a call to DAQ_DB_HalfReady to check when the data is ready, and then a call to DAQ_DB_Transfer to transfer the data). The NIDAQ internally keeps track of which half-buffer is being filled, and transfers out the appropriate half-buffer automatically. The array to use for the circular buffer and its size are specified by a call to SCAN_Start.

Two analog outputs are available on the NIDAQ. Each contains a 12-bit DAC and can be hardware jumper configured for unipolar or bipolar operation. (The factory setting is for bipolar operation with two's compliment being written to the DAC. Refer to [65, p.2-12] if this is changed.) An on-board voltage reference of 10 volts is currently used (giving

[5] A common error which occurs when the sample rate is too high is that data is overwritten before it can be transferred out of the half-buffer.

an output voltage range of -10 to 10 volts), but an external voltage reference signal may also be used (refer to [65]).

A.2.3 Ariel DSP-96 Floating-Point Processor Board

The Ariel DSP-96 Floating-Point Processor Board (hereafter referred to as the DSP) is a general purpose digital signal processing board which performs the actual task of beamforming. It features one 33.3 MHz DSP96002 processor (capable of 50 Mflops), two banks of 16 Kwords static memory and 256 Kwords of dynamic memory. See [3] for a full description of the DSP.

A.3 Software

Several tools are provided by the MATS software. These include full beamforming operations, calibration of the microphones, and simple data acquisition in which the sampled input data is dumped to a file for off-line processing. The calibration and data acquisition routines should be reasonably self-explanatory once the principle beamforming software is understood. For this reason, only the beamforming software is discussed in detail. Full source code (containing extensive comments) can be found in Section A.7.

A.3.1 Overview of Beamforming Software

The beamforming software is divided into two separate directories: one for 8 (or less) channels and the other for 16 (or less) channels. This separation was done to minimise the amount of software modification required when changing the number of channels.

The operations performed by the beamforming software are shown in Fig. A.9. The 16 input channels are sampled by the NIDAQ, and the sampled data is automatically stored sequentially in the appropriate half-buffer in the PC. When one of the half-buffers is full, it is transfered to the DSP for beamforming. The beamformed data is then returned to the PC and copied to the appropriate output half-buffer where it is automatically transferred to the NIDAQ for digital-to-analog conversion.

Configuration and Data Files

The actual number of channels to be used and the number of filter coefficients in each channel filter are stored in a configuration file, “filcof.cfg”. The actual channel filter

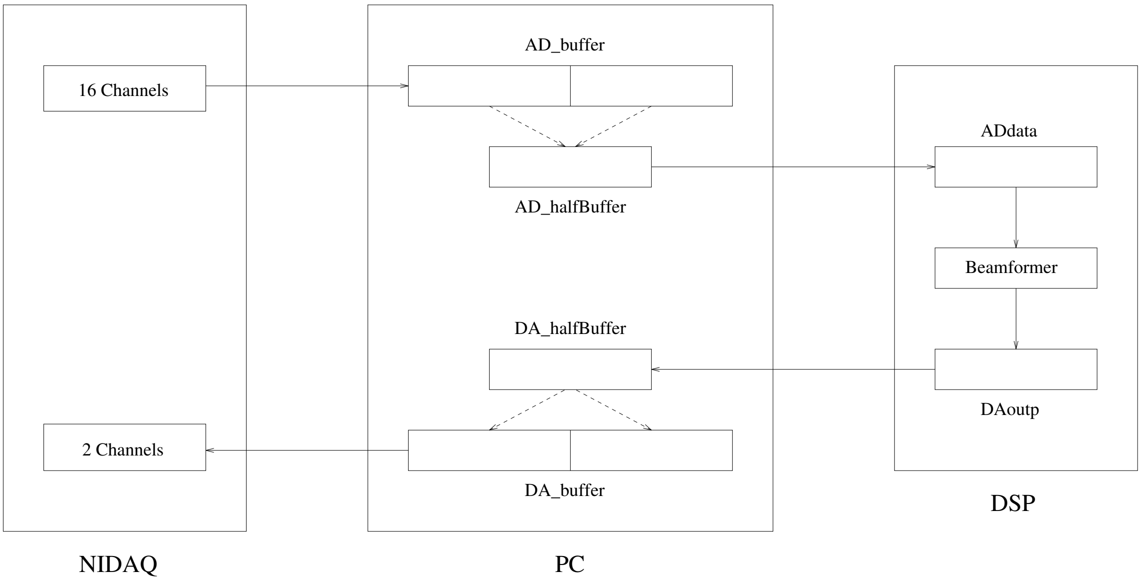


Figure A.9: Block diagram of beamforming software.

coefficients are stored in a data file, "filcof.dat". These files are read at run-time. Thus, changes to the filter coefficients only requires modification of these files; the source code does not need to be recompiled.

An example configuration file is shown in Fig. A.10(a). The first number (top line) defines the number of channels, N (either 8 or 16). The next N numbers define the number of coefficients for each of the filters on the N channels. There is no requirement to have the same number of coefficients on each channel. Note that, because of the structure of the DSP software, each channel must have at least 2 coefficients, even if the channel has only zero or one coefficient. The last number in the configuration file defines the number of filter coefficients in the secondary filter.

An example data file corresponding to this configuration file is shown in Fig. A.10(b). A blank line has been placed between each set of filter coefficients for clarity only: these blank lines would not be included in a real data file. Note that there are actually only five channels included in this beamformer. The last three channels are excluded since they have zeros as their filter coefficients.

A.3.2 Details of Beamforming Software

Each of the software components is briefly described below. Full descriptions of the system calls noted in this section may be found in Section A.5.

adda8.h and adda16.h

These header files define the high level routines to control the data acquisition process. They contain several important constants, including AD_Buffer_size and AD_halfBuffer_size (which are used for the input channels), DA_Buffer_size and DA_halfBuffer_size (which are used for the output channel), and numChans, ADtimebase, WFtimebase, ADsampleInt and update_int. It is *highly* recommended that these constants *are not changed*, as this will also require changes to the corresponding DSP code. This is the main reason that there is separate (but practically identical) software for 8 and 16 channel operation. The (approximate)[6] sample rate is stored in the constant sampleRate. This constant may be safely modified if required.

ad.c

This file contains the implementation of the input acquisition routines defined in adda8.c or adda16.c. The routines are briefly summarised below (full details can be found in the comments of the source code in Section A.7).

AD_setup: Configures the NIDAQ for data acquisition with calls to Get_DA_Brds_Info, AI_Mux_Config, DAQ_Config, DAQ_Trigger_Config and DAQ_DB_Config. Then sets the channel scanning order[7] and the channel gains, allocates memory for the buffers, and finally initialises the NIDAQ with a call to SCAN_Setup.

AD_start: Starts the NIDAQ scanning with a call to SCAN_start.

AD_check_ready: Checks if one of the half-buffers is full with a call to DAQ_DB_HalfReady.

AD_get_half: Transfers the data from one of the half-buffers to the DSP with a call to DAQ_DB_Transfer. This function should be called once AD_check_ready returns true.

AD_stop: Stops the data acquisition with a call to DAQ_Clear and then releases the memory used by the buffers.

da.c

This file contains the implementation of the D/A operations of the NIDAQ board. Full details of the routines can be found in the source code (see Section A.7).

[6] The reason that this is only an approximate sample rate is explained by the fact that the sample rate is actually determined by the value of ADsampleInt, which must be an integer. The actual sample rate is given by $1/(\text{ADsampleInt} \times \text{numChans} \times 200 \times 10^{-9})$, and is displayed at run-time.

[7] The scanning order is not sequential due to the hardware implementation of the MPAB

DA_setup: Initialises the NIDAQ D/A outputs with a call to Init_DA_brds. Then allocates memory for the data buffers, configures the D/A operation (WFM_DB_Config), sets up the buffers (WFM_Load), and sets the output rate (WFM_ClockRate).

DA_start: Starts the D/A process with a call to WFM_Group_Control.

DA_check_ready: Checks if one of the half-buffers is full with a call to WFM_DB_Halfready.

DA_put_half: Transfers the data from one of the half-buffers to the NIDAQ with a call to WFM_DB_Transfer. This function should be called once DA_check_ready returns true.

DA_stop: Stops the D/A operation (WFM_Group_Control) and frees the memory used by the buffers.

pc8.c **and** pc16.c

These are the main programs which control the data acquisition process and transfer data to and from the DSP (which performs the actual beamforming operation). The files are practically identical, so only pc8.c is considered in detail.

pc.8 contains several important constants used in the DSP memory map including adr_addata (the location of the acquired channel data), adr_fil_config_record (the location of the filter configuration record), fil_config_len (the size of the filter configuration record), adr_debbuf (the location of a debugging area), adr_filcof (the location of the filter coefficients), and adr_daoutp (the location of the beamformer output data). These constants *should not be modified* as this would require a change to the DSP memory map.

There are several high-level functions defined by pc.8.

merror: Displays a relevant error message if a DSP call returns an error.

DSP_init: Defines the DSP configuration (with a call to GetConfigFile)and resets the DSP and loads the assembler program (InitializeDsp).

debug_filcon: Uploads the current filter configuration record to the PC (with UploadDspToMemory) and displays it.

test_chan: Clears the current filter coefficients, and sets up a single channel for data acquisition.

load_filcof_cfg: Loads the filter configuration from the configuration file.

load_filcofs: Loads the filter coefficients from the filter data file, and optionally writes them to a binary file for inputting next time.

menu_control: Provides a menu of options. This subroutine can be easily modified to include further functionality. Currently it includes options to test a single channel, reload the filter coefficients from the data file, and to quit.

A block diagram of the main program is shown in Fig. A.11.

dp.asm

This is the assembler program which performs the actual beamforming operation. There are several important constants which define the DSP memory map: again, these constants should not be changed as this would require a major modification to the rest of the code.

The structure of the beamformer implemented by the DSP is shown in Fig. A.12. This is the required structure for a broadband frequency invariant beamformer. However, it is a very general structure which can be used for almost any broadband (or narrowband) beamformer. The design of the filters for a frequency invariant beamformer is described in Chapter 3. For full details of the DSP software, refer to the commented source code in Section A.7.

A.4 Conclusions and Possible Extensions

As stated in the introduction, the aim of MATS was to provide a flexible system for testing beamforming algorithms. Since MATS has already been successfully used to experimentally verify some beamforming algorithms,[8] it may be concluded that the system has proved successful in its aim.

Some possible extensions to MATS are listed below.

- Provide short circuit and over-voltage protection for the MPAB as suggested in §A.2.1.
- Use both output channels of the NIDAQ to provide a stereo effect such that the direction to which the beamformer is steered is mirrored by the two channel output. A simple way to do this is to apply the same data to each output channel, but change the relative volume of each channel to reflect the beam steering direction.

[8] See Chapter 3.

- Implement an adaptive beamforming algorithm. One simple candidate algorithm would be to form several beams pointed at different directions simultaneously. At each adaptation step, the beam with the greatest output power is chosen. This scheme has already been successfully applied in practice [29, 51].

- Allow on-line beam steering. This could be done by storing a set of pre-calculated filter coefficients which steer the beam to a set of angles, and then loading the appropriate set at run-time.

A.5 System Function Calls

This Section details the system calls used in the MATS software. For each function call a brief description of its functionality is given, along with an outline of the parameters required and the appropriate setting in brackets. Full details of these calls may be found in [66] for the NIDAQ and [3] for the DSP.

A.5.1 NIDAQ Function Calls

A/D Functions

AI_Configure [66, p.3-5]: Sets up the analog inputs of the NIDAQ. Parameters include inputMode (=1, i.e. referenced single-ended), inputRange (=0, i.e. -5v to 5v), and polarity (=0, i.e. bi-polar).

AI_Mux_Config [66, p.3-8]: Configures the number of external multiplexer boards connected to the NIDAQ (zero in our case).

