Direction Finding Using Cumulant in Multipath Environment with an Arbitrary Array

Maryam Johnny¹, Vahid Tabataba Vakili², Mansour Sheikhan³

¹Electrical Engineering Department, Islamic Azad University-South Tehran Branch, Tehran, Iran; ²Electrical Engineering Department, Iran University of Science and Technology, Tehran, Iran.

Email: Maryam.Johnny@gmail.com

Received January 3rd, 2012; revised February 8th, 2012; accepted March 13th, 2012

ABSTRACT

In this paper, the problem of estimating the direction of arrival of signals of which some may be perfectly correlated is considered. This method can be applied in the situation that the non-Gaussian independent and coherent signals coexist with unknown Gaussian noise. In this method at first via mappings, the virtual uniform linear array (ULA) and also the shifted versions of this virtual ULA by assuming that all the DOAs are located in one section are constructed. In order to avoid coloring the noise because of these mappings we use a cumulant matrix instead of a covariance ones. In this method since we construct all the subarrays virtually for detection of coherent signals we do not need the array with regular configuration. The advantages of this method are: increasing the array aperture, having the ability to find the DOAs with fewer sensors and also avoiding the coupling between sensors as much as possible in contrast to conventional spatial smoothing.

Keywords: Array Interpolation; Direction of Arrival; MUSIC; Cumulant; Uniform Linear Array

1. Introduction

Direction finding techniques based on the eigendecomposition of the covariance matrix of the vector of the signals received by an array of sensors, have received considerable attention in years. The main drawback of these techniques like MUSIC [1] is their inability to handle perfectly correlated (or highly correlated) signals which arise quite often in practice due to multipath propagation. This fact has motivated various researchers to look for variations of the MUSIC algorithm which do not suffer from this problem. Under the certain conditions, spatial smoothing technique introduced in [2,3] makes it possible to use a MUSIC type algorithm in the presence of arbitrary signal correlation. The most restrictive aspect of the spatial smoothing technique as presented in [2] and discussed further in [3,4], is that it requires a linear uniformly spaced array and also has a significant loss of array aperture. Other techniques, such as the one presented in [5], appear to have similar restrictions. Also these methods all assume that the additive noise is a white Gaussian process or the noise covariance matrix is known in advanced. On the contrary, in many practical situations, the additive noise is colored Gaussian process and a priori estimate of the noise covariance matrix is not available therefore these methods will suffer severe performance degradation. Owing to the attractive property of Gaussian processes that all cumulant spectra of order greater than two are identical to zero, many effective algorithms based on cumulants [6,7] have been proposed. An attempt to generalize the spatial smoothing technique based on cumulant to arbitrary array geometries (Sparse Array) by using the idea of interpolated array [8] is the main topic here. In this method we need no extra sensors and subarrays so array aperture significantly increased means that we need only \( K + 1 \) sensors for detection of \( K \) coherent signals in contrast to conventional spatial smoothing that needs \( 2K \) sensors or FBSS that use \( \frac{3}{2} K \) sensors [9]. This paper is organized as follows. A definition of cumulant and some properties of that is introduced in Section 2. Narrowband signal model and also the proposed method of DOA estimation are then considered and analyzed in detail in Sections 3 and 4, respectively. The simulation results to validate the effectiveness of our method are shown in Section 5 and some concluding remarks are given in Section 6.

2. Cumulant for Array Processing

Conventional array processing techniques utilize only the second order statistics of received signals. Second-order statistics are sufficient whenever the signals can be completely characterized by knowledge of the first two mo-
ments, as in the Gaussian case; however, in real applications, practical far field sources often emit non-Gaussian signals, e.g., as in a communication scenario. Whenever second-order statistics can not completely characterize all of the statistical properties of underlying signals, it is beneficial to consider information embedded in higher than second-order moments. Higher order prove to be a rewarding alternative to second order statistics, and there are many signal processing problems that are not solvable without access to HOS [10]. Particular cases of higher order statistics (Cumulant) are the third and fourth order statistics. In most of the application that deal with cumulant we use fourth order statistic, a logical question to ask is “why do we need fourth order cumulant?” If a random process is symmetrically distributed like Laplace, Uniform, Gaussian, etc. then its third order cumulant equals zero. Additionally, some processes have extremely small third order cumulants and much larger fourth order so for such a process we must use fourth order cumulant. We list properties of cumulants that are useful to us in the sequel [10]:

