4

Broadband Processing

4.1 Tapped-Delay Line Structure
   4.1.1 Description
   4.1.2 Frequency Response
   4.1.3 Optimization
   4.1.4 Adaptive Algorithm
   4.1.5 Minimum Mean Square Error Design
      4.1.5.1 Derivation of Constraints
      4.1.5.2 Optimization

4.2 Partitioned Realization
   4.2.1 Generalized Side-Lobe Canceler
   4.2.2 Constrained Partitioned Realization
   4.2.3 General Constrained Partitioned Realization
      4.2.3.1 Derivation of Constraints
      4.2.3.2 Optimization

4.3 Derivative Constrained Processor
   4.3.1 First-Order Derivative Constraints
   4.3.2 Second-Order Derivative Constraints
   4.3.3 Optimization with Derivative Constraints
      4.3.3.1 Linear Array Example
   4.3.4 Adaptive Algorithm
   4.3.5 Choice of Origin

4.4 Correlation Constrained Processor

4.5 Digital Beamforming

4.6 Frequency Domain Processing
   4.6.1 Description
   4.6.2 Relationship with Tapped-Delay Line Structure Processing
      4.6.2.1 Weight Relationship
      4.6.2.2 Matrix Relationship
      4.6.2.3 Derivation of $R_f(k)$
      4.6.2.4 Array with Presteering Delays
      4.6.2.5 Array without Presteering Delays
      4.6.2.6 Discussion and Comments
   4.6.3 Transformation of Constraints
      4.6.3.1 Point Constraints
      4.6.3.2 Derivative Constraints

4.7 Broadband Processing Using Discrete Fourier Transform Method
   4.7.1 Weight Estimation
   4.7.2 Performance Comparison
4.7.2.1 Effect of Filter Length
4.7.2.2 Effect of Number of Elements in Array
4.7.2.3 Effect of Interference Power
4.7.3 Computational Requirement Comparison
4.7.4 Schemes to Reduce Computation
4.7.4.1 Limited Number of Bins Processing
4.7.4.2 Parallel Processing Schemes
4.7.4.2.1 Parallel Processing Scheme 1
4.7.4.2.2 Parallel Processing Scheme 2
4.7.4.2.3 Parallel Processing Scheme 3
4.7.5 Discussion
4.7.5.1 Higher SNR with Less Processing Time
4.7.5.2 Robustness of DFT Method

4.8 Performance
Notation and Abbreviations
References

The beamformer structure of Figure 2.1 discussed earlier is for narrowband signals. As the signal bandwidth increases, beamformer performance using this structure starts to deteriorate [Rod79]. For processing broadband signals, a tap delay line (TDL) structure shown in Figure 4.1 is normally used [Rod79, May81, Voo92, Com88, Ko81, Ko87, Nun83, Yeh87, Sco83]. A lattice structure consisting of a cascade of simple lattice filters sometimes is also used [Ale87, Lin86, Iig85, Soh84], offering certain processing advantages.

Although the TDL structure with constrained optimization is the commonly used structure for broadband array signal processing, alternative methods have been proposed.

FIGURE 4.1
Broadband processor with tapped delay line structure.
These include adaptive nonlinear schemes, which maximizes the signal-to-noise ratio (SNR) subject to additional constraints [Win72]; a variation of the Davis beamformer [Dav67], which adapts one filter at a time to speed up convergence [Ko90]; a composite system that also utilizes a derivative of beam pattern in the feedback loop to control the weights [Tak80] to reject wideband interference; optimum filters that specify rejection response [Sim83]; master and slave processors [Hua90]; a hybrid method that uses an orthogonal transformation on data available from the TDL structure before applying weights [Che95] to improve its performance in multipath environments; the weighted Tschebysheff method [Nor94]; and the two-sided correlation transformation method [Val95].

In this chapter, details on an array processor using the TDL structure and its partitioned realization to process broadband array signals are provided, the time domain and frequency domain methods are described, and details on deriving various constraints are given [God95, God97, God99]. The treatment presented here is for solving a constrained beamforming problem, assuming that the look direction is known. It can easily be extended to the case when a reference signal is available.

4.1 Tapped-Delay Line Structure

In this section, a TDL structure for broadband antenna array processing is described, its frequency response and optimization are discussed, an LMS algorithm to estimate the solution of the point-constrained optimization problem is developed, and a design using minimum mean square error (MSE) between the frequency response of the processor and the desired response is presented.

4.1.1 Description

Figure 4.1 shows a general structure of a broadband antenna array processor consisting of L antenna elements, steering delays \(T_l(\phi_0, \theta_0), l = 1, \ldots, L\) and a delay line section of \(J - 1\) delays with inter-tap delay spacing \(T\). The steering delays \(T_l(\phi_0, \theta_0), l = 1, \ldots, L\) in front of each element are pure time delays and are used to steer the array in a given look direction \((\phi_0, \theta_0)\). If \(\tau_l(\phi_0, \theta_0)\) denotes the time taken by the plane wave arriving from direction \((\phi_0, \theta_0)\), and measured from the reference point to the \(l\)th element, then the steering delay \(T_l(\phi_0, \theta_0)\) may be selected using

\[
T_l(\phi_0, \theta_0) = T_0 + \tau_l(\phi_0, \theta_0), \quad l = 1, \ldots, L \tag{4.1.1}
\]

where \(T_0\) is a bulk delay such that \(T_l(\phi_0, \theta_0) > 0\) for all \(l\).

If \(s(t)\) denotes the signal induced on an element present at the center of the coordinate system due to a broadband source of power density \(S(f)\) in direction \((\phi, \theta)\), then the signal induced on the \(l\)th element is given by \(s(t + \tau_l(\phi, \theta))\), as discussed in Chapter 2.

Let \(x_{l}(t)\) denote the output of the \(l\)th sensor presteered in \((\phi_0, \theta_0)\). It is given by

\[
x_{l}(t) = s(t + \tau_l(\phi, \theta) - T_l(\phi_0, \theta_0)) \tag{4.1.2}
\]

For a source in \((\phi_0, \theta_0)\) it becomes \(x_{l}(t) = s(t - T_0)\), yielding identical wave forms after presteering delays.
The TDL structure shown in Figure 4.1 following the steering delay on each channel is a finite impulse response (FIR) filter. The coefficients of these filters are constrained to specify the frequency response in the look direction. It should be noted that these coefficients are real compared to complex weights of the narrowband processor.

It follows from Figure 4.1 that the output \( y(t) \) of the processor is given by

\[
y(t) = \sum_{l=1}^{L} \sum_{k=1}^{J} x_l(t-(k-1)T)w_{lk}
\]  

(4.1.3)

where \( w_{lk} \) denotes the weight on the \( k \)-th tap of the \( l \)-th channel. Note that the \( k \)-th tap output corresponds to the output after \((k-1)\) delays. Thus, first tap output corresponds to the output of presteering delays and before any tapped delays section, the second tap output corresponds to the output after one delay and \( J \)-th tap output corresponds to the output after \( J-1 \) delays.

Let \( \mathbf{W} \) defined by

\[
\mathbf{W}^T = [w_1^T, w_2^T, \ldots, w_J^T]
\]  

(4.1.4)

denote \( LJ \) weights of the filter structure, with \( w_m \) denoting the column of \( L \) weights on the \( m \)-th tap.

Define an \( L \)-dimensional vector \( \mathbf{x}(t) \) to denote array signals after presteering delays, that is,

\[
\mathbf{x}(t) = [x_1(t), x_2(t), \ldots, x_L]^T
\]  

(4.1.5)

and an \( LJ \)-dimensional vector \( \mathbf{X}(t) \) to denote array signals across the TDL structure, that is,

\[
\mathbf{X}^T(t) = [\mathbf{x}^T(t), \mathbf{x}^T(t-T), \ldots, \mathbf{x}^T(t-(J-1)T)]
\]  

(4.1.6)

It follows from (4.1.3) to (4.1.6) that the output \( y(t) \) of the processor in the vector notation becomes

\[
y(t) = \mathbf{W}^T\mathbf{X}(t)
\]  

(4.1.7)

If \( \mathbf{X}(t) \) can be modeled as a zero-mean stochastic process, then the mean output power of the processor for a given \( \mathbf{W} \) is given by

\[
P(\mathbf{W}) = \mathbb{E}[y^2(t)]
\]  

(4.1.8)

\[
= \mathbf{W}^T\mathbf{R}\mathbf{W}
\]

where

\[
\mathbf{R} = \mathbb{E}[\mathbf{X}(t)\mathbf{X}^T(t)]
\]  

(4.1.9)

is an \( LJ \times LJ \) dimensional real matrix and denotes the array correlation matrix with its elements representing the correlation between various tap outputs. The correlation between the outputs of \( m \)-th tap on the \( l \)-th channel and \( n \)-th tap on the \( k \)-th channel is given by

\[
(R_{m,n})_{l,k} = \mathbb{E}[x_l(t-(m-1)T)x_k(t-(n-1)T)]
\]  

(4.1.10)
Note that the $L \times L$ matrix $R_{mn}$ denotes the correlation between the array outputs at the mth and nth taps, that is, after $(m-1)$ and $(n-1)$ delays.

Substituting from (4.1.2), it follows that

$$
(R_{m,n})_{l,k} = \rho[(m-n)T + T_k(\phi_0, \theta_0) - T_k(\phi_0, \theta_0) + \tau_k(\phi, \theta) - \tau_j(\phi, \theta)]
$$

(4.1.11)

where $\rho(\tau)$ denotes the correlation function of $s(t)$, that is,

$$
\rho(\tau) = E[s(t)s(t+\tau)]
$$

(4.1.12)

The correlation function is related to the spectrum of the signal by the inverse Fourier transform, that is,

$$
\rho(\tau) = \int_{-\infty}^{\infty} S(f) e^{i2\pi \tau f} df
$$

(4.1.13)

Thus, from known spectra of sources and their arrival directions, the correlation matrix may be calculated. In practice, it can also be estimated by measuring signals at the output of various taps.

For $M$ uncorrelated directional sources, the array correlation matrix is the sum of correlation matrices due to each source, that is,

$$
R = \sum_{l=1}^{M} R_l
$$

(4.1.14)

where $R_l$ is the array correlation matrix due to the $l$th source in direction $(\phi, \theta)$.

Let $R_S$ denote the array correlation matrix due to the signal source, that is, a source in the look direction, and $R_N$ denote the array correlation matrix due to noise, that is, unwanted directional sources and other noise. The mean output signal power $P_S(W)$ and mean output noise power $P_N(W)$ for a given weight vector are, respectively, given by

$$
P_S(W) = W^T R_S W
$$

(4.1.15)

and

$$
P_N(W) = W^T R_N W
$$

(4.1.16)

The output SNR for given weights is

$$
\text{SNR}(W) = \frac{P_S(W)}{P_N(W)} = \frac{W^T R_S W}{W^T R_N W}
$$

(4.1.17)
4.1.2 Frequency Response

Assume that the signal induced on an element at the center of the coordinate system due to a monochromatic plane wave of frequency \( f \) can be represented in complex notation as \( e^{j2\pi ft} \). Thus, the induced signal on the \( l \)th element after the steering delay due to a plane wave arriving in direction \((\phi, \theta)\) becomes \( e^{j2\pi f(t + \tau_l(\phi, \theta) - T_l(\phi_0, \theta_0))} \). The frequency response \( H(f, \phi, \theta) \) of the processor to a plane wave front arriving in direction \((\phi, \theta)\) is then given by

\[
H(f, \phi, \theta) = \sum_{l=1}^{L} e^{j2\pi f(T_l(\phi, \theta))} e^{-j2\pi fT_l(\phi_0, \theta_0)} \sum_{k=1}^{J} w_k e^{-j2\pi(f(k-1)T_k)}
\]

\[
= \mathbf{S}^T(f, \phi, \theta)\mathbf{T}(f) \sum_{k=1}^{J} w_k e^{-j2\pi f(k-1)T_k}
\]  

(4.1.18)

where \( \mathbf{T}(f) \) is a diagonal matrix of steering delays given by

\[
\mathbf{T}(f) = \begin{bmatrix}
e^{-j2\pi fT_l(\phi_0, \theta_0)} & 0 \\
e^{-j2\pi fT_l(\phi, \theta)} & 0 \\
0 & e^{-j2\pi fT_l(\phi_0, \theta_0)}
\end{bmatrix}
\]

(4.1.19)

and \( \mathbf{S}(f, \phi, \theta) \) is an L-dimensional vector defined as

\[
\mathbf{S}^T(f, \phi, \theta) = \begin{bmatrix} e^{j2\pi fT_1(\phi, \theta)} & e^{j2\pi fT_2(\phi, \theta)} & \ldots & e^{j2\pi fT_L(\phi, \theta)} \end{bmatrix}
\]

(4.1.20)

It follows from (4.1.1), (4.1.19), and (4.1.20) that

\[
\mathbf{S}^T(f, \phi_0, \theta_0)\mathbf{T}(f) = a(f)[1, 1, \ldots, 1]
\]

(4.1.21)

where

\[
a(f) = e^{-j2\pi fT_0}
\]

(4.1.22)

In this case, the frequency response of the array steered in the look direction \((\phi_0, \theta_0)\) is given by

\[
H(f, \phi_0, \theta_0) = a(f) \sum_{k=1}^{J} f_k e^{-j2\pi f(k-1)T_k}
\]

(4.1.23)

where

\[
f_k = \mathbf{T}^T \mathbf{w}_k, \quad k = 1, 2, \ldots, J
\]

(4.1.24)

with \( \mathbf{1} \) denoting a vector of ones.
Let \( f \) be a \( J \)-dimensional constraint vector defined as
\[
\mathbf{f} = \begin{bmatrix} f_1, f_2, \ldots, f_J \end{bmatrix}^T
\]  
(4.1.25)
and \( C \) be an \( L J \times J \) constraint matrix defined as
\[
\mathbf{C} = \begin{bmatrix}
1 & 0 & \ldots & 0 \\
0 & 1 & \ldots & 0 \\
& \ddots & \ddots & \ddots \\
0 & \ldots & 0 & 1
\end{bmatrix}
\]  
(4.1.26)
The \( J \) constraints defined by (4.1.24) can now be expressed as
\[
\mathbf{C}^T \mathbf{W} = \mathbf{f}
\]  
(4.1.27)
Since \( a(\mathbf{f}) \) given by (4.1.22) corresponds to a pure time delay, the \( J \) constraints \( \{f_k\} \) can be used to specify the frequency response in the direction \((\phi_0, \theta_0)\).

The processor can be forced to have a flat frequency response in the look direction by selecting \( \mathbf{f} \) as follows:
\[
f_i = \begin{cases} 
1 & i = k_0 \\
0 & i \neq k_0
\end{cases}
\]  
(4.1.28)
where \( k_0 \) is a parameter, which can itself be optimized. Frequently, \( k_0 \) is taken as \( J/2 \) for \( J \), an even number, and \((J + 1)/2\) for \( J \), an odd number, since for a sufficiently large \( J \) this gives close to optimum performance.

### 4.1.3 Optimization

The frequency response of an array processor in the look direction can be fixed using the \( J \) constraints in (4.1.27). The processor can minimize the non–look direction noise when weights are selected by minimizing the total mean output power such that (4.1.27) is satisfied. Thus, in situations where one is interested in finding array weights, such that the array processor minimizes the total noise and has the specified response in the look direction, the following constrained beamforming problem is considered:

\[
\begin{align*}
\text{minimize} & \quad \mathbf{W}^T \mathbf{R} \mathbf{W} \\
\text{subject to} & \quad \mathbf{C}^T \mathbf{W} = \mathbf{f}
\end{align*}
\]  
(4.1.29)
where \( \mathbf{f} \) is a \( J \)-dimensional vector that specifies the frequency response in the look direction and \( \mathbf{C} \) is an \( L J \times J \) constraint matrix.

Let \( \hat{\mathbf{W}} \) denote the solution of the above problem. The solution is obtained by the Lagrange multipliers method [Bry69, Lue69, Pie69]. This method transforms the constrained problem into an unconstrained problem by adding the constraint function to the cost function using a \( J \)-dimensional vector of undetermined Lagrange multipliers \( \lambda \) to generate a new cost function. Let \( J(\mathbf{W}) \) denote the cost function for the present problem. It is given by
where 1/2 is added to simplify the mathematics.

Taking the gradient of (4.1.30) with respect to $W$,

$$\nabla_W J(W) = RW + C\lambda$$  \hfill (4.1.31)

At the solution point, the cost function gradient is zero. Thus,

$$RW + C\lambda = 0$$  \hfill (4.1.32)

Assuming that the inverse of the array correlation matrix $R$ exists, $\hat{W}$ may be expressed in terms of Lagrange multipliers as

$$\hat{W} = -R^{-1}C\lambda$$  \hfill (4.1.33)

Since $\hat{W}$ satisfies the constraint $C\hat{W} = f$, it follows from (4.1.33) that

$$-C^T R^{-1}C\lambda = f$$  \hfill (4.1.34)

An expression for Lagrange multipliers may be found from (4.1.34), yielding

$$\lambda = -(C^T R^{-1}C)^{-1} f$$  \hfill (4.1.35)

Substituting for Lagrange multipliers in (4.1.33) from (4.1.35), an expression for the optimal weights [Fro72] follows:

$$\hat{W} = R^{-1}C(C^T R^{-1}C)^{-1} f$$  \hfill (4.1.36)

Let $\hat{P}$ denote the mean output power of the processor using optimal weights, that is,

$$\hat{P} = \hat{W}^T R\hat{W}$$  \hfill (4.1.37)

Substituting for $\hat{W}$ from (4.1.36),

$$\hat{P} = f^T(C^T R^{-1}C)^{-1} f$$  \hfill (4.1.38)

The point-constraint minimization problem (4.1.29) specifies $J$ constraints on the weights such that the sum of $L$ weights on all channels before the $j$th delay is equal to $f_j$. For all pass frequency responses in the look direction, all but one $f_i$, $i = 1, 2, \ldots, J$ are selected to be equal to zero. For $i$'s close to $J/2$, $f_j$ is taken to be unity. Thus, the constraints specify that the sum of weights across the array is zero, except one near the middle of the filter that is equal to unity.
Thus, for all pass frequency responses when \( f_i, i = 1, \ldots, J \) are selected as

\[
 f_i = \begin{cases} 
 1 & i = k_0 \\
 0 & i \neq k_0 
\end{cases}
\]  
(4.1.39)
equation (4.1.38) becomes

\[
 \hat{p} = (C^T R^{-1} C)^{-1}_{k_0,k_0}
\]  
(4.1.40)

Application of broadband beamforming structures using TDL filters to mobile communications has been considered in [Win94, Des92, Ish95, Koh92] to overcome multipath fading and large delay spread in TDMA as well as in CDMA systems.

