

ENEE634 Space-Time Signal Processing: Project 2

Adaptive MVDR Beamforming and the QRD-RLS Algorithm

Steven Tjoa

Dept. of Electrical and Computer Engineering, University of Maryland

April 4, 2005

Adaptive beamformers distinguish the properties of signals and noise through spatial filtering, where an array of independent sensors provide a means of sampling the received signal in space. The sensor outputs are then processed by a transversal filter to produce the beamformer output. The primary purpose of the adaptive beamformer is to protect a target signal while cancelling an interference signal.

In the following experiment, we will use the QRD-RLS algorithm to create an adaptive beamformer that processes a target signal without magnitude distortion, and we will show how the QRD-RLS algorithm can be implemented using a systolic array. Then we will evaluate the performance of the beamformer in the presence of correlated and uncorrelated interferences.

1 MVDR Beamforming

Before we examine the algorithmic aspects of the MVDR beamformer, we first present a general scenario where array processing is used. Consider a linear transversal filter $w(n)$ convolved with an input signal $u(n)$ to form the output signal $y(n)$:

$$y(n) = \sum_k w^*(k)u(n-k) \quad (1)$$

Now suppose that the input signal $u(n)$ is spatial signal that represents the inputs to a linear array of M sensors, as illustrated in Figure 1, where $M = 5$. The array receives a plane wave along a direction specified by the angle ϕ_0 , also known as the look direction. We assume that the distance between adjacent sensors in the array is $d = \lambda/2$, where λ is the wavelength of the received signal. As a result, we find that for any instant in time, the only difference among the sensor inputs is a phase difference θ_0 from the reference input $u_0(n)$:

$$y(n) = u_0(n) \sum_{k=0}^{M-1} w_k^* e^{-jk\theta_0} \quad (2)$$

The phase difference θ is related to the angle of incidence ϕ as follows:

$$\theta = \frac{2\pi d}{\lambda} \sin \phi = \pi \sin \phi \quad (3)$$

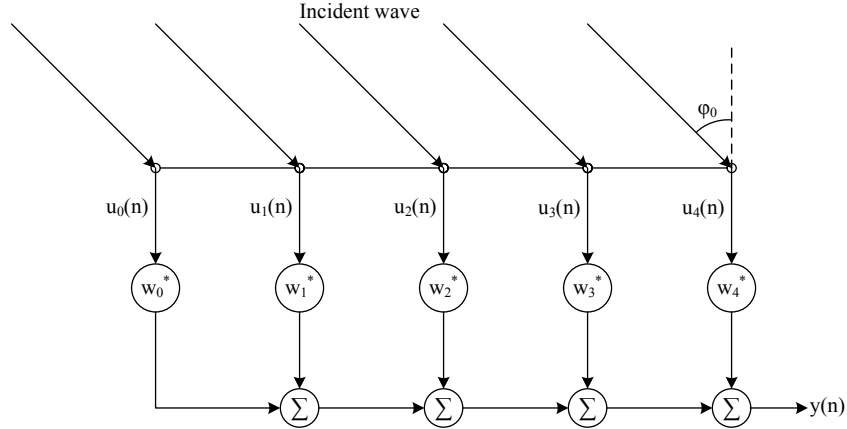


Figure 1: Configuration of a linear sensor array for beamforming.

Now suppose we wish to “protect” all signals that are received from the look direction. We can state the following linear constraint:

$$\sum_{k=0}^{M-1} w_k^* e^{-jk\theta_0} = \mathbf{w}^H(n) \mathbf{s}(\theta_0) = g \quad (4)$$

where $\mathbf{w}(n)$ is the tap weight vector of the transversal filter, and $\mathbf{s}(\theta)$ is the steering vector:

$$\mathbf{s}(\theta) = [1 \ e^{-j\theta} \ e^{-j2\theta} \ \dots \ e^{-j(M-1)\theta}]^T \quad (5)$$

Note how the quantity $\mathbf{w}^H \mathbf{s}(\theta)$ above is the same as the Fourier transform of $w(k)$. Therefore, we can interpret the constraint by saying that we require the gain at the look direction to be held constant. Furthermore, we wish to minimize the mean-square value of the filter output, subject to the above linear constraint. A spatial filter that achieves this task is called a linearly-constrained minimum variance (LCMV) beamformer. If we simply choose the constraint $g = 1$, where signals received at the look direction are processed with unity gain, then the response at the look direction is distortionless. This special case of the LCMV beamformer is known as a minimum-variance distortionless response (MVDR) beamformer.