[CP1]—If \( \{\alpha_i\}_{i=1}^n \) are constants and \( \{x_i\}_{i=1}^n \) are random variables, then

\[
\text{cum}(\alpha_1 x_1, \alpha_2 x_2, \ldots, \alpha_n x_n) = \left( \prod_{i=1}^n \alpha_i \right) \text{cum}(x_1, x_2, \ldots, x_n)
\]

[CP2]—Cumulants are additive in their arguments

\[
\text{cum}(x_1 + y_1, x_2, \ldots, x_n)
\]

\[
= \text{cum}(x_1, x_2, \ldots, x_n) + \text{cum}(y_1, x_2, \ldots, x_n)
\]

[CP3]—If the random variable \( \{x_i\}_{i=1}^n \) are independent of the random variables \( \{y_i\}_{i=1}^n \), then

\[
\text{cum}(x_1 + y_1, x_2 + y_2, \ldots, x_n + y_n)
\]

\[
= \text{cum}(x_1, x_2, \ldots, x_n) + \text{cum}(y_1, y_2, \ldots, y_n)
\]

[CP4]—Cumulants suppress Gaussian noise of arbitrary covariance, i.e., if \( \{z_i\}_{i=1}^n \) are Gaussian random variables independent of \( \{x_i\}_{i=1}^n \) and \( n > 2 \), we have

\[
\text{cum}(x_1 + z_1, x_2 + z_2, \ldots, x_n + z_n) = \text{cum}(x_1, x_2, \ldots, x_n)
\]

[CP5]—If a subset of random variables \( \{x_i\}_{i=1}^n \) are independent of the rest, then

\[
\text{cum}(x_1, x_2, \ldots, x_n) = 0
\]

[CP6]—The permutation of the random variables does not change the value of the cumulant.

In this paper we use Fourth order Cumulant defined as follows:

\[
\text{Cum}(r_1, r_2, r_3, r_4)
\]

\[
= E\{r_1 r_2 r_3 r_4\} - E\{r_1 r_2\} E\{r_3 r_4\} - E\{r_1 r_3\} E\{r_2 r_4\} - E\{r_1 r_4\} E\{r_2 r_3\}
\]

### 3. Problem Formulation

Consider an arbitrary array composed of \( M \) sensors. Let \( q \) narrowband and non-Gaussian plane waves with zero mean, centered at frequency \( w_c \), impinge on the array with \( M \) sensors like Figure 1 from directions \( \{\theta_1, \ldots, \theta_q\} \). Using complex signal representation, the received signals at the \( m \)th sensor can be expressed as

\[
r(t) = \sum_{n=1}^{q} a_n s_n(t) e^{j\theta_c} + n_m(t)
\]

where \( s_n(t) \) is the signal associated with the \( n \)th wavefront, \( a_n \) is the complex response of the sensor to the \( n \)th wavefront, \( (x_n, y_n) \) are the coordinates of the \( n \)th sensor measured in half wave-length units and \( n_m(t) \) is an additive noise at the \( m \)th sensor. In addition, the unknown noise is assumed to be Gaussian with variance \( \sigma^2 \), uncorrelated with the signals and uncorrelated from sensor to sensor. Rewriting (1) in a vector notation, assuming for simplicity that the sensors are omnidirectional with unit gain, i.e., \( a_n = 1 \), we obtain:

\[
r(t) = \sum_{n=1}^{q} a(\theta_n) s_n(t) + n(t)
\]

where \( r(t), n(t) \) are the \( M \times 1 \) vector

\[
r(t) = \begin{bmatrix} r_1(t), \ldots, r_m(t) \end{bmatrix}^T
\]

\[
n(t) = \begin{bmatrix} n_1(t), \ldots, n_m(t) \end{bmatrix}^T
\]

