14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, September 4-8, 2006, copyright by EURASIP

ON THE EFFICIENT IMPLEMENTATION AND TIME-UPDATING OF THE LINEARLY CONSTRAINED MINIMUM VARIANCE BEAMFORMER

Andreas Jakobsson† *and Stephen R. Alty*‡

† Department of Electrical Engineering, Karlstad University, Sweden.
‡ Centre for Digital Signal Processing Research, King's College London, UK.

ABSTRACT

The linearly constrained minimum variance (LCMV) method is an extension of the classical minimum variance distortionless response (MVDR) filter, allowing for multiple linear constraints. Depending on the spatial filter length and the desired frequency grid, a direct computation of the resulting spatial beampattern may be prohibitive. In this paper, we exploit the rich structure of the LCMV expression to find a non-recursive computationally efficient implementation of the LCMV beamformer with fixed constraints. We then extend this implementation by means of its time-varying displacement structure to derive an efficient time-updating algorithm of the spatial spectral estimate. Numerical simulations indicate a dramatic computational gain, especially for large arrays and fine frequency grids.

1. INTRODUCTION

The area of sensor array signal processing has received a considerable interest in the recent literature, and numerous algorithms addressing different aspects of the topic have been proposed (see, e.g., [1, 2] and the references therein). Typically, these algorithms exploits the difference in propagation delay recorded at the different sensor array elements to form a parametric or non-parametric spatial spectral estimate. Recently, non-parametric spatial spectrum estimators have received renewed interest, mainly as these have the benefit of not explicitly assuming an *a priori* known signal model, and as a result tend to be more robust to variations in the measured signal than parametric counterparts. A traditional non-parametric method for spatial spectral estimation is to apply the simple beamformer for all directions of interest and use the beamformer outputs to estimate the spatial power spectral distribution. However, as is well-known, such a *non-adaptive* beamformer suffers from either low resolution or high leakage, or both [2,3]. Another common approach is the minimum variance distortionless response (MVDR) beamformer. The MVDR beamformer has several desirable properties, and is often applied when high spatial resolution is desired. It is a filterbank method that forms a *data-adaptive* weight vector for each direction of interest, weighting the array elements in an adaptive manner so as to minimize the beamformer output power while passing signals from a given direction of interest undistorted. This effectively places deep nulls cancelling interference from sources in directions other than the one of interest, resulting in a very high resolution spatial estimate. The linearly constrained minimum variance (LCMV) beamformer is a generalized form of the MVDR beamformer, allowing for additional constraints, for example to counter the influence of known jamming signals (see, e.g., [2], and the references therein, for a more general discussion on LCMV and on different typical constraints).

In general, both the MVDR and the LCMV beamformers suffer from being computationally cumbersome, especially for large data sets, large arrays and/or for evaluation over a fine frequency grid; this as both methods suffer from requiring the evaluation of a vector-matrix product containing the inverse of the possibly large dimensional data covariance matrix, say $\mathbf{R}_y$, for each frequency grid point of interest. For this reason, there has been a substantial interest in finding efficient *recursive* implementations of the LCMV and the related generalized sidelobe canceller (see, e.g., [4–9] and the references therein). However, given the recursive nature of these algorithms, one often needs to consider step-size selection, internal error propagation and convergence rate. Furthermore, in several applications, only a limited amount of data is available for a particular setup; in such cases, often only the resulting spatial spectral estimate is of interest.

In this paper, we propose a non-recursive efficient implementation of the LCMV beamformer with fixed non-frequency dependent constraints exploiting the rich structure of the LCMV expression. Via the use of the matrix inversion lemma, the LCMV structure allows for the evaluation of the beamformer using the fast Fourier transform (FFT), dramatically reducing the required computational complexity. Different from the above mentioned recursive algorithms, the proposed method provides an *exact* implementation of the LCMV beamformer. The presented implementation, together with its time-variant displacement (TVD) structure, is then exploited to derive an efficient forward-backward averaged (FBA) sliding-window time-updating of the LCMV spatial spectrum (we refer the reader to [10, 11] for a further discussion on displacement theory). The algorithm has the benefit of not requiring any step-size selection and offers the exact LCMV spatial spectrum over the examined time window.

2. THE LCMV BEAMFORMER

Consider a uniform linear sensor array consisting of m elements, measuring signals impinging on the array with approximately planar wavefronts. Let $\mathbf{y}_t$, for $t = 1, \ldots, N$, denote the measured m-dimensional data vectors. The LCMV beamformer can, for a *given* spatial frequency ω_1, be found as the m-tap spatial finite impulse response (FIR) filter, $\mathbf{h}_{\omega_1}$, minimizing (see, e.g., [2])

$$\mathbf{h}_{\omega_1} = \arg\min_{\mathbf{h}_{\omega_1}} \mathbf{h}^*_{\omega_1} \mathbf{R}_y \mathbf{h}_{\omega_1} \quad \text{subject to} \quad \mathbf{C}^*_{\omega} \mathbf{h}_{\omega_1} = \mathbf{f}, \quad (1)$$

where $(\cdot)^*$ denotes the conjugate transpose, and $\mathbf{f}$ is a d-dimensional constraint vector, with $d \ll m$. Furthermore,

This work was supported in part by the Royal Society, UK.