2 QRD-RLS Algorithm

Figure 1 illustrates a scenario where one incident wave arrives at the sensor array. However, the sensor array will usually receive more than one signal, each coming from a potentially different direction. For example, a target signal will arrive at an angle of incidence of ϕ_0 , and an interference signal will arrive at an angle of incidence ϕ_1 . Of course, the direction of interference ϕ_1 is most likely unknown and possibly varying. Therefore, we must perform adaptive processing on the data received by the sensor array to cancel the interference. One recursive algorithm that is well-suited for adaptive beamforming is the QR decomposition-based recursive least-squares (QRD-RLS) algorithm.

Suppose we have an input signal $u_k(i)$ from sensor k at time i , a desired response $d(i)$, and the error $e(i)$. We can define the following vectors:

$$\mathbf{u}(i) = \begin{bmatrix} u_1(i) \\ u_2(i) \\ \vdots \\ u_M(i) \end{bmatrix}, \quad \mathbf{d}(n) = \begin{bmatrix} d(1) \\ d(2) \\ \vdots \\ d(n) \end{bmatrix}, \quad \boldsymbol{\epsilon}(n) = \begin{bmatrix} e(1) \\ e(2) \\ \vdots \\ e(n) \end{bmatrix} \quad (6)$$

where $\mathbf{u}(i)$ is an input vector in the spatial domain (i.e. a snapshot of the array at time i), $\mathbf{d}(n)$ is the desired response vector, and $\boldsymbol{\epsilon}(n)$ is the error vector consisting of the a posteriori errors $e(i) = d(i) - \mathbf{w}^H(n)\mathbf{u}(i)$. Let us also define an input data matrix $\mathbf{A}(n)$ in terms of a series of snapshots:

$$\mathbf{A}^H(n) = \begin{bmatrix} u_1(1) & u_1(2) & \cdots & u_1(n) \\ u_2(1) & u_2(2) & \cdots & u_2(n) \\ \vdots & \vdots & \ddots & \vdots \\ u_M(1) & u_M(2) & \cdots & u_M(n) \end{bmatrix} = [\mathbf{u}(1) \ \mathbf{u}(2) \ \cdots \ \mathbf{u}(n)] \quad (7)$$

Then we can define the error vector as the following:

$$\boldsymbol{\epsilon}(n) = \mathbf{d}(n) - \mathbf{A}(n)\mathbf{w}(n) \quad (8)$$

We can now define the cost function:

$$\mathcal{E}(n) = \sum_{i=1}^n \lambda^{n-i} |e(i)|^2 = \boldsymbol{\epsilon}^H(n) \boldsymbol{\Lambda}(n) \boldsymbol{\epsilon}(n) \quad (9)$$

where λ is a forgetting factor slightly less than 1, and $\boldsymbol{\Lambda}(n) = \text{diag}(\lambda^{n-1}, \lambda^{n-2}, \dots, 1)$. If we introduce an arbitrary unitary matrix $\mathbf{Q}(n)$, then we can express the cost function as a vector norm:

$$\mathcal{E}(n) = \|\mathbf{Q}(n)\boldsymbol{\Lambda}^{\frac{1}{2}}(n)\boldsymbol{\epsilon}(n)\|^2 \quad (10)$$

Our goal is to minimize the cost function $\mathcal{E}(n)$, which is equivalent to minimizing the quantity $\|\mathbf{Q}(n)\boldsymbol{\Lambda}^{\frac{1}{2}}(n)\boldsymbol{\epsilon}(n)\|$. Note that we can expand the error vector:

$$\mathbf{Q}(n)\boldsymbol{\Lambda}^{\frac{1}{2}}(n)\boldsymbol{\epsilon}(n) = \mathbf{Q}(n)\boldsymbol{\Lambda}^{\frac{1}{2}}(n)\mathbf{d}(n) - \mathbf{Q}(n)\boldsymbol{\Lambda}^{\frac{1}{2}}(n)\mathbf{A}(n)\mathbf{w}(n) \quad (11)$$