DAQ_Clear [66, p.3-38]: Cancels the current data acquisition operation.

DAQ_Config [66, p.3-38]: Configures the data acquisition operation. Parameters include startTrig (=0, i.e. generate a software trigger to start the DAQ sequence) and extConv (=0, i.e. use on-board clocks to control DAQ A/D conversions).

DAQ_DB_Config [66, p.3-40]: Determines whether double buffering is enabled for the data acquisition operation. Parameters include DBmode (=1, i.e. enable double-buffering).

DAQ_DB_HalfReady [66, p.3-40]: Checks whether the next half buffer of data is ready. Parameters are halfReady (whether next buffer is ready, 0=not ready, 1=ready) and daqStopped (not used).

DAQ_DB_Transfer [66, p.3-42]: Transfers the next half buffer of data to a storage buffer. Parameters include halfBuffer (the address of the array to which the data is to be transferred), ptsTfr (number of points transferred, not used), and daqStopped (ignored).

DAQ_Trigger_Config [66, p.3-51]: Configures the pre-triggering mode of data acquisition. Parameters include stopTrig (=0, i.e. disable pre-triggering) and ptsAfterStopping (not required).

Get_DA_Brds_Info [66, p.3-79]: Given the Slot number of a board, returns the board code, i.e., the type of board associated with that particular board number, and other information. We only use this call to give the board code.

SCAN_Setup [66, p.3-125]: Initialises circuitry for the scanned data acquisition operation. Parameters include numChans (8 or 16), chanVector (array containing on-board channel scan sequence – this is not sequential because of the channel setup on the MPAB), and gainVector (array containing gain setting to be used for each channel – options are -1 (for a gain of 0.5),1,2,5,10,20,50,100 – the same gain should be used for every channel, with individual channel gains normalised using the trim-pots on the MPAB).

SCAN_Start [66, p.3-126]: Initialises the data acquisition process. Parameters include buffer (the buffer into which the data should be stored), count (the number of samples to acquire, i.e., the size of the buffer), sampTimeBase (=-1, use 5MHz clock as timebase, i.e., 200nsec resolution), sampInterval (sample rate = 1/(sampInterval × numChans × 200e-9)), scanTimebase (ignored), and scanInterval (=0, the scan sequence is restarted one sample after it has completed, i.e., continuous sampling).

D/A Functions

Init_DA_Brds [66, p.3-85]: Initialises the NIDAQ board to its default state. Used to guarantee that the output voltage at DAC0 is 0.0 volts.

WFM_ClockRate [66, p.3-150]: Configures the output rate. Parameters include whichclock (=0, i.e. uses the update clock), timebase (=-1, use 5MHz clock as timebase, i.e., 200nsec resolution), interval (number of timebase units which elapse between updates of the analog output, interval = numChans × ADsampleInt), and mode (=0, not used).

WFM_DB_Config [66, p.3-152]: Configures the D/A for double-buffered operation. Parameters include numChans (=1), chanVect (=0, i.e. use channel zero), dbMode (=1, i.e. enable double-buffering), oldDataStop (=0, i.e. allow regeneration of data), and partialTransferStop (=0, i.e. allow partial half buffer transfers).

WFM_DB_HalfReady [66, p.3-154]: Checks if the double buffer is ready to receive the next half buffer of data. Parameters include halfReady (0=not ready, 1=ready).

WFM_DB_Transfer [66, p.3-156]: Transfers an array of data to the output buffer. Parameters include buffer (the array of data to transfer) and count (the number of data points, i.e., the size of the data buffer).

WFM_Group_Control [66, p.3-159]: Control analog output operation. Parameters include operation (0=terminate operation, 1=start operation).

WFM_Load [66, p.3-161]: Assigns a buffer to a selected analog output. Parameters include numChans (=1), chanVect (=0), buffer (array of values to output), count (number of points to output, i.e., size of buffer), iterations (=0, i.e. output continues indefinitely), and mode (=0, not used).

A.5.2 DSP Function Calls

These are function calls which are used within the C code on the PC. They are used to set up the DSP and transfer information between the DSP and the PC. The functions are executed with a call to m96(), which is passed a single parameter which is a pointer to the *parameter block* (a storage area that holds the actual parameters). In C, the parameter block is defined as a structure (see [3, p.163] for a definition of this structure).

DownloadMemoryToDsp [3, p.252]: Transfer data from a data array to the DSP memory. Used to transfer an array of data acquired from the analog inputs to the DSP for the beamforming to be applied. Parameters include a transfer format code (=0xa0000000, i.e., block I/O transfer to the DSP Y-memory), the memory address (=0x00100000), and a pointer to the data array which is to be downloaded.

GetConfigFile [3, p.243]: Reads the specified file and sets up a DSP configuration structure. Parameters include a pointer to the configuration storage area and a pointer to the configuration file name (M96.CFG).

InitializeDsp [3, p.202]: Resets the DSP and boot-loads an assembler programs to the DSP. Parameters include the object file format (=0, i.e., Motorola load file format) and a pointer to the file name to boot-load.

UploadDspToMemory [3, p.253]: Transfer data from DSP memory to a data array. This is used to transfer the beamformer output to the PC, where it is then sent to the analog output. Parameters include a transfer format code (=0xa0000000, i.e., block I/O transfer from DSP Y-memory), the DSP address (=0x00101800), and a pointer to the array to which the data is to be uploaded.

A.6 MPAB Test Results

This Section details the testing performed on the MPAB. Full details of the testing procedures are given for reproducibility.

A.6.1 Input High-pass Filter

The gain of all channels was set to 41.6dB, and a sine wave of 50 mv peak-to-peak (v_{in}) was applied to the input of channel 1 with the cutoff frequency of the MF6-50 Switched Capacitor Lowpass Filter set to 8 kHz (so as not to influence the measurements over the frequency range used); all other inputs were left open. The output voltage on channel 1 (v_{out}) was measured over a range of frequencies and the gain calculated as

$$\text{Gain (dB)} = 20\log_{10}\left(\frac{v_{\text{out}}}{v_{\text{in}}}\right) - 41.6$$

and compared with the response of an ideal first order Butterworth high-pass filter (see Fig. A.13). The test was also performed on channel 2 and the results obtained were practically identical to those obtained for channel 1.

A.6.2 Anti-aliasing Filter

Cutoff Frequency of 4 kHz

The gain of all channels was set to 24.6dB, and a sine wave of 400 mv peak-to-peak (v_{in}) was applied to the input of channel 1 with the cutoff frequency of the MF6-50 Switched Capacitor Lowpass Filter set to 4 kHz (i.e., the clock rate was set to 200 kHz); all other inputs were left open. The output voltage on channel 1 (v_{out}) was measured over a range of frequencies and the gain calculated as

$$\text{Gain (dB)} = 20\log_{10}\left(\frac{v_{\text{out}}}{v_{\text{in}}}\right) - 24.6$$

and compared with the response of an ideal 6th order Butterworth low-pass filter (see Fig. A.14). The test was also performed on channel 2 and the results obtained were practically identical to those obtained for channel 1.

Cutoff Frequency of 8 kHz

The procedure above was followed with the cutoff frequency set to 8 kHz (i.e., the clock rate was set to 400 kHz). Again the gain was calculated and compared with the response of an ideal 6th order Butterworth low-pass filter (see Fig. A.15)

A.6.3 Reconstruction Filter

A 10 volt peak-to-peak sine wave was applied to the input of the 4th order Butterworth reconstruction filter, and the output measured over a range of frequencies. The results obtained were compared with the response of an ideal 4th order Butterworth response (see Fig. A.16).

A.7 Source Code

This Section lists the source code for the beamforming software. A description of the programs is given in §3.2. The programs are:

adda8.h: header file which defines the high level routines used by the data acquisition process

ad.c: implementation of the input routines defined in adda8.h

da.c: implementation of the output routines defined in adda8.h

pc8.c: main program which controls the data acquisition process and transfers data to and from the DSP

dp.asm: DSP program which performs the actual beamforming

count10.src: GAL program to divide the clock signal by either 10 or 20

```
8
4 4 4 4 4 2 2 2
5
```

(a)

```
  -1.0095
  -4.2779
   4.2779
   1.0095

  -5.0458
  -2.9437
   2.9437
   5.0458

   1.5395
  -1.9642
   1.9642
  -1.5395

   1.5866
  -2.0242
   2.0242
  -1.5866

  -5.0373
  -2.8227
   2.8227
   5.0373

   0
   0

   0
   0

   0
   0

  -0.2004
   0.1339
   0.3745
   0.1339
  -0.2004
```

(b)

Figure A.10: (a) Example configuration file, and (b) corresponding data file.

```
Load filter configuration record (load_filcof_cfg)
            and allocate memory for filter coefficients
Initialise DSP and NIDAQ (DSP_init, DA_setup, AD_setup)
Process filter configuration and download to DSP (DownloadMemoryToDsp)
Load filter coefficients (load_filcofs) and display
Download filter coefficients to DSP (DownloadMemoryToDsp)
Set up link between the DSP buffers and the NIDAQ buffer
Start D/A and A/D operations (DA_start, AD_start)
while(AD_cnt)) {
  if (AD_check_ready) {
    AD_get_half
    Download AD_halfbuffer to DSP (DownloadMemoryToDsp)
    AD_cnt--
  }
  if (DA_check_ready) {
    Upload DSP to DA_halfbuffer (UploadDspToMemory)
    DA_put_half
  }
  if (kbhit) {
    Stop D/A and A/D operations (DA_stop, AD_stop)
    Get menu option and take appropriate action (menu_control)
    Setup D/A and A/D operations (DA_setup, AD_setup)
    Download filter coefficients to DSP (DownloadMemoryToDsp)
    Start D/A and A/D operations (DA_start, AD_start)
  }
}
Stop D/A and A/D operations (DA_stop, AD_stop)
```

Figure A.11: Block diagram of main program.

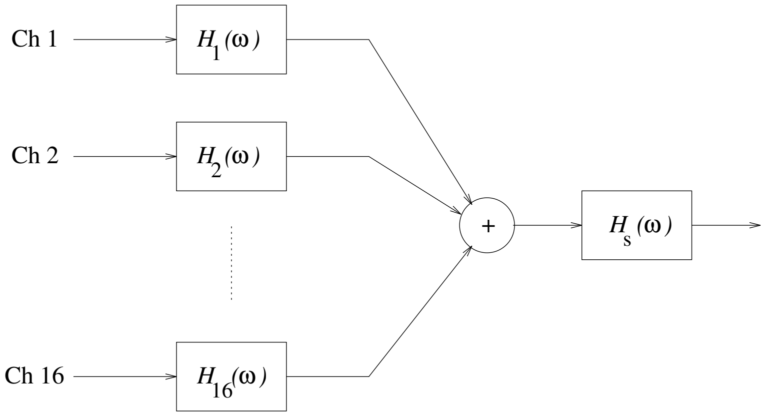


Figure A.12: Block diagram of operations performed by the DSP board.

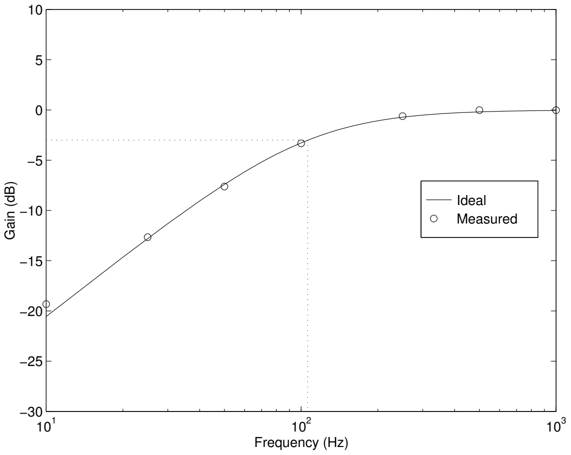


Figure A.13: Frequency response of Butterworth 1st order high-pass filter stage. Cutoff frequency is 106.1 Hz (i.e., a 1.5 μF capacitor and a 1 KΩ resistor).