And \( a(\theta_n) \) is the steering vector of the array in the direction \( \theta_n \):

\[
a(\theta_n) = \begin{bmatrix} e^{j\theta_n \sin \theta_1 \cos \theta_2}, \ldots, e^{j\theta_n \sin \theta_1 \cos \theta_q} \end{bmatrix}^T
\]

To further simplify the notation, we rewrite (2) as

\[
r(t) = A(\theta) S(t) + n(t)
\]

where \( S(t) \) is the \( q \times 1 \) vector as:

\[
S(t) = \begin{bmatrix} s_1(t), \ldots, s_q(t) \end{bmatrix}^T
\]

and \( A(\theta) \) is the \( M \times q \) matrix

\[
A(\theta) = \begin{bmatrix} a(\theta_1), \ldots, a(\theta_q) \end{bmatrix}
\]

also the fourth order cumulant of \( r(t) \) can be computed as (8):

\[
\text{Cum}\begin{bmatrix} r_1, r_2, r_3, r_4 \end{bmatrix} = E\{r_1 r_2 r_3 r_4\} - E\{r_1 r_2\} E\{r_3 r_4\} - E\{r_1 r_3\} E\{r_2 r_4\} - E\{r_1 r_4\} E\{r_2 r_3\}
\]

For \( 1 \leq i, j, k, l \leq M \).

The computation of cumulant and some algorithm based on are come in [11]. In this paper for solving noise
Direction Finding Using Cumulant in Multipath Environment with an Arbitrary Array

In fact we replace every member of covariance matrix with its cumulant counterpart which is computed in the sufficient snapshots by (9).

\[
E\{r_n^*(t)r_t(t)\}
= \frac{\sigma^2}{\gamma_{_{A_n}}} a_1
\begin{array}{c}
\text{cum}\{r_n^*(t), r_t(t), r_n^*(t), r_t(t)\}
= \frac{1}{N_p} \sum_{i=1}^{N_p} (r_n^*(t)r_t(t) - r_n^*(t)r_t(t))
- \frac{1}{N_p} \sum_{i=1}^{N_p} (r_n^*(t)r_t(t) - r_n^*(t)r_t(t))
- \frac{1}{N_p} \sum_{i=1}^{N_p} (r_n^*(t)r_t(t) - r_n^*(t)r_t(t))
- \frac{1}{N_p} \sum_{i=1}^{N_p} (r_n^*(t)r_t(t) - r_n^*(t)r_t(t))
\end{array}
\]

4. Proposed DOA Estimation

4.1. Spatial Smoothing

Communication systems operating in a mobile communication environment may encounter multipath propagation caused by various reflected surfaces (e.g., buildings, hills, cars, etc.). When a signal wavefront reflects off of a surface, the original wavefront and the reflected wavefront will both impinge on the receiving array of sensors, although from different directions. Since the original signal and reflected signal come from the same radiating source, if the delay difference between the two paths is sufficiently small, then the signals are coherent (i.e., fully correlated) and their covariance matrix or cumulant matrix is singular. As was pointed out in the previous section, the nonsingularity of the signal covariance matrix or cumulant matrix is the key to successful application of eigenstructure techniques. Many effective decorrelated methods are then proposed to overcome this problem. Among these methods, the spatial smoothing techniques are relatively more effective, which are first introduced in [2] and extensively studied by Shan et al. [3] and Pillai and Kwon [9]. Their solutions are based on a preprocessing scheme that divides the original array into number of subarrays. Then the average of the subarray covariance matrices is computed and in conjunction with the high-resolution methods to resolve the signals. Examination of the smoothing technique described in [13,14] reveals that it is not limited to uniform linear arrays. The forward and also forward-backward spatial smoothing can be performed on any array which can be subdivided into subarrays which all have the same configuration, but are shifted with respect to each other like Figure 2. But when it is urgent to use Sparse array or any array that has not a suitable construction for using spatial smoothing like Figure 1 what should we do for the detection of the \(K\) coherent signals?