### 4.1.4 Adaptive Algorithm

A constrained LMS algorithm to estimate the optimal weights of a narrowband element space processor is discussed in Chapter 3. The corresponding algorithm to estimate the optimal weights of the broadband processor given by (4.3.36) may be developed as follows [Fro72].

Let \( W(n) \) denote the weights estimated at the \( n \)th iteration. At this stage, a new array sample \( X(n + 1) \) is available and the array output using weights \( W(n) \) is given by

\[
y(n) = W^T(n)X(n + 1)
\]  
(4.1.41)

For notational simplicity it is assumed that the \( n \)th iteration coincides with the \( n \)th time sample. The new weight vector \( W(n + 1) \) is calculated by moving in the negative direction of the cost function gradient, that is,

\[
W(n + 1) = W(n) - \mu \nabla_W J(W(n))
\]  
(4.1.42)

where \( J(W(n)) \) is the cost function given by (4.1.30), with \( W \) replaced by \( W(n) \) and \( \mu \) is a positive scalar. Replacing \( R \) with its noisy sample \( X(n + 1)X^T(n + 1) \), it follows from (4.1.31) that

\[
\nabla_W J(W(n)) = X(n + 1)X^T(n + 1)W(n) + C\lambda(n) = y(n)X(n + 1) + C\lambda(n)
\]  
(4.1.43)

where \( \lambda(n) \) denotes the Lagrange multipliers at the \( n \)th iteration. Substituting from (4.1.43) in (4.1.42),

\[
W(n + 1) = W(n) - \mu y(n)X(n + 1) - \mu C\lambda(n)
\]  
(4.1.44)

Assuming that the estimated weights satisfy the constraints at each iteration, it follows from the second equation of (4.1.29) that
Multiplying by $C^T$ on both sides of (4.1.44) and using (4.1.45), it follows that

$$C^T y(n)X(n+1) + C^T C \lambda(n) = 0$$

(4.1.46)

Solving for $\lambda(n)$

$$\lambda(n) = -\left( C^T C \right)^{-1} C^T y(n)X(n+1)$$

(4.1.47)

Substituting in (4.1.44),

$$W(n+1) = W(n) - \mu y(n) PX(n+1)$$

(4.1.48)

where

$$P = I - C \left( C^T C \right)^{-1} C^T$$

(4.1.49)

is a projection operator. It follows from (4.1.45) and (4.1.49) that

$$PW(n) = W(n) - C \left( C^T C \right)^{-1} f$$

(4.1.50)

Thus,

$$W(n) = PW(n) + C \left( C^T C \right)^{-1} f$$

(4.1.51)

and after substitution for $W(n)$, (4.1.48) becomes

$$W(n+1) = P \left[ W(n) - \mu y(n) X(n+1) \right] + F$$

(4.1.52)

where

$$F = C \left( C^T C \right)^{-1} f$$

(4.1.53)

Thus, knowing the array weights $W(n)$, array output, and array sample $X(n+1)$, the new weights $W(n+1)$ can be calculated using the constrained LMS algorithm given by (4.1.52), (4.1.53), and (4.1.49).

The algorithm is initialized at $n = 0$ using

$$W(0) = F$$

(4.1.54)

The initialization of the algorithm using weights equal to $F$ is selected because it denotes the optimal weights in the presence of only white noise, that is, no directional interference. This follows from the fact that the array correlation matrix $R$ in this case is given by

$$R = \sigma_n^2 I$$

(4.1.55)
Substituting in (4.1.36), it follows that

\[ \hat{W} = C(C^T c)^{-1} f \]

\[ = F \]  

(4.1.56)

The convergence analysis of the algorithm may be carried out similar to that for the narrowband case discussed in Chapter 3.

A substantial amount of computation in (4.1.52) is required to compute a multiplication between an \( L \)-dimensional vector and matrix \( P \). The sparse nature of matrix \( C \) allows simplification of the algorithm with reduced computation as follows.

It follows from (4.1.26) that

\[ C^T C = LI \]

(4.1.57)

where \( I \) is an identity matrix.

Substituting in (4.1.53) and (4.1.49) yields

\[ F = \frac{1}{L} C F \]

\[ = \frac{1}{L} \begin{bmatrix} f_1 & 1 & 0 \\ & \ddots & \ddots \\ 0 & & 0 \end{bmatrix} \]

(4.1.58)

and

\[ P = I - \frac{CC^T}{L} \]

\[ = I - \frac{1}{L} \begin{bmatrix} 11^T & 0 \\ & \ddots & \ddots \\ 0 & & 11^T \end{bmatrix} \]

(4.1.59)

From (4.1.52), (4.1.58), and (4.1.59) an update equation in \( w_j(n), j = 0, 1, \ldots, J - 1 \) may be expressed as [Buc86]

\[ w_j(n+1) = \left[ I - \frac{11^T}{L} \right] \left[ w_j(n) - \mu y(n)x(n+1-j) \right] + \frac{f_j}{L} 1 \]

(4.1.60)

where \( w_j(n) \) denotes the \( L \) weights after the \( j \)th tap computed at the \( n \)th iteration, and \( x(n + 1 - j) \) denotes the array signal after the \( j \)th tap. Thus, (4.1.60) allows iterative computation of \( J \) columns of weights separately.

Noting that for an \( L \)-dimensional vector, \( a \)

\[ 1^T a = \sum_{i=1}^{L} a_i \]

(4.1.61)

(4.1.60) may be implemented in summation form as [Fro72]:

© 2004 by CRC Press LLC
\[ w_j(n+1) = w_j(n) - \mu y(n)x(n+1-j) \]

\[ -\frac{1}{L} \sum_{l=1}^{L} [w_j(n) - \mu y(n)x_l(n+1-j)] + \frac{f}{L} 1 \]

(4.1.62)

where \((w_j(n))_l\) denotes the \(l\)th component of the weight vector \(w_j(n)\).

### 4.1.5 Minimum Mean Square Error Design

The processor design considered in Section 4.1.3 by solving constrained optimization problems given by (4.1.29) minimizes the mean output power while maintaining a specified frequency response in the look direction. In this section, a processor design discussed in [Er85] is presented. This processor uses the TDL structure similar to that shown in Figure 4.1. The weights of the processor are estimated to minimize the MSE \(\varepsilon_0\) between the frequency response of the processor in the look direction and the desired look direction response over a frequency band of interest \([f_L, f_H]\), defined as

\[ \varepsilon_0 = \int_{f_L}^{f_H} |A(f, \phi_0, \theta_0) - H(f, \phi_0, \theta_0)|^2 df \]  

(4.1.63)

where \(A(f, \phi, \theta)\) denotes the desired frequency response in direction \((\phi, \theta)\). For a processor to have a flat frequency response in the look direction, it is given by

\[ A(f, \phi_0, \theta_0) = \exp(2\pi ft) \]  

(4.1.64)

where \(\tau\) denotes a delay parameter that may be optimized [Er85].

As the constraints on the weights are designed to minimize the deviation of the processor response from the desired response in the means squared sense, the presteering delays are not necessary. In this case, the presteering delays \(T_l(\phi_0, \theta_0), l = 1, 2, \ldots, L\) are set to zero. This is equivalent to the situation when matrix \(T(f)\) is not included in the frequency response expression (4.1.18).

The processor also allows exact presteering as well as coarse presteering. For the exact presteering case, the steering delays are given by (4.1.1). This case is useful in comparing the performance of the processor using the minimum MSE design with that of the optimal processor discussed in Section 4.1.3. Coarse presteering arises when sampled signals are processed and steering delays are selected as the integer multiples of the sampling time closest to the exact delays required to steer the array in look direction.

In the treatment that follows, it is assumed that steering delays \(T_l(\phi_0, \theta_0), l = 1, 2, \ldots, L\) are included in the design and the frequency response of the processor is given by (4.1.18). However, the values of \(T_l(\phi_0, \theta_0)\) will depend on the case under consideration, that is, no presteering, coarse presteering, or exact presteering.

#### 4.1.5.1 Derivation of Constraints

It follows from (4.1.63) that

\[ \varepsilon_0 = \sigma_0 + W^TQW - 2P^T W \]  

(4.1.65)
where $\sigma_0$ is a scalar given by

$$\sigma_0 = \frac{1}{L} \int H^*(f, \phi_0, \theta_0) H(f, \phi_0, \theta_0) df$$  \hspace{1cm} (4.1.66)$$

$$W^T Q W = \frac{1}{L} \int H^*(f, \phi_0, \theta_0) H(f, \phi_0, \theta_0) df$$  \hspace{1cm} (4.1.67)$$

$$P^T W = \frac{1}{2} \int \left\{ A^*(f, \phi_0, \theta_0) H(f, \phi_0, \theta_0) + H^*(f, \phi_0, \theta_0) A(f, \phi_0, \theta_0) \right\} df$$  \hspace{1cm} (4.1.68)$$

$Q$ is an $L \times L$ dimensional positive, semidefinite symmetrical matrix, and $P$ is an $L$-dimensional vector.

Substituting for $H(f, \phi_0, \theta_0)$ in (4.1.67) and (4.1.68) leads to the following expressions for $Q$ and $P$ [Er85]:

$$Q_{k,1} = \psi \left[ (\tau_i - \tau_j) + (T_i - T_j) + (n - m) T \right]$$  \hspace{1cm} (4.1.69)$$

$$k = i + (m-1)L, \quad l = j + (n-1)L, \quad i, j = 1, 2, \ldots, L, \quad m, n = 1, 2, \ldots, J$$

where

$$\psi(\tau) = \left[ f_i \sin c(2\pi f_i \tau) - f_j \sin c(2\pi f_j \tau) \right]$$  \hspace{1cm} (4.1.70)$$

with

$$\sin c(\alpha) = \frac{\sin \alpha}{\alpha}$$  \hspace{1cm} (4.1.71)$$

and

$$P = \left[ P_1^T, P_2^T, \ldots, P_J^T \right]^T$$  \hspace{1cm} (4.1.72)$$

where

$$\left[ P_{k,1} \right] = \frac{1}{2} \int \left\{ A^*(f, \phi_0, \theta_0) e^{2\pi i (\tau_i - T_i)(k-1)l} + A(f, \phi_0, \theta_0) e^{-2\pi i (\tau_j - T_j)(k-1)l} \right\} df$$  \hspace{1cm} (4.1.73)$$

$$l = 1, 2, \ldots, L, \quad k = 1, 2, \ldots, J$$

Let $\bar{W}$ denote an $L$-dimensional vector that minimizes $\epsilon_0$. Thus,

$$\frac{\partial \epsilon_0}{\partial \bar{W}} \bigg|_{\bar{W} = \bar{W}} = 0$$  \hspace{1cm} (4.1.74)$$
It follows from (4.1.65) and (4.1.74) that \( \tilde{W} \) satisfies

\[
Q\tilde{W} = P
\]  

(4.1.75)

Rewrite (4.1.65) using (4.1.75) as

\[
\varepsilon_0 = (\tilde{W} - W)^T Q(\tilde{W} - W) - \tilde{W}^T Q\tilde{W} + \sigma_0
\]  

(4.1.76)

As the signal distortion depends on the allowed MSE between the desired look direction response and the processor response in the look direction over the frequency band of interest, the processor weights can be constrained to limit the MSE less than or equal to some threshold value \( \delta_0 \). Thus, an optimization problem can be formulated as discussed below.

**4.1.5.2 Optimization**

Consider the following optimization problem:

\[
\begin{align*}
\text{minimize} & \quad W^H R W \\
\text{subject to} & \quad \varepsilon_0 \leq \delta_0
\end{align*}
\]  

(4.1.77)

Defining an error vector

\[
V = \tilde{W} - W
\]  

(4.1.78)

and using (4.1.76), the optimization problem (4.1.77) becomes

\[
\begin{align*}
\text{minimize} & \quad (\tilde{W} - V)^T R (\tilde{W} - V) \\
\text{subject to} & \quad V^T Q V \leq \xi
\end{align*}
\]  

(4.1.79)

where

\[
\xi = \tilde{W}^T Q \tilde{W} + \delta_0 - \sigma_0
\]  

(4.1.80)

Note that (4.1.80) follows from (4.1.76), (4.1.77), and (4.1.79).

Let \( \tilde{W} \) be the solution of the optimization problem (4.1.77). It can be obtained using the Lagrange multipliers method as follows [Er85].

Let \( J(V, \lambda) \) be the cost function defined as

\[
J(V, \lambda) = (\tilde{W} - V)^T R (\tilde{W} - V) + \lambda (V^T Q V - \xi)
\]  

(4.1.81)

where \( \lambda \geq 0 \) is the Lagrange multiplier. As \( J(V, \lambda) \) is a convex function of \( V \), the solution for any \( \lambda \) is given by
Substituting from (4.1.81) it follows that

\[ (R + \lambda Q)\hat{V}(\lambda) = R\hat{W} \]  \hspace{1cm} (4.1.83)

which implies

\[ \hat{V}^T(\lambda)R\hat{V}(\lambda) + \lambda \hat{V}^T(\lambda)Q\hat{V}(\lambda) = \hat{V}^T(\lambda)R\hat{W} \]  \hspace{1cm} (4.1.84)

Substituting for \( V = \hat{V}(\lambda) \) in (4.1.81) and rewriting it as

\[ J(\hat{V}(\lambda), \lambda) = \hat{W}^T R\hat{W} - \hat{W}^T R\hat{V}(\lambda) - \hat{V}^T(\lambda) R\hat{W} + \hat{V}^T(\lambda) R\hat{V}(\lambda) + \lambda \hat{V}^T(\lambda) Q\hat{V}(\lambda) - \lambda \xi \]  \hspace{1cm} (4.1.85)

and using (4.1.84), (4.1.85) becomes

\[ J(\hat{V}(\lambda), \lambda) = \hat{W}^T R\hat{W} - \hat{W}^T R\hat{V}(\lambda) - \lambda \xi \]  \hspace{1cm} (4.1.86)

It follows from (4.1.83) that

\[ \hat{V}(\lambda) = (R + \lambda Q)^{-1} R\hat{W} \]  \hspace{1cm} (4.1.87)

Substituting for \( \hat{V}(\lambda) \) from (4.1.87) in (4.1.86),

\[ \hat{j}(\lambda) \Delta J(\hat{V}(\lambda), \lambda) \]

\[ = \hat{W}^T R\hat{W} - \hat{W}^T R(R + \lambda Q)^{-1} R\hat{W} - \lambda \xi \]  \hspace{1cm} (4.1.88)

It follows from the duality theorem [Lue69] that the optimum Lagrange multiplier \( \hat{\lambda} \) can be obtained by maximizing \( \hat{j}(\lambda) \). Thus, it follows that

\[ \left. \frac{\partial \hat{j}(\lambda)}{\partial \lambda} \right|_{\hat{\lambda}} = 0 \]  \hspace{1cm} (4.1.89)

Substituting (4.1.88) in (4.1.89) yields

\[ -\hat{W}^T R \left. \frac{\partial}{\partial \lambda} (R + \lambda Q)^{-1} \right|_{\hat{\lambda}} R\hat{W} = \xi \]  \hspace{1cm} (4.1.90)
To carry out the partial differentiation of \((R + \lambda Q)^{-1}\), define an invertible matrix:

\[
A(\lambda) = (R + \lambda Q)
\]  

(4.1.91)

Thus,

\[
A(\lambda)A^{-1}(\lambda) = I
\]  

(4.1.92)

Carrying out the partial differentiation with respect to \(\lambda\) results in

\[
\frac{\partial A(\lambda)}{\partial \lambda} A^{-1}(\lambda) + A(\lambda) \frac{\partial A^{-1}}{\partial \lambda} = 0
\]  

(4.1.93)

Hence,

\[
\frac{\partial A^{-1}(\lambda)}{\partial \lambda} = -A^{-1}(\lambda) \frac{\partial A(\lambda)}{\partial \lambda} A^{-1}(\lambda)
\]  

(4.1.94)

Substituting for \(A(\lambda)\) yields

\[
\frac{\partial}{\partial \lambda} (R + \lambda Q)^{-1} = -(R + \lambda Q)^{-1} Q (R + \lambda Q)^{-1}
\]  

(4.1.95)

(4.1.90) and (4.1.95) imply that \(\hat{\lambda}\) is the solution of

\[
\tilde{W}^T R \left( R + \hat{\lambda} Q \right)^{-1} Q \left( R + \hat{\lambda} Q \right)^{-1} R \tilde{W} = \xi
\]  

(4.1.96)

(4.1.87) and (4.1.78) imply that \(\hat{W}_c\), the solution of (4.1.77), is given by

\[
\hat{W}_c = \tilde{W} - \left( R + \hat{\lambda} Q \right)^{-1} R \tilde{W}
\]  

(4.1.97)

where \(\tilde{W}\) satisfies (4.1.75).

See [Er85] for discussion of the processor when it has exact presteering and is designed for flat response over the entire frequency range \((0, 1/2T)\). In this case, processor performance approaches that of the TDL processor discussed in Section 4.1.3, as \(\delta_0 \to 0\).

### 4.2 Partitioned Realization

The broadband processor structure shown in Figure 4.1 is sometimes referred to as an element space processor or direct form of realization compared to a beam space processor or partitioned form of realization. In the partitioned form, the processor is generally realized using two blocks as shown in Figure 4.2. The upper block forms a fixed main beam to receive the signal from the look direction and the lower block form auxiliary beams also known as secondary beams to estimate the noise (interferences and other unwanted noise) in the main beam. The lower block is designed to have no look direction.
signal so that when its output is subtracted from the main beam it reduces the noise. The blocking of signal from the lower section may be achieved in several ways. In one case, the array signals are processed through a signal blocking filter before processing. Signal processing in this case solves an unconstrained optimization problem. This unconstrained partitioned processor is referred to as the generalized side-lobe canceler and shown in Figure 4.3.
The signal in the lower section may also be blocked using constraints on its weights. In this case, the weights of the lower section are estimated by solving a constrained optimization. This form of realization is referred to as the constrained partitioned realization. Its block diagram is shown in Figure 4.4. Both forms of realization are discussed in this section.

4.2.1 Generalized Side-Lobe Canceler

The structure shown in Figure 4.3, also referred to as the generalized side-lobe canceler for broadband signals [Gri82], is discussed here for a point constraint, that is, the response is constrained to be unity in the look direction. Steering delays are used to align the wave form arriving from the look direction as discussed in the previous section for the element space processor. The array signals after the steering delays are passed through two sections. The upper section is designed to produce a fixed beam with a specified frequency response and the lower section consists of adjustable weights. The output of the lower section is subtracted from the output of the fixed beam to produce the processor output.