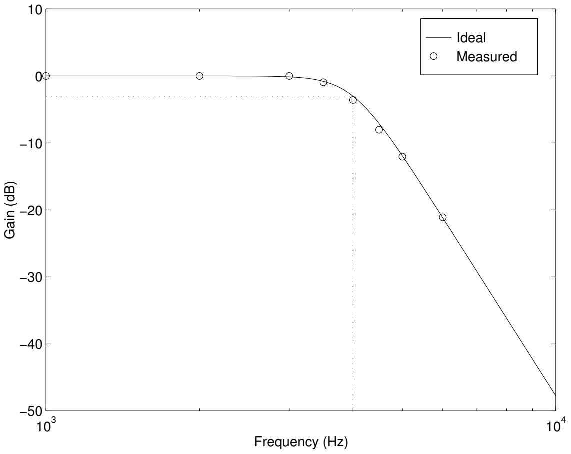


Figure A.14: Frequency response of Butterworth 6th order anti-aliasing filter stage. Cutoff frequency is 4kHz.

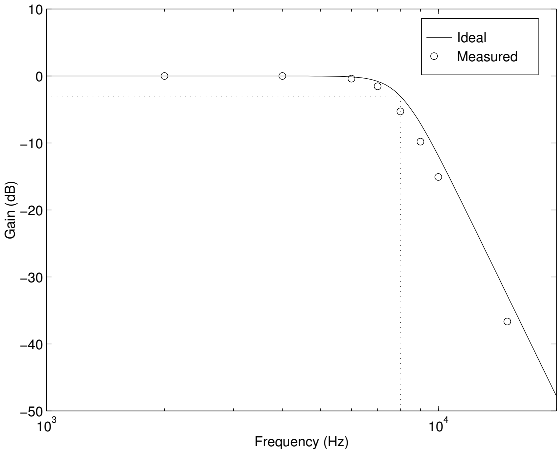


Figure A.15: Frequency response of Butterworth 6th order anti-aliasing filter stage. Cutoff frequency is 8kHz.

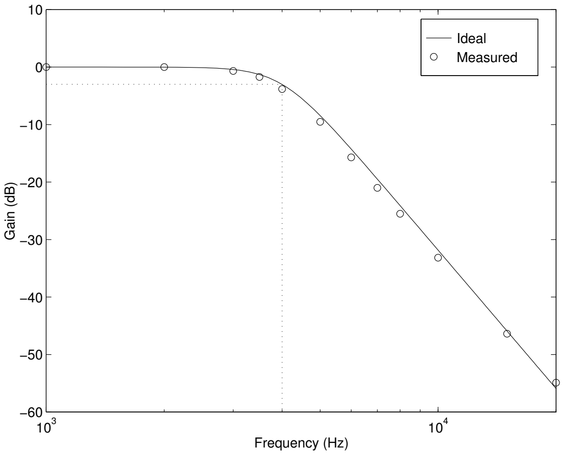


Figure A.16: Frequency response of Butterworth 4th order reconstruction filter stage.

A.7.1 adda8.h

```
/*************************************************************************/
/*      Definitions for A/D & D/A conversion and double buffering       */
/*                                                                       */
/*      Actual sampling rate is defined here also, and D/A sampling      */
/*              count is derived from that later in this file            */
/*                                                                       */
/*      ADDA.H                                  Nov 1994 (Martin Buss)   */
/*      Version 1.0    works only for 16 channels                        */
/*                                                                       */
/*      Version 1.1                             Dec 1994 (Martin Buss)   */
/*      - modification for even number of channels                       */
/*      - comments added                                                 */
/*                                                                       */
/*      Version 1.2                             May 1995 (Darren Ward)   */
/*      - renamed to adda8.h                                             */
/*      - suitable only for 8 channel operation                          */
/* - allows definition of sample rate directly */
/*                                                                       */
/*************************************************************************/

/*************************************************************************/
/*      A/D conversion                                                   */
/*************************************************************************/

/* size of the circular buffer in number of samples */
/* the number of samples is also the number of (int)s for the circular
buffer. */
/* !!! Make sure the buffer is an even multiple of the number of channels
   times 2 being sampled. */
/* configured for 8 channels * 2 * 0x200 (=512) samples */
#define AD_Buffer_size (0x2000)
/* size of the half buffer in number of samples  */
/* The AD_halfBuffer is one half as big as the AD_Buffer. */
#define AD_halfBuffer_size (AD_Buffer_size/2)

/* !!!DSP!!! AD_halfBuffer_size/2 is the number of 32 bit words transferred
to the DSP memory, this is very important to remember and
if the buffer sizes here are changed, the loop-counter in
dp.asm has to be changed as well. Not doing so results in
unpredictable results.  */

/* Version 1.1: The number of channels must be an even number */
/* !!!DSP!!! changing the number of channels requires a manual change
of dp.asm, where also the number of channels has to be adjusted */
#define numChans (8)

/* select the 200 nsec timebase for timing of A/D conversion */
#define ADtimebase (-1)

/* select a sampling rate in Hz */
#define sampleRate 16000

/* use the defined sample rate to determine ADsampleInt */
#define ADsampleInt ( (int)(1.0 / ( (double)sampleRate*(double)numChans* \
    200.0e-9 )) )

/* buffer used to get data from AD_Buffer */
extern int huge *AD_halfBuffer;

extern void AD_setup();
extern void AD_start();
extern int AD_check_ready();
extern void AD_get_half();
extern void AD_stop();

/*************************************************************************/
```

```
/*      A/D conversion                                                  */
/************************************************************************/

/* Version 1.1: DA_Buffer_size is AD_Buffer_size/numChans */
#define DA_Buffer_size (AD_Buffer_size/numChans)
#define DA_halfBuffer_size (DA_Buffer_size/2)

/* set waveform timebase to 200 nsec MIO-16F-5 */
#define WFtimebase (-1)
#define update_int (numChans*ADsampleInt)

extern int huge *DA_halfBuffer;

extern void DA_setup();
extern void DA_start();
extern int DA_check_ready();
extern void DA_put_half();
extern void DA_stop();
```

A.7.2 ad.c

```
/***************************************************************************


    Program to handle the NIDAQ board for A/D conversion using
double buffered operation


void AD_setup()
setup double buffers and do some initialization for
AD sampling rate and number of channels, etc.
buffer sizes and sampling rate are defined in the
include file "adda.h"
Remark: A change of AD buffer sizes involves also
adaptation of everything else from DA buffer
size to actual DSP software change for there
will be a different number of data per transfer
and execution cycle in the DSP. It is therefore
not recommended to change buffer sizes.
The buffer sizes as currently set are appropriate
for most applications, including a change of
sampling rate and number of channels.
void AD_start();
start A/D conversion process, AD double buffer will be
continuously filled with converted signals
int AD_check_ready();
check whether a half buffer is available for transfer
to DSP.
void AD_get_half();
copies from double buffer to half buffer, which then
can be transferred to DSP
void AD_stop();
stops A/D conversion and cleans up heap memory

Martin Buss, Nov  8, 1994
    Jan 11, 1995

    Modifications:
    (1) - changed scan order to comply with h/ware scanning order
    (2) - changed names of some routines, constants, etc.
Darren Ward, May  4, 1995

***************************************************************************/


#include <stdio.h>
#include <malloc.h>
#include <ctype.h>
#include <conio.h>
#include <fcntl.h>
#include <sys\types.h>
#include <sys\stat.h>
#include "nidaq.h"          /* function definitions   */
#include "nidaqerr.h"       /* error code definitions */
#include "adda8.h"

int huge *AD_Buffer;          /* buffer used to store data cyclically */
int huge *AD_halfBuffer;      /* buffer used to get data from AD_Buffer */

void        ErrClean();         /* error message and clean up procedure */

   int   i,              /* index                                        */
 errNum,        /* error returned from driver function calls      */
 boardCode,     /* board code                                     */
 chans[16],     /* the analog input channels to be sampled        */
 gains[16],     /* the gains for input channels                   */
 status,        /* flag indicating completion of data acquisition */
 halfReady,     /* flag indicating a half buffer of data is ready */
 ptsTfr,        /* number of points transferred to AD_halfBuffer  */
 dum;           /* a place-holder for Get_DA_Brds_Info            */
```

```
/*   int timebase,sampleInt; */
        unsigned int
 no_bits,       /* Number of A to D bits in the corresponding board */
 numBytesToWrite,  /* number of bytes to write to the file          */
 numBytesWritten;  /* number of bytes actually written to the file  */

/***************************************************************************/
/*      AD_setup() sets up all the buffers up to the sampling rate
/*              for continuous sampling of a number of channels
/***************************************************************************/

void AD_setup()
{
/* Initialize array pointers to NULL */
   AD_Buffer = NULL;
   AD_halfBuffer = NULL;

     /*-------------------------------------------------------------*
      *                C O N F I G U R A T I O N                    *
      *                                                             *
      * Step 1 in the configuration phase is to configure the board.  *
      * Please note that, except for the DAQ_B_Config call, these    *
     * configuration calls are included for illustration only - they *
     * merely re-establish the default settings. The DAQ_B_Config   *
    * call is required as it enables the double-buffered mode       *
    * needed for this program.
    *-------------------------------------------------------------*/

 errNum = Get_DA_Brds_Info (board, &boardCode, &dum, &dum, &dum, &dum,
&dum, &dum, &dum);
 ErrPrint ("Get_DA_Brds_Info", errNum);

 for (i=0; i<numChans; i++)
```

```
   {
/* inputRange is ignored for AT-MIO-16F-5, AT-MIO-64F-5, AT-MIO-16X, and \
   Lab-PC */
/* syntax:  AI_Configure (board, chan, inputMode, inputRange, polarity, \
   driveAIS); */
   errNum = AI_Configure (board,    i,         1,          0,        0, \
          0);
      ErrClean ("AI_Configure", errNum);
   }

   errNum = AI_Mux_Config (board, 0);
   ErrClean ("AI_Mux_Config", errNum);

/* syntax:  DAQ_Config (board, startTrig, extConv);                      */
   errNum = DAQ_Config (board, 0,        0);
   ErrClean ("DAQ_Config", errNum);

   errNum = DAQ_Trigger_Config (board, 0,        0);
   ErrClean ("DAQ_Trigger_Config", errNum);

/* syntax:  DAQ_B_Config (board, DBmode);                               */
  errNum = DAQ_B_Config (board, 1);
 ErrClean ("DAQ_B_Config", errNum);

  /*-------------------------------------------------------------*
   *                C O N F I G U R A T I O N                    *
   *                                                             *
   * Step 2 in the configuration phase configures the data buffers *
   * and opens the file. Since the size of the data buffers       *
   * must be an even multiple of the number of channels sampled,   *
   * we set the number of channels to sample at this point. We     *
   * also adjust the buffer sizes according to the sample rate to  *
   * avoid waiting a very long time between disk writes when the   *
   * sampling rate is slow. So we prompt for the sampling rate at  *
```

```
      * this point as well. We also prompt for the number of samples  *
      * to save in the disk file.                                      *
      *---------------------------------------------------------------*/

/* Set up the channel and gain arrays. */
   for (i=0; i<numChans; i++)
   {
      if ( (float)(i/2)==((float)i/2.0) )
chans[i] = i/2;
      else
chans[i] = (i-1)/2 + 8;
      gains[i] = 1;
   }

/* allocate space for the array of acquired data */
   AD_Buffer = (int huge *) halloc ((unsigned long)AD_Buffer_size,
 sizeof(int));
   if (AD_Buffer == NULL)
   {
      printf("\nInsufficient memory for circular buffer.");
      exit(1);
   }

   AD_halfBuffer = (int huge *) halloc ((unsigned long)AD_halfBuffer_size,
     sizeof(int));
   if (AD_halfBuffer ==  NULL)
   {
      printf("\nInsufficient memory for circular and half buffers.");
      hfree (AD_Buffer);
      exit(1);
   }

/*
errNum = DAQ_Rate (128000, 0, &timebase, &sampleInt);
   ErrClean ("DAQ_Rate", errNum);
*/

   errNum = SCAN_Setup (board, numChans, chans, gains);
   if (errNum) ErrClean ("SCAN_Setup", errNum);

}

/***************************************************************************/
/*      AD_start() starts the scanning into the double buffers
/***************************************************************************/

void AD_start()
{
/*
   printf("\nADsampleInt = %d, Sample Rate = %g (Hz)\n",ADsampleInt,
1.0/((float)ADsampleInt*(float)numChans*200.0e-9) );
*/

   errNum = SCAN_Start (board, AD_Buffer, AD_Buffer_size,
ADtimebase, ADsampleInt, 2, 0);
   if (errNum) ErrClean ("SCAN_Start", errNum);
}

/***************************************************************************/
/*      AD_check_ready()
/***************************************************************************/

int AD_check_ready()
{

   errNum = DAQ_B_HalfReady (board, &halfReady, &status);
  if (errNum)
  {
        ErrPrint ("DAQ_B_HalfReady", errNum);
        exit(-1);
     }
     return(halfReady);
  }

  /**************************************************************************/
  /*      AD_get_half()
  /**************************************************************************/
  void AD_get_half()
  {
     errNum = DAQ_B_Transfer (board, AD_halfBuffer, &ptsTfr, &status);
    if (errNum)
    {
       ErrPrint ("DAQ_B_Transfer", errNum);
      exit(-1);
   }
}

void AD_stop()
{
     /*----------------------------------------------------------------*
      *                 C L E A N I N G   U P                          *
      *                                                                *
      * Cleaning up involves clearing the acquisition, freeing the    *
      * buffers and closing the data file. Also, reporting why the    *
      * acquisition terminated is a friendly thing to do. Note that   *
      * we leave the file open here because of the following plotting *
      * section.                                                       *
      *----------------------------------------------------------------*/

/* clearing the acquisition halts the board, disables interrupts and DMA
   and restores the interrupt vector table. */
   errNum = DAQ_Clear (board);
```

```
   ErrPrint ("DAQ_Clear", errNum);

   hfree (AD_Buffer);
   hfree (AD_halfBuffer);
}


/***************************************************************************
   void ErrClean( proc_name, errNum )

   This function prints the error,if any, from a NI-DAQ DOS function.
   In case of an error memory allocated for buffer is freed and the
   program is ended.
      'proc_name': string of the name of the NI-DAQ DOS function
      'errNum': the error returned from the NI-DAQ DOS function
      'value_buf': space for binary values read from input channel
***************************************************************************/
void ErrClean(procname,errNum)
char   *procname;
int   errNum;
{
   ErrPrint(procname,errNum);
   if (errNum != noErr)
   {
      if (AD_Buffer != NULL) hfree(AD_Buffer );
      if (AD_halfBuffer != NULL) hfree(AD_halfBuffer );
      printf("\nSorry! Exiting program because of the error.\n");
      exit(1);
   }
}
```