$\mathbf{R}_y = E\left\{\mathbf{y}_t\mathbf{y}_t^*\right\}$, with $E\left\{\cdot\right\}$ denoting expectation,

$$\mathbf{C}_\omega = \left[\begin{array}{cc} \mathbf{a}_{\omega_1} & \mathbf{A}_{\tilde{\omega}} \end{array}\right] \quad (2)$$

$$\mathbf{a}_\omega = \left[\begin{array}{cccc} 1 & e^{i\omega} & \dots & e^{i\omega(m-1)} \end{array}\right]^T \quad (3)$$

where $(\cdot)^T$ denotes the transpose and $\mathbf{A}_{\tilde{\omega}}$ represents the constraint matrix (see, e.g., [2] for further details on how to select the constraints and the constraint matrix). Herein, we will focus on the common case when the $m \times (d-1)$ constraint matrix, $\mathbf{A}_{\tilde{\omega}}$, does not depend on the frequency of interest, ω_1; this is, for instance, the case when the constraints are used to cancel known jammer directions. Further, the constraint corresponding to the frequency of interest, $\omega_1 \in [-\pi/2, \pi/2]$, is typically selected equal to unity to pass this frequency undistorted, i.e., $f_1 = 1$, where

$$\mathbf{f} = \left[\begin{array}{cc} f_1 & \underline{\mathbf{f}}^T \end{array}\right]^T. \quad (4)$$

As is well known, the beamformer minimizing (1) is found as [2,3]

$$\mathbf{h}_{\omega_1} = \mathbf{R}_y^{-1}\mathbf{C}_\omega\left(\mathbf{C}_\omega^*\mathbf{R}_y^{-1}\mathbf{C}_\omega\right)^{-1}\mathbf{f}, \quad (5)$$

yielding the spatial spectral estimate

$$\varphi_y(\omega_1) \triangleq \mathbf{h}_{\omega_1}^*\mathbf{R}_y\mathbf{h}_{\omega_1} = \mathbf{f}^*\left(\mathbf{C}_\omega^*\mathbf{R}_y^{-1}\mathbf{C}_\omega\right)^{-1}\mathbf{f}. \quad (6)$$

Commonly, $\mathbf{R}_y$ in (6) is unknown, and is replaced by a consistent estimate; herein, we use the FBA covariance matrix estimate as it is known to yield preferable spectral estimates [12], i.e.,

$$\hat{\mathbf{R}}_y = \frac{1}{2}\left(\hat{\mathbf{R}}_y^f + \mathbf{J}\hat{\mathbf{R}}_y^{fT}\mathbf{J}\right), \quad (7)$$

where $\mathbf{J}$ denotes the exchange matrix, and

$$\hat{\mathbf{R}}_y^f = \frac{1}{N}\sum_{t=1}^{N}\mathbf{y}_t\mathbf{y}_t^*. \quad (8)$$

We note that the computational cost of evaluating $\varphi_y(\omega_1)$, with the $d \times d$ matrix

$$\mathbf{Q}_\omega \triangleq \mathbf{C}_\omega^*\hat{\mathbf{R}}_y^{-1}\mathbf{C}_\omega, \quad (9)$$

might well be prohibitive for large m, especially as $\varphi_y(\omega)$ often needs to be evaluated over a very fine frequency grid, requiring $\mathbf{Q}_\omega$ to be computed over P frequency grid points, with typically $P \gg N$; a brute-force evaluation of $\mathbf{Q}_\omega$, using (7), for P frequencies, requires $\mathscr{O}(m^3 + (N+dP)m^2 + d^2mP)$ operations. Here, we will for simplicity assume that the frequency grid covers the full frequency range, remarking that frequencies such that $\mathbf{C}_\omega$ loses rank should be omitted from the resulting estimate. We note that should only a limited frequency region be of interest, the algorithm can easily be modified accordingly.