The upper section consists of a broadband conventional beam with a required frequency response obtained by selecting the coefficients $f_j$, $j = 1, \ldots, J$ of the FIR filter. Signals from all channels are equally weighted and summed to produce the output $y_C(t)$ of the conventional beam. For this realization to be equivalent to the direct form of realization, all weights need to be equal to $1/L$ and the filter coefficients $f_j$, $j = 1, \ldots, J$ need to be specified as discussed in the previous section. The output of the fixed beam is given by

$$y_F(t) = \sum_{k=0}^{J-1} y_C(t - Tk) \quad (4.2.1)$$

with
\[ y_c(t) = \frac{x^T(t) 1}{L} \]  

(4.2.2)

where \( x(t) \) denotes the array signal after presteering delays.

The fixed beam output can be expressed using the vector notation as

\[ y_f(t) = W_f^T X(t) \]  

(4.2.3)

where \( X(t) \) is an \( LJ \)-dimensional array signal vector defined by (4.1.6), \( W_f \) is an \( LJ \)-dimensional fixed weight given by

\[ W_f = C C^T C^{-1} f \]  

(4.2.4)

and \( C \) is the constraint matrix given by (4.1.26). Note that \( W_f \) is identical to \( F \) defined by (4.1.53).

The lower section consists of a matrix prefilter and a TDL structure. The matrix prefilter shown in the lower section is designed to block the signal arriving from the look direction. Since these signal wave forms after the steering delays are alike, the signal blocking can be achieved by selecting the matrix \( B \) such that the sum of its each row is equal to zero. For the partitioned processor to have the same degree of freedom as that of the direct form, the \( L - 1 \) rows of the matrix \( B \) need to be linearly independent. The output \( e(t) \) after the matrix prefilter is an \( L - 1 \) dimensional vector given by

\[ e(t) = Bx(t) \]  

(4.2.5)

and can be thought of as outputs of \( L - 1 \) beams that are then shaped by the coefficients of the FIR filter of each TDL section. Let an \( L - 1 \) dimensional vector \( v_k \) denote these coefficients before the \( k \)th delay. The \( J \) vectors \( v_1, v_2, \ldots, v_J \) correspond to the \( J \) columns of weights in the tapped delay line filter in the lower section. The lower filter output is then given by

\[ y_A(t) = \sum_{k=0}^{L-1} v^T_{k+1} e(t-kT) \]  

(4.2.6)

The output may be expressed in the vector notation as

\[ y_A(t) = V^T E(t) \]  

(4.2.7)

where \( (L - 1)J \) dimensional vector \( V \) denotes the weights of the lower section defined as

\[ V = [v_1^T, v_2^T, \ldots, v_J^T] \]  

(4.2.8)

and \( (L - 1)J \) dimensional vector \( E(t) \) denotes the array signals in the lower section defined as

\[ E(t)^T = [e^T(t), e^T(t-T), \ldots, e^T(t-(J-1)T)] \]  

(4.2.9)
It follows from (4.2.3) and (4.2.7) that the array output is then given by

\[ y(t) = y_f(t) - y_A(t) \]
\[ = W_f^T X(t) - V^T E(t) \]  \hspace{1cm} (4.2.10)

For a given weight \( V \), the mean output power of the processor is given by

\[ P(V) = E\left[y^2(t)\right] \]
\[ = E\left[\{W_f^T X(t) - V^T E(t)\}^2\right] \]  \hspace{1cm} (4.2.11)
\[ = W_f^T R_f W_f - W_f^T R_{XE} V - V^T R_{XE}^T W_f + V^T R_{EE} V \]

where

\[ R_{XE} = E\left[X(t)E^T(t)\right] \]  \hspace{1cm} (4.2.12)
and

\[ R_{EE} = E\left[E(t)E^T(t)\right] \]  \hspace{1cm} (4.2.13)

As the array signal vectors \( E(t) \) and \( X(t) \) are related through matrix \( B \), both matrices \( R_{XE} \) and \( R_{EE} \) could be rewritten in terms of \( R \) and \( B \).

Since the response of the processor in the look direction is fixed due to the fixed beam, and the lower section contains no signal from the look direction due to the presence of the matrix prefilter, nonlook direction noise may be minimized by adjusting weights of the lower section to minimize the mean output power. Thus, the optimal weights denoted by \( \hat{V} \) are the solution of the following unconstrained beamforming problem:

\[ \text{minimize} \quad P(V) \]  \hspace{1cm} (4.2.14)

Since the mean output power surface \( P(V) \) is a quadratic function of \( V \), the solution of the above problem can be obtained by taking the gradient of the of \( P(V) \) with respect to \( V \) and setting it equal to zero. Thus,

\[ \nabla_V P(V)_{V=0} = 0 \]  \hspace{1cm} (4.2.15)

Substituting for \( P(V) \) from (4.2.11),

\[ R_{EE} \hat{V} = R_{XE}^T W_f \]  \hspace{1cm} (4.2.16)

When the array correlation matrix \( R \) is invertible, the matrix \( R_{EE} \) is invertible and (4.2.16) yields

\[ \hat{V} = R_{EE}^{-1} R_{XE}^T W_f \]  \hspace{1cm} (4.2.17)
It can be shown [Gri82] that when the weights in the array processors in Figure 4.1 and Figure 4.2 are optimized, the performance of the two processors is identical. The weights $\tilde{V}$ may be expressed using array correlation matrix as follows.

Let $\tilde{B}$ be a matrix defined as

$$
\tilde{B} = \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix}
$$

(4.2.18)

It follows from (4.2.4), (4.2.9) and (4.2.18) that

$$
\begin{bmatrix}
\begin{bmatrix}
    e(t) \\
    e(t+T) \\
    M \\
    e(t-(J-1)T)
\end{bmatrix}
\end{bmatrix}
= \begin{bmatrix}
\begin{bmatrix}
    Bx(t) \\
    Bx(t-T) \\
    M \\
    Bx(t-(J-1)T)
\end{bmatrix}
\end{bmatrix}
$$

(4.2.19)

$$
= \tilde{B}X(t)
$$

Substituting in (4.2.12) and (4.2.13) yields

$$
R_{xe} = E\left[X(t)X^T(t)\right]\tilde{B}^T
$$

(4.2.20)

and

$$
R_{ee} = \tilde{B}R\tilde{B}^T
$$

(4.2.21)

It follows from (4.2.17), (4.2.20), and (4.2.21) that

$$
\tilde{V} = (\tilde{B}R\tilde{B}^T)^{-1}\tilde{B}W_f
$$

(4.2.22)

Substituting in (4.2.10) from (4.2.19) and (4.2.22), the output of the processor with optimized weights becomes

$$
y(t) = W_f^T X(t) - W_f^T \tilde{B} (\tilde{B}R\tilde{B}^T)^{-1}\tilde{B}X(t)
$$

(4.2.23)

$$
= W_f^T \left[I - \tilde{B} (\tilde{B}R\tilde{B}^T)^{-1}\tilde{B}\right]X(t)
$$
4.2.2 Constrained Partitioned Realization

Figure 4.4 shows a structure of the constrained partitioned realized processor [Jim77]. The main difference between the constrained processor and unconstrained processor (also referred to as the generalized side-lobe canceler in the previous section) is that the latter uses a signal blocking matrix to stop the signal from entering the lower section and solves an unconstrained beamforming problem, whereas the constrained processor uses constraints on the weights of the lower section to eliminate the signal at the output of the lower section. Consequently, the optimization problem solved to estimate the weights of the lower section is a constrained one.

Let the LJ-dimensional vector \( \mathbf{W}_F \) given by Figure 4.4 denote the weights of the fixed beam (upper section). Thus, the output of the upper section \( y_F(t) \) is given by

\[
y_F(t) = \mathbf{W}_F^T \mathbf{x}(t)
\]  (4.2.24)

Let the LJ-dimensional vector \( \mathbf{W} \) denote the weights of the lower section. Thus, the output of the lower section \( y_A(t) \) is given by

\[
y_A(t) = \mathbf{W}^T \mathbf{x}(t)
\]  (4.2.25)

The processor output \( y(t) \) is the difference of the two outputs. Thus,

\[
y(t) = y_F(t) - y_A(t) = (\mathbf{W}_F - \mathbf{W})^T \mathbf{x}(t)
\]  (4.2.26)

The mean output power \( P(\mathbf{W}) \) for given weights is given by

\[
P(\mathbf{W}) = (\mathbf{W}_F - \mathbf{W})^T \mathbf{R} (\mathbf{W}_F - \mathbf{W})
\]  (4.2.27)

The lower section is designed such that its output does not contain the look direction signal. This is achieved by selecting its weights to be the solution of the following beamforming problem:

\[
\begin{align*}
\text{minimize} \quad & (\mathbf{W}_F - \mathbf{W})^T \mathbf{R} (\mathbf{W}_F - \mathbf{W}) \\
\text{subject to} \quad & \mathbf{C}^T \mathbf{W} = 0
\end{align*}
\]  (4.2.28)

It follows from (4.1.26) and the second equation of (4.2.28) that

\[
\mathbf{1}^T \mathbf{w}_j = 0, \quad j = 1, 2, \ldots, J
\]  (4.2.29)

where \( \mathbf{w}_j \) denotes the weights of the jth column, that is, before the jth delay in the lower section.

Since the look direction signal waveforms on all elements after presteering delays are alike, the constraints of (4.2.29) ensure that the lower section has a null response in the look direction. Thus, the constraint in (4.2.28) achieves a null in the look direction similar to that achieved by the matrix prefilter \( \mathbf{B} \) discussed in the previous section.
Let $\hat{W}_0$ denote the solution of (4.2.28). Using the method of Lagrange multipliers discussed in Section 4.1,

$$\hat{W}_0 = W_f - R^{-1}C^T(R^{-1}C)^{-1}C^TW_f$$

$$= W_f - \hat{W} \tag{4.2.30}$$

where $\hat{W}$ is given by (4.1.36).

### 4.2.3 General Constrained Partitioned Realization

In this section, a processor realization in general constrained form is presented where the upper section is designed to minimize the MSE between the look direction desired response and the frequency response of the processor in the look direction over a frequency band of interest $[f_L, f_H]$, as discussed in Section 4.1.5. The lower section is designed such that its weights are constrained to yield a zero power response over the frequency band of interest to prevent signal suppression. Design details may be found in [Er86].

Let an $L_J$-dimensional vector $\tilde{W}$ denote the weight of the upper section. These are designed using minimum MSE design and satisfy (4.1.75). The output of the upper section $y_f(t)$ is given by

$$y_f(t) = \tilde{W}^T X(t) \tag{4.2.31}$$

Let an $L_J$-dimensional vector $W$ denote the weights of the lower section. Thus, the output of the lower section $y_A(t)$ is given by

$$y_A(t) = W^T X(t) \tag{4.2.32}$$

and the processor output $y(t)$ is given by

$$y(t) = (\tilde{W} - W)^T X(t) \tag{4.2.33}$$

The mean output power $P(W)$ for a given $W$ is given by

$$P(W) = (\tilde{W} - W)^T \tilde{R}(\tilde{W} - W) \tag{4.2.34}$$

#### 4.2.3.1 Derivation of Constraints

Let weight vector $W$ be constrained such that the power response of the lower section in the look direction is zero over the frequency band of interest, that is,

$$\int_{f_l}^{f_H} H^*(f,\phi_0,\theta_0)H(f,\phi_0,\theta_0) df = 0 \tag{4.2.35}$$

It follows from (4.2.35) and (4.1.67) that
As $Q$ is a positive semidefinite matrix, it can be factorized using its eigenvalues and eigenvectors as

$$Q = U \Lambda U^T$$

(4.2.37)

where $\Lambda$ is a diagonal matrix with its elements being $\lambda_i(Q), i = 1, 2, \ldots, LJ,$ the eigenvalues of $Q,$ such that

$$\lambda_1(Q) \geq \lambda_2(Q) \geq \cdots \geq \lambda_{LJ}(Q) \geq 0$$

(4.2.38)

and $U$ is an $LJ \times LJ$ matrix of the eigenvector of $Q,$ that is,

$$U = [U_1, U_2, \ldots, U_{LJ}]$$

(4.2.39)

where $U_i, i = 1, 2, \ldots, LJ$ are the orthonormal eigenvectors of $Q$ with the property that

$$U_i^T U_j = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$$

(4.2.40)

Substituting (4.2.37) in (4.2.36)

$$W^T U \Lambda U^T W = 0$$

(4.2.41)

Assume that $Q$ has rank $\eta_0.$ Thus, it follows from (4.2.39) and (4.2.41) that the necessary and sufficient conditions to satisfy (4.2.41) are

$$W^T U_i = 0, \quad i = 1, 2, \ldots, \eta_0$$

(4.2.42)

Thus, the linear constraints of the form (4.2.42) can be used to ensure that the lower section has a zero power response in the look direction over the frequency range of interest. It should be noted that signal blocking in the lower section using these constraints is independent of presteering delays, that is, the processor may include exact presteering, coarse presteering, or no presteering.

### 4.2.3.2 Optimization

Let the optimum weight vector $\hat{W}$ be the solution of the following constrained beamforming problem

$$\text{minimize} \quad (\hat{W} - W)^T R (\hat{W} - W)$$

subject to \quad $U_{\eta_0}^T W = 0$

(4.2.43)

where $U_{\eta_0}$ is the $LJ \times \eta_0$ dimensional matrix given by

$$U_{\eta_0} = [U_1, U_2, \ldots, U_{\eta_0}]$$

(4.2.44)
As $U_i$ for $i = 1, 2, \ldots, \eta_0$ are linearly independent, the matrix $U_{\eta_0}$ has full rank. Using the method of Lagrange multipliers,

$$
\hat{W} = \hat{W} - R^{-1}U_{\eta_0}\left(U_{\eta_0}^{T}R^{-1}U_{\eta_0}\right)^{-1}U_{\eta_0}^{T}\hat{W}
$$

(4.2.45)

### 4.3 Derivative Constrained Processor

The implication of the point constraint considered in Section 4.1 is that the array pattern has a unity response in the look direction. It can be broadened using additional constraints, such as derivative constraints, along with the point constraint [Er83, Er90, Er86a, Thn93]. The derivative constraints set the derivatives of the power pattern with respect to $\phi$ and $\theta$ equal to zero in the look direction. The higher the order of derivatives, that is, first order, second order, and so on, the broader the beam in the look direction normally becomes. A broader beam is useful when the actual signal direction and known direction of the signal are not precisely the same. In such situations, the processor with the point constraint in the known direction of the signal would cancel the desired signal as if it were interference. Other directional constraints to improve the performance of the beamformer in the presence of the look direction error include multiple linear constraints [Tak85, Buc87, Gri87] and inequality constraints [Ahm83, Ahm84, Er90a, Er93].

In this section, some of these constraints are derived, a beamforming problem using these constraints is formulated, an algorithm to estimate solution of the optimization problem is presented, and the effect that choice of coordinate system origin has on the performance of an array system using derivative constraints is discussed.

Derivative constraints are derived by setting derivatives of the power response $\rho(f, \phi, \theta)$ with respect to $\phi$ and $\theta$ to zero in direction $(\phi_0, \theta_0)$. Since $H(f, \phi, \theta)$ denotes the frequency response of the processor, it follows that

$$
\rho(f, \phi, \theta) = H^*(f, \phi, \theta)H(f, \phi, \theta)
$$

(4.3.1)

The first-order derivative constraints are now derived [Er83].

#### 4.3.1 First-Order Derivative Constraints

It follows from (4.3.1) that the partial derivative of the power response with respect to $\phi$ is given by

$$
\frac{\partial \rho}{\partial \phi} = H^*\frac{\partial H}{\partial \phi} + \frac{\partial H^*}{\partial \phi}H
$$

(4.3.2)

where the parameters of $\rho(f, \phi, \theta)$ and $H(f, \phi, \theta)$ are omitted for ease of notation. It follows from (4.1.18) that

$$
\frac{\partial H}{\partial \phi} = \frac{\partial S^T(f, \phi, \theta)}{\partial \phi}T(f)\sum_{i=1}^{l}w_ke^{-2\pi f[i-1]}T
$$

(4.3.3)
Differentiating (4.1.20) with respect to $\phi$,
\[
\frac{\partial S^T(f, \phi, \theta)}{\partial \phi} = j2\pi f \left[ \frac{\partial \tau_l(\phi, \theta)}{\partial \phi} e^{2\pi i n_l(\phi, \theta)} \right] + L \frac{\partial \tau_l(\phi, \theta)}{\partial \phi} e^{i2\pi n_l(\phi, \theta)} \right] \tag{4.3.4}
\]

\[
= j2\pi f S^T(f, \phi, \theta) \Lambda_\phi(\phi, \theta)
\]

where
\[
\Lambda_\phi(\phi, \theta) = \begin{bmatrix}
\frac{\partial \tau_l(\phi, \theta)}{\partial \phi} & 0 \\
0 & \frac{\partial \tau_l(\phi, \theta)}{\partial \phi}
\end{bmatrix} \tag{4.3.5}
\]

and $\tau_l(\phi, \theta)$ is given by (2.1.1). It can also be expressed as
\[
\tau_l(\phi, \theta) = \frac{1}{c} \left\{ (x_l \cos \phi + y_l \sin \phi) \sin \theta + z_l \cos \theta \right\} \tag{4.3.6}
\]

where $x_l$, $y_l$, and $z_l$ denote the components of the $l$th element along the $x$, $y$, and $z$ axis, respectively, and $c$ denotes the speed of propagation.