We now incorporate a concept from linear algebra known as QR decomposition (QRD): if \mathbf{A} is a $n \times M$ matrix of rank M for $n > M$, then \mathbf{A} can be expressed as $\mathbf{A} = \mathbf{QR}$, where \mathbf{Q} is a unitary matrix and \mathbf{R} is upper triangular. Then for some unitary $\mathbf{Q}(n)$,

$$\mathbf{Q}(n)\boldsymbol{\Lambda}^{\frac{1}{2}}(n)\mathbf{A}(n) = \begin{bmatrix} \mathbf{R}(n) \\ \mathbf{O} \end{bmatrix} \quad (12)$$

where $\mathbf{R}(n)$ is upper triangular of dimension $M \times M$. Next, define the following partition:

$$\mathbf{Q}(n)\boldsymbol{\Lambda}^{\frac{1}{2}}(n)\mathbf{d}(n) = \begin{bmatrix} \mathbf{p}(n) \\ \mathbf{v}(n) \end{bmatrix} \quad (13)$$

where $\mathbf{p}(n)$ is a vector of dimension $M \times 1$, and $\mathbf{v}(n)$ is a vector of dimension $N - M \times 1$. Finally, we obtain the following result:

$$\mathbf{Q}(n)\mathbf{\Lambda}^{\frac{1}{2}}(n)\boldsymbol{\epsilon}(n) = \begin{bmatrix} \mathbf{p}(n) \\ \mathbf{v}(n) \end{bmatrix} - \begin{bmatrix} \mathbf{R}(n) \\ \mathbf{O} \end{bmatrix} \mathbf{w}(n) = \begin{bmatrix} \mathbf{p}(n) - \mathbf{R}(n)\mathbf{w}(n) \\ \mathbf{v}(n) \end{bmatrix} \quad (14)$$

We conclude that the cost $\mathcal{E}(n) = \|\mathbf{Q}(n)\mathbf{\Lambda}^{\frac{1}{2}}(n)\boldsymbol{\epsilon}(n)\|^2$ is minimized iff. $\mathbf{R}(n)\mathbf{w}(n) = \mathbf{p}(n)$. Although this equation closely resembles the normal equations from the least-squares problem, QR decomposition allows a more stable and computationally efficient solution for $\mathbf{w}(n)$.

Now we derive the adaptive QRD-RLS algorithm by deriving the necessary recursion equations. Note the following recursions based on the definitions of $\mathbf{d}(n)$, $\mathbf{A}(n)$, and $\mathbf{\Lambda}(n)$:

$$\mathbf{d}(n) = \begin{bmatrix} \mathbf{d}(n-1) \\ d^*(n) \end{bmatrix}, \quad \mathbf{A}(n) = \begin{bmatrix} \mathbf{A}(n-1) \\ \mathbf{u}^H(n) \end{bmatrix}, \quad \mathbf{\Lambda}(n) = \begin{bmatrix} \lambda\mathbf{\Lambda}(n-1) & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \quad (15)$$

Define the matrix $\bar{\mathbf{Q}}(n)$ as a padded version of $\mathbf{Q}(n)$:

$$\bar{\mathbf{Q}}(n) = \begin{bmatrix} \mathbf{Q}(n) & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \quad (16)$$

Then we obtain the following partially triangularized matrix:

$$\bar{\mathbf{Q}}(n-1)\mathbf{\Lambda}^{\frac{1}{2}}(n)\mathbf{A}(n) = \begin{bmatrix} \lambda^{\frac{1}{2}}\mathbf{Q}(n-1)\mathbf{\Lambda}^{\frac{1}{2}}(n-1)\mathbf{A}(n-1) \\ \mathbf{u}^H(n) \end{bmatrix} = \begin{bmatrix} \lambda^{\frac{1}{2}}\mathbf{R}(n-1) \\ \mathbf{O} \\ \mathbf{u}^H(n) \end{bmatrix} \quad (17)$$

Define the $n \times n$ matrix $\mathbf{T}(n)$ as a unitary product of M Givens rotation matrices that are applied to annihilate all M elements in the last row:

$$\begin{bmatrix} \lambda^{\frac{1}{2}}\mathbf{R}(n) \\ \mathbf{O} \end{bmatrix} = \mathbf{T}(n) \begin{bmatrix} \lambda^{\frac{1}{2}}\mathbf{R}(n-1) \\ \mathbf{O} \\ \mathbf{u}^H(n) \end{bmatrix} \quad (18)$$

This equation is the update recursion for the upper triangular matrix $\mathbf{R}(n)$. Finally, we compute the update recursion for $\mathbf{p}(n)$:

$$\begin{bmatrix} \mathbf{p}(n) \\ \mathbf{v}(n) \end{bmatrix} = \mathbf{T}(n) \begin{bmatrix} \lambda^{\frac{1}{2}}\mathbf{p}(n-1) \\ \lambda^{\frac{1}{2}}\mathbf{v}(n-1) \\ d^*(n) \end{bmatrix} \quad (19)$$

These two update recursions form the basis of the QRD-RLS algorithm as shown in Table 1. Because the matrix $\mathbf{R}(n)$ is upper triangular, $\mathbf{w}(n)$ can be solved easily using back substitution.