A.7.3 da.c

```
/**************************************************************************

    Program to handle the NIDAQ board for D/A conversion using
double buffered operation

void DA_setup();
setup double buffers and do some initialization for
DA sampling rate and number of channels, etc.
buffer sizes and sampling rate are defined in the
include file "adda.h"
Remark: A change of DA buffer sizes involves also
adaptation of everything else from AD buffer
size to actual DSP software change for there
will be a different number of data per transfer
and execution cycle in the DSP. It is therefore
not recommended to change buffer sizes.
The buffer sizes as currently set are appropriate
for most applications, including a change of
sampling rate and number of channels.
void DA_start();
start DA conversion as waveform generation
int DA_check_ready();
check if half buffer has been converted
return 1 if a new half buffer is needed for next cycle
in double buffer
void DA_put_half();
copies half buffer into appropriate position in double buffer
void DA_stop();
stops DA conversion

Martin Buss, Nov  8, 1994
     Jan 11, 1995
**************************************************************************/

#include <stdio.h>
#include <malloc.h>
#include "nidaq.h"      /* function definitions */
#include "nidaqerr.h"   /* error code definitions */
#include "adda8.h"

/* set up variables for double buffered WFM generation */
int huge *DA_Buffer;
int huge *DA_halfBuffer;
int DA_HalfReady;

int boardCode,err;
int chanVect[1];

void       errCheck();      /* for printing error messages and exiting */

void DA_setup()
{
   int i;

/* initialize the board to be sure that the voltage at DAC0 is 0.0 volts */
   err = Init_DA_Brds (board, &boardCode);
   ErrPrint ("Init_DA_Brds", err);

/* allocate space for the array of acquired data */
   DA_Buffer = (int huge *) halloc ((unsigned long)DA_Buffer_size,
 sizeof(int));
   if (DA_Buffer == NULL)
   {
      printf("\nInsufficient memory for DA_Buffer.");
      hfree(DA_Buffer);
      exit(1);
      }

      DA_halfBuffer = (int huge *) halloc ((unsigned long)DA_halfBuffer_size,
        sizeof(int));
      if (DA_halfBuffer ==  NULL)
      {
         printf("\nInsufficient memory for DA_halfBuffer.");
         hfree (DA_halfBuffer);
         exit(1);
      }

      /* clear DA_Buffer */
      for (i=0; i<DA_Buffer_size; i++) DA_Buffer[i]=0;
   /* clear DA_halfBuffer */
      for (i=0; i<DA_halfBuffer_size; i++) DA_halfBuffer[i]=0;

      chanVect[0] = 0;
   /*      WFM_B_Config(board,numChans,chanVect,dbMode,oldDataStop, \
        partialTransferStop); */
     err = WFM_B_Config(board,       1,chanVect,     1,          0, \
                       0);
    errCheck ("WFM_B_Config", err);

/*      WFM_Load (board,numChans, chanVect,    buffer, \
                             count, iterations,mode); */
   err = WFM_Load (board,       1, chanVect, DA_Buffer, \
      (unsigned long) DA_Buffer_size,         0L, 0);
   errCheck ("WFM_Load", err);

/*      WFM_ClockRate(board,group,whichclock,   timebase, \
                       interval,mode);   */
   err = WFM_ClockRate(board,    1,         0, WFtimebase,
      (unsigned long) update_int,   0);
   errCheck("WFM_ClockRate", err);
```

```
     }

     void DA_start()
     {
        /* start the waveform generation process */
        err = WFM_Group_Control (board, 1, 1);
        errCheck ("WFM_Group_Control", err);
     }

     int DA_check_ready()
     {
        err = WFM_B_HalfReady(board,1,chanVect,&DA_HalfReady);
       errCheck ("WFM_B_HalfReady", err);
      return(DA_HalfReady);
   }

   void DA_put_half()
   {
      /*    WFM_B_Transfer(board,numChans,chanVect,       buffer, \
                                    count);    */
     err = WFM_B_Transfer(board,       1,chanVect,DA_halfBuffer, \
       (unsigned long)DA_halfBuffer_size);
    errCheck ("WFM_B_Transfer", err);
}

void DA_stop()
{
   /* stop the waveform generation process */
   err = WFM_Group_Control (board, 1, 0);
   errCheck ("WFM_Group_Control", err);

   hfree(DA_Buffer);
   hfree(DA_halfBuffer);
}
```

```
/*
 * This procedure calls ErrPrint to print an error message. If the error is
 * a serious one (as opposed to a warning) this procedure terminates the
 * program. Note that WFM_Group_Control is called before termination.
 */

void errCheck (proc_name, err_num)
char *proc_name;
int err_num;
{
   ErrPrint (proc_name, err_num);
   if (err_num < 0)
   {
      WFM_Group_Control (board,1,0);
      exit(1);
   }
}
```