Substituting (4.3.4) in (4.3.3) and noting that $T(f)$ and $\Lambda_\phi(\phi, \theta)$ are diagonal matrices,
\[
\frac{\partial H}{\partial \phi} = j2\pi f S^T(f, \phi, \theta) T(f) \Lambda_\phi(\phi, \theta) \sum_{l=1}^{1} w_l e^{-j2\pi (l-1)T} \tag{4.3.7}
\]

which implies
\[
\frac{\partial H}{\partial \phi} \bigg|_{\phi_0, \theta_0} = j2\pi f S^T(f, \phi_0, \theta_0) T(f) \Lambda_\phi(\phi_0, \theta_0) \sum_{l=1}^{1} w_l e^{-j2\pi (l-1)T} \tag{4.3.8}
\]

Noting from (4.1.21) that
\[
S^T(f, \phi_0, \theta_0) T(f) = a(f) T^T \tag{4.3.9}
\]

(4.3.7) yields
\[
\frac{\partial H}{\partial \phi} \bigg|_{\phi_0, \theta_0} = j2\pi f a(f) \sum_{l=1}^{1} \Lambda_\phi(\phi_0, \theta_0) w_l e^{-j2\pi (l-1)T} \tag{4.3.10}
\]

It follows from (4.1.23) that
\[
H^*(f, \phi_0, \theta_0) = a^*(f) \sum_{k=1}^{1} f_k e^{i2\pi (k-1)T} \tag{4.3.11}
\]
Thus,

\[
H^* \frac{\partial H}{\partial \phi} \bigg|_{(\phi_0, \theta_0)} = j2\pi f a(f)a^*(f) \sum_{i=1}^{J} \sum_{k=1}^{L} f_k I^T \Lambda_a(\phi_0, \theta_0) w_i e^{-j2\pi f(1-k)T}
\] (4.3.12)

Noting from (4.1.22) that \(a(f)a^*(f) = 1\) and using this in (4.3.12),

\[
H^* \frac{\partial H}{\partial \phi} \bigg|_{(\phi_0, \theta_0)} = j2\pi f \sum_{i=1}^{J} \sum_{k=1}^{L} f_k I^T \Lambda_a(\phi_0, \theta_0) w_i e^{-j2\pi f(1-k)T}
\] (4.3.13)

Similarly,

\[
\frac{\partial H^*}{\partial \phi} \bigg|_{(\phi_0, \theta_0)} = -j2\pi f \sum_{i=1}^{J} \sum_{k=1}^{L} f_k I^T \Lambda_a(\phi_0, \theta_0) w_i e^{j2\pi f(1-k)T}
\] (4.3.14)

Substituting in (4.3.2),

\[
\frac{\partial \rho}{\partial \phi} \bigg|_{(\phi_0, \theta_0)} = 2\pi f \sum_{i=1}^{J} \sum_{k=1}^{L} f_k I^T \Lambda_a(\phi_0, \theta_0) w_i \left[ j e^{-j2\pi f(1-k)T} - j e^{j2\pi f(1-k)T} \right]
\]
\[
= 4\pi f \sum_{i=1}^{J} \sum_{k=1}^{L} f_k I^T \Lambda_a(\phi_0, \theta_0) w_i \sin 2\pi f(1-k)T
\] (4.3.15)

Similarly,

\[
\frac{\partial \rho}{\partial \theta} \bigg|_{(\phi_0, \theta_0)} = 4\pi f \sum_{i=1}^{J} \sum_{k=1}^{L} f_k I^T \Lambda_a(\phi_0, \theta_0) w_i \sin 2\pi f(1-k)T
\] (4.3.16)

where

\[
\Lambda_a(\phi, \theta) = \begin{bmatrix}
\frac{\partial \tau_1(\phi, \theta)}{\partial \theta} & 0 \\
0 & \frac{\partial \tau_0(\phi, \theta)}{\partial \theta}
\end{bmatrix}
\] (4.3.17)

It follows from (4.3.15) and (4.3.16), respectively, that sufficient conditions for \(\frac{\partial \rho}{\partial \phi} \bigg|_{(\phi_0, \theta_0)} = 0\) for all \(f > 0\) are

\[
I^T \Lambda_a(\phi_0, \theta_0) w_i = 0, \quad i = 1, 2, \ldots, J
\] (4.3.18)

and sufficient condition for \(\frac{\partial \rho}{\partial \theta} \bigg|_{(\phi_0, \theta_0)} = 0\) for all \(f > 0\) are
Equations (4.3.18) and (4.1.19) denote 2J linear constraints on the weights of the broadband processor. These constraints are sufficient for the first-order derivatives of the power response with respect to $\phi$ and $\theta$ evaluated at $(\phi, \theta)$ to be zero. These are referred to as the first-order derivative constraints and are imposed along with the point constraint discussed previously.

Using a similar approach to the derivation of the first-order constraints presented in this section, higher-order derivative constraints may be derived by setting the higher-order derivatives of the power response with respect to $\phi$ and $\theta$ evaluated at $(\phi_0, \theta_0)$ to zero.

4.3.2 Second-Order Derivative Constraints

The equations describing the second-order derivative constraints follow [Er83]:

$$
1^T \Lambda_{\phi}(\phi_0, \theta_0)w_i = 0, \quad 1 = 1, 2, \ldots, J
$$

(4.3.19)

Equations (4.3.18) and (4.1.19) denote 2J linear constraints on the weights of the broadband processor. These constraints are sufficient for the first-order derivatives of the power response with respect to $\phi$ and $\theta$ evaluated at $(\phi, \theta)$ to be zero. These are referred to as the first-order derivative constraints and are imposed along with the point constraint discussed previously.

Using a similar approach to the derivation of the first-order constraints presented in this section, higher-order derivative constraints may be derived by setting the higher-order derivatives of the power response with respect to $\phi$ and $\theta$ evaluated at $(\phi_0, \theta_0)$ to zero.

4.3.2 Second-Order Derivative Constraints

The equations describing the second-order derivative constraints follow [Er83]:

$$
1^T \frac{\partial \Lambda_{\phi}(\phi, \theta)}{\partial \phi} \bigg|_{(\phi_0, \theta_0)} w_i = 0, \quad 1 = 1, 2, \ldots, J
$$

(4.3.20)

$$
1^T \Lambda_{\phi}(\phi_0, \theta_0)w_i = 0, \quad 1 = 1, 2, \ldots, J
$$

(4.3.21)

$$
1^T \frac{\partial \Lambda_{\phi}(\phi, \theta)}{\partial \theta} \bigg|_{(\phi_0, \theta_0)} w_i = 0, \quad 1 = 1, 2, \ldots, J
$$

(4.3.22)

$$
1^T \Lambda_{\phi}(\phi_0, \theta_0)w_i = 0, \quad 1 = 1, 2, \ldots, J
$$

(4.3.23)

$$
1^T \frac{\partial \Lambda_{\phi}(\phi, \theta)}{\partial \theta} \bigg|_{(\phi_0, \theta_0)} w_i = 0, \quad 1 = 1, 2, \ldots, J
$$

(4.3.24)

and

$$
1^T \Lambda_{\phi}(\phi_0, \theta_0)\Lambda_{\phi}(\phi_0, \theta_0)w_i = 0, \quad 1 = 1, 2, \ldots, J
$$

(4.3.25)

These equations denote 6J linear constraints that are sufficient for the second-order derivatives with respect to $\phi$ and $\theta$ evaluated at $(\phi_0, \theta_0)$ to be zero. These are imposed along with the point constraint and first-order derivative constraints.

It should be noted that these constraints depend on array geometry and are not necessarily linearly independent. In the next section, a beamforming problem with derivative constraints is considered.

4.3.3 Optimization with Derivative Constraints

A beamforming problem using derivative constraints may be formulated similar to the constrained beamforming problem considered previously by adding derivative constraints
specified by (4.3.18) to (4.3.25) to the point constraint given by the second equation of (4.1.29).

In this case (4.1.29) becomes

\[
\begin{align*}
\text{minimize} & \quad W^T RW \\
\text{subject to} & \quad D^T W = g
\end{align*}
\]  

(4.3.26)

where

\[
g^T = [f^T, 0^T, \ldots, 0^T]
\]  

(4.3.27)

and

\[
D = [C_0; C_1; \ldots; C_8]
\]  

(4.3.28)

with \(LJ \times J\) matrices \(C_0\) to \(C_8\) given by

\[
C_0 = \text{diag}[1]
\]  

(4.3.29)

\[
C_1 = \text{diag}[1^T \Lambda_0(\phi_0, \theta_0)]
\]  

(4.3.30)

\[
C_2 = \text{diag}[1^T \Lambda_0(\phi_0, \theta_0)]
\]  

(4.3.31)

\[
C_3 = \text{diag}
\begin{bmatrix}
1^T \frac{\partial \Lambda_0(\phi, \theta)}{\partial \phi} |_{(\phi_0, \theta_0)}
\end{bmatrix}
\]  

(4.3.32)

\[
C_4 = \text{diag}[1^T \Lambda_2^2(\phi_0, \theta_0)]
\]  

(4.3.33)

\[
C_5 = \text{diag}[1^T \frac{\partial \Lambda_0(\phi, \theta)}{\partial \theta} |_{(\phi_0, \theta_0)}]
\]  

(4.3.34)

\[
C_6 = \text{diag}[1^T \Lambda_0^2(\phi_0, \theta_0)]
\]  

(4.3.35)

\[
C_7 = \text{diag}
\begin{bmatrix}
1^T \frac{\partial \Lambda_0(\phi, \theta)}{\partial \theta} |_{(\phi_0, \theta_0)}
\end{bmatrix}
\]  

(4.3.36)

and

\[
C_8 = \text{diag}[1^T \Lambda_0(\phi_0, \theta_0)]
\]  

(4.3.37)
The notation $\text{diag}[x]$ in (4.3.29) to (4.3.37) is defined as

$$\text{diag}[x] = \begin{bmatrix} x & 0 \\ 0 & x \end{bmatrix}$$  \hspace{1cm} (4.3.38)

For example, in (4.3.29)

$$x = 1$$  \hspace{1cm} (4.3.39)

and $C_0$ is given by

$$C_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$  \hspace{1cm} (4.3.40)

It can easily be verified from (4.3.26) to (4.3.37) that

$$C_i^T W = f$$  \hspace{1cm} (4.3.41)

and

$$C_i^T W = 0, \quad i = 1, \ldots, 8$$  \hspace{1cm} (4.3.42)

Equation (4.3.41) is the second equation of (4.1.29) and defines the point constraint, whereas (4.3.42) defines derivative constraints given by (4.3.18) to (4.3.25).

The optimization problem (4.3.26) is similar in form to (4.1.29). Thus, it follows from (4.1.36) that if $D$ is of full rank, then the optimal weight $\hat{W}$, the solution of (4.3.26), is given by

$$\hat{W} = R^{-1} D (D^T R^{-1} D)^{-1} R$$  \hspace{1cm} (4.3.43)

The rank of $D$ is dependent on array geometry. This is explained in the following example using a linear array [Er83].

**4.3.3.1 Linear Array Example**

Consider a linear array along the $x$-axis with $x_l$ denoting the position of the $l$th element. Assume that the directional sources are in the $x$-$y$ plane with the look direction making an angle $\phi_0$ with the array. In view of these assumptions, it follows that

$$\theta = 90^\circ, \quad y_l = 0, \quad z_l = 0, \quad l = 1, 2, \ldots, L$$  \hspace{1cm} (4.3.44)

These equations along with (4.3.6) imply that

$$\tau_i(\phi) = \frac{x_l \cos \phi}{c}$$  \hspace{1cm} (4.3.45)
\[
\frac{\partial \tau_1(\phi)}{\partial \phi} = -x_1 \sin \phi \quad (4.3.46)
\]

and
\[
\frac{\partial^2 \tau_1(\phi)}{\partial^2 \phi} = -x_1 \cos \phi \quad (4.3.47)
\]

Now consider the constrained Equation (4.3.18) to Equation (4.3.25). Using (4.3.44) and the fact the time delay \(\tau_l(\phi)\) is not a function of \(\theta\), one notes that the constraint equations (4.3.19), (4.3.22), (4.3.23), (4.3.24), and (4.3.25) vanish. The only constraints remaining are those given by (4.3.18), (4.3.20), and (4.3.21), that is,

\[
1^T \Lambda_\phi(\phi_0) \mathbf{w}_1 = 0, \quad 1 = 1, 2, \ldots, J \quad (4.3.48)
\]

\[
1^T \frac{\partial \Lambda_\phi(\phi_0)}{\partial \phi} \mathbf{w}_1 = 0, \quad 1 = 1, 2, \ldots, J \quad (4.3.49)
\]

and

\[
1^T \Lambda_\phi(\phi_0) \mathbf{w}_1 = 0, \quad 1 = 1, 2, \ldots, J \quad (4.3.50)
\]

where \(\Lambda_\phi(\phi), \frac{\partial \Lambda_\phi(\phi)}{\partial \phi}\) and \(\Lambda_\phi^2(\phi)\) are diagonal matrices given by

\[
\Lambda_\phi(\phi) = \begin{bmatrix}
\frac{\partial \tau_1(\phi)}{\partial \phi} & 0 \\
0 & \frac{\partial \tau_1(\phi)}{\partial \phi}
\end{bmatrix} \quad (4.3.51)
\]

\[
\frac{\partial \Lambda_\phi(\phi)}{\partial \phi} = \begin{bmatrix}
\frac{\partial^2 \tau_1(\phi)}{\partial^2 \phi} & 0 \\
0 & \frac{\partial^2 \tau_1(\phi)}{\partial^2 \phi}
\end{bmatrix} \quad (4.3.52)
\]

and

\[
\Lambda_\phi^2(\phi) = \begin{bmatrix}
\left(\frac{\partial \tau_1(\phi)}{\partial \phi}\right)^2 & 0 \\
0 & \left(\frac{\partial \tau_1(\phi)}{\partial \phi}\right)^2
\end{bmatrix} \quad (4.3.53)
\]
To simplify the notation, define three $L$ vectors $\mathbf{\lambda}(\phi)$, $\mathbf{\sigma}(\phi)$, and $\mathbf{\psi}(\phi)$ as

\begin{equation}
\mathbf{\lambda}(\phi) = 1^T \mathbf{\Lambda}_\phi(\phi) \quad (4.3.54)
\end{equation}

\begin{equation}
\mathbf{\sigma}(\phi) = 1^T \frac{\partial \mathbf{\Lambda}_\phi}{\partial \phi}(\phi) \quad (4.3.55)
\end{equation}

and

\begin{equation}
\mathbf{\psi}(\phi) = 1^T \mathbf{\Lambda}_\phi^2(\phi) \quad (4.3.56)
\end{equation}

Using (4.3.45) to (4.3.47) and (4.3.51) to (4.3.53), these become

\begin{equation}
\mathbf{\lambda}_\phi(\phi_0) = -\frac{\sin \phi_0}{c} \begin{bmatrix} x_1 \\ M \\ x_L \end{bmatrix} \quad (4.3.57)
\end{equation}

\begin{equation}
\mathbf{\sigma}_\phi(\phi_0) = -\frac{\cos \phi_0}{c} \begin{bmatrix} x_1 \\ M \\ x_L \end{bmatrix} \quad (4.3.58)
\end{equation}

and

\begin{equation}
\mathbf{\psi}_\phi(\phi_0) = -\frac{\sin^2 \phi_0}{c^2} \begin{bmatrix} x_1^2 \\ M \\ x_L^2 \end{bmatrix} \quad (4.3.59)
\end{equation}

The three constraint equations (4.3.48) to (4.3.50) are then given by

\begin{equation}
\mathbf{\lambda}_\phi^T(\phi_0) \mathbf{w}_i = 0, \quad 1 = 1, 2, \ldots, J \quad (4.3.60)
\end{equation}

\begin{equation}
\mathbf{\sigma}_\phi^T(\phi_0) \mathbf{w}_i = 0, \quad 1 = 1, 2, \ldots, J \quad (4.3.61)
\end{equation}

and

\begin{equation}
\mathbf{\psi}_\phi^T(\phi_0) \mathbf{w}_i = 0, \quad 1 = 1, 2, \ldots, J \quad (4.3.62)
\end{equation}

Note that (4.3.60) denotes $J$ first-order constraints equations, and (4.3.61) and (4.3.62) denote $2J$ second-order constraints. For a general array, there are $2J$ linear constraints and $6J$ derivative constraints as discussed previously. Thus, the constraints for a linear array are much less than those for a general array. It should be noted that these constraints are functions of the look direction.
For look direction in broadside to the array \( \phi_0 = 90^\circ \), it follows from (4.3.58) that \( \sigma_0(\phi_0) = 0 \) and (4.3.61) vanish, reducing the constraints from \( 3J \) to \( 2J \) for a linear array. Similarly, for an endfire array where the look direction is parallel to the array, \( \phi_0 = 0^\circ \) or \( \phi_0 = 180^\circ \), (4.3.57) and (4.3.59) imply that both (4.3.60) and (4.3.62) vanish. Thus, for a linear array, only \( J \) second-order constraints given by (4.3.61) remain; first-order constraints have vanished.

When a beamforming problem is considered using derivative constraints, the constraints equations specifying only linearly independent constraints need to be considered. It follows from (4.3.57) and (4.3.58) that vectors \( \lambda_0(\phi_0) \) and \( \sigma_0(\phi_0) \) are not linearly independent; thus constraints (4.3.60) and (4.3.61) are not independent. Hence, only \( 2J \) constraints given by (4.3.60) and (4.3.62) need to be used in the optimization process.

For this case beamforming problem given by (4.3.26) to (4.3.37) reduce to

\[
\begin{align*}
\text{minimize} & \quad W^T R W \\
\text{subject to} & \quad D^T W = g
\end{align*}
\]

where

\[
g^T = [\dot{f}^T, \theta^T, \phi^T]
\]

and

\[
D = [C_0; C_1; C_2]
\]

with

\[
C_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}
\]

\[
C_1 = \begin{bmatrix} \lambda_0(\phi_0) & 0 \\ 0 & \lambda_0(\phi_0) \end{bmatrix}
\]

and

\[
C_2 = \begin{bmatrix} \psi_0(\phi_0) & 0 \\ 0 & \lambda_0(\phi_0) \end{bmatrix}
\]

For linearly independent vectors \( 1 \), \( \lambda_0(\phi_0) \) and \( \psi_0(\phi_0) \), the constraint matrix \( D \) has full rank [Er83] and the beamforming solution is given by (4.3.43).
4.3.4 Adaptive Algorithm

An estimate of the solution of the beamforming problem (4.3.63), which converges in mean to the optimal weights given by (4.3.43), may be made using a constrained LMS algorithm similar to that given by (4.1.52), (4.1.53), and (4.1.49). In this case, it becomes

$$W(n+1) = P[W(n) - \mu y(n)x(n+1)] + G$$  \hspace{1cm} (4.3.69)

where the projection operator

$$P = I - D(D^TD)^{-1}D^T$$  \hspace{1cm} (4.3.70)

and

$$G = D(D^TD)^{-1}g$$  \hspace{1cm} (4.3.71)

The algorithm is initialized at \(n = 0\) with

$$W(0) = G$$  \hspace{1cm} (4.3.72)

Note that the initial weight vector \(W(0)\) correspond to the optimal weight given by (4.1.43) in the presence of white noise only.