A primary advantage of the QRD-RLS algorithm is its ability to be implemented as a systolic array, as shown in Figure 2. The inputs are fed into each cell in skewed order from top to bottom, and the rotations corresponding to each Givens matrix is fed from left to right. In other words, the systolic array rotates each row in such a way that the leading element of the received data vector is annihilated. The rotated outputs from the first row are then fed to the second row, which undergoes another rotation to annihilate the leading element of the data vector received by the second row, and so on. The error $e(n)$ is formed as a product between the output from the two data cells in the last row.

-
1. Initialization: $\mathbf{R}(0) = \mathbf{O}$, $\mathbf{p}(0) = \mathbf{0}$
 2. Updating $\mathbf{R}(n)$: $\begin{bmatrix} \lambda^{\frac{1}{2}}\mathbf{R}(n) \\ \mathbf{O} \end{bmatrix} = \mathbf{T}(n) \begin{bmatrix} \lambda^{\frac{1}{2}}\mathbf{R}(n-1) \\ \mathbf{O} \\ \mathbf{u}^H(n) \end{bmatrix}$
 3. Updating $\mathbf{p}(n)$: $\begin{bmatrix} \mathbf{p}(n) \\ \mathbf{v}(n) \end{bmatrix} = \mathbf{T}(n) \begin{bmatrix} \lambda^{\frac{1}{2}}\mathbf{p}(n-1) \\ \lambda^{\frac{1}{2}}\mathbf{v}(n-1) \\ d^*(n) \end{bmatrix}$
 4. Tap weight computation: $\mathbf{w}(n) = \mathbf{R}^{-1}(n)\mathbf{p}(n)$
-

Table 1: The standard QRD-RLS algorithm.

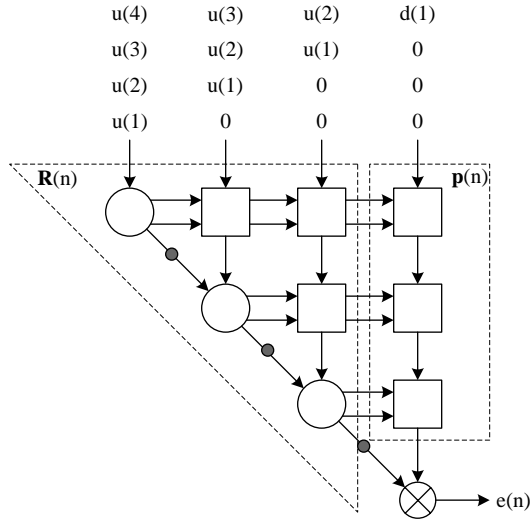


Figure 2: A systolic array structure of the QRD-RLS algorithm.

3 Experimental Results

3.1 Problem Formulation

Now that we have established the theory and implementation behind the QRD-RLS algorithm, we can apply the algorithm to adaptively obtain a MVDR beamformer. Suppose an array of $M = 5$ sensors receives an input signal that consists of a target signal, an interference signal, and noise:

$$u_k(n) = A_0 e^{jn\theta_0} + A_1 e^{jn\theta_1 + j\psi(n)} + v(n), \quad k = 0, 1, 2, 3, 4 \quad (20)$$

where A_0 is the gain of the target signal, A_1 is the gain of the interference signal, θ_0 is the phase difference in target signals received by adjacent sensors, θ_1 is the phase difference in interference signals received by adjacent sensors, $\psi(n)$ is a random phase uniformly distributed over $[0, 2\pi)$, and $v(n)$ is complex white Gaussian noise with zero mean and unit variance.

In various pieces of literature, there are many ways to set up the beamforming problem in terms of the desired response and the input signal vector. For example, a formulation by Haykin [1] sets $d(n)$ to zero, thus minimizing the mean-square value of the filter output. Here is one simpler way to develop this scenario in Matlab:

```

% initialize filter parameters
taps = 5;
lambda = 1.0;
trials = 1;
iterations = 20;
N = iterations+taps-1;
n = (0:N-1)';

% initialize signal parameters
SNR = 10;
INR = 40;
A0 = sqrt(10^(SNR/10));
A1 = sqrt(10^(INR/10));
theta0 = -0.15*pi;
theta1 = 0.00*pi;

% synthesize signals
psi = 2*pi*rand(N,1);
target = A0*exp(j*n*theta0);
noise = randn(N,1) + j*randn(N,1);

% create desired response vector and input matrix
d = target + noise;
for i=1:taps
    target = A0*exp(j*(n-i)*theta0);
    interf = A1*exp(j*(n-i)*theta1 + j*psi);
    noise = randn(N,1) + j*randn(N,1);
    A(:,i) = target + interf + noise;
end

% use a QRD-RLS adaptive filter
[w, mse] = qdrdls(A,d,lambda,taps,iterations);

The function qdrdls uses an implementation of a systolic array. The main loop of this function
follows:

for n=1:N
    if (n <= M)
        U(1,:) = [A(n,:) 0];
    else
        U(1,:) = [A(n,:) d(n)];
    end

    err(n) = delay(M,1)*U(M+1,M+1);
    for i=1:M
        for j=i:M+1
            if (i == j)