A.7.4 pc8.c

```
/************************************************************************/
/*      main program for the PC which controls all functions            */
/*                                                                      */
/*      PC.C                                    Nov 1994 (Martin Buss)  */
/*                                                                      */
/*      Version 1.1                             Dec 1994 (Martin Buss)  */
/*      modified for the new memory map and on-line modification
of filter configuration record and filter coefficients  */
/*      Version 1.2                             Jan 1995 (Martin Buss)  */
/*      - on-line modification and download of filter coefficients
controllable by keyboard commands
- some display of functions on screen has been added, which might
have to be excluded for very high sampling rates near
the PC-speed limits
- if MENUCONTROL is defined AD/DA conversion is stopped upon
a key pressed on the keyboard and a menu will pop up
for control of filter coefficients and such
this part of the program is under development and just
shows how a menu might be implemented to control the
functions of the software. In future a change of sampling
interval and number of channels, or whatever might be
implemented as menu functions as well                   */
/*      Version 1.3                             May 1995 (Darren Ward)  */
/*      - filter configuration is stored in a file and loaded at runtime
- added function for testing a single channel
*/
#define MENUCONTROL
/*                                                                      */
/************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <dos.h>
#include <conio.h>
#include <malloc.h>
#include "dspm96.h"
#include "adda8.h"

#define PROG    "DP"
#define filename_filcof "filcof.dat"
#define filename_filcofbin "filcof.bin"
#define filename_filcofcfg "filcof.cfg"

struct m96parms mcall, mcall_AD, mcall_DA, mcall_filcof;
/************************************************************************/
/*      define all memory addresses corresponding to dp.asm
(see dp.asm for explanation)                            */
/************************************************************************/
#define adr_addata 0x0000
#define adr_fil_config_record 0x0800
#define adr_debbuf 0x0880
#define adr_filcof 0x0900
#define adr_daoutp 0x1800
/************************************************************************/
/*      define buffers for filter configuration record and coefficients */
/************************************************************************/
float fbuf[0x80];
#define fil_config_len 0x80
unsigned long fil_config[fil_config_len];
unsigned long micN;
unsigned long filN[numChans];
unsigned long ofilN;
#define filcofsetN 1L
float huge *fil_coffs;
int filcof_ind=0;
int temp_int=0;
```

```
FILE *f;
unsigned long cofreclen;
/************************************************************************/
/*      print error which occured calling m96()                         */
/*      !!! this also stops all A/D and D/A conversion                  */
/*              when not stopping this properly PC may hang             */
/************************************************************************/
merror( q )
struct m96parms *q;
{
    int fun,i1,i2, err;
    err = q->mret;
    fun=(*q).mfun;
    i1=(*q).i1;
    i2=(*q).i2;
    printf("\nError %d in DSP call %d!\n", (*q).mret, (*q).mfun );
    printf("#1: %ld #2: %ld #3: %ld #4: %ld #5: %ld #6: %ld\n", (*q).i1, \
      (*q).i2, (*q).i3, (*q).i4, (*q).i5, (*q).i6 );
    printf("Control word: %x\n",(*q).mconfig);
    mcall.mfun = ShowError;
    mcall.i1=(*q).mret;
    mcall.i2=YES;
    m96(&mcall);
    (*q).mfun=fun;
    (*q).i1=i1;
    (*q).i2=i2;
    AD_stop();
    DA_stop();
    exit(err);
}

/************************************************************************/
/*      DSP_init()
Initializes the DSP and gets configuration words        */
/************************************************************************/
void DSP_init()
{
    struct mconfigfile mcon[1];
    int rec;

    mcall.mfun = GetConfigFile;
    mcall.i1 = 1;
    mcall.p1 = &mcon[0];
    mcall.s1 = CONFIG_FILE_NAME;
    if(! m96(&mcall)) {
printf("\nFatal error: Bad or missing M96.CFG configuration file\n");
merror(&mcall);
exit(1);
}
    rec = mcall.i2;      // i2 returns record to use based on M96SEL env var

    printf("Configuration word: 0x%4.4x\n", mcon[rec].configuration_word);
    printf("Base address: %d (decimal)\n", mcon[rec].base_address);
    mcall.mconfig = mcon[rec].configuration_word;

    mcall.mfun = InitializeDsp;
    mcall.i1 = 0;
    mcall.s1 = PROG;
    if(! m96(&mcall))
merror(&mcall);
    printf("Boot loaded %s (0x%04x words) to %s\n", mcall.s1, mcall.i3, \
      mcall.s2);

    printf("DSP96002 PC Driver version %ld.%ld\n", mcall.i1, mcall.i2);
    // Seed the random number generator
    srand( (unsigned)time( NULL ) );

    // mcall_AD and mcall_DA use same configuration word
    mcall_AD.mconfig=mcall.mconfig;
    mcall_DA.mconfig=mcall.mconfig;
    mcall_filcof.mconfig=mcall.mconfig;
}
/************************************************************************/
/*      debug_filcon()                                                  */
/*              upload and display filter configuration record          */
/************************************************************************/
void debug_filconf()
{
    int i;
    mcall.mfun=UploadDspToMemory;
    mcall.i2=0xa0000030L;
    mcall.i3=0x00100000L|((unsigned long)adr_fil_config_record);
    mcall.p1=fil_config;
    if (!m96(&mcall)) merror(&mcall);
    printf("\n");
    printf("  stateN-1  interl.   filN-1    stateptr  daoutp\n");
    for (i=0; i<5*(numChans+1); i+=5) {
printf("  %08lx  %08lx  %08lx  %08lx  %08lx\n",
fil_config[i],fil_config[i+1],fil_config[i+2],
fil_config[i+3],fil_config[i+4]);
    }
    printf("\n");
}

/************************************************************************/
/*      test_chan(int chanNum)                                          */
/*              acquire data from a single channel only                 */
/************************************************************************/
void test_chan(int chanNum)
/* chanNum is the channel no to acquire data from, in range 1-numChans */
{
    int i,j;

/* Clear all the filter coefficients */
    temp_int = 0;
    for (j=0; j<micN; j++) {
      for (i=0; i<filN[j]; i++) fil_coffs[temp_int+i]=0.0;
      temp_int += filN[j];
    }

/* And set up the secondary filter as an all pass */
    temp_int=0;
    for (j=0; j<micN; j++) temp_int += filN[j];
    fil_coffs[temp_int] = 1.0;
    for (i=1; i<ofilN; i++) fil_coffs[temp_int+i]=0.0;

/* Then we set up the Nth filter as an all-pass*/
  for (i=1; i<micN; i++)
  {
    if (chanNum>0) {
      temp_int = 0;
      for (j=0; j<chanNum-1; j++) temp_int+=filN[j];
      fil_coffs[temp_int] = 1.0;
    }
  }

/* Now print the coefficients of all the filters */
/*
    temp_int = 0;
    for (j=0; j<micN; j++) {
      printf("Filter %d\n",j+1);
      for (i=0; i<filN[j]; i++) printf("%10.7f",fil_coffs[temp_int+i]);
      printf("\n");
      temp_int += filN[j];
    }
    printf("Secondary Filter\n");
    for (i=0; i<ofilN; i++) printf("%10.7f",fil_coffs[temp_int+i]);
    printf("\n");
*/


/* These are the coefficients for an 11 tap reconstruction compensation
   filter, which can be used instead of the secondary filters above */
/*    i=0;
    fil_coffs[temp_int+i] = 2.82e-3; i++;
    fil_coffs[temp_int+i] = -1.03e-2; i++;
    fil_coffs[temp_int+i] = 2.36e-3; i++;
    fil_coffs[temp_int+i] = 1.00e-1; i++;
    fil_coffs[temp_int+i] = -4.91e-1; i++;
    fil_coffs[temp_int+i] = 1.79; i++;
    fil_coffs[temp_int+i] = -4.91e-1; i++;
    fil_coffs[temp_int+i] = 1.00e-1; i++;
    fil_coffs[temp_int+i] = 2.36e-3; i++;
    fil_coffs[temp_int+i] = -1.03e-2; i++;
    fil_coffs[temp_int+i] = 2.82e-3;
*/
}
/************************************************************************/
/*      load_filcof_cfg()                                               */
/*              load filter coefficient configuration from cfg file     */
/************************************************************************/
void load_filcof_cfg()
{
    int i;

    /* read filter configurtion from file */
    printf("Reading filcof configuration.");
    f = fopen(filename_filcofcfg,"rt");
    if (f==NULL) {
      printf("\n Could not open '%s'",filename_filcofcfg);
      exit(-1);
    }
    fscanf(f,"%u",&micN);
    printf("\nmicN: %u, filN:",micN);
    for (i=0; i<micN; i++)
    {
      fscanf(f,"%u",&filN[i]);
      printf(" %u",filN[i]);
    }
    fscanf(f,"%u",&ofilN);
    printf(", ofilN: %u\n",ofilN);
    fclose(f);
}

/************************************************************************/
/*      load_filcofs()                                                  */
/*              load filter coefficients from data file                 */
/************************************************************************/
void load_filcofs()
{
    int i,j;

    /* read filter coefficients from file */
    printf("Reading fil_coffs");
    f=fopen(filename_filcofbin,"rb");
    if (f==NULL) {
printf(" from text file...");
f=fopen(filename_filcof,"rt");
if (f==NULL) {
    printf("Could not open filter coefficient file %s\n", \
            filename_filcof);
    exit(-1);
}
for (i=0; i<filcofsetN*cofreclen; i++)
    fscanf(f,"%f",&(fil_coffs[i]));
fclose(f);
/*        printf("done\nwriting to binary file...\n");
f=fopen(filename_filcofbin,"wb");
i=fwrite((void*)fil_coffs,(size_t)cofreclen*sizeof(float), \
          (size_t)filcofsetN,f);
fclose(f);
if (i!=filcofsetN) {
    printf("error writing to binary file, could only write %d \
              filcof sets\n",i);
    exit(-1);
}
*/        printf("done\n");
    } else {
printf(" from bin file...");
i=fread((void*)fil_coffs,(size_t)cofreclen*sizeof(float), \
          (size_t)filcofsetN,f);
printf("done\n");
fclose(f);
if (i!=filcofsetN) {
    printf("error reading from binary file, could only read %d \
              filcof sets\n",i);
    exit(-1);
}
    }
}

#ifdef MENUCONTROL
/**************************************************************************/
/*      control menu for software package
only active if MENUCONTROL is defined
/**************************************************************************/
void menu_control()
```

```
{
    int i;
    int c;

    printf("\n\nMenu for DSP software control");
    printf("\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");
    printf("\n[1-%d] test a single channel",micN);
    printf("\n[R] reload filter coefficients");
    printf("\n[Q] quit program");
    printf("\nAny other, continue.");
    printf("\nCommand -> ");
    switch (c=getche()) {
    case 'n':
    case 'N':
printf("\nEnter null angle [10-90 deg]: ");
scanf("%d",&i);
if (i<10) i=10;
if (i>90) i=90;
filcof_ind=i-10;
printf("Continuing with null.\n");
break;
    case '1':
    case '2':
    case '3':
    case '4':
    case '5':
    case '6':
    case '7':
    case '8':
i=(int)c - (int)'0';
if ( (i<1)||(i>micN) ) i=1;
printf("\nTesting channel %d.\n",i);
test_chan(i);
break;
```

```
    case 'r':
    case 'R':
printf("\n");
load_filcofs();
break;
    case 'q':
    case 'Q':
printf("\nExiting.\n");
exit(0);
break;
    default:
printf("\nContinuing.\n");
    }
}
#endif

/**********************************************************************/
/*      main program initializing everything
also contains the main control loop                    */
/**********************************************************************/
main(int argc, char *argv[])
{
    int err,i,j;
    int AD_cnt=-1,DA_cnt=0;

    /* load filter configuration info */
    printf("\nLoading filter configuration info.");
    load_filcof_cfg();

    /* initialize fil_coffs array by malloc */
    cofreclen=0;
    for(i=0;i<micN; i++) {
      cofreclen += filN[i];
    }
    cofreclen+=(numChans-micN)*2L+ofilN;
    printf("Filter coefficient record length = %ld floats\n",cofreclen);
    printf("Allocating %ld Bytes for fil_coffs\n",4L*filcofsetN*cofreclen);
    fil_coffs = (float huge *) halloc (filcofsetN*cofreclen,sizeof(float));
    if (fil_coffs ==  NULL)
    {
printf("\nInsufficient memory for fil_coffs array.");
hfree (fil_coffs);
exit(1);
    }
/*    if (argc<2) {
printf("Usage:  pc <0..15> <AD_cnt>\n\n<0..15> denotes filter \
          coefficients to be used\n<AD_cnt> denotes the number of AD \
          packets\n");
exit(0);
    }
*/
    /* If there is a command line argument, use it as the number of times
       the AD buffer should be filled before exiting, otherwise scan
       continually */
    if (argc==2) AD_cnt=atoi(argv[1]);

    /* Initialize DSP and mcall configuration word */
    DSP_init();

    /* setup D/A stuff and continuous waveform generation
with double buffering */
    DA_setup();

    /* setup A/D stuff and continuous scanning
with double buffering */
    AD_setup();

    /* process and download filter configuration record for DSP
```

```
in Y-memory     y:pars */
    /* filter configuration record is explained in dp.asm */
    for (i=0,j=0; i<numChans+1; i++) {
if (i<micN) {
    /* filter configuration for micN active microphones */
    fil_config[j++]=filN[i]-1;
    /* filter interleave factor */
    fil_config[j++]=1;
    fil_config[j++]=filN[i]-1;
    /* filter state pointers */
    fil_config[j++]=0x00100000L|((unsigned long)(i*0x80L));
    fil_config[j++]=0;
} else {
    if (i<numChans) {
/* filter configuration for rest of active channels */
fil_config[j++]=1;
/* filter interleave factor */
fil_config[j++]=1;
fil_config[j++]=1;
/* filter state pointers */
fil_config[j++]=0x00100000L|((unsigned long)\
                  (micN*0x80L+i*0x8L));
/* filter interleave factor */
fil_config[j++]=0;
}
    else {
/* output filter configuration */
fil_config[j++]=ofilN-1;
/* filter interleave factor */
fil_config[j++]=1;
fil_config[j++]=ofilN-1;
/* filter state pointers */
fil_config[j++]=0x00100000L|((unsigned long)\
                  ((micN+1)*0x80L));
fil_config[j++]=0;
    }
}
    }
/*
#define INTERLEAVE_TEST 1
    fil_config[0]=INTERLEAVE_TEST*filN-1;
    fil_config[1]=INTERLEAVE_TEST;
*/


    mcall.mfun=DownloadMemoryToDsp;
    mcall.i2=0xa0000000L|((unsigned long)fil_config_len);
    mcall.i3=0x00100000L|((unsigned long)adr_fil_config_record);
    mcall.p1=fil_config;
    if (!m96(&mcall)) merror(&mcall);

    debug_filconf();

/* Load filter coefficients from data file */
    load_filcofs();

    printf("\nPress any key to continue."); getch();

/* Now print the coefficients of all the filters */
    temp_int = 0;
    for (j=0; j<micN; j++) {
      printf("Filter %d\n",j+1);
      for (i=0; i<filN[j]; i++) printf("%10.7f",fil_coffs[temp_int+i]);
      printf("\n");
      temp_int += filN[j];
    }

    printf("Secondary Filter\n");
    for (i=0; i<ofilN; i++) printf("%10.7f",fil_coffs[temp_int+i]);
    printf("\n");

    /* setup filcof down load mcall_filcof & download initial coeffs */
    mcall_filcof.mfun=DownloadMemoryToDsp;
    mcall_filcof.i2=0xa0000000L|((unsigned long)(cofreclen));
    mcall_filcof.i3=0x00100000L|((unsigned long)(adr_filcof));
    mcall_filcof.p1=fil_coffs;
    if (!m96(&mcall_filcof)) merror(&mcall_filcof);

    /* AD data goes to DSP Y-memory */
    mcall_AD.mfun=DownloadMemoryToDsp;
    mcall_AD.i2=0xa0000000L|((unsigned long)AD_halfBuffer_size/2);
    mcall_AD.i3=0x00100000L|((unsigned long)adr_addata);
    mcall_AD.p1=AD_halfBuffer;

    /* DA data from DSP Y-memory */
    mcall_DA.mfun=UploadDspToMemory;
    mcall_DA.i2=0xa0000000L|((unsigned long)DA_halfBuffer_size/2);
    mcall_DA.i3=0x00100000L|((unsigned long)adr_daoutp);
    mcall_DA.p1=DA_halfBuffer;

    printf("\nmcall_AD.i2= %08lx\n",mcall_AD.i2);
    printf("mcall_AD.i3= %08lx\n",mcall_AD.i3);
    printf("mcall_DA.i2= %08lx\n",mcall_DA.i2);
    printf("mcall_DA.i3= %08lx\n",mcall_DA.i3);

    printf("\nPress a key to start sampling."); getch();
    printf("\nSampling Rate: %g Hz\n",
1.0/((float)ADsampleInt*(float)numChans*200.0e-9) );

    DA_start();
    AD_start();

    while (AD_cnt) {
```

```
/* Now wait until half buffer is full and transfer to DSP */
/*       while( !AD_check_ready() ) ;
*/
if (AD_check_ready()) {
    AD_get_half();

    /* download halfBuffer to DSP Y-memory */
    if (!m96(&mcall_AD)) merror(&mcall_AD);

    /* get FIR filters going */
    mcall.mfun=SendDsp;
    if (!m96(&mcall)) merror(&mcall);

    AD_cnt--;
}

if (DA_check_ready()) {
    /* upload DA-values from DSP */
    if (!m96(&mcall_DA)) merror(&mcall_DA);

    DA_put_half();
    DA_cnt++;
}

if (kbhit()) {
    switch(getch()) {
    case '+':
filcof_ind++;
if (filcof_ind>=filcofsetN) filcof_ind=filcofsetN-1;
break;
    case '-':
filcof_ind--;
if (filcof_ind<0) filcof_ind=0;
break;
    }

#ifdef MENUCONTROL
    DA_stop();
    AD_stop();
    menu_control();
    DA_setup();
    AD_setup();
    /* download filter coefficients to DSP */
    mcall_filcof.p1=fil_coffs+filcof_ind*cofreclen;
    if (!m96(&mcall_filcof)) merror(&mcall_filcof);
/*
    debug_filconf();
    printf("\nPress any key to continue."); getch(); printf("\n");
*/
    DA_start();
    AD_start();
#endif
}
    }

    AD_stop();
    DA_stop();

    printf("DA_cnt= %d\n",DA_cnt);

    debug_filconf();

    mcall.mfun=UploadDspToMemory;
    mcall.i2=0xa0000030L;
    mcall.i3=0x00100000L|((unsigned long)adr_debbuf);
    mcall.p1=fil_config;
    if (!m96(&mcall)) merror(&mcall);
    printf("\n");
    for (i=0; i<0x30; i++) printf("  %08lx",fil_config[i]);
    printf("\n");
    exit(0);

}
```