3. PROPOSED EFFICIENT IMPLEMENTATION

Using the block-matrix inversion lemma (see, e.g., [3]), $\mathbf{Q}_\omega^{-1}$ can be expressed as (10), given at the top of next page, where $\mathbf{I}$ denotes the identity matrix (of appropriate dimension), and

$$\mu_{\omega_1} = \mathbf{a}_{\omega_1}^*\hat{\mathbf{R}}_y^{-1}\mathbf{a}_{\omega_1} \quad (11)$$

$$\nu_\omega = \mathbf{A}_{\tilde{\omega}}^*\hat{\mathbf{R}}_y^{-1}\mathbf{a}_{\omega_1} \quad (12)$$

$$\mathbf{G}_{\tilde{\omega}} = \mathbf{A}_{\tilde{\omega}}^*\hat{\mathbf{R}}_y^{-1}\mathbf{A}_{\tilde{\omega}} \quad (13)$$

By again using the matrix inversion lemma to rewrite the factor $(\mathbf{G}_{\tilde{\omega}} - \mu_{\omega_1}^{-1}\nu_\omega\nu_\omega{}^*)^{-1}$, (10) can be expressed as

$$\mathbf{Q}_\omega^{-1} = \left[\begin{array}{cc} z_1 & \mathbf{z}_2^* \\ \mathbf{z}_2 & \mathbf{Z}_3 \end{array}\right] \quad (14)$$

where

$$z_1 = \mu_{\omega_1}^{-1} + \mu_{\omega_1}^{-2}\phi_\omega + \frac{\phi_\omega^2}{\mu_{\omega_1}^2(\mu_{\omega_1} - \phi_\omega)} \quad (15)$$

$$\mathbf{z}_2 = -\mu_{\omega_1}^{-1}\mathbf{G}_{\tilde{\omega}}^{-1}\nu_\omega - \frac{\phi_\omega}{\mu_{\omega_1}^2 - \mu_{\omega_1}\phi_\omega}\mathbf{G}_{\tilde{\omega}}^{-1}\nu_\omega \quad (16)$$

$$\mathbf{Z}_3 = \mathbf{G}_{\tilde{\omega}}^{-1} + \frac{1}{\mu_{\omega_1} - \phi_\omega}\mathbf{G}_{\tilde{\omega}}^{-1}\nu_\omega\nu_\omega^*\mathbf{G}_{\tilde{\omega}}^{-1} \quad (17)$$

with $\phi_\omega \triangleq \nu_\omega^*\mathbf{G}_{\tilde{\omega}}^{-1}\nu_\omega$. Thus, assuming that the frequency of interest is passed undistorted, $\varphi_y(\omega_1)$ in (6) can be expressed as

$$\varphi_y(\omega_1) = z_1 + 2\,\mathrm{Re}\left(\underline{\mathbf{f}}^*\mathbf{z}_2\right) + \underline{\mathbf{f}}^*\mathbf{Z}_3\underline{\mathbf{f}}, \quad (18)$$

where $\mathrm{Re}(x)$ denotes the real part of x. It is worth noting that the LCMV beamformer is often used to null known jammer directions; in such a case, $\underline{\mathbf{f}} = \mathbf{0}$, allowing the evaluation of $\varphi_y(\omega_1)$ to be simplified to $\varphi_y(\omega_1) = z_1$. We proceed to note that for a Hermitian matrix $\Lambda \in \mathbb{C}^{m\times m}$, it holds that [13]

$$\mathbf{a}_{\omega_1}^*\Lambda\mathbf{a}_{\omega_1} = 2\,\mathrm{Re}\left(\sum_{s=0}^{m-1}\lambda_s e^{i\omega_1 s}\right) - \lambda_0, \quad (19)$$

where

$$\lambda_s = \sum_{k=s}^{m-1}\Lambda_{k,k-s}, \quad (20)$$

with $\Lambda_{k,p}$ denoting the (k,p)th index of Λ. Thus, given $\hat{\mathbf{R}}_y^{-1}$, both the quadratic forms μ_{ω_1} and ϕ_ω, and thus z_1, can be efficiently evaluated *over all P frequency grid points simultaneously* using the FFT. Similarly, $\underline{\mathbf{f}}^*\mathbf{G}_{\tilde{\omega}}^{-1}\nu_\omega$ can be computed for all P frequencies using the FFT. Further, we note that the $(d-1)\times(d-1)$ matrix $\mathbf{G}_{\tilde{\omega}}^{-1}$ only needs to be evaluated once for all frequencies. Thus, the evaluation of (18) requires $\mathscr{O}(Nm^2 + m^3 + P\log P + m^2(d-1) + m(d-1)^2)$ operations. We note that in cases when only a narrow band, or a subset, of the spatial spectrum is of interest, the complexity can be significantly reduced by exploiting a local Fourier transform (see, e.g., [14,15]) in place of the FFT in (19). In the following section, we will proceed to examine how to update the LCMV beamformer as additional data becomes available.