```

```

    gamma(i+1) = delay(i);
    if (U(i,j) == 0)
        C(i,j+1) = 1;
        S(i,j+1) = 0;
        R(i,j) = sqrt(lambda)*R(i,j);
        delay(i) = gamma(i);
    else
        rtemp = sqrt(lambda*R(i,j)^2 + abs(U(i,j))^2);
        C(i,j+1) = sqrt(lambda)*R(i,j)/rtemp;
        S(i,j+1) = U(i,j)/rtemp;
        R(i,j) = rtemp;
        delay(i) = C(i,j+1)*gamma(i);
    end
elseif (i < j)
    U(i+1,j) = C(i,j)*U(i,j) - S(i,j)*sqrt(lambda)*R(i,j);
    R(i,j) = C(i,j)*sqrt(lambda)*R(i,j) + S(i,j)*U(i,j);
    C(i,j+1) = C(i,j);
    S(i,j+1) = S(i,j);
end
end
end
end
end

```

3.2 Target and Interference Directions-of-Arrival

First, we investigate Problem 11.11 in Haykin [1], which asks to plot the spatial response of the beamformer as the look direction approaches the interference direction. Figure 3 presents plots for the amplitude responses of various beamformers. We find that the beamformer performs very well for a look direction of $\phi_0 = \sin^{-1}(-0.15)$ and $\phi_0 = \sin^{-1}(-0.10)$. However, for $\phi_0 = \sin^{-1}(-0.05)$, the direction of the interference is so close to the look direction that there is a small amount of attenuation at the look direction; for $n = 20$ there is approximately a 7 dB attenuation, and for $n = 100$ there is approximately a 5 dB attenuation.

The attenuation of the MVDR beamformer at the look direction occurs as a result of inadequate spatial resolution. In particular, the number of sensors M in the array governs the degrees of freedom in the beamformer. From standard FIR filter design techniques [5], the width of the sidelobe decreases as M increases. Therefore, by increasing M we can also narrow the notch that occurs at the interference direction such that nearby directions are not unfairly attenuated.

We also generally find that an increase in the number of iterations increases the attenuation at all directions outside the look direction. Recall that the MVDR beamformer tries to minimize the mean-square value of the filter output subject to the linear “distortionless response” constraint. Naturally, as n increases, the MSE decreases and the tap weight vector $\mathbf{w}(n)$ approaches the optimal tap weight vector \mathbf{w}_o . Subsequently, we expect better attenuation at all directions outside the look direction. From Figure 3, for $n = 100$ we observe an attenuation at the interference direction of at least 10 dB greater than the case where $n = 20$.

3.3 Correlation Between Target and Interference Signals

We now investigate Problem 11.12 in Haykin [1]. In Eq. 20, the random variable $\psi(n)$ changes values over n . However, in the event that the target and the interference are correlated, $\psi(n)$ is held constant:

$$u_k(n) = A_0 e^{jn\theta_0} + A_1 e^{jn\theta_1 + j\psi_1} + v(n), \quad k = 0, 1, 2, 3, 4 \quad (21)$$

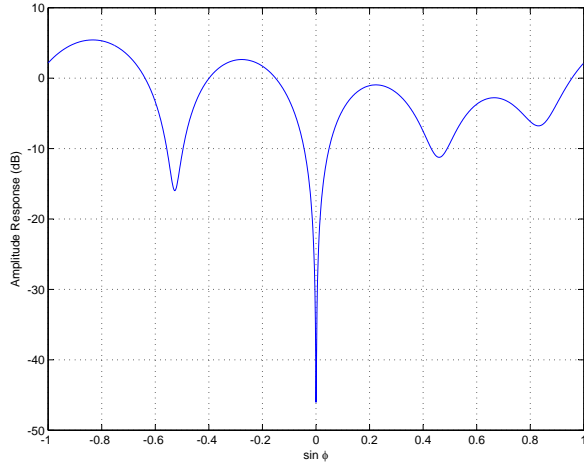
Given this new input signal, the amplitude responses of the MVDR beamformer are illustrated in Figure 4. The effect of correlation between target and interference signals has been studied extensively [3], [6]. Correlated sources can lead to cancellation of the desired signal due to high correlation of reflected signal components with the target signal, resulting in target signal distortion. Some techniques decorrelate the signal prior to beamforming to avoid such target distortion. In Figure 4, it is difficult to find a significant difference with Figure 3, however there is some additional target attenuation that is visible particularly for the case where $\phi_0 = \sin^{-1}(-0.05)$.