Due to the sparse nature of matrices \(C_0\), \(C_1\), and \(C_2\), the projection operator \(P\) is sparse and allows development of a temporally decoupled update equation to estimate the \(J\) columns of \(L\) weights similar to that discussed earlier for the point constraint. For this case, the algorithm is given by [Buc86]

$$w_j(n+1) = P[w_j(n) - \mu y(n)x(n+1-j) + \tilde{f}\tilde{C}^{\top}(\tilde{C}^{\top}\tilde{C})^{-1}e_j]$$  \hspace{1cm} (4.3.73)

\(j = 1, 2, \ldots, J\)

where

$$P = I - \tilde{C}(\tilde{C}^{\top}\tilde{C})^{-1}\tilde{C}^{\top}$$  \hspace{1cm} (4.3.74)

$$\tilde{C} = [1, \lambda_{\phi}(\phi_0), \psi_{\phi}(\phi_0)]$$  \hspace{1cm} (4.3.75)

and

$$e_j^{\top} = [1, 0, 0]$$  \hspace{1cm} (4.3.76)

4.3.5 Choice of Origin

In array system design, location of the time reference point (origin of the coordinate system) with respect to the array elements is chosen for notational convience. In most
cases, it is one element of the array or array’s center of gravity. These origin choices do not affect the array beam pattern or output SNR. However, this is not the case when derivative constraints are involved. The reason is that the constraint matrix $D$ is a function of $\tau_l(\phi, \theta)$, which in turn depends on origin choice, as it denotes the time taken by a plane wave arriving from direction $(\phi, \theta)$ and measured from the origin to the array’s $l$th element. This dependence of the constraint matrix $D$ on the choice of origin affects the solution of the constrained beamforming problem. Hence, the beam pattern and the output SNR of the beamformer using optimal weights depends on the choice of origin.

The vector $G$ used to initialize the adaptive algorithm is the optimal weight under white noise conditions and the output noise power is proportional to the norm of this weight, that is, $G^T G$. In view of this, the chosen origin should minimize $G^T G$ [Buc86].

It follows from (4.3.71) that

$$G^T G = g^T (D^T D)^{-1} g$$  \hspace{1cm} (4.3.77)

The first-order and the second-order derivative constraints discussed in this section so far are sufficient to ensure that the power response derivatives evaluated at the look direction are zero. However, these constraints are not the necessary and sufficient conditions for the derivatives to be zero. In what follows is a discussion on the first-order derivative constraints for a flat-response processor. For this case, these constraints are necessary and sufficient conditions which ensure that the array beam pattern is independent of the choice of origin [Er90].

The constraint vector $f$ for the case of a flat frequency response in the look direction is given by (4.1.28), that is,

$$f_i = \begin{cases} 1 & i = k_0 \\ 0 & i \neq k_0 \end{cases}$$  \hspace{1cm} (4.3.78)

Substituting (4.3.78) in (4.3.15), it follows that

$$\frac{\partial \rho}{\partial \phi} \bigg|_{(\phi_0, \theta_0)} = 4\pi f \sum_{i=1}^{J} \Lambda^T_l(\phi, \theta) w_i \sin 2\pi f (1-k_0) T$$  \hspace{1cm} (4.3.79)

If $J$ is odd and $k_0 = (J + 1)/2$, then (4.3.79) can be rewritten as

$$\frac{\partial \rho}{\partial \phi} \bigg|_{(\phi_0, \theta_0)} = 4\pi f \sum_{i=1}^{k_0-1} \Lambda^T_i(\phi, \theta) (w_{k_0+1} - w_{k_0-1}) \sin 2\pi f 1 T$$  \hspace{1cm} (4.3.80)

As the right hand side of (4.3.80) is a finite Fourier series, it follows that the necessary and sufficient conditions for

$$\frac{\partial \rho}{\partial \theta} \bigg|_{(\phi_0, \theta_0)} = 0$$

for all $f > 0$ are that all series coefficients are simultaneously equal to zero, that is,
Similarly, the necessary and sufficient conditions for

\[ I^T A_k(\phi_0, \theta_0)(w_{k+1} - w_{k-1}) = 0, \]

are

\[ \begin{cases} 1 = 1, 2, \ldots, k_0 - 1, \\ k_0 = \frac{J+1}{2} \end{cases} \] (4.3.81)

Note that (4.3.81) and (4.3.82) denote \( J - 1 \) linear constraints compared to \( 2J \) linear constraints given by (4.3.18) and (4.3.19). Discussion on second-order derivative constraints may be found in [Er90], and an unconstrained partitioned realization of the processor with derivative constraints is provided by [Er86a].

4.4 Correlation Constrained Processor

A set of nondirectional constraints to improve the performance of a broadband array processor using a TDL structure under look direction errors is discussed in [Kik89]. These are referred to as correlation constraints, and they use known characteristics of the desired signal to estimate an \( \text{LJ} \)-dimensional correlation vector \( r_d \) between the desired signal and the array signal vector due to the desired signal, that is,

\[ r_d = E[s_d(t)X_d(t)] \] (4.4.1)

where \( s_d(t) \) denotes the desired signal induced on the reference element, and \( \text{LJ} \)-dimensional vector \( X_d(t) \) denotes the array signal across the TDL structure due to the desired signal only.

The beamforming problem in this case becomes

\[
\begin{align*}
\text{minimize} & \quad W^T R W \\
\text{subject to} & \quad r_d^T W = \rho_0
\end{align*}
\] (4.4.2)

where \( \rho_0 \) is a scalar constant that specifies the correlation between the desired signal and array output due to the desired signal, that is,

\[ \rho_0 = [s_d(t) y_d(t)] \] (4.4.3)
where
\[ y_d(t) = W^T x_d(t) \]  
(4.4.4)

For the desired signal with a flat spectrum over the frequency band of interest the constraint in (4.4.2) becomes \( P W = I \) [Er93]. It can easily be verified that the solution \( \hat{W}_C \) of the beamforming problem (4.4.2) is given by
\[ \hat{W}_C = \left( R^{-1} r_d (r_d^T R^{-1} r_d)^{-1} \right)^{-1} \]  
(4.4.5)

4.5 Digital Beamforming

In this section, in a brief review of digital beamforming, the process of forming beams in various directions is described [God97]. First, consider the analog beamformer structure shown in Figure 4.5, where signals from all elements are weighted, delayed, and summed to form a beam. The output of the beamformer is given by
\[ y(t) = \sum_{l=1}^{L} w_l x_l (t - \tau_l(\phi)) \]  
(4.5.1)

The delay in front of each element is adjusted such that the signals induced from a given direction, where the beam needs to be pointed to, are aligned after the delays. The weights are adjusted to shape the beam.

In digital beamforming [Pri78, Dud77, Muc84, Pri79, Fan84, Mar89, Rud69, Gab84, Bra80, Syl86, DeM77], the weighted signals from various elements are sampled, stored, and summed after appropriate delays to form beams. The required delay is provided by selecting samples from different elements such that the selected samples are taken at different times. Each sample is delayed by an integer multiple of the sampling interval \( \Delta \). The process is shown in Figure 4.6 for a linear array of equispaced elements where the samples of weighted signals are shown as circles. Weights on each element are not shown.

![FIGURE 4.5](DELAY AND SUM PROCESSOR STRUCTURE)
Assume that a beam is to be formed in direction $\phi_2$. Let the direction be such that

$$\tau_l(\phi_2) = (l - 1)\Delta\quad (4.5.2)$$

Thus, the signal from the $l$th element needs to be delayed by $(l - 1)\Delta$ seconds. This may be accomplished by summing the samples on a line marked with symbol A in Figure 4.6. For this case, the samples from Element 1 are not delayed, samples from Element 2 are delayed by one sample, and so on.

Similarly, a beam may be steered in direction $\phi_3$ by summing the samples connected by the line marked with symbol B in Figure 4.6, where the signals from $L$th element are not delayed, samples from element $L - 1$ are delayed by one sample, and so on. The beam formed in direction $\phi_4$, by summing the samples connected by the line marked with symbol C, does not require any delay.

It follows from the above discussion that when using this process, one can only form beams in directions that require delays equal to some integer multiple of the sampling interval, that is,

$$\tau_l(\phi) = k\Delta\quad (4.5.3)$$

where $k, l = 1, 2, ..., L$ are integers. The number of discrete directions where a beam can be exactly pointed increases with increased sampling as shown in Figure 4.7, where the sampling interval is $\Delta/2$. The figure shows that additional beams in directions $\phi_4$ and $\phi_5$ may be formed. These exact beams are normally referred to as synchronous or natural beams [Pri78], and it is possible to form a number of these beams simultaneously using a separate summing network for each beam.
The practical requirement of an adequate set of directions where simultaneous beams need to be pointed implies that the array signals be sampled at much higher rates than required by Nyquist criteria to reconstruct the wave form back from the samples [Pap75]. The high sampling rate means a large number of storage requirements along with high-speed input-output devices, analog-to-digital converters, and large bandwidth cables [Pri78].

The high sampling rate requirement may be overcome by digital interpolation [Pri78, Pri79, Syl86], which basically simulates the samples generated by high sampling rates and thus increases the effective sampling rate. The process works by sampling the array signal at a Nyquist rate or higher and padding with zeros between each sample to form a new sequence. The number of zeros padded decides the effective sampling rate. For the sampling rate to increase by $L$-fold, $L - 1$ zeros are padded to create a sequence as big as if it were created by high-speed sampling. The padded sequences then are used for digital beamforming by selecting appropriate samples as required and the beam output is passed through an FIR filter to remove the unwanted spectrum. This filter is normally referred to as an interpolation filter. The beams formed by interpolation beamformers have a slightly higher side-lobe level.

A tutorial introduction to digital interpolation beamformers is given in [Pri78], whereas some additional fundamentals of digital array processing may be found in [Dud77]. A comparison of many approaches to digital beamforming implementations is discussed in [Muc84, Mar89], who show how a real-time implementation is a trade-off between various conflicting requirements of hardware complexities, memory requirements, and system performance.

The shape of a beam, particularly its beam width, is controlled by the size of the array. Generally, a narrow beam results from a larger array. In practice, the array size is fixed and its extent is limited. A process known as extrapolation may be used [Fan84] during digital beamforming to simulate a large array extent resulting in improved beam pattern.
As the interpolation increases the effective sampling rate, the extrapolation extends the effective array length. More information on signal extrapolation schemes may be found in [Pap75, Sul91, Cad79, Son82, Jai81, Sna83].

Digital beamforming techniques for mobile satellite communications are examined in [Chu90] by studying a configuration of a digital beamforming system capable of working in transmit and receive modes. Digital beamforming for mobile satellite communications has also been reported in [Geb95, Chu90]. An introduction to digital beamforming for mobile communications may be found in [Ste87].

### 4.6 Frequency Domain Processing

A general structure of the element-space frequency domain processor is shown in Figure 4.8, where broadband signals from each element are transformed into a frequency domain using the discrete Fourier transform (DFT), and each frequency bin is processed by a narrowband processor structure. The weighted signals from all elements are summed to produce an output at each bin. The weights are selected independently by minimizing the mean output power at each frequency bin subject to steering direction constraints. Thus, the weights required for each frequency bin are selected independently and this selection may be performed in parallel, leading to faster weight update. When an adaptive algorithm such as the LMS algorithm is used for weight updating, a different step size may be used for each bin leading to faster convergence.

![Frequency domain processor structure.](image.png)

**FIGURE 4.8**
Frequency domain processor structure.
Various aspects of array signal processing in a frequency domain are reported in the literature [Hod79, Arm74, Den78, Nar81, Web84, Shy85, Flo88, Ber86, Ree85, Man82, Kum90, Cla83, Zhu90, God95, Hin81]. The optimum performance of the time domain and frequency domain processors are the same only when the signals in various frequency bins are independent. This independence assumption is mostly made in the study of frequency domain processing. When the assumption does not hold, the frequency domain processor may be suboptimal. Some of the tradeoffs and a comparison of the two processors are discussed in [Hod79, God95].

A study of the frequency domain algorithm [Web84] for coherent signals indicates that the frequency domain method is insensitive to the sampling rate, and may be able to reduce the effects of element malfunctioning on the beam pattern. A study in [Shy85] shows that due to its modular parallel structure, beam forming in the frequency domain is well suited for VLSI implementation and is less sensitive to the coefficient quantization. Computational advantages of the frequency domain method (FDM) for bearing estimation are discussed in [Ree85, Kum90, Hin81], and for correlated data are considered in [Man82, Zhu90]. A general treatment of time and frequency domain realization with a view to compare the structure of various algorithms of weight estimation in a unified manner is provided in [God95].

In this section, frequency domain processing is studied in detail using a constrained element space processor, and relationships between the time domain processor and the frequency domain processor are established [God95].

4.6.1 Description

Consider an L-element array immersed in a noise field consisting of uncorrelated broadband directional sources and white noise. Let \( s(t) \) be a broadband real signal, with the power spectral density \( S(f) \) induced on a reference element due to a source. The autocorrelation function

\[
\rho(\tau) = E[s(t)s(t+\tau)]
\]  

(4.6.1)

is the inverse Fourier transform of \( S(f) \), that is,

\[
\rho(\tau) = \int_{-\infty}^{\infty} S(f)e^{2\pi if\tau} df
\]  

(4.6.2)

Let \( x_i(t) \) denote the time wave form derived from the \( i \)-th element after presteering. Let these wave forms be sampled at frequency \( f_s \). Denoting the sampling interval by \( T \), the sampled wave form derived from \( i \)-th element becomes \( x_i(nT) \). As the sampling period does not play any role in the treatment that follows, it has been omitted for ease of notation.

Let \( x(n) \) denote the \( L \) samples after presteering delays, that is,

\[
x(n) = [x_1(n), x_2(n), \ldots, x_L(n)]^T
\]  

(4.6.3)

Now consider \( N \) array samples \( x(n-i+1), i = 1, \ldots, N \), with \( x(n) \) denoting the most recent samples. Let these be processed by the frequency domain processor structure shown in Figure 4.8, where these are first converted into \( N \) frequency bins using discrete Fourier transforms and then processed using \( N \) narrowband processors.
Let \( \tilde{y}(k) \) denote the output of the kth bin. From Figure 4.8, it follows that

\[
\tilde{y}(k) = h^H(k)\tilde{x}(k) \tag{4.6.4}
\]

where an L-dimensional complex vector \( h(k) \) denotes the L weights of the narrowband processor for the kth bin, that is,

\[
h(k) = [h_1(k), \ldots, h_L(k)]^T \tag{4.6.5}
\]

with \( h_l(k) \) denoting the weight on the lth channel.

The L-dimensional complex vector \( \tilde{x}(k) \) denotes the L-frequency domain samples, that is,

\[
\tilde{x}(k) = [\tilde{x}_1(k), \ldots, \tilde{x}_L(k)]^T \tag{4.6.6}
\]

with \( \tilde{x}_l(k) \) denoting the frequency domain samples from the lth channel. The N frequency samples of the lth channel \( \tilde{x}_l(k), k = 0, 1, \ldots, N - 1 \) are related to the N time samples \( x_l(n), n = 1, 2, \ldots, N \) by the discrete Fourier transform [Bur85], that is,

\[
\tilde{x}_l(k) = \sum_{n=1}^{N} x_{l,i} e^{-\frac{2\pi}{N}(i-1)k} \quad k = 0, 1, \ldots, N - 1 \tag{4.6.7}
\]

where \( x_{l,i} \equiv x_l(n - i + 1), i = 1, 2, \ldots, N \) and \( x_{l,1} \equiv x_l(n) \) denotes the most recent sample.

Thus, using N array samples \( x(n - i + 1), i = 1, 2, \ldots, N \), the frequency domain processor produces N frequency domain outputs \( \tilde{y}(k), k = 0, 1, \ldots, N - 1 \). These are converted into N output time samples \( y(n - i + 1), i = 1, 2, \ldots, N \) using the inverse DFT, that is,

\[
y(n - i + 1) = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{y}(k) e^{\frac{2\pi}{N}(i-1)k} \tag{4.6.8}
\]

where \( y(n) \) denote the most recent output.

The most recent output corresponds to \( i = 1 \) in the LHS of (4.6.8). Thus, it follows from (4.6.8) that

\[
y(n) = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{y}(k) \tag{4.6.9}
\]

Thus, the most recent output sample may be obtained by averaging the output of N narrowband processors without computing N-point inverse DFT. This aspect is exploited in sliding window processing, where N most recent input samples are converted into frequency domain using DFT, and the time domain output is obtained by averaging the N outputs. In this scheme, every time a new input sample arrives, a full cycle involving conversion to frequency domain using DFT, narrowband processing, and computation of output using (4.6.9) needs to be carried out.
The other processing scheme discussed previously where N input time samples are collected, converted to the frequency domain, processed using N narrowband processors, and converted back to N output time samples using the inverse DFT is referred to as block processing [Com88]. Thus, in summary, in block processing a block of N input samples is collected to be processed using narrowband processing to obtain N output time samples. On the other hand, in sliding window processing every time a new sample arrives, the complete processing cycle is invoked. The difference in the processing cycle for the two schemes is that the sliding window processing does not use the inverse DFT.

In both cases, once the N time samples are converted into N frequency domain samples, any of the narrowband processing schemes discussed in previous chapters may be used. In the next section, the relationship between the frequency domain processing discussed in this section and the time domain processing using the TDL structure discussed earlier is established.

4.6.2 Relationship with Tapped-Delay Line Structure Processing

Assume that the N array samples \(x(n - i + 1), i = 1, 2, \ldots, N\) are processed by two processor structures, namely, the TDL structure shown in Figure 4.1 where the processing is carried out in the time domain and frequency domain processor structure shown in Figure 4.8 where the processing is carried out in frequency domain. In the following, the conditions are derived for the two processors to produce identical outputs.