4. TIME-UPDATING THE LCMV BEAMFORMER

Given the centrohermitian structure of $\hat{\mathbf{R}}_y$, one may form the decomposition $\hat{\mathbf{R}}_y = \mathbf{K}\mathbf{B}\mathbf{K}^*$ [16], where $\mathbf{K}$ can be selected to be column conjugate symmetric[1] and unitary. In particular, for even dimension $\hat{\mathbf{R}}_y$,

$$\mathbf{K} = \frac{1}{\sqrt{2}}\left[\begin{array}{cc} \mathbf{I} & i\mathbf{I} \\ \mathbf{J} & -i\mathbf{J} \end{array}\right], \quad (21)$$

where $\mathbf{K}$ is a square matrix with the same dimensions as $\hat{\mathbf{R}}_y$.

[1]A matrix $\mathbf{K}$ is said to be column conjugate symmetric if $\mathbf{K} = \mathbf{J}\mathbf{K}^*$.

$$\mathbf{Q}_{\omega}^{-1} = \begin{bmatrix} \mu_{\omega_1} & \nu_{\omega}^* \\ \nu_{\omega} & \mathbf{G}_{\tilde{\omega}} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{I} \\ \mathbf{0} \end{bmatrix} \mu_{\omega_1}^{-1} \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} -\mu^{-1}\nu_{\omega}^* \\ \mathbf{I} \end{bmatrix} (\mathbf{G}_{\tilde{\omega}} - \mu_{\omega_1}^{-1}\nu_{\omega}\nu_{\omega}^*)^{-1} \begin{bmatrix} -\mu_{\omega_1}^{-1}\nu_{\omega} & \mathbf{I} \end{bmatrix} \tag{10}$$

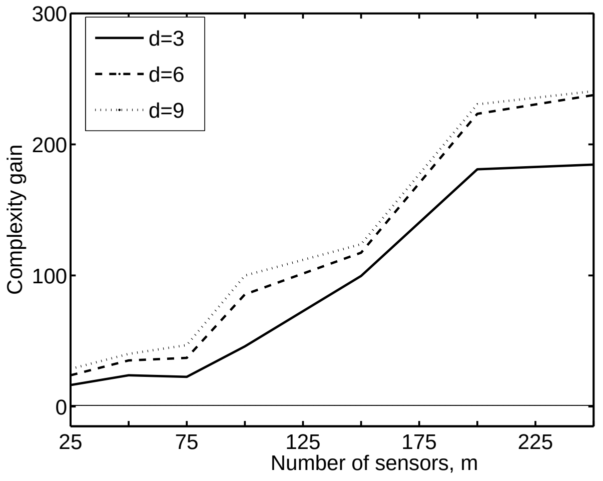

Figure 1: Average complexity gain as a function of the array size, *m*, for varying number of constraints, *d*. Here, the data size is $N = 3m$ and P is selected as the next power of two larger than N.

Similarly, for odd dimension $\hat{\mathbf{R}}_y$,

$$\mathbf{K} = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{I} & 0 & i\mathbf{I} \\ 0 & i\sqrt{2} & 0 \\ \mathbf{J} & 0 & -i\mathbf{J} \end{bmatrix}. \tag{22}$$

For this choice of $\mathbf{K}$, $\mathbf{B}$ is a real symmetric matrix. As shown in [16], this decomposition offers a significant complexity reduction for the most common operations on FBA covariance matrices, such as the time-updating of $\hat{\mathbf{R}}_y$. Let the FBA covariance matrix estimate at time t be denoted $\hat{\mathbf{R}}_y(t)$. Then, an efficient *sliding window* time-update of $\hat{\mathbf{R}}_y(t)$, such that

$$\hat{\mathbf{R}}_y(t) = \hat{\mathbf{R}}_y(t-1) + \hat{\mathbf{Y}}_t\hat{\mathbf{Y}}_t^* - \check{\mathbf{Y}}_t\check{\mathbf{Y}}_t^*, \tag{23}$$

where $\hat{\mathbf{Y}}_t$ and $\check{\mathbf{Y}}_t$ are the updating and downdating data matrices, i.e.,

$$\hat{\mathbf{Y}}_t = \begin{bmatrix} \mathbf{y}_t & \mathbf{J}\mathbf{y}_t^* \end{bmatrix} \tag{24}$$

$$\check{\mathbf{Y}}_t = \begin{bmatrix} \mathbf{y}_{t-N} & \mathbf{J}\mathbf{y}_{t-N}^* \end{bmatrix}, \tag{25}$$

with N denoting the length of the sliding window, can preferably[2] be formed as $\hat{\mathbf{R}}_y(t) = \mathbf{K}\mathbf{B}_t\mathbf{K}^*$, where

$$\mathbf{B}_t = \mathbf{B}_{t-1} + \hat{\mathbf{Z}}_t\hat{\mathbf{Z}}_t^T - \check{\mathbf{Z}}_t\check{\mathbf{Z}}_t^T, \tag{26}$$

with the *compact* updating and