4 Conclusion

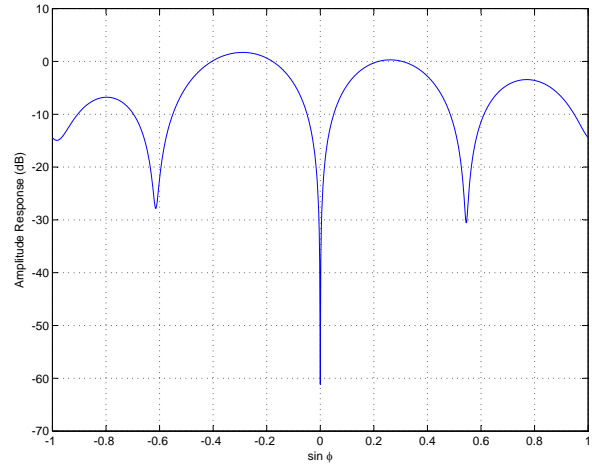
In this report, we analyzed the operation of the QRD-RLS filter in a beamforming application. First, we formulated the beamforming problem and the MVDR constraint, which requires unity gain at the look direction ϕ_0 . Second, we derived the QRD-RLS algorithm, a recursive algorithm that minimizes a sum of squared error through QR decomposition. In creating an upper-triangular matrix $\mathbf{R}(n)$, we can easily attain the tap weight vector $\mathbf{w}(n)$ through back-substitution. We also found that this algorithm can be implemented in a systolic array, which rotates the matrix $\mathbf{R}(n)$ element-by-element. Finally, we examined the effect of spatial and temporal interference correlation on the amplitude response of a MVDR beamformer, and found that correlation between target and interference signals results in an attenuation at the look direction.

References

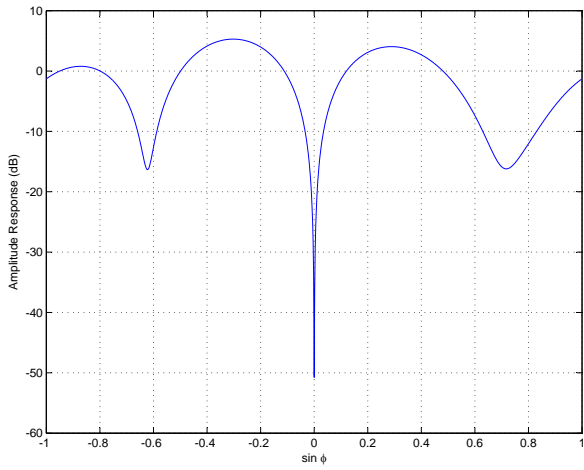
- [1] S. Haykin. *Adaptive Filter Theory*, Upper Saddle River, N.J.: Prentice Hall, 2002.
- [2] M. Hayes. *Statistical Digital Signal Processing and Modeling*, New York, N.Y.: Wiley, 1996.
- [3] W. Herbordt and W. Kellermann. "Adaptive Beamforming for Audio Signal Acquisition."
- [4] J. G. McWhirter and T. J. Shepherd. "Systolic array processor for MVDR beamforming," *Proceedings of the IEE*, part F, vol. 136, pp. 15-80, 1989.
- [5] A.V. Oppenheim and R.W. Schaffer. *Discrete-Time Signal Processing*, Upper Saddle River, N.J.: Prentice Hall, 1999.
- [6] M. D. Zoltowski. "On the Performance Analysis of the MVDR Beamformer in the Presence of Correlated Interference," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, no. 6, pp. 945-947, 1988.