4.6.2.1 Weight Relationship

The output of the time domain processor shown in Figure 4.1 is given by

\[
y(n) = W^T X(n)
\]

where \(W\) is defined in (4.1.4) and \(X(n)\) is defined in (4.1.6) with \(t\) replaced by \(n\). Rewrite (4.6.10) as

\[
y(n) = \sum_{i=1}^{L} \sum_{m=1}^{j} w_{lm} x_{mn} (n - (m - 1))
\]

\[
= \sum_{i=1}^{L} \sum_{m=1}^{j} w_{lm} x_{1}
\]

(4.6.11)

It follows from (4.6.11) that the output at time \(n\) depends on the present input \(x_i(n)\) and \(J - 1\) previous inputs, namely, \(x_i(n - 1), \ldots, x_i(n - J + 1)\). Thus, for a given set of N samples under consideration, one is able to obtain only \(N - (J + 1)\) output samples, namely \(y(n), y(n - 1), \ldots, y(n - N + J)\). This implies that for \(J = N\), these samples only produce one output sample, given by

\[
y(n) = \sum_{i=1}^{L} \sum_{m=1}^{N} w_{lm} x_{mn}
\]

(4.6.12)

Now, consider the frequency domain processor processing the same N samples. The most recent time sample for the frequency domain processor is given by (4.6.9). For the
two processors to produce identical outputs, the time samples given by (4.6.9) and (4.6.12) must be equal, that is,

\[ \sum_{l=1}^{L} \sum_{m=1}^{N} w_{lm} x_{lm} = \frac{1}{N} \sum_{k=0}^{N-1} y(k) \]  

(4.6.13)

Rewrite (4.6.4) as

\[ \hat{y}(k) = \sum_{l=1}^{L} h^*(l) \tilde{x}(k), \quad k = 0, 1, \ldots, N - 1 \]  

(4.6.14)

It follows from (4.6.13), (4.6.14), and (4.6.7) that

\[ \sum_{l=1}^{L} \sum_{m=1}^{N} w_{lm} x_{lm} = \frac{1}{N} \sum_{k=0}^{N-1} h^*(l) \sum_{l=1}^{L} x_{l} e^{-j \frac{2\pi}{N} (l-1)k} \]  

(4.6.15)

\[ = \sum_{l=1}^{L} \sum_{m=1}^{N} x_{l} \left( \frac{1}{N} \sum_{k=0}^{N-1} h^*(k) e^{-j \frac{2\pi}{N} (l-1)k} \right) \]

The identity holds if

\[ w_{lm} = \frac{1}{N} \sum_{k=0}^{N-1} h^*(k) e^{-j \frac{2\pi}{N} (m-1)k}, \quad l = 1, 2, \ldots, L, \quad m = 1, 2, \ldots, N \]  

(4.6.16)

Thus,

\[ w_{lm}, \quad m = 1, 2, \ldots, N = \text{DFT} \left\{ \frac{h^*(k)}{N}, \quad k = 0, 1, \ldots, N - 1 \right\} \]  

(4.6.17)

It follows then that both processors produce identical outputs when the TDL structure has length equal to N and the two sets of weights are related by (4.6.16).

### 4.6.2.2 Matrix Relationship

Consider the output sequence of the frequency domain structure of Figure 4.8. Assume that \( M_0 \) sets, each of N samples, are being processed. Let \( \hat{y}(k, m) \) denote the output of the kth frequency bin due to the mth data block. For a given \( h(k) \), the mean output power of the kth bin is given by

\[ P(k) = \frac{1}{M_0} \sum_{m=0}^{M_0-1} \hat{y}(k,m) \hat{y}^*(k,m) \]  

(4.6.18)

Following (4.6.4), the output of kth bin due to mth set is given by
\[ \hat{y}(k, m) = h^H(k) \tilde{x}(k, m) \]  

(4.6.19)

This along with (4.6.18) implies that

\[ P(k) = h^H(k) R_x(k) h(k) \]  

(4.6.20)

where

\[ R_x(k) = \frac{1}{M_b} \sum_{m=0}^{M_b-1} \hat{X}(k, m) \tilde{X}^H(k, m) \]  

(4.6.21)

is an estimate of the array correlation matrix for the kth bin.

It follows from (4.6.21) that

\[ (R_x(k))_{l,n} = \frac{1}{M_b} \sum_{m=0}^{M_b-1} \tilde{X}_l(k, m) \tilde{X}_n^*(k, m) \]  

(4.6.22)

Since

\[ \tilde{X}_l(k, m) = \sum_{i=1}^{N} x_{n}(m) e^{-2\pi j i k} \]

it follows from (4.6.22) that

\[ (R_x(k))_{l,n} = \frac{1}{M_b} \sum_{m=0}^{M_b-1} \sum_{i=1}^{N} x_{n}(m) e^{-2\pi j i k} \sum_{i=1}^{N} x_{m}(m) e^{2\pi j i k} \]  

(4.6.23)

Note that \( x_i \) is a real variable and \( \tilde{x}_i \) is a complex variable. Define an N-dimensional vector \( x_i(m) \) representing N samples in the tapped delay line structure on the lth channel as

\[ x_i(m) = \begin{bmatrix} x_{n}(m) \\ M \\ x_{\Delta N}(m) \end{bmatrix}, \quad l = 1, 2, \ldots, L \]  

(4.6.24)

and an N-dimensional vector \( e(k) \) representing N phasers at kth bin as

\[ e(k) = \begin{bmatrix} 1 \\ M e^{2\pi j k} \\ M e^{2\pi j (N-1) k} \\ e^{2\pi j N k} \end{bmatrix} \]  

(4.6.25)
From (4.6.23) to (4.6.25) it follows that

\[
(R_i(k))_{mn} = \frac{1}{M_0} \sum_{m=0}^{M_0-1} e^{iH(k)x_i(m)x_n^T(m)e(k)}
\]

(4.6.26)

where

\[
\hat{R}_{i,n} = \frac{1}{M_0} \sum_{m=0}^{M_0-1} x_i(m)x_n^T(m)
\]

(4.6.27)

is an $N \times N$ matrix denoting the correlation between the $l$th and $n$th elements for the tapped delay line structure, estimated from $M_0$ sets of samples, each of length $N$. It is an unbiased estimate for the correlation between the $l$th and $n$th elements for given $M_0$ samples. As $M_0$ increases, the estimate asymptotically approaches the true correlation. Therefore, the relationship between the frequency domain and time domain matrices holds for the true correlation matrices.

Throughout the chapter, $R_f$ and $R$ are used to denote the frequency domain and time domain array correlation matrices, respectively, as well as their unbiased estimates. Furthermore, the correlation between the $m$th and $n$th taps is denoted by the matrix $(R_{mn})$, and the correlation between $l$th and $i$th elements is denoted by the matrix $(\hat{R}_{l,i})$.

### 4.6.2.3 Derivation of $R_f(k)$

Let $(R_{mn})_{ij}$ denote the correlation between $l$th and $i$th elements after $m$th and $n$th taps due to a source in direction $(\phi, \theta)$. An expression for $(R_{mn})_{ij}$ from (4.1.11) is given by

\[
\left(R_{mn}\right)_{ij} = \rho[(m-n)T + T_j + \tau_i - \tau_j]
\]

(4.6.28)

where the arguments $\phi$ and $\theta$ have been suppressed for the ease of notation.

As the correlation function is symmetrical for real signals, it follows from (4.6.2) and (4.6.28) that

\[
\left(R_{mn}\right)_{ij} = \int_{-\infty}^{\infty} S(f)e^{-2\pi f (m-n)}e^{i2\pi f(T_j - T_i)}e^{i2\pi f(\tau_i - \tau_j)}df
\]

(4.6.29)

Define an $N$-dimensional vector $e(f)$ denoting $N$ phasors at frequency $f$ as

\[
\begin{bmatrix}
1 \\
e^{-i2\pi fT} \\
M \phantom{1}
\end{bmatrix}
\]

(4.6.30)

It follows from (4.6.29) that the $N \times N$ matrix denoting the correlation between $l$th and $i$th elements is given by
Equation (4.6.31) along with (4.6.26) implies that

\[
\left( \hat{R}_{i,i} \right) = \int_{-\infty}^{\infty} S(f)e^{i\eta(f)}e^{i2\pi(t_i - \tau_i + T_i)} df
\]  

(4.6.31)

Substituting for \( e(f) \) and \( e(k) \), (4.6.33) becomes

\[
\left( \hat{R}_{i,i} \right) = e^{i\eta(k)}(\hat{R}_{i,i})e(k)
\]

(4.6.32)

where

\[
a(f,k) = e^{i\eta(k)}e^{i\eta}(f)e^{i\eta}(f)e^{i\eta}(k)
\]

(4.6.33)

Using steering vector notation, one obtains from (4.6.32) the following compact expression for \( R_f(k) \):

\[
R_f(k) = \int_{-\infty}^{\infty} S(f)a(f,k)\tilde{S}(f,\phi,\theta)\tilde{S}^{\dagger}(f,\phi,\theta) df
\]

(4.6.35)

where \( \tilde{S}(f,\phi,\theta) \) denotes the steering vector in \((\phi,\theta)\) direction for an array presteered in \((\phi_0,\theta_0)\).

### 4.6.2.4 Array with Presteering Delays

Noting that the steering vector in \((\theta_0,\theta_0)\) direction for an array presteered in \((\phi_0,\theta_0)\) is identical to \(1\), it follows from (4.6.33) and (4.6.35) that the matrix \( R_f(k) \) due to a source in a presteered direction is given by

\[
R_f(k) = \alpha(k)11^T
\]

(4.6.36)

where

\[
\alpha(k) = e^{i\eta(k)} \left[ \int_{-\infty}^{\infty} S(f)e^{i\eta}(f) df \right] e(k)
\]

(4.6.37)

The matrix in the square brackets on the right side of (4.6.37) is a spectrum-dependent quantity. Let it be denoted by \( A \). Its \((m,n)\)th element \( A_{mn} \) is given by
Am,n can be evaluated for a specific spectrum using (4.6.38). For example, for a brick-wall type of spectrum given by

\[ S(f) = \begin{cases} a_0 & f_L < |f| < f_H \\ 0 & \text{otherwise} \end{cases} \]  

(4.6.39)

it becomes

\[ A_{m,n} = 2a_0 \left[ \frac{\sin 2\pi f_H (m-n)}{2\pi (m-n)} - \frac{\sin 2\pi f_L (m-n)}{2\pi (m-n)} \right] \]  

(4.6.40)

where \( f_H \) and \( f_L \) are assumed to be normalized with respect to the sampling frequency.

### 4.6.2.5 Array without Presteering Delays

For this case, the steering delays \( T_i = 0, i = 1, 2, \ldots, L \). Thus, it follows from (4.6.32) that

\[ R_i(k) = \int S(f) a(f, k) S(f, \phi, \theta) S^H(f, \phi, \theta) df \]  

(4.6.41)

where \( S(f, \phi, \theta) \) denotes the steering vector in \((\phi, \theta)\) direction for an array without presteering. Note that this matrix in general is not equal to a matrix that depends on the energy from the kth bin only, namely

\[ \tilde{R}_i(k) = \int_{k-\Delta f}^{(k+1)\Delta f} S(f) S(f, \phi, \theta) S^H(f, \phi, \theta) df \]  

(4.6.42)

with \( \Delta f = 1/N \) denoting the bandwidth of a frequency bin.

### 4.6.2.6 Discussion and Comments

The results presented here show that when a broadband correlation matrix is transformed into narrowband matrices, these matrices depend on the spectrum of the signal beyond the bandwidth of their particular frequency bins, which is controlled by the parameter \( a(f, k) \) given by (4.6.34). Figure 4.9 and Figure 4.10 show how this parameter behaves as a function of the frequency for \( N = 10 \) and \( N = 100 \), respectively. The plots are for \( k = 0 \), and show the normalized value of the parameter with respect to its maximum value \( N^2 \).

### 4.6.3 Transformation of Constraints

As discussed in Section 4.3, the weights of the broadband element space processor using TDL are subjected to various constraints to make the processor robust against various uncertainties. In this section, some of these constraints are transformed for narrowband processors operating in the frequency domain.
4.6.3.1 Point Constraints

Assume that the weights of the TDL are constrained in the look direction, such that

$$\sum_{l=1}^{L} w_{lm} = f_m, \quad m = 1, 2, ..., N$$  \hspace{1cm} (4.6.43)

where $f_m$, $m = 1, 2, ..., N$ specifies the frequency response of the processor in the look direction as discussed in Section 4.1.2. Note that (4.6.43) is obtained by rewriting (4.1.24) with $J$ replaced by $N$. Summing on both sides of (4.6.16) over $l$,

$$\sum_{l=1}^{L} w_{lm} = \frac{1}{N} \sum_{k=0}^{N-1} \sum_{l=1}^{L} h_l^{*}(k)e^{2\pi i (m-1)k}, \quad m = 1, 2, ..., N$$  \hspace{1cm} (4.6.44)

This along with (4.6.43) implies that

$$f_m = \frac{1}{N} \sum_{k=0}^{N-1} \sum_{l=1}^{L} h_l^{*}(k)e^{2\pi i (m-1)k}, \quad m = 1, 2, ..., N$$  \hspace{1cm} (4.6.45)
Taking the inverse DFT on both sides, after rearrangements

\[
\sum_{l=1}^{L} h_l^i(k) = \sum_{m=1}^{L} f_m e^{\frac{2\pi j(m-1)k}{N}}, \quad k = 0, 1, \ldots, N-1
\]  \hspace{1cm} (4.6.46)

Thus, the equivalent constraints on the weights of the kth bin processor are given by

\[
h^H(k) = \tilde{f}_k, \quad k = 0, 1, 2, \ldots, N-1
\]  \hspace{1cm} (4.6.47)

where \( \tilde{f}_k \) specifies the constraint on the weights of the kth bin processor. It follows from (4.6.46) that

\[
\tilde{f}_k = \sum_{m=1}^{N} f_m e^{\frac{2\pi j(m-1)k}{N}}
\]  \hspace{1cm} (4.6.48)

Thus, \( \tilde{f}_k, k = 0, 1, 2, \ldots, N-1 \) are the coefficients of inverse DFT of \( N_{fm} \), \( m = 1, 2, \ldots, N \).

### 4.6.3.2 Derivative Constraints

The derivative constraints for the broadband processor are discussed in detail in Section 4.3. These are imposed alongside the point constraints to broaden the beamwidth, which

\[\text{FIGURE 4.10}\]

The parameter \( a(f,k) \) defined by (4.6.33), normalized with respect to its maximum value, vs. frequency for \( N = 100 \) and \( k = 0 \). (From Godara, L.C., Application of the fast Fourier transform to broadband beamforming, *J. Acoust. Soc. Am.*, 98, 230–240, 1995. With permission.)
helps to overcome the pointing errors. First-order constraints are given by (4.3.18) and (4.3.19). Rewriting,

\[ 1^T \Lambda_{\phi}(\phi_0, \theta_0) w_m = 0, \quad m = 1, 2, \ldots, N \]  
(4.6.49)

and

\[ 1^T \Lambda_{\theta}(\phi_0, \theta_0) w_m = 0, \quad m = 1, 2, \ldots, N \]  
(4.6.50)

where \( \Lambda_{\phi}(\phi, \theta) \) and \( \Lambda_{\theta}(\phi, \theta) \) are diagonal matrices given by

\[ \Lambda_{\phi}(\phi, \theta) = \begin{bmatrix} \frac{\partial \tau_1(\phi, \theta)}{\partial \phi} & 0 \\ 0 & \frac{\partial \tau_1(\phi, \theta)}{\partial \theta} \end{bmatrix} \]  
(4.6.51)

and

\[ \Lambda_{\theta}(\phi, \theta) = \begin{bmatrix} \frac{\partial \tau_1(\phi, \theta)}{\partial \theta} & 0 \\ 0 & \frac{\partial \tau_1(\phi, \theta)}{\partial \phi} \end{bmatrix} \]  
(4.6.52)

Rewrite (4.6.16) in vector notation as

\[ w_m = \frac{1}{N} \sum_{k=0}^{N-1} h^*(k) e^{-j2\pi(m-1)k}, \quad m = 1, 2, \ldots, N \]  
(4.6.53)

Substituting in (4.6.49) and (4.6.50),

\[ \frac{1}{N} \sum_{k=0}^{N-1} 1^T \Lambda_{\phi}(\phi_0, \theta_0) h^*(k) e^{-j2\pi(m-1)k} = 0, \quad m = 1, 2, \ldots, N \]  
(4.6.54)

and

\[ \frac{1}{N} \sum_{k=0}^{N-1} 1^T \Lambda_{\theta}(\phi_0, \theta_0) h^*(k) e^{-j2\pi(m-1)k} = 0, \quad m = 1, 2, \ldots, N \]  
(4.6.55)

Taking the inverse DFT on both sides of (4.6.54) and (4.6.55), the following equivalent constraints on the narrowband weights result:

\[ 1^T \Lambda_{\phi}(\phi_0, \theta_0) h^*(k) = 0, \quad k = 0, 1, \ldots, N-1 \]  
(4.6.56)
and

\[ \mathbf{1}^T \Lambda_0(\phi_0, \theta_0) \mathbf{h}^* (k) = 0, \quad k = 0, 1, \ldots, N - 1 \]  

(4.6.57)

Alternatively, these may be expressed as

\[ \mathbf{h}^{(i)}(k) \Lambda_0(\phi_0, \theta_0) \mathbf{1} = 0, \quad k = 0, 1, \ldots, N - 1 \]  

(4.6.58)

and

\[ \mathbf{h}^{(i)}(k) \Lambda_0(\phi_0, \theta_0) \mathbf{1} = 0, \quad k = 0, 1, \ldots, N - 1 \]  

(4.6.59)

Following a similar procedure, the second-order derivative constraints for the weights of the broadband processor given by (4.3.20) to (4.3.25) can be transformed for the weights of the narrowband processors. These are given by

\[ \mathbf{h}^{(i)}(k) \frac{\partial \Lambda_0(\phi, \theta)}{\partial \phi} \bigg|_{(\phi_0, \theta_0)} = 0, \quad k = 0, 1, \ldots, N - 1 \]  

(4.6.60)

\[ \mathbf{h}^{(i)}(k) \Lambda_0^2(\phi_0, \theta_0) \mathbf{1} = 0, \quad k = 0, 1, \ldots, N - 1 \]  

(4.6.61)

\[ \mathbf{h}^{(i)}(k) \frac{\partial \Lambda_0(\phi, \theta)}{\partial \theta} \bigg|_{(\phi_0, \theta_0)} = 0, \quad k = 0, 1, \ldots, N - 1 \]  

(4.6.62)

\[ \mathbf{h}^{(i)}(k) \Lambda_0^2(\phi_0, \theta_0) \mathbf{1} = 0, \quad k = 0, 1, \ldots, N - 1 \]  

(4.6.63)

\[ \mathbf{h}^{(i)}(k) \frac{\partial \Lambda_0(\phi, \theta)}{\partial \theta} \bigg|_{(\phi_0, \theta_0)} = 0, \quad k = 0, 1, \ldots, N - 1 \]  

(4.6.64)

and

\[ \mathbf{h}^{(i)}(k) \Lambda_0(\phi_0, \theta_0) \Lambda_0(\phi_0, \theta_0) \mathbf{1} = 0, \quad k = 0, 1, \ldots, N - 1 \]  

(4.6.65)

### 4.7 Broadband Processing Using Discrete Fourier Transform Method

In the previous section, an FDM to process broadband signals was discussed in which broadband time domain data are transformed into narrowband frequency domain data using DFT, and are then processed using narrowband processing schemes. The processed signals are transformed into broadband time domain signals using the inverse DFT. Thus,
the implementation is done using narrowband processors operating at different frequency bins.

In contrast, in the time domain method (TDM) discussed in Section 4.1.3, the processor is implemented in a time domain using a TDL structure, as shown in Figure 4.1. The weights of the broadband processor are obtained by solving the constrained beamforming problem when the look direction information is available.