A.7.5 dp.asm

```
;*********************************************************************
;*      DSP main routine for filters                                 *
;*                                                                   *
;*      DP.ASM                              Nov 1994 (Martin Buss)  *
;*                                                                   *
;*      Version 1.1                         Dec 1994 (Martin Buss)  *
;*      change of memory map to enable a change of configuration and *
;*              filter coefficients on-line                          *
;*********************************************************************

dp              IDENT   1,0
;

;J_SPARE_1_I            ;ISR for Janus Inner Port spare function 1
J_SPARE_1_I     MACRO
MOVEP   #$FFFFFFFF,X:<<MI_HTX
ENDM

;J_SPARE_2_I            ;ISR for Janus Inner Port spare function 2
J_SPARE_2_I     MACRO
nop
nop
ENDM

;J_SPARE_3_I            ;ISR for Janus Inner Port spare function 3
J_SPARE_3_I     MACRO
nop
nop
ENDM

;       IfJ_RUN_ME is set to any value, Janus will begin execution of this
;       program after reset.  IfJ_RUN_ME is not defined, Janus will stay in
;       a 'do nothing' loop and await commands.

J_RUN_ME        EQU     1               ;Run this program after reset

;       This will assemble the monitor for the correct port configuration
:
INCLUDE "JANUS.ASM"


;       Do not use an ORG statement after 'including' Janus.
;       The following code will assemble to the address following the
;       end of the Janus monitor.  If you need to know the origin of your
;       program, use the Janus WHEREIS command.  This program begins with
;       the word following Janus.


;Version 1.1: number of channels has to be even and any change
;       must be reflected in adda.h
numch   equ     8
numch2  equ     (numch/2)
;number of loops for filter execution (see below)
loopcnt equ     $1000/numch

;pars is the address of parameters used for filter calculation
;Version 1.1 has new memory map and enables on-line change of
;       configuration record and filter coefficients
;
;!!! All changes to the memory map must also be done in pc.c
;
;y-memory start from $00100000
;offset from $00100000
;0000-07ff      A/D data 0x800 words is equivalent to 0x1000 words of
;               (int)s which has to coincide with the buffer sizes
;               defined in adda.h
```

```
;
;0800-087f      parameters: filter configuration record and some memory
;0880-08ff      debbuf for debugging purposes
;0900-17ff      filter coefficients
;1800-????      D/A output data (length depending on the number of channels)
;               example: 8 channels are from $1800-$18ff
;                        (0x100 words or 0x200 (int)s)
;                        which is 0x1000 (int)s A/D / numch
;!!!!!!!!!      This is also the number of loops to be executed
;                       for the filters
ymem    equ     $00100000
addata  equ     ymem
pars    equ     ymem+$0800
;
;pars is organized sequentially for the filters
;the output filter also uses this parameter set
;
;       filter state order-1 = filter order * interleave factor - 1
;       interleave factor
;       filter order-1
;       filter state pointer
;       filter output
debbuf  equ     ymem+$0880
filcof  equ     ymem+$0900
daoutp  equ     ymem+$1800

;------------------------------------------------------------------------
;------------------------------------------------------------------------
;       filter macros
;------------------------------------------------------------------------
;------------------------------------------------------------------------
FMAC0   MACRO
move    y:(r0)+,m2              ;get filter state order-1 into m2
                                        ;  (mask for addressing)
```

```
move    y:(r0)+,n2              ;filter interleave factor
ENDM

FMAC1   MACRO
move    y:(r0)+,d2.l            ;get filter order-1
move    y:(r0),r2               ;get ptr for filter states
float.s d0
fclr    d1                      d0.s,x:-(r2)
fclr    d0                      x:(r2)+n2,d4.s  y:(r5)+,d6.s
rep     d2.l
fmpy    d4,d6,d1  fadd.s d1,d0  x:(r2)+n2,d4.s  y:(r5)+,d6.s
fmpy    d4,d6,d1  fadd.s d1,d0
;save index for filter states
  fadd.s d1,d0                  r2,y:(r0)+
;sum up all outputs in d5 and save to filout (filter output)
  fadd.s d0,d5                  d0.s,y:(r0)+
ENDM

FMAC2   MACRO
FMAC0
move    y:(r7),d0.l
ext     d0.l
FMAC1
ENDM

FMAC3   MACRO
FMAC0
move    y:(r7)+,d0.l
split   d0.l,d0.l
ext     d0.l
FMAC1
ENDM
;------------------------------------------------------------------------
;------------------------------------------------------------------------
```

```
 ;       Clear internal filter states x:$00100000..4000

 move    #-1,m4
 move    #$00100000,r4
 fclr    d0.s
 rep     #$4000
 move    d0.s,x:(r4)+
 move    #$00100000,r4
 clr     d0.l
 rep     #$4000
 move    d0.l,y:(r4)+

 done
 andi    #$cf,mr                          ;enable all interrupts

 get_cmd brclr   B_HRDF,x:<<M_HSR,get_cmd        ;wait for command
movep   x:<<M_HRX,r4                     ;copy param to r4

ori     #$30,mr                          ;disable all interrupts
;-----------------------------------------------------------------------
;-----------------------------------------------------------------------
filgoo
move    #-1,m5
move    m5,m6
move    m5,m7
move    m5,m4

move    #pars,r0
move    #5*(numch+1)-1,m0

move    #daoutp,r6              ;D/A data in y:$00100800
move    #addata,r7              ;A/D data in y:$00100000
```

```
clr     d3.l

do      #loopcnt,loopdat
;-----------------------------------------------------------------------
;-----------------------------------------------------------------------
;do main filters on each channel of A/D and sum up their outputs in d5.s
fclr    d5                      ;clear sum
move    #filcof,r5
move    #pars,r0
move    #-1,m0
do      #numch2,chloop1
FMAC2
FMAC3
chloop1
;sum of all filter outputs is in d5
;now run output filter
FMAC0
move    y:(r0)+,d2.l            ;get filter order-1
move    y:(r0),r2               ;get ptr for filter states
move    d5.s,d0.s
fclr    d1                      d0.s,x:-(r2)
fclr    d0                      x:(r2)+n2,d4.s  y:(r5)+,d6.s
rep     d2.l
fmpy    d4,d6,d1  fadd.s d1,d0  x:(r2)+n2,d4.s  y:(r5)+,d6.s
fmpy    d4,d6,d1  fadd.s d1,d0
;save index for filter states
  fadd.s d1,d0                  r2,y:(r0)+
move    d0.s,y:(r0)+

int     d0
brset   #0,d3.l,da2nd
;process first half of D/A
move    d0.l,d7.l
bset    #0,d3.l
```

```
bra     <daend
;process second half of D/A
da2nd
join    d0.l,d7.l
move    d7.l,y:(r6)+
bclr    #0,d3.l
daend
nop
loopdat
move    r4,y:debbuf
bra     <done
;-----------------------------------------------------------------------
;-----------------------------------------------------------------------
END
```