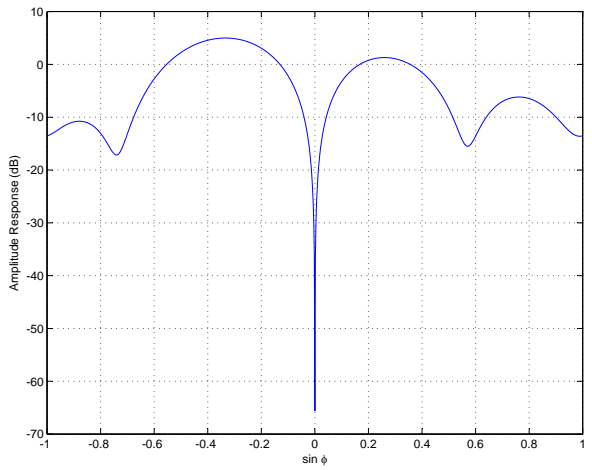
(a) $\phi_0 = \sin^{-1}(-0.15)$, $n = 20$



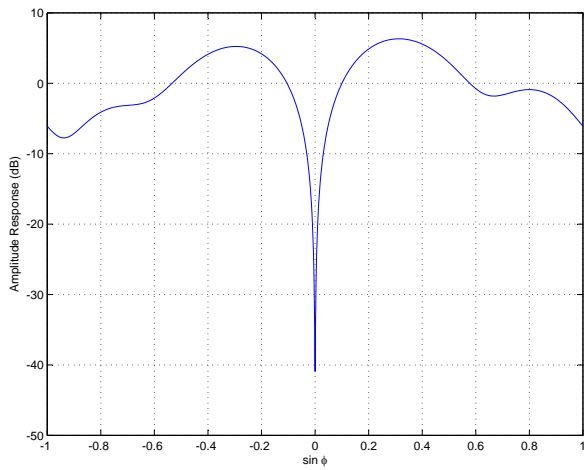
(b) $\phi_0 = \sin^{-1}(-0.15)$, $n = 100$



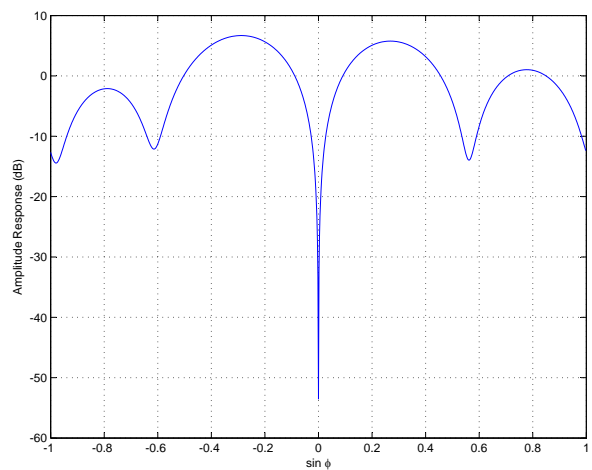
(c) $\phi_0 = \sin^{-1}(-0.10)$, $n = 20$