In this section, the DFT method for estimating the weights of the broadband processor using a TDL structure of Figure 4.1 is discussed, and the performance of the broadband processor using the DFT method is compared with that using the time domain method [God99]. The method is discussed by considering the beamforming problem with the point constraint. In this case the TDM solves the following beamforming problem:

$$\begin{align*}
\text{minimize} & \quad W^T R W \\
\text{subject to} & \quad C^T W = f
\end{align*}$$

(4.7.1)

where $C$ is the constraint matrix defined in (4.1.26) and $f$ is a J-dimensional vector selected to specify the frequency response in the look direction. The weights $\hat{W}$ estimated by the TDM are the solution of (4.7.1), and are given by

$$\hat{W} = R^{-1} C (C^T R^{-1} C)^{-1} C^T f \quad (4.7.2)$$

The DFT method estimates the weights of the broadband processor of Figure 4.1 in two steps. First, it estimates the weights of narrowband processors by minimizing the mean output power of each frequency bin, and then uses the relations developed in the last section between the time domain and frequency domain structures for identical outputs to transform these into the required weights. It also maintains the same frequency response in the look direction as is done by the TDM using the appropriate constraints developed in the last section. Figure 4.11 shows a schematic diagram of the DFT method.

![Schematic diagram of broadband processor using DFT methods.](image-url)
The similarity between the TDM and DFT methods is that both estimate the weights of the TDL structure of Figure 4.1. The main difference between the two is that this method minimizes the mean output power of each frequency bin, rather than minimizing the mean output power of the processor, as is done by the TDM. This implies that if the sum of the mean output powers from all frequency bins is not equal to the mean output power of the processor shown in Figure 4.1, then the realized processor using the DFT method does not maximize the mean output SNR in the absence of errors, as is case with the processor using the TDM to estimate the weights. However, this method offers the potential for a large amount of computational savings for real-time applications due to its parallel nature of implementation as discussed later in this section.

As the DFT method minimizes the mean output power of each frequency bin and then uses the relations between the time domain and the frequency domain structures for the identical outputs, the performance of the realized processor in the absence of implementation errors is the same as the processor implemented in the frequency domain. However, there are important differences.

The main difference between the DFT method and the FDM is that this method uses the optimized weights of the narrowband processors operating at different frequency bins to estimate the optimal weights of the time domain broadband processor. The processor is implemented in the time domain and the received signal flows in the time domain structure without encountering the delay associated with the frequency domain implementation. This may be important for some applications.

As broadband processor performance using the DFT method to estimate the weights when implemented in the time domain is identical to that implemented in the frequency domain, this fact presents a framework for comparing the performance of time domain and frequency domain implementations under identical conditions.

### 4.7.1 Weight Estimation

The DFT method uses the following procedure to estimate the weights of the time-domain broadband-constrained processor using a TDL structure of length J.

1. Estimate narrowband array correlation matrices $R_i(k)$, $k = 0, \ldots, J - 1$ using

   \[ R_i(k)_{ji} = e^{H}(k)\bar{R}_i(k) e(k), \quad 1, i = 1, \ldots, L \]  \hspace{1cm} (4.7.3)

   where

   \[ e(k) = \left[ 1, e^{\frac{2\pi}{J}k}, \ldots, e^{\frac{2\pi}{J}(J-1)k} \right]^T \]  \hspace{1cm} (4.7.4)

   and $(\bar{R}_i(k))$ is a $J \times J$ matrix denoting the correlation between samples from $l$th and $i$th elements given by (4.6.27).

2. Estimate $\hat{h}(k)$, $k = 0, \ldots, (J - 1)$ using

   \[ \hat{h}(k) = \frac{R_i^{-1}(k)\bar{f}_k^*}{T^*R_i^{-1}(k)\bar{f}_k} \]  \hspace{1cm} (4.7.5)
which are the solutions of the following narrowband beamforming problems:

\[
\text{minimize} \quad h^H(k)R_e(k)h(k)
\]
\[
h(k) \quad k = 0, \ldots, J - 1
\]
\[
\text{subject to} \quad h^H(k)I = \tilde{f}_k
\]

where

\[
\tilde{f}_k = \sum_{m=1}^{J} f_m e^{j2\pi(m-1)k} , \quad k = 0, \ldots, J - 1
\]  

Equation (4.7.7) ensures that the required frequency response in the desired direction is maintained. It should be noted that due to the symmetry property of the Fourier transform, one only needs to estimate \( \hat{J} \) narrowband weights \( \hat{h}(k) \), \( k = 0, \ldots, (\hat{J} - 1) \), where

\[
\hat{J} = \begin{cases} 
\frac{J+1}{2}, & \text{when } J \text{ is odd} \\
\frac{J}{2}, & \text{when } J \text{ is even}
\end{cases} 
\]  

3. Estimate the weights of the time domain structure of Figure 4.1 using

\[
\hat{w}_{m1} = \frac{1}{J} \sum_{k=0}^{J-1} \hat{h}^*_{\ell}(k)e^{-j2\pi(m-1)k} , \quad m = 1, \ldots, J, \quad \ell = 1, \ldots, L
\]  

The block diagram shown in Figure 4.12 summarizes the method to estimate the weights of the broadband processor using the proposed technique.

4.7.2 Performance Comparison

In this section, examples are presented to compare the output SNR of the processor using the weights estimated by the DFT method and the TDM. The weights for the TDM are computed using (4.7.2), whereas for the DFT method, they are computed using (4.7.3) to (4.7.9). Both methods use actual LJ \( \times \) LJ dimensional array correlation matrix \( R \), and produce LJ weights of the TDL structure.

A linear array of equispaced elements is used in the presence of one interference source. The element spacing is measured in wavelengths of the desired signal at the highest frequency. The signal bandwidth is expressed in terms of the normalized frequency with respect to sampling frequency. The sampling frequency is taken to be equal to twice the highest frequency of the desired signal. Thus, the normalized highest frequency of the desired signal is identical to 0.5. All directional sources are assumed to be of the brickwall type spectrum. The directional sources considered for the study are assumed to be of two bandwidths, referred to as the large bandwidth and the small bandwidth. The normalized frequency band for the large bandwidth is from \([0.15, 0.5]\), whereas for the small bandwidth it is \([0.45, 0.5]\). The desired signal of unit power is assumed to be present broadside to the array.
The output SNR is computed using

\[
\text{SNR} = \frac{\hat{W}^H R_{\text{s}} \hat{W}}{\hat{W}^H R_{\text{n}} \hat{W}}
\]

with \( R_{\text{s}} \) denoting the actual array correlation matrix due to signal only, and \( R_{\text{n}} \) denoting the actual array correlation matrix due to interference and background noise only. The SNR is plotted as a function of the angle of the interference by varying it from 0° to 180°. The array is constrained to have the all-pass response in the desired signal direction by selecting

\[
f_i = \begin{cases} 
1 & i = \frac{J+1}{2} \\
0 & \text{otherwise}
\end{cases}
\]

(4.7.10)

where the filter-length parameter \( J \) is assumed to be an odd integer.

The performance comparison is carried out by varying the length of the filter, number of elements in the array, the signal bandwidth, and interference-to-background-noise ratio to see how various parameters affect the result.

4.7.2.1 Effect of Filter Length

In order to compare the performance of the two methods for a different number of taps, a five-element array is used in the presence of a directional interference of power 10 dB above the signal level and the white noise power 10 dB below the signal level. Figure 4.13 shows \( \text{SNR}_T(\text{dB}) - \text{SNR}_D(\text{dB}) \), or equivalently, \( 10\log_{10}(\text{SNR}_T/\text{SNR}_D) \), as a function of interference angle for various filter lengths with \( \text{SNR}_T \) and \( \text{SNR}_D \), respectively, denoting the output SNR of the processor using the TDM and DFT methods.

© 2004 by CRC Press LLC
The two bullets on each curve indicate the beamwidth of the antenna array used. An expression for the beamwidth of the main lobe for the narrowband array with a large number of elements is given by [Col85]

\[ \text{Beamwidth} = 2 \arcsin \left( \frac{\lambda_0}{Ld} \right) \]

where \( \lambda_0 \) is the wavelength of the narrowband signal, and \( d \) is the element spacing in meters. Beamwidth for the broadband arrays has been taken to be the average of the two beamwidths computed at the lowest and the highest frequencies of the signal.

Figure 4.13 shows that the difference between the two SNRs is smaller when the interference is outside the main lobe than the case when it is within the main lobe, except when the interference is close to the look direction, in which case the processor generally is not used for its interference canceling capability and thus the situation is of no practical significance.

Above certain values of filter length, the results for the two bandwidths are different. For a small bandwidth signal, the difference between the two SNRs is very small when the interference is outside the main lobe, whereas it is reasonably high when it is within the main lobe, except when the interference is close to the look direction.

In the case of large bandwidth signals, the difference between the two output SNRs does not become as small as that for the small bandwidth case when the interference is outside the main lobe. The difference is more sensitive to the filter length above certain values for the large bandwidth case compared to the small bandwidth case and decreases as the filter length is increased.

### 4.7.2.2 Effect of Number of Elements in Array

For this example, the array element numbers are varied to study their effect on the performance difference of the two methods. Figure 4.14 shows the difference in the two SNRs for the both bandwidth sources. When the interference is outside the main lobe, an increase in the number of elements causes a decrease in the difference between the SNRs.
obtained by the two methods. This implies that increasing the number of elements improves the output SNR of the DFT method more than that of the TDM. Thus, when the interference is away from the look direction, the output SNR achievable by the DFT method approaches that of the TDM as the number of elements are increased. It should be noted that an increase in the number of elements in the array causes a decrease in the array beamwidth. Thus, as the number of elements in the array increases, the sector outside the main lobe increases. When an interference is present in this sector, the difference in the two SNRs is small.

Figure 4.14 also shows that the maximum value of the difference between the SNRs of the two methods increases as the number of elements in the array is increased. Furthermore, the direction of interference where the maximum difference between the SNRs occurs moves closer to the look direction as the number of elements is increased. Thus, it means that as the number of elements is increased, the interference canceling capability of the DFT method decreases relative to the TDM when the interference is close to the look direction.

This is a very interesting result. It says that the interference-canceling capability of the DFT method decreases, as the interference is very close to the look direction. In practice, a situation in which interference is close to the look direction rarely occurs, and even if it did, the interference-canceling capability of a processor is low for all practical purposes. However, in the presence of the look direction error, situations do occur when the desired source is not in the look direction and a processor treats it as interference. Extra precautions are necessary to overcome such situations. It appears from these results that the DFT method provides this beam-broadening capability naturally. This aspect of the DFT method is further explored in a later section to show that it is robust against look direction errors.

4.7.2.3 Effect of Interference Power

Figure 4.15 shows the difference in SNRs achievable by the TDM and DFT methods for various interference power levels at a given background noise. This figure shows that the performance of the processor using the DFT method deteriorates relative to the one using the TDM as interference power increases. This is true for small as well as large bandwidth signals. However, the deterioration is comparatively low when the interference is outside the main lobe. For the small bandwidth case, it is hardly noticeable.
4.7.3 Computational Requirement Comparison

In this section, examples are presented to compare the two methods based on their computational requirements to estimate weights of the TDL filter once the time domain array correlation matrix has been computed. The computation count reflects the floating-point operations required for weight estimation. Denoting the computation count for the TDM and the DFT method by \( O_T \) and \( O_D \), respectively, one obtains from (4.7.2) and (4.7.3) to (4.7.9) that

\[
O_T = 2J^2(L^3 + L^2 + 2L + 1) + 2LJ^2
\]  
(4.7.11)

and

\[
O_D = 4J^3L^2 + 4J^2L^2 + 4JL^3 + 8J^2L + 4JL^2 + 10JL
\]  
(4.7.12)

It should be noted that no allowance has been made in either of the methods for any special matrix structure that might be used to reduce computation count.

Figure 4.16 shows the ratio of the floating-point operation for the TDM to the DFT method, \( O_T/O_D \), as a function of the filter length for a varying number of elements. The TDM requires more computation than the DFT method, and a reduction of the order of 50 is possible using an array of 100 elements with a tapped delay line filter of length 100. It should be noted that an increase in filter length does not increase the computational savings as much as that achievable by increasing the number of elements. This is also evident from approximations of \( O_T \) and \( O_D \) for large \( J \) and \( L \). Approximating (4.7.11) and (4.7.12) for large \( J \) and \( L \) lead to

\[
O_T = 2J^3L^3
\]  
(4.7.13)

and

\[
O_D = 4J^3L^2
\]  
(4.7.14)
It follows from (4.7.13) and (4.7.14) that

\[
\frac{O_T}{O_D} = \frac{L}{2}
\]

(4.7.15)

Thus, it follows that the reduction of the order of \( L/2 \) is possible using the DFT method.

### 4.7.4 Schemes to Reduce Computation

In this section, a number of schemes are discussed to reduce the computational requirements for weight estimation using the DFT method.

#### 4.7.4.1 Limited Number of Bins Processing

The DFT method basically divides the entire spectrum into a number of frequency bins and processes signals in each bin. The weights at each bin are selected by minimizing the mean output power of each bin subject to constraints. In practice, the processing of all bins is not necessary, as the desired signal only covers a part of the spectrum, and thus one is only interested in canceling the interference that overlaps the signal bandwidth. Hence, one only needs to select weights by minimizing the mean output power of those bins that are in the vicinity of the signal bandwidth. The weights for bins outside this range may be selected to provide the maximum SNR under no directional sources. The conventional processor maximizes the output SNR in the absence of a directional source environment. Thus, selecting the weights of the antenna array is done by solving the optimal beamforming problem for those bins in the vicinity of the signal bandwidth and using equal weighting for other bins.

Since the equal weighting process does not require any computation, processing a limited number of bins reduces the computation load substantially, depending on the signal bandwidth. Computer analyses have shown that good results are obtained by processing two extra bins, one on each side of the signal bandwidth. Let \( \hat{J} \) denote the number of bins in the vicinity of the signal bandwidth that need to be processed by solving the optimization problem. Thus, \( \hat{J} \) is given by
where \( \lfloor x \rfloor \) denotes an integer greater than or equal to \( x \), and \( B_S \) denotes the normalized signal bandwidth, that is,

\[
\hat{J} = \begin{cases} 
\left[ (J+1)B_S \right] + 2 & \text{J is odd} \\
\left[ JB_S \right] + 2, & \text{J is even}
\end{cases}
\]  \hspace{1cm} (4.7.16)

Figure 4.17 shows the improvement in output SNR using this method compared with the DFT method vs. interference angle with number of elements = 5, \( J = 101 \), signal power = 1.0, interference power = 10.0, and white noise power = 0.1. (From Godara, L.C. and Jahromi, M.R.S., IEEE Trans. Signal Process., 47, 2386–2395, 1999. ©IEEE. With permission.)

\[
B_S = \frac{f_i - f_i}{f_i}
\]  \hspace{1cm} (4.7.17)

In order to illustrate the computational efficiency and performance improvement provided by this scheme, consider the parameters of Figure 4.17. For this case, the bin elimination scheme requires the processing of eight bins for the small bandwidth signal and 38 bins for the large bandwidth case, compared with 51 bins by the normal DFT method. The floating-point operations required to process these bins reduce to 16% and 75% of the normal DFT method for the two cases, respectively.

Figure 4.17 shows the improvement in output SNR using this method compared with the normal DFT method, which processes all the bins. The SNR improvement is evident for all interference directions. Thus, the processor using this method not only requires less computation time but also attains higher output SNR compared to the normal DFT method.

### 4.7.4.2 Parallel Processing Schemes

It is possible to increase the computation speed of the FDM by carrying out many computations in parallel. Hardware complexity and, thus, system cost, is expected to increase as more and more parallel processing is carried out to increase processing speed. Thus, there is a tradeoff between speed, which is vital in real-time operations, and system hardware cost. In this section, selected schemes are discussed, and their computational requirements are compared with the TDM.

#### 4.7.4.2.1 Parallel Processing Scheme 1

A block diagram showing the steps involved in this scheme to estimate the weights is shown in Figure 4.18. The scheme processes all frequency bins in parallel. The number of bins \( \hat{J} \) required to be processed for a \( J \)-tap filter is given by (4.7.8).
It should be noted that when weights are estimated for real-time operations, the time taken by the processor is an important measure of its performance, and the parallel processing scheme minimizes this time. Let \( OD_1 \) denote the computation count that reflects this fact, and which represents the time taken to estimate the weights rather than to measure total computation requirements. Then, the number of floating-point operations \( OD_1 \) required to estimate the weights is given by

\[
4.6.18
\]

4.7.4.2.2 Parallel Processing Scheme 2

This scheme not only processes all frequency bins in parallel but carries out matrix multiplications in parallel. Computation of each element of matrix \( R_f(k) \) requires the following operation:

\[
R_f(k)_{ij} = \mathbf{e}^H(k)(\hat{R}_{i,j})\mathbf{e}(k), \quad 1, 1, \ldots, L
\]

The scheme carries out multiplication of \( \mathbf{e}(k) \) with each column of \( (\hat{R}_{i,j}) \) in parallel to reduce computation time from \( J \) vector multiplications to \( 1 \) vector multiplication. The resulting vector is then multiplied with \( \mathbf{e}^H(k) \). The total time to compute each element of \( R_f(k) \) reduces from \( J + 1 \) complex vector multiplications to two complex vector multiplications. A block diagram of the scheme is shown in Figure 4.19.

Let the number of floating-point operations required to estimate weights with this scheme be denoted by \( OD_2 \). The solution, then, is

\[
OD_2 = 16L^2J + 8L^3 + 8J^2L + 8L^2 + 20L \tag{4.7.20}
\]

FIGURE 4.18

It should be noted that when weights are estimated for real-time operations, the time taken by the processor is an important measure of its performance, and the parallel processing scheme minimizes this time. Let \( OD_3 \) denote the computation count that reflects this fact, and which represents the time taken to estimate the weights rather than to measure total computation requirements. Then, the number of floating-point operations \( OD_3 \) required to estimate the weights is given by

\[
OD_3 = 8J^2L^2 + 8JL^2 + 8L^3 + 8J^2L + 8L^2 + 20L \tag{4.6.18}
\]

\[
4.7.4.2.2 \text{ Parallel Processing Scheme 2}
\]

This scheme not only processes all frequency bins in parallel but carries out matrix multiplications in parallel. Computation of each element of matrix \( R_f(k) \) requires the following operation:

\[
R_f(k)_{ij} = \mathbf{e}^H(k)(\hat{R}_{i,j})\mathbf{e}(k), \quad 1, 1, \ldots, L
\]

The scheme carries out multiplication of \( \mathbf{e}(k) \) with each column of \( (\hat{R}_{i,j}) \) in parallel to reduce computation time from \( J \) vector multiplications to \( 1 \) vector multiplication. The resulting vector is then multiplied with \( \mathbf{e}^H(k) \). The total time to compute each element of \( R_f(k) \) reduces from \( J + 1 \) complex vector multiplications to two complex vector multiplications. A block diagram of the scheme is shown in Figure 4.19.