A.7.6 GAL Source Code

```
TITLE Counter10;
ENGINEER Bob Edwards;
COMPANY ANU;
COMMENT This is the counter for the 16 channel mic filter;
"
" This circuit is clocked by a 4Mhz crystal
"
GROUP USING p16v8
    clk:1,
end;

FUNCTION count10;

    INPUT cnt, clk; " Phase 1 & 2 input
    OUTPUT A, B, C, D, E CLOCKED_BY clk;

   TRUTH_TABLE
cnt, A, B, C, D, E :: A, B, C, D, E;

0,   0, x, 0, 0, 0 :: 0, 0, 0, 0, 1 ;
0,   0, x, 0, 0, 1 :: 0, 0, 0, 1, 0 ;
0,   0, x, 0, 1, 0 :: 0, 0, 0, 1, 1 ;
0,   0, x, 0, 1, 1 :: 0, 0, 1, 0, 0 ;
0,   0, x, 1, 0, 0 :: 1, 0, 0, 0, 0 ;
0,   1, x, 0, 0, 0 :: 1, 0, 0, 0, 1 ;
0,   1, x, 0, 0, 1 :: 1, 0, 0, 1, 0 ;
0,   1, x, 0, 1, 0 :: 1, 0, 0, 1, 1 ;
0,   1, x, 0, 1, 1 :: 1, 0, 1, 0, 0 ;
0,   1, x, 1, 0, 0 :: 0, 0, 0, 0, 0 ;

1,   0, 0, 0, 0, 0 :: 0, 0, 0, 0, 1 ;
1,   0, 0, 0, 0, 1 :: 0, 0, 0, 1, 0 ;
1,   0, 0, 0, 1, 0 :: 0, 0, 0, 1, 1 ;
1,   0, 0, 0, 1, 1 :: 0, 0, 1, 0, 0 ;
1,   0, 0, 1, 0, 0 :: 0, 1, 0, 0, 0 ;
1,   0, 1, 0, 0, 0 :: 0, 1, 0, 0, 1 ;
1,   0, 1, 0, 0, 1 :: 0, 1, 0, 1, 0 ;
1,   0, 1, 0, 1, 0 :: 0, 1, 0, 1, 1 ;
1,   0, 1, 0, 1, 1 :: 0, 1, 1, 0, 0 ;
1,   0, 1, 1, 0, 0 :: 1, 0, 0, 0, 0 ;
1,   1, 0, 0, 0, 0 :: 1, 0, 0, 0, 1 ;
1,   1, 0, 0, 0, 1 :: 1, 0, 0, 1, 0 ;
1,   1, 0, 0, 1, 0 :: 1, 0, 0, 1, 1 ;
1,   1, 0, 0, 1, 1 :: 1, 0, 1, 0, 0 ;
1,   1, 0, 1, 0, 0 :: 1, 1, 0, 0, 0 ;
1,   1, 1, 0, 0, 0 :: 1, 1, 0, 0, 1 ;
1,   1, 1, 0, 0, 1 :: 1, 1, 0, 1, 0 ;
1,   1, 1, 0, 1, 0 :: 1, 1, 0, 1, 1 ;
1,   1, 1, 0, 1, 1 :: 1, 1, 1, 0, 0 ;
1,   1, 1, 1, 0, 0 :: 0, 0, 0, 0, 0 ;

x,   x, x, 1, 0, 1 :: 0, 0, 0, 0, 0 ;
x,   x, x, 1, 1, 0 :: 0, 0, 0, 0, 0 ;
x,   x, x, 1, 1, 1 :: 0, 0, 0, 0, 0 ;
    END; "Truth Table

END count10;
```

Bibliography

[1] K.M. Ahmed and R.J. Evans, "An adaptive array processor with robustness and broadband capabilities," *IEEE Trans. Antennas Propagat.*, vol. AP-32, pp. 944–950, Sept. 1984.

[2] S.P. Applebaum, "Adaptive arrays," *IEEE Trans. Antennas Propagat.*, vol. AP-24, pp. 585–598, Sept. 1976.

[3] Ariel Corporation, *User's Manual for the DSP-96 DSP96002 Floating-Point Attached Processor Board with Dual-Channel Analog I/O*, First ed., 1991.

[4] H. Bach and J.E. Hansen, "Uniformly spaced arrays," in *Antenna Theory, Pt 1* (R.E. Collin and F.J. Zucker, eds.), ch. 5, pp. 138–206, New York: McGraw-Hill Inc., 1969.

[5] G.R. Baldcock and T. Bridgeman, *The Mathematical Theory of Wave Motion.* Chichester, England: Ellis Horwood Ltd, 1981.

[6] A.J. Barabell, "Improving the resolution performance of eigenstructure based direction finding algorithms," in *Proc. IEEE Int. Conf. Acoust. Speech Sig. Process.(ICASSP-83)*, Boston, USA, pp. 336–339, 1983.

[7] M.F. Berger and H.F. Silverman, "Microphone array optimization by stochastic region contraction," *IEEE Trans. Sig. Proc.*, vol. 39, pp. 2377–2386, Nov. 1991.

[8] G. Bienvenu and L. Kopp, "Decreasing high resolution method sensitivity by conventional beamformer processing," in *Proc. IEEE Int. Conf. Acoust. Speech Sig. Process. (ICASSP-84)*, pp. 33.2.1–33.2.4, 1984.

[9] K.M. Buckley, "Spatial/spectral filtering with linearly constrained minimum variance beamformers," *IEEE Trans. Acoust. Speech Sig. Proc.*, vol. ASSP-35, pp. 249–266, Mar. 1987.

[10] K.M. Buckley and L.J. Griffiths, "Broad-band signal-subspace spatial-spectrum (BASS-ALE) estimation," *IEEE Trans. Acoust. Speech Sig. Proc.*, vol. 36, pp. 953–964, July 1988.

[11] K.M. Buckley and X.L. Xu, "Reduced-dimension broad-band source localization: Preprocessor design," in *Advanced Algorithms for Sig. Proc. III* (F.T. Luk, ed.), vol. 975, San Diego, California, USA, pp. 368–376, SPIE – The International Society for Optical Engineering, Aug. 1988.

[12] D.K. Cheng, "Optimization techniques for antenna arrays," *Proc. IEEE*, vol. 59, pp. 1664–1674, Dec. 1971.

[13] D.K. Cheng, *Field and Wave Electromagnetics.* Reading, Massachusetts: Addison-Wesley, second ed., 1989.

[14] T. Chou, "Broadband frequency-independent beamforming," Master's thesis, MIT, Jan. 1995.

[15] T. Chou, "Frequency-independent beamformer with low response error," in *Proc. IEEE Int. Conf. Acoust. Speech Sig. Process. (ICASSP-95)*, Detroit, USA, pp. 2995–2998, May 1995.

[16] Y.L. Chow, "On grating plateaux of nonuniformly spaced arrays," *IEEE Trans. Antennas Propagat.*, vol. AP-13, pp. 208–215, Mar. 1965.

[17] K.L. Chung, *A Course in Probability Theory.* New York: Harcourt, Brace & World, Inc., 1968.

[18] C.A. Coulson and A. Jeffrey, *Waves: A mathematical approach to the common types of wave motion.* London: Longman Group Ltd, 1977.

[19] P.J. Davis and P. Rabinowitz, *Methods of Numerical Integration.* San Diego: Academic Press Inc., 1984.

[20] J.H. Doles III and F.D. Benedict, "Broad-band array design using the asymptotic theory of unequally spaced arrays," *IEEE Trans. Antennas Propagat.*, vol. 36, pp. 27–33, Jan. 1988.

[21] C.L. Dolph, "A current distribution for broadside arrays which optimizes the relationship between beam width and side-lobe level," *Proc. IRE*, vol. 34, pp. 335–348, June 1946.

[22] R.S. Elliot, "Design of line source antennas for narrow beamwidth and asymmetric sidelobes," *IEEE Trans. Antennas Propagat.*, vol. AP-23, pp. 100–107, Jan. 1975.

[23] R.S. Elliot, "Design of line source antennas for sum patterns with sidelobes of individually arbitrary heights," *IEEE Trans. Antennas Propagat.*, vol. AP-24, pp. 76–83, Jan. 1976.

[24] R.S. Elliot, "On discretizing continuous aperture distributions," *IEEE Trans. Antennas Propagat.*, vol. AP-25, pp. 617–621, Sept. 1977.

[25] M.H. Er, “Technique for antenna array pattern synthesis with controlled broad nulls,” *IEE Proc. H, Microwaves, Opt. & Antennas*, vol. 135, pp. 375–380, Dec. 1988.

[26] M.H. Er and A. Cantoni, “Derivative constraints for broadband element space antenna array processors,” *IEEE Trans. Acoust. Speech Sig. Proc.*, vol. ASSP-31, pp. 1378–1393, Dec. 1983.

[27] R.J. Evans and K.M. Ahmed, “Robust adaptive array antennas,” *J. Acoust. Soc. Amer.*, vol. 71, pp. 384–394, Feb. 1982.

[28] J.L. Flanagan, D.A. Berkeley, G.W. Elko, J.E. West, and M.M. Sondhi, “Autodirective microphone systems,” *Acustica*, vol. 73, pp. 58–71, 1991.

[29] J.L. Flanagan, J.D. Johnston, R. Zahn, and G.W. Elko, “Computer steered microphone arrays for sound transduction in large rooms,” *J. Acoust. Soc. Amer.*, vol. 78, pp. 1508–1518, Nov. 1985.

[30] P. Forster and G. Vezzosi, “Application of spheroidal sequences to array processing,” in *Proc. IEEE Int. Conf. Acoust. Speech Sig. Process. (ICASSP-87)*, pp. 2267–2271, 1987.

[31] B. Friedlander and A.J. Weiss, “Direction finding for wide-band signals using an interpolated array,” *IEEE Trans. Sig. Proc.*, vol. 41, pp. 1618–1634, Apr. 1993.

[32] O.L. Frost III, “An algorithm for linearly constrained adaptive array processing,” *Proc. IEEE*, vol. 60, pp. 926–935, Aug. 1972.

[33] W.F. Gabriel, “Adaptive processing array systems,” *Proc. IEEE*, vol. 80, pp. 152–162, Jan. 1992.

[34] M.M. Goodwin and G.W. Elko, “Constant beamwidth beamforming,” *Proc. IEEE Int. Conf. Acoust. Speech Sig. Process. (ICASSP-93)*, vol. 1, pp. 169–172, 1993.

[35] Y. Grenier, “A microphone array for car environments,” *Speech Communication*, vol. 12, pp. 25–39, Mar. 1993.

[36] J.W.R. Griffiths, “Adaptive array processing: a tutorial,” *IEE Proc. H, Microwaves, Opt. & Antennas*, vol. 130, pp. 3–10, Feb. 1983.

[37] L.J. Griffiths and C.W. Jim, “An alternative approach to linearly constrained adaptive beamforming,” *IEEE Trans. Antennas Propagat.*, vol. AP-30, pp. 27–34, Jan. 1982.

[38] T.P. Guella and R.M. Davis, “Synthesis of notched antenna patterns for wideband processing,” *IEEE Trans. Antennas Propagat.*, vol. 43, pp. 1465–1471, Dec. 1995.

[39] P.S. Hacker and H.E. Schrank, “Range distance requirements for measuring low and ultralow sidelobe antenna patterns,” *IEEE Trans. Antennas Propagat.*, vol. AP-30, pp. 956–965, Sept. 1982.

[40] R.C. Hansen, “Measurement distance effects on low sidelobe patterns,” *IEEE Trans. Antennas Propagat.*, vol. AP-32, pp. 591–594, June 1984.

[41] R.C. Hansen, “Array pattern control and synthesis,” *Proc. IEEE*, vol. 80, pp. 141–151, Jan. 1992.

[42] S. Haykin, ed., *Array Signal Processing.* Englewood Cliffs, New Jersey: Prentice-Hall Inc., 1985.

[43] S. Haykin and J. Kesler, “Relation between the radiation pattern of an array and the two-dimensional discrete Fourier transform,” *IEEE Trans. Antennas Propagat.*, vol. AP-23, pp. 419–420, May 1975.

[44] E. Hixson and K. Au, “Broadband constant beamwidth acoustical arrays,” Tech. Rep. 19, Acoustics Research Lab, U.T. Austin, 1970.

[45] J.E. Hudson, *Adaptive Array Principles.* Stevenage, UK: Peregrinus, 1981.

[46] H. Hung and M. Kaveh, “Focussing matrices for coherent signal-subspace processing,” *IEEE Trans. Acoust. Speech Sig. Proc.*, vol. 36, pp. 1272–1281, Aug. 1988.

[47] A. Ishimaru, “Theory of unequally-spaced arrays,” *IRE Trans. Antennas Propagat.*, vol. AP-10, pp. 691–702, Nov. 1962.

[48] A. Ishimaru and Y.S. Chen, “Thinning and broadbanding antenna arrays by unequal spacings,” *IEEE Trans. Antennas Propagat.*, vol. AP-13, pp. 34–42, Jan. 1965.

[49] T.H. Ismail and M.M. Dawoud, “Null steering in phased arrays by controlling the element positions,” *IEEE Trans. Antennas Propagat.*, vol. 39, pp. 1561–1566, Nov. 1991.

[50] D.H. Johnson and D.E. Dudgeon, *Array Signal Proessing: Concepts and Techniques.* Englewood Cliffs, New Jersey: Prentice-Hall Inc., 1993.