(d) $\phi_0 = \sin^{-1}(-0.10)$, $n = 100$

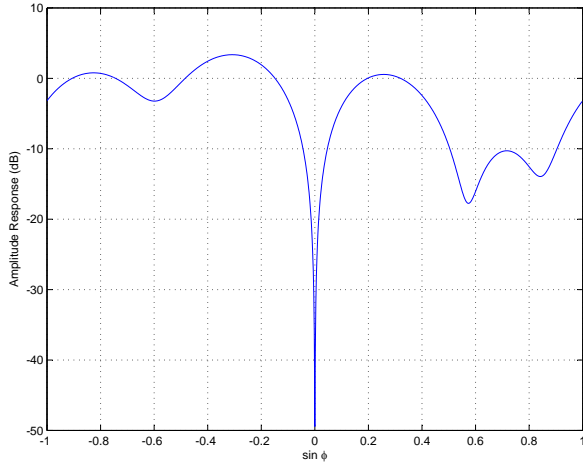


(e) $\phi_0 = \sin^{-1}(-0.05)$, $n = 20$

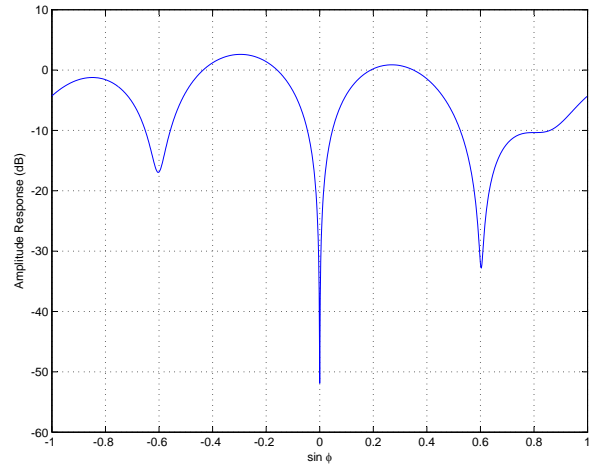


(f) $\phi_0 = \sin^{-1}(-0.05)$, $n = 100$

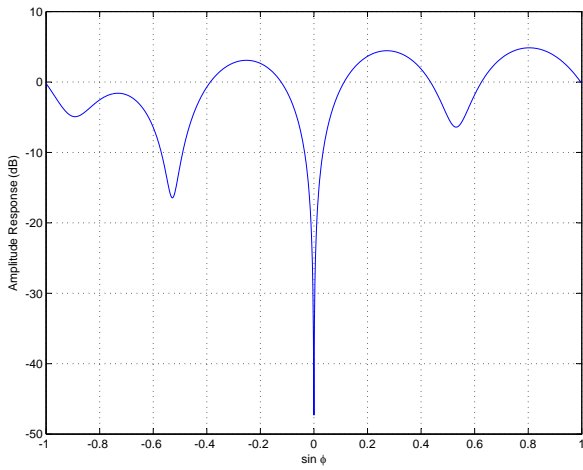
Figure 3: Amplitude responses for the MVDR beamformer with uncorrelated sources, varying look directions ϕ_0 , and iterations n .



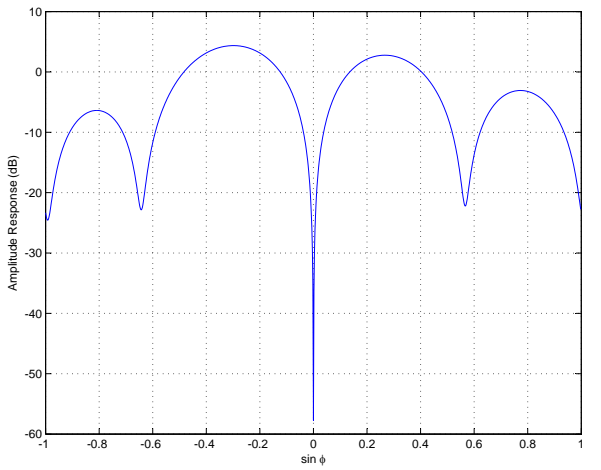
(a) $\phi_0 = \sin^{-1}(-0.15)$, $n = 20$



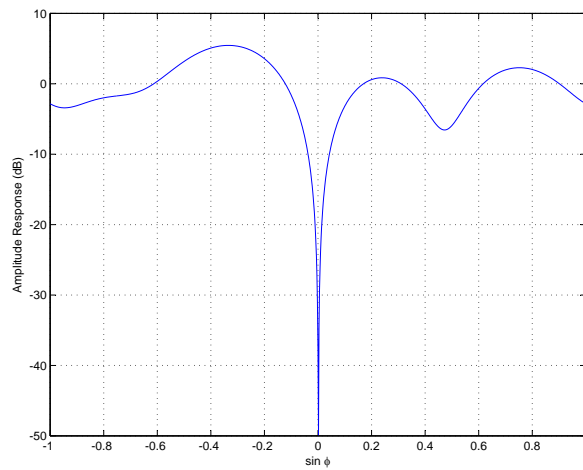
(b) $\phi_0 = \sin^{-1}(-0.15)$, $n = 100$



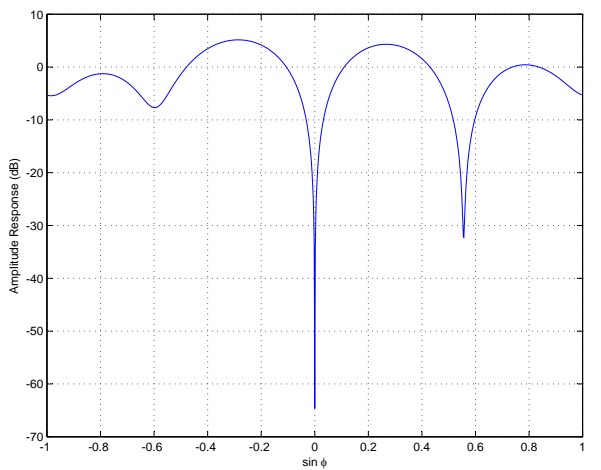
(c) $\phi_0 = \sin^{-1}(-0.10)$, $n = 20$



(d) $\phi_0 = \sin^{-1}(-0.10)$, $n = 100$



(e) $\phi_0 = \sin^{-1}(-0.05)$, $n = 20$



(f) $\phi_0 = \sin^{-1}(-0.05)$, $n = 100$

Figure 4: Amplitude responses for the MVDR beamformer with correlated sources, varying look directions ϕ_0 , and iterations n .