4.2. Array Interpolation

1) The first step in designing an interpolated array is to divide the field of view of the real arbitrary array into sectors.

2) For each sector we define a set of angles like \(\theta_i = [\theta_i(1), \theta_i(2) + \Delta \theta, \theta_i(2) + 2\Delta \theta, \ldots, \theta_i(2)^n] \) where \(\Delta \theta = \frac{\theta_{2} - \theta_{1}}{n}\). These angles are used only in the design of the interpolation matrix.

3) Compute the steering vectors associated with the set \(\theta_i\) for the given real array and arrange them in a matrix as follows:

\[
A_i(\theta_i) = [a(\theta_i(1)), \ldots, a(\theta_i(2^n))]
\]

Copyright © 2012 SciRes.
Next we decide where we want to place the virtual elements of the interpolated array, having decided on their locations we can compute the array manifold of the interpolated array. We denote by $\overline{A}(\theta)$ the section of this array manifold computed for the angles $\theta_i$ also as follows:

$$\overline{A}(\theta) = \left[ \overline{a}(\theta^{(1)}), \ldots, \overline{a}(\theta^{(2)}) \right]$$ (11)

In other words $A_i(\theta)$, $\overline{A}(\theta)$ are the responses of the real and virtual arrays to the signals arriving from the set $\theta_i$ respectively. In this paper we try to create virtual subarrays having a suitable geometry for the application of the spatial smoothing technique. The idea in this paper is to select the uniform linear arrays as the virtual subarrays all having the same number of sensors as the original real array but are shifted versions of each other like Figure 3. The displacement vector of these virtual subarrays and also between their sensors is equal to $\frac{\lambda}{2}$.

4) The basic assumption is that the array manifold of the virtual array can be obtained by linear interpolation of the array manifold of the real array, within each sector. In other words, we assume that there exist a constant $M \times M$ matrix $B_i$ such that

$$B_i A_i(\theta) = \overline{A}(\theta)$$ (12a)

Of course, the interpolation is not exact and therefore the equality above does not really hold. The “best” interpolation matrix $B_i$ is the one which will give the best fit between the interpolated response $B_i A(\theta)$ and the desired response $\overline{A}(\theta)$.

In this paper we assume that all the coherent DOAs are located in one section like $[\theta_1, \theta_2]$, this section is divided with the calibration angles $\delta \theta$.

$$\tilde{\theta} = \left[ \theta_1, \theta_1 + \delta \theta, \ldots, \theta_1 + i \delta \theta, \theta_2 \right]$$

where $\delta \theta = \frac{\theta_2 - \theta_1}{n}, i = 1, \ldots, n$

where $n$ is the division index. $A_k(\tilde{\theta}), A_k(\hat{\theta})$ are the manifold matrixes for the $k$th desired virtual subarray and real array respectively constructed from these calibration angles $\hat{\theta}$.

Because every virtual subarray is the shifted version of the first virtual subarray. So:

$$A_k(\tilde{\theta}) = A_k(\hat{\theta}) D^{-1} B_k A_k(\hat{\theta})$$ (12b)

where $D^k$ denotes the $k$th power of the $n \times n$ diagonal matrix $D$ as follows:

$$D = \text{diag}\left\{ e^{-j \frac{\nu_0 f_0}{c} \sin \theta}, e^{-j \frac{\nu_0 f_0}{c} \sin (\theta + \delta \theta)}, \ldots, e^{-j \frac{\nu_0 f_0}{c} \sin (\theta_i)} \right\},$$

$$w_0 = 2\pi f_0$$ (13)

In (13) parameter $c$ is the speed of light and $d$ is the distance between adjacent sensors of virtual ULA that for avoiding grating lobe it must be equal to half a wavelength $\left( d = \frac{\lambda}{2} \right)$. By the idea of interpolation as described in 4.2 we can relate every virtual subarray to the real ones by solving a following linear equation:

$$A_k(\tilde{\theta}) = B_k A_k(\hat{\theta})$$ (14)