[51] W. Kellerman, “A self-steering digital microphone array,” *Proc. IEEE Int. Conf. Acoust. Speech Sig. Process. (ICASSP-91)*, vol. 5, pp. 3581–3584, 1991.

[52] F. Khalil, J.P. Jullien, and A. Gilloire, “Microphone array for sound pickup in teleconference systems,” *J. Audio Engineering Society*, vol. 42, pp. 691–700, Sept. 1994.

[53] R. Kincaid, “RC filter design by the numbers,” *The Electronic Engineer*, pp. 57–61, Oct. 1968.

[54] C.C. Ko, "Broadband power inversion array with maximally flat response at null directions," *IEE Proc. F, Commun., Radar & Signal Process.*, vol. 136, pp. 161–167, Aug. 1989.

[55] J. Krolik and D. Swingler, "Focused wide-band array processing by spatial resampling," *IEEE Trans. Acoust. Speech Sig. Proc.*, vol. 38, pp. 356–360, Feb. 1990.

[56] T.I. Laakso, V. Välimäki, M. Karjalainen, and U.K. Laine, "Splitting the unit delay," *IEEE Sig. Proc. Mag.*, vol. 13, pp. 30–60, Jan. 1996.

[57] J. Lardies and J.P. Guilhot, "Realization of a broadband constant beamwidth endfire line array," *Acoust. Letters*, vol. 10, no. 8, pp. 122–127, 1987.

[58] H.B. Lee and M.S. Wengrovitz, "Resolution threshold of beamspace MUSIC for two closely spaced emitters," *IEEE Trans. Acoust. Speech Sig. Proc.*, vol. 38, pp. 1545–1559, Sept. 1990.

[59] T.S. Lee, "Efficient wideband source localization using beamforming invariance technique," *IEEE Trans. Sig. Proc.*, vol. 42, pp. 1376–1387, June 1994.

[60] M.T. Ma, *Theory and Application of Antenna Arrays.* New York: Wiley, 1974.

[61] R.J. Mailloux, *Phased Array Antenna Handbook.* Boston: Artech House Inc., 1994.

[62] R.J. Mailloux, "Covariance matrix augmentation to produce adaptive array pattern troughs," *Electronics Letters*, vol. 31, pp. 771–772, May 1995.

[63] R.A. Monzingo and T.W. Miller, *Introduction to Adaptive Arrays.* New York: Wiley, 1980.

[64] D. Mustafa and K. Glover, "Controllers which satisfy a closed-loop H_∞-norm bound and maximize an entropy integral," in *27th Conference on Decision and Control*, vol. 2, Austin, Texas, pp. 959–964, Dec. 1988.

[65] National Instruments Corporation, *AT-MIO-16F-5 User Manual*, June 1992 ed., 1992.

[66] National Instruments Corporation, *NI-DAQ Function Reference Manual for DOS/Windows/Lab Windows, version 4.3*, 1992.

[67] B.P. Ng, M.H. Er, and C. Kot, "A flexible array synthesis method using quadratic programming," *IEEE Trans. Antennas Propagat.*, vol. 41, pp. 1541–1550, Nov. 1993.

[68] B.P. Ng, M.H. Er, and C. Kot, "Linear array geometry synthesis with minimum sidelobe level and null control," *IEE Proc., Microw. Antennas Propag.*, vol. 141, pp. 162–166, June 1994.

[69] C.A. Olen and R.T. Compton Jr, "A numerical pattern synthesis algorithm for arrays," *IEEE Trans. Antennas Propagat.*, vol. 38, pp. 1666–1676, Oct. 1990.

[70] A.V. Oppenheim and R.W. Schafer, *Discrete-Time Signal Processing.* Englewood Cliffs, New Jersey: Prentice-Hall Inc., 1989.

[71] A.V. Oppenheim, A.S. Willsky, and I.T. Young, *Signals and Systems.* Englewood Cliffs, New Jersey: Prentice-Hall Inc., 1983.

[72] H.J. Orchard, R.S. Elliot, and G.J. Stern, "Optimizing the synthesis of shaped beam antenna patterns," *IEE Proc. H, Microwaves, Opt. & Antennas*, vol. 132, pp. 63–68, Feb. 1985.

[73] R.E.A.C. Paley and N. Weiner, *Fourier Transforms in the Complex Domain.* Providence, RI: American Mathematical Society, 1934.

[74] T.W. Parks and C.S. Burrus, *Digital Filter Design.* New York: Wiley, 1987.

[75] R.G. Pridham and R.A. Mucci, "A novel approach to digital beamforming," *J. Acoust. Soc. Amer.*, vol. 63, pp. 425–434, Feb. 1978.

[76] R.W. Redlich, "Iterative least squares synthesis of nonuniformly spaced linear arrays," *IEEE Trans. Antennas Propagat.*, vol. 21, pp. 106–108, Jan. 1973.

[77] L. Robin, *Fonctions Sphériques de Legendre et Fonctions Sphéroïdales.* Paris: Gauthier-Villars, 1957.

[78] R. Roy and T. Kailath, "ESPRIT – estimation of signal paramters via rotational invariance techniques," *IEEE Trans. Acoust. Speech Sig. Proc.*, vol. 37, pp. 984–995, July 1989.

[79] S.A. Schelkunoff, "A mathematical theory of linear arrays," *Bell Sys. Tech. J.*, vol. 22, pp. 80–107, 1943.

[80] R.O. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. Antennas Propagat.*, vol. AP-34, pp. 276–280, Mar. 1986.

[81] H.F. Silverman, "Some analysis of microphone arrays for speech data acquisition," *IEEE Trans. Acoust. Speech Sig. Proc.*, vol. ASSP-35, pp. 1699–1712, Dec. 1987.

[82] M.I. Skolnik, "Nonuniform arrays," in *Antenna Theory, Pt 1* (R.E. Collin and F.J. Zucker, eds.), ch. 6, pp. 207–234, New York: McGraw-Hill Inc., 1969.

[83] R. Smith, "Constant beamwidth receiving arrays for broad band sonar systems," *Acustica*, vol. 23, pp. 21–26, 1970.

[84] H. Steyskal, "Synthesis of antenna patterns with prescribed nulls," *IEEE Trans. Antennas Propagat.*, vol. AP-30, pp. 273–279, Mar. 1982.

[85] H. Steyskal, "Wide-band nulling performance versus number of pattern constraints for an array antenna," *IEEE Trans. Antennas Propagat.*, vol. AP-31, pp. 159–163, Jan. 1983.

[86] H. Steyskal, R.A. Shore, and R.L. Haupt, “Methods for null control and their effects on the radiation pattern,” *IEEE Trans. Antennas Propagat.*, vol. AP-34, pp. 404–409, Mar. 1986.

[87] P. Stoica and A. Nehorai, “MUSIC, maximum likelihood, and Cramer-Rao bound,” *IEEE Trans. Acoust. Speech Sig. Proc.*, vol. 37, pp. 720–741, May 1989.

[88] P. Stoica and A. Nehorai, “Comparative performance study of element-space and beam-space MUSIC estimators,” *Circuits Systems Sig. Proc.*, vol. 10, no. 3, pp. 285–292, 1991.

[89] B.E. Stuckman and J.C. Hill, “Method of null steering in phased array antenna systems,” *Electronics Letters*, vol. 26, pp. 1216–1218, July 1990.

[90] D.N. Swingler and J. Krolik, “Source location bias in the coherently focused high-resolution broad-band beamformer,” *IEEE Trans. Acoust. Speech Sig. Proc.*, vol. 37, pp. 143–145, Jan. 1989.

[91] K. Takao and K. Komiyama, “An adaptive antenna for rejection of wideband interference,” *IEEE Trans. Aerosp. Electron. Sys.*, vol. AES-16, pp. 452–459, July 1980.

[92] T.T. Taylor, “Design of line-source antennas for narrow beamwidth and low sidelobes,” *IRE Trans. Antennas Propagat.*, vol. AP-3, pp. 16–28, Jan. 1955.

[93] A. Tennant, M.M. Dawoud, and A.P. Anderson, “Array pattern nulling by element position perturbations using a genetic algorithm,” *Electronics Letters*, vol. 30, pp. 174–176, Feb. 1994.

[94] F.I. Tseng, “Design of array and line-source antennas for Taylor patterns with a null,” *IEEE Trans. Antennas Propagat.*, vol. AP-27, pp. 474–479, July 1979.

[95] D.G. Tucker, “Arrays with constant beam-width over a wide frequency range,” *Nature*, vol. 180, pp. 496–497, Sept. 1957.

[96] P.P. Vaidyanathan, *Multirate Systems and Filter Banks.* Englewood Cliffs, New Jersey: Prentice-Hall Inc., 1993.

[97] B.D. Van Veen and K.M. Buckley, “Beamforming: a versatile approach to spatial filtering,” *IEEE ASSP Mag.*, vol. 5, pp. 4–24, Apr. 1988.

[98] A.T. Villeneuve, “Taylor patterns for discrete arrays,” *IEEE Trans. Antennas Propagat.*, vol. AP-32, pp. 1089–1093, Oct. 1984.

[99] H. Wang and M. Kaveh, “Coherent signal-subspace processing for the detection and estimation of angles of arrival of multiple wide-band sources,” *IEEE Trans. Acoust. Speech Sig. Proc.*, vol. ASSP-33, pp. 823–831, Aug. 1985.

[100] M. Wax, T.J. Shan, and T. Kailath, “Spatio-temporal spectral analysis by eigenstructure methods,” *IEEE Trans. Acoust. Speech Sig. Proc.*, vol. ASSP-32, pp. 817–827, Aug. 1984.

[101] B. Widrow, P.E. Mantey, L.J. Griffiths, and B.B. Goode, “Adaptive antenna systems,” *Proc. IEEE*, vol. 55, pp. 2143–2159, Dec. 1967.

[102] C. Winter, “Using continuous apertures discretely,” *IEEE Trans. Antennas Propagat.*, vol. AP-25, pp. 695–700, Sept. 1977.

[103] P. Woodward, “A method of calculating the field over a plane aperture required to produce a given polar diagram,” *J. IEE*, vol. 93, pp. 1554–1558, 1947.

[104] P. Woodward and J. Lawson, “The theoretical pecision with which an arbitrary radiation pattern may be obtained from a source of finite size,” *J. IEE*, vol. 95, pp. 363–369, 1948.

[105] L. Wu and A. Zielinski, “An iterative method for array pattern synthesis,” *IEEE J. Oceanic Eng.*, vol. 18, pp. 280–286, July 1993.

[106] X.L. Xu and K.M. Buckley, “Statistical performace comparison of MUSIC in element-space and beam-space,” in *Proc. IEEE Int. Conf. Acoust. Speech Sig. Process.(ICASSP 89)*, pp. 2124–2127, 1989.

[107] X.L. Xu and K.M. Buckley, “Bias analysis of the MUSIC location estimator,” *IEEE Trans. Sig. Proc.*, vol. 40, pp. 2559–2569, Oct. 1992.

[108] X.L. Xu and K.M. Buckley, “An analysis of beam-space source localization,” *IEEE Trans. Sig. Proc.*, vol. 41, pp. 501–504, Jan. 1993.

[109] Q.T. Zhang, “Probability of resolution of the MUSIC algorithm,” *IEEE Trans. Sig. Proc.*, vol. 43, pp. 978–987, Apr. 1995.

[110] Q.T. Zhang, “A statistical resolution theory of the beamformer-based spatial spectrum for determining the directions of signals in white noise,” *IEEE Trans. Sig. Proc.*, vol. 43, pp. 1867–1873, Aug. 1995.

[111] M.D. Zoltowski, G.M. Kautz, and S.D. Silverstein, “Beamspace root-MUSIC,” *IEEE Trans. Sig. Proc.*, vol. 41, pp. 344–364, Jan. 1993.