Let the number of floating-point operations required to estimate weights with this scheme be denoted by \( OD_2 \). The solution, then, is

\[
OD_2 = 16L^2J + 8L^3 + 8J^2L + 8L^2 + 20L \tag{4.7.20}
\]
4.7.4.2.3 Parallel Processing Scheme 3

The FDM requires estimation of $L^2$ elements of matrix $R_f(k)$. This scheme estimates these elements in parallel, as shown in Figure 4.20. Thus, by computing $R_f(k)_{i,j}$, $i = 1, \ldots, L$ in parallel, it saves time of the order of $L^2$ in the matrix estimation. Let the floating-point operations required to estimate weights using this scheme be denoted by $O_{F3}$. Then

$$O_{F3} = 8J^2 + 8J + 8L^3 + 8J^2L + 8L^2 + 20L$$  \hspace{1cm} (4.7.21)$$

It should be noted that this scheme incorporates the processing of all frequency bins in parallel but does not carry out the multiplications of $e(k)$ with $(\hat{R}_i)$ in parallel, as suggested by Scheme 2. However, it is possible to carry out these operations in parallel by combining all of the above schemes to get the maximum speed for real-time operations. The floating-point operations required to estimate the weights using the combined scheme are given by the following expression:

$$O_{DC} = 16J + 8L^3 + 8J^2L + 8L^2 + 20L$$  \hspace{1cm} (4.7.22)$$

Figure 4.21 compares the ratios of floating-point operations required to estimate the optimal weights using the TDM to the FDM using various parallel processing schemes. Figure 4.21(a) shows the results for an array with 100 elements as a function of filter length. Figure 4.21(b) shows the floating-point operations ratio as a function of the number of elements using 100 taps. The successive parallel processing schemes require less processing time, and thus, a substantial increase in computation speed is possible using them. Using a 100-element array with a filter length of 100 taps, a 50-fold computational savings is
possible without any parallel processing, and 125,620-fold using all parallel processing
schemes, compared to the TDM.

It should be noted that the schemes discussed in this section to increase processing speed
tend to do so by increasing system complexity. The limited bin-processing scheme not
only reduces the computation requirements of the DFT method but also has a potential
to improve its performance without increasing system complexity, as is the case with
parallel processing schemes.
4.7.5 Discussion

It follows from the results presented so far that the DFT method is computationally more efficient than the TDM, the output SNR of the beamformer is lower when the weights are estimated by the DFT method compared to the case when the weights are estimated by the TDM, and the interference-canceling capability of the DFT method decreases more than the TDM when the interference approaches the look direction. Some of these issues are reexamined in this section with the view to show that by appropriate choice of filter length and number of elements in the array, it is possible to achieve a higher output SNR with less processing time using the DFT method than the TDM. The DFT method is also robust against look direction errors.

4.7.5.1 Higher SNR with Less Processing Time

It is possible to obtain better SNR using the DFT method by increasing the number of elements or filter length such that the required processing time remains less than when using the TDM. Two examples are presented to demonstrate this fact.

In the first example, an array with 20 elements uses the DFT method and an array with 10 elements uses the TDM to estimate the weights. Performance of the two methods is compared in Figure 4.22, where results are displayed for both small and large bandwidth cases. The figure shows that the DFT method yields better performance than the TDM. For this case, computational savings of 14% are possible without using any parallel processing, and when using combined parallel processing, 99%.

The second example uses a five-element array and a filter of 17 taps for the TDM and 177 taps for the DFT method. Results for both bandwidth sources displayed in Figure 4.23 indicate that the DFT method performance is almost equal to that of the TDM. The DFT method for this case requires about 21% less computation time than the TDM. It should be noted that the computational savings have been achieved by using parallel processing, which increases hardware cost. Reduction in hardware cost could be achieved by using the bin elimination method, which reduces 89 parallel stages to 11 stages for the small bandwidth case and to 65 parallel stages for the large bandwidth case.

© 2004 by CRC Press LLC
4.7.5.2 Robustness of DFT Method

Processor performance is compared when weights are estimated using the two methods in the presence of the look direction error (LDE). It is assumed that the actual signal direction is different from the look direction. The weights in both cases are constrained in the look direction.

Figure 4.24 shows the output SNR of the processor using the two methods as a function of look direction error. The error is measured relative to the look direction and is assumed positive in the counterclockwise direction. Thus, errors $1^\circ$ and $-1^\circ$ mean that the signal direction is, respectively, $91^\circ$ and $89^\circ$ relative to the line of the array.
Figure 4.24 shows that the DFT method is robust against the look direction error in the presence of a single interference. The SNR of the processor using the DFT is about 10 dB more than the one using the TDM in the presence of look direction error of less than 0.5°. Although computer simulation shows the robustness of the DFT method against pointing error, a theoretical explanation does not seem to exist.

4.8 Performance

Performance of broadband arrays as a function of the number of various parameters such as the number of taps, tap spacing, array geometry, array aperture, and signal bandwidth has been considered in the literature [May81, Voo92, Com88, Ko81, Ko87, Nun83, Yeh87, Sco83] to understand their influence on the behavior of arrays. An analysis [May81] of broadband array using eigenvalues of the array correlation matrix indicates that the product of the array aperture and fractional bandwidth (FBW) of the signal is an important parameter of the broadband array in determining its performance. The FBW is defined as the ratio of the bandwidth to the center frequency of the signal. The number of taps required on each element depends on this parameter as well as on the shape of the array, with more taps needed for a complex shape. A study [Voo92, Com88] of the SNR as a function of inter-tap spacing indicates that there is a range of spacing that yields close to maximum attainable SNR and depends on the FBW of the signal. This range includes quarter wavelength spacing at the center frequency \( f_0 \). The quarter wavelength spacing produces a 90° phase shift at \( f_0 \) and is equal to \( 1/4f_0 \). By measuring the tap spacing as a multiple of this delay, the inter-tap spacing with the multiple around 1/FBW yields close to the highest attainable SNR. With the multiple between 1/FBW to 4/FBW, a larger number of taps for an equivalent performance is necessary.

A study of the jamming rejection capability [Ko81] and tracking performance of the array in nonstationary environment [Ko87] also indicates that when tap spacing is measured in terms of the signal’s center frequency, the best performance is achieved when the spacing is \( 1/4f_0 \). For this tap spacing, the array correlation matrix has less eigenvalue spread, which is the reason for this performance. The eigenvalue spread of a matrix indicates the range of values that its eigenvalues take. A bigger ratio of the largest eigenvalue to the smallest eigenvalue indicates a larger spread.

The TDL filter tends to increase the degrees of freedom of the array that may be traded against the number of elements such that an array with \( L \) elements is able to suppress more than \( L - 1 \) directional interferences provided that their center frequencies are not the same and fall within the FBW of the signal [Yeh87].

Acknowledgments

Notation and Abbreviations

\((\phi, \theta)\) direction in three-dimensional coordinate system, Figure 2.2

\((\phi)\) direction with respect to line array

\((\phi_0, \theta_0)\) look direction

\(\Delta f\) bandwidth of frequency bin

\([f_L, f_H]\) frequency band of interest

\(a(f,k)\) scalar defined in (4.6.33)

DFT discrete Fourier transform

FDM frequency domain method

FIR finite impulse response

LDE look direction error

MMSE minimum mean square error

MSE mean square error

TDL tapped delay line

TDM time domain method

\(A(f, \phi, \theta)\) desired frequency response in direction \((\phi, \theta)\)

\(B\) matrix prefilter

\(\tilde{B}\) matrix with \(B\) as diagonal elements

\(B_S\) normalized signal bandwidth

\(C\) \(L \times J\) dimensional constraint matrix

\(C_k\) constraint matrix

\(D\) constraint matrix

\(\text{diag}[x]\) matrix with \(x\) as diagonal elements

\(E(t)\) column of matrix prefilter outputs across TDL structure

\(e(t)\) column of \(L - 1\) outputs of matrix prefilter

\(e(k)\) column of \(N\) phasers at \(k\)th bin defined in (4.6.25)

\(e(f)\) column of \(N\) phasers at frequency \(f\) defined in (4.6.30)

\(F\) optimal weights with point constraints, only white noise present

\(f\) \(J\)-dimensional constraint vector

\(f_k\) \(k\)th component of \(f\)

\(i_k\) \(k\)th coefficient of inverse DFT of \(Nf_m\) \(m = 1, 2, \ldots, N\)

\(f_s\) sampling frequency

\(G\) optimal weights with directional constraints, only white noise present

\(g\) constraint vector

\(H, H(f, \phi, \theta)\) frequency response of TDL processor in direction \((\phi, \theta)\)

\(h(k)\) \(L\) weights of narrowband processor for \(k\)th bin

\(h_l(k)\) weight on \(l\)th channel for \(k\)th bin

\(J\) number of taps in tapped delay line filter

\(\tilde{J}\) number of bins that need processing in bin elimination method
\( \hat{J} \) number of bins that need processing due to DFT properties
\( J(W) \) cost function
\( J(V, \lambda) \) cost function

\( L \) number of elements
\( M \) number of directional sources
\( M_0 \) number of data sets of \( N \) samples
\( N \) Number of samples processed by frequency domain method
\( O_T \) Number of floating-point operations using TDM
\( O_D \) Number of floating-point operations using DFT method

\( P \) projection operator
\( P(W) \) mean out power of a processor for given \( W \)
\( P_S(W) \) mean output signal power for given \( W \)
\( P_N(W) \) mean output noise power for given weight
\( P(k) \) mean out power of narrowband processor for \( k \)th bin
\( \hat{P} \) mean output power of TDL processor using optimal weights

\( P \) \( J \times J \)-dimensional column vector defined by (4.1.68)
\( Q \) \( J \times J \) matrix defined by (4.1.67)

\( R \) array correlation matrix
\( R_S \) array correlation matrix due to signal source
\( R_N \) array correlation matrix due to noise
\( R_i \) array correlation matrix due to \( i \)th source in direction \( (\phi_i, \theta_i) \)

\( (R_{mn}) \) matrix denoting correlation after \((m - 1)\) and \((n - 1)\) delays

\( (\hat{R}_{ij}) \) matrix denoting correlation between \( i \)th and \( j \)th elements
\( R_S(k) \) array correlation matrix in frequency domain for \( k \)th bin

\( \hat{R}_i(k) \) array correlation matrix using energy from \( k \)th bin only

\( R_{XE} \) matrix of correlation between \( X(t) \) and \( E(t) \)
\( R_{EE} \) matrix of correlation between \( E(t) \) and \( E(t) \)

\( r_d \) correlation between desired signal and array signal vector
\( S(f) \) power spectral density of \( s(t) \)
\( S(f, \phi, \theta) \) steering vector at frequency \( f \) in direction \( (\phi, \theta) \)

\( \hat{S}(f, \phi, \theta) \) steering vector in \( (\phi, \theta) \) direction for array presteered in \( (\phi_0, \theta_0) \)

\( s(t) \) signal induced on reference element

\( \text{SNR} \) signal-to-noise ratio
\( \text{SNR}(W) \) \( \text{SNR} \) for given \( W \)
\( \text{SNR}_T \) \( \text{SNR} \) using TDM
\( \text{SNR}_D \) \( \text{SNR} \) using DFT method

\( T \) inter-tap spacing, sampling interval
\( T(f) \) diagonal matrix of steering delays

\( T_i(\phi, \theta) \) steering delay on \( i \)th element

\( T_0 \) bulk delay to make \( T_i(\phi, \theta) \) a positive quantity

\( U \) \( J \times J \) matrix of the eigenvector of \( Q \)
\( U_{\eta_0} \) matrix of eigenvectors associated with \( \eta_0 \) nonzero eigenvalues of \( Q \)

\( U_i \) eigenvector associated with \( i \)th eigenvalue of \( Q \)

\( V \) error vector, column of \( (L - 1)J \) weights of TDL structure

\( \hat{V} \) \( (L - 1)J \) dimensional optimal weights of TDL structure

\( v_k \) column of \( L - 1 \) weights on the \( k \)th tap of TDL structure

\( W \) column of \( LJ \) weights of TDL structure

\( W_F \) column of \( LJ \) fixed weight

\( \hat{W} \) optimal weights of TDL processor

\( \hat{W}_0 \) optimal weights of constrained partitioned processor

\( \hat{W} \) weight vector which minimizes \( \varepsilon_0 \)

\( W(n) \) weights estimated at the \( n \)th iteration

\( w_m \) column of \( L \) weights on the \( m \)th tap of TDL structure

\( w_{lk} \) weight on the \( k \)th tap of the \( l \)th channel

\( X(t) \) column of array signals across the TDL structure

\( X(n) \) array signals at \( n \)th instant of time

\( x(t) \) column of array signals after presteering delays

\( \tilde{x}(k) \) column of frequency domain array signals for \( k \)th bin

\( \tilde{x}(k,m) \) array signals for \( k \)th bin from \( m \)th data set

\( x_i(t) \) output of \( i \)th sensor presteered in \( (\phi_0, \theta_0) \)

\( x_{ii} \) output of \( i \)th sensor before \( i \)th tap

\( x_{ii}(m) \) output of \( i \)th sensor before \( i \)th tap from \( m \)th data set

\( x_i(m) \) \( N \) outputs of \( i \)th sensor across TDL filter from \( m \)th data set

\( x_i \) components of the \( i \)th element along \( x \)-axis

\( \tilde{x}_i(k) \) output of \( i \)th sensor for \( k \)th bin

\( \tilde{x}_i(k,m) \) output of \( i \)th sensor from \( m \)th data set for \( k \)th bin

\( y(t) \) output of processor

\( y_i \) components of \( i \)th element along \( y \)-axis

\( y(n) \) output at \( n \)th instant of time

\( \tilde{y}(k) \) output of processor at \( k \)th bin

\( \tilde{y}(k,m) \) output of processor at \( k \)th bin from \( m \)th data set

\( y_A(t) \) output of auxiliary beams

\( y_f(t) \) output of fixed beam

\( z_i \) components of \( i \)th element along \( z \)-axis

\( \Lambda \) diagonal matrix with elements being eigenvalues of \( Q \)

\( \Lambda_{\phi}(\phi, \theta) \) diagonal matrix defined by (4.3.5)

\( \Lambda_{\phi}(\phi, \theta) \) diagonal matrix defined by (4.3.17)

\( \Lambda_{\phi}(\phi) \) diagonal matrix defined by (4.3.51)

\( \Delta \) sampling interval

\( \eta_0 \) rank of \( Q \)

\( \delta_0 \) threshold value

\( \sigma_0 \) normalizing constant
\( \sigma(\phi) \) vector defined by (4.3.55)

\( \psi(\phi) \) vector defined by (4.3.56)

\( \varepsilon_0 \) MSE between \( A(f,\phi_0,\theta_0) \) and \( H(f,\phi_0,\theta_0) \)

\( \lambda \) Lagrange multiplier

\( \lambda_i(Q) \) \( i \)th eigenvalue of \( Q \)

\( \lambda(n) \) Lagrange multipliers at \( n \)th iteration

\( \lambda(\phi) \) vector defined by (4.3.54)

\( \rho(\tau) \) correlation function of \( s(t) \)

\( \rho, \rho(f,\phi,\theta) \) power response of TDL processor in direction \( (\phi,\theta) \)

\( \rho_0 \) correlation between desired signal and array output

\( \tau_l(\phi,\theta) \) delay faced by signal from source in \( (\phi,\theta) \) on \( l \)th element

\( \tau_l(\phi) \) delay faced by signal from source in \( (\phi) \) on \( l \)th element

\( \tau \) delay parameter

References


Bry69 Bryson, Jr., A.E. and Ho, Y.C., \( \text{Applied Optimal Control} \), Blaisdell, Waltham, MA, 1969.


Chu90 Chujo, W. and Yasukawa, K., Design study of digital beam forming antenna applicable to mobile satellite communications, \( IEE \) Antennas and Propagation Symposium Digest, 400–403, 1990.


Com88 Compton, Jr., R.T., The bandwidth performance of a two element adaptive array with

DeM77 DeMuth, G.J., Frequency domain beamforming techniques, in *Proceedings of IEEE Interna-

Dav67 Davies, D.E.N., Independent angular steering of each zero of the directional pattern for a


Des92 Despins, C.L.B., Falconer, D.D. and Mahmoud, S.A., Compound strategies of coding, equal-
ization and space diversity for wideband TDMA indoor wireless channels, *IEEE Trans. Vehicu-


Er83 Er, M.H. and Cantoni, A., Derivative constraints for broadband element space antenna

Er85 Er, M.H. and Cantoni, A., A new approach to the design of broadband element space

Er86 Er, M.H. and Cantoni, A., A new set of linear constraints for broadband time domain

Er86a Er, M.H. and Cantoni, A., An unconstrained portioned realization for derivative constrained
1986.

Er90 Er, M.H. and Ng, B.P., On derivative constrained broadband beamforming, *IEEE Trans.

Er90a Er, M.H., An alternative implementation of quadratically constrained broadband beam-

Er93 Er, M.H., On the limiting solution of quadratically constrained broadband beamformers,

Fan84 Fan, H., El-Masry, E.I. and Jenkins, W.K., Resolution enhancement of digital beamforming,

Flo88 Florian, S. and Bershad, N.J., A weighted normalized frequency domain LMS adaptive


Gab84 Gabel, R.A. and Kurth, R.R., Hybrid time-delay/phase-shift digital beamforming for uni-

Geb95 Gebauer, T. and Gockler, H.G., Channel-individual adaptive beamforming for mobile sat-


God97 Godara, L.C., Application to antenna arrays to mobile communications. Part II: Beamform-

God99 Godara, L.C. and Jahromi, M.R.S., Limitations and capabilities of frequency domain con-

Gri82 Griffiths, L.J. and Jim, C.W., An alternative approach to linearly constrained adaptive

Gri87 Griffiths, L.J. and Buckley, K.M., Quiescent pattern control in linearly constrained adaptive


Hod79 Hodgkiss, W.S., Adaptive array processing: time vs. frequency domain, *International Con-

Hua90 Huang, K.C., Chang, S.H. and Chen, Y.H., An alternative structure for adaptive broadband

Iig85 Iiguni, Y., Sakai, H. and Tokumaru, H., Convergence properties of simplified gradient