In following we use the cumulant’s properties in the direction finding. Since the Cumulant or Fourth order moment is blind to additive Gaussian noise (color or white) as shown in (CP4) the Fourth order Cumulant matrix counterpart of the real array is obtained as follows:

$$C = (A_i(\theta)) S_i (A_i(\theta))^H$$ (15a)

where $S_i$ is the $q \times q$ Cumulant matrix of signals that impinge on the real array. After obtaining interpolation matrix $B_i$ for every virtual subarray now we can compute the cumulant of these subarrays as follows:

$$C_k = B_i C B_i^H$$ (15b)

Now the spatially smoothed cumulant matrix is defined as the sample means of the virtual subarrays cumulants:

$$\overline{C} = \frac{1}{K} \sum_{k=1}^{K} C_k$$ (16)

where $K$ is the number of virtual subarrays that for nonsingularity of $\overline{C}$ it should be larger than the number of coherent signals. If mapping matrices $(B)_h$ best selected then $\overline{C}$ becomes:

$$\overline{C} = (A_{11}(\theta)) \left( \frac{1}{K} \sum_{k=1}^{K} (E^{k-1}) S_h (E^{k-1})^H \right) (A_{11}(\theta))^H$$ (17)

And $E^K$ is the $k$th power of $E$.

$$E = \text{diag}\left\{ e^{-j \frac{\nu_0 f_0}{c} \sin \theta_1}, e^{-j \frac{\nu_0 f_0}{c} \sin \theta_2}, \ldots, e^{-j \frac{\nu_0 f_0}{c} \sin \theta_n} \right\}$$

where $w_0 d = \frac{2\pi f_0 \lambda}{c} = \pi$

From (17) we can define $\overline{S}_h$ as follows:
\[ \tilde{S}_c = \frac{1}{K} \sum_{k=1}^{K} (E^{k-1}) \tilde{S}_e (E^{k-1})^T \]

\[ = \left[ (I)(E) \cdots (E^{k-1}) \right] \left( \frac{1}{K} \tilde{S}_e \right) \left( \frac{1}{K} \tilde{S}_e \right)^T \left( E^{k-1} \right) \left( E^{k-1} \right)^T \]  

(18)

Which can be further simplified to:

\[ \tilde{S}_c = G G^H \]  

(19)

where \( G \) is the \( q \times Kq \) block matrix:

\[ G = \left[ C(E)C \cdots (E^{k-1})C \right] \]  

(20)

In which \( C \) denoting the Hermitian square root of \( \frac{1}{K} S_e \). It is clear that the rank of \( C \) is equal to the rank of \( \tilde{S}_c \). Thus our task is to prove that \( G \) has rank \( q \) even when this \( q \) signals are coherent. Recalling that the rank of a matrix is unchanged by a permutation of its columns, it can be easily verified that \( G \) can be written in the form of (21).

\[ G = \begin{bmatrix} c_{11} b_1 & c_{12} b_1 & \cdots & c_{1q} b_1 \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ c_{q1} b_q & c_{q2} b_q & \cdots & c_{qq} b_q \end{bmatrix} \]  

(21)

where \( c_{ij} \) is the \( ij \)th element of the matrix \( C \) and \( b_i (i = 1, \ldots, q) \) is the \( K \times 1 \) column vector:

\[ b_i = [1 e^{j \theta_i}, e^{2j \theta_i}, \ldots e^{qj \theta_i}]^T \]  

(22)

To show that the matrix \( G \) is of rank \( q \), it suffices to show that each row of the matrix \( S \) has at least one non-zero element and that all the vectors \( b_i \) are linearly independent. The first fact follows by contradiction. Assume that a row of \( C \), say the \( k \)th, is composed of all zeros this implies that the \( k \)th signal has zero cumulant, in contradiction to the definition of \( C \) as the cumulant matrix of nonvanishing non-Gaussian signals. The linear independence of the vectors \( b_i \) follows by observing that these vectors can be embedded in a vandermonde matrix. Since rank of \( \tilde{S}_c \) is equal to the rank of \( G \) and rank of \( G \) is \( q \) so the rank of \( C \) is \( q \). At the end we apply one of the subspace methods to \( C \).

5. Simulation and Experimental Results

In this section, we illustrate the performance of our method through simulations. We select conventional spatial smoothing in the presence of white noise with variance \( \sigma_w^2 \) as a comparative method. In the first simulation, we consider three non-Gaussian coherent signals received by two arrays, ULA for the Conventional Spatial Smoothing and the known arbitrary one for our Cumulant Spatial Smoothing each with 7 sensors. The amplitudes and phases of the complex fading coefficient of coherent signals are \([1,0.7,0.6]\) and \([1,12,22]\) respectively. The unknown colored Gaussian additive noises are generated by passing the complex white Gaussian random processes with zero mean and \( \sigma_n^2 \) variance through a spatial moving average (MA) filter of order 2 with coefficients \([a_0,a_1,a_2]\), where \([a_0,a_1,a_2]\) are selected to be \([1,0.8,0.6]\) in the simulations. The input SNR is defined as \( 10 \log_{10} \left( \frac{\sigma_n^2}{\sigma_w^2} \right) \). Note that the noise covariance matrices for colored and white noises have the same trace, i.e. total noise power introduced to these methods have the same power. 200 Monte Carlo trials were performed for each experiment, and the root mean square error (RMSE) of the DOA estimates is used as the performance index:

\[ \text{RMSE} = \sqrt{\frac{1}{200q} \sum_{n=1}^{200} \left( \hat{\theta}_n - \theta_k \right)^2} \]

where \( \hat{\theta}_n \) is an estimate of \( \theta_k \) for the \( n \)th Monte Carlo trial, and \( q \) is the number of all the independent or coherent signals. In Figure 4 the RMSE of the DOA estimates against input SNR is shown with the \( N_s = 500 \) and 1000 snapshots for the two methods. Also the RMSE of the DOA estimate of two methods versus the number of snapshots with input SNR equaling to zero is shown in Figure 5. In these two simulations we assume that the division index is \( n = 50 \) for our method. As it is clearly seen, the performance of our method (Cumulant Spatial Smoothing) is better than Conventional spatial smoothing.
specially in low SNR but against the increasing of snapshots. In the third simulation we show the performance of our method versus division index $n$ in the presence of three predefined coherent signals. As it is obvious in Figures 6(a)-(c), by choosing $n$ as large as possible we can get a better resolution (targets are distinguished clearly). To investigate the performance of our cumulant based method, we performed additional experiments by changing both the SNR and the data length. Each mean and standard deviation pair in each table are obtained from 100 independent realizations. Table 1(a) reports the results of Spatial Smoothing for 200 snapshots with different SNRs in which the conventional spatial smoothing based on sample covariance means is selected as a direction finder method. Table 1(b) also shows the performance of conventional spatial smoothing versus two different snapshots with SNR equaling to zero.

From Tables 1(a), (b) it is obvious that the conventional spatial smoothing is more dependent on SNR than snapshot and can estimate DOAs with fewer snapshots by increasing the SNR accurately. Also we repeat the experiment again for our method based on the sample cumulant means with division index $n = 50$, as it is seen in Table 2(a) we can obtain better estimations in the low SNR by increasing the number of snapshots or data lengths. Table 2(b) also shows the performance of our method versus two different SNRs with a number of snapshots equaling to 200.

As shown in Table 2(b) our cumulant spatial smoothing fails in general in a few snapshots, it means that the effect of snapshots in our method is more than the SNR in contrast to conventional spatial smoothing. In the latter simulation we do the detection probability for these two
obtained from solving Equation (14). We see this detection probability for our cumulant spatial smoothing with three different smoothing versus SNR. As it is clearly seen by increasing these parameter the resolution and our method against the probability of detection of our method will be better. We see by increasing these parameter the resolution and the probability of detection of our method will be better.

6. Conclusion

In this paper, DOA estimation method in multipath environment with fewer sensors has been presented. We have taken the effects of cumulant and interpolated array into account in spatial smoothing. In this paper, by the use of mapping and interpolation techniques in an arbitrary array geometry we tried to approach to the ULA and also shifted versions of this ULA so that the spatial smoothing can be performed on it, therefore the need for extra real subarrays has been removed. Because this mapping can color the white noise and also in practical situation we don’t have any knowledge of Gaussian noise, we use cumulant instead of the covariance of data. The simulation results presented illustration of the performance of our method against n, SNR and the number of snapshots. We see by increasing these parameter the resolution and the probability of detection of our method will be better.

Table 1. (a) (b), DOA estimation with conventional spatial smoothing with 7 sensor versus SNR and snapshot.

<table>
<thead>
<tr>
<th></th>
<th>SNR = -5 dB</th>
<th>SNR = 0 dB</th>
<th>SNR = 5 dB</th>
<th>SNR = 10 dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOA</td>
<td>7°</td>
<td>13°</td>
<td>22°</td>
<td></td>
</tr>
<tr>
<td>Snapshot = 200</td>
<td>Mean = 10.8546</td>
<td>Mean = 16.3767</td>
<td>Mean = 25.0914</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Std = 2.2275</td>
<td>Std = 2.3773</td>
<td>Std = 2.2781</td>
<td></td>
</tr>
<tr>
<td>Snapshot = 500</td>
<td>Mean = 8.3541</td>
<td>Mean = 14.3715</td>
<td>Mean = 23.6741</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Std = 1.1636</td>
<td>Std = 1.0297</td>
<td>Std = 1.1233</td>
<td></td>
</tr>
<tr>
<td>Snapshot = 1000</td>
<td>Mean = 7.2890</td>
<td>Mean = 13.5066</td>
<td>Mean = 22.44</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Std = 0.2917</td>
<td>Std = 0.2954</td>
<td>Std = 0.2840</td>
<td></td>
</tr>
</tbody>
</table>

Table 2. (a) (b), DOA estimation with our cumulant method with 7 sensors versus SNR and snapshots.

<table>
<thead>
<tr>
<th></th>
<th>SNR = 0, 0.5</th>
<th>SNR = 0</th>
<th>SNR = 0</th>
<th>SNR = 200</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOA</td>
<td>7°</td>
<td>13°</td>
<td>22°</td>
<td></td>
</tr>
<tr>
<td>Snapshot = 200</td>
<td>Mean = 9.9831</td>
<td>Mean = 14.5400</td>
<td>Mean = 24.6120</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Std = 1.7519</td>
<td>Std = 1.6910</td>
<td>Std = 1.7063</td>
<td></td>
</tr>
<tr>
<td>Snapshot = 500</td>
<td>Mean = 8.1503</td>
<td>Mean = 14.1648</td>
<td>Mean = 23.6741</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Std = 1.1636</td>
<td>Std = 1.1600</td>
<td>Std = 1.1215</td>
<td></td>
</tr>
<tr>
<td>Snapshot = 1000</td>
<td>Mean = 7.6748</td>
<td>Mean = 13.7687</td>
<td>Mean = 22.9125</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Std = 0.3010</td>
<td>Std = 0.3943</td>
<td>Std = 0.4016</td>
<td></td>
</tr>
<tr>
<td>Snapshot = 1500</td>
<td>Mean = 7.2568</td>
<td>Mean = 13.4101</td>
<td>Mean = 22.1988</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Std = 0.2961</td>
<td>Std = 0.2655</td>
<td>Std = 0.2531</td>
<td></td>
</tr>
<tr>
<td>Snapshot = 2000</td>
<td>Mean = 8.4111</td>
<td>Mean = 14.6851</td>
<td>Mean = 23.5630</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Std = 1.2683</td>
<td>Std = 1.1725</td>
<td>Std = 1.2031</td>
<td></td>
</tr>
</tbody>
</table>

REFERENCES


Direction Finding Using Cumulant in Multipath Environment with an Arbitrary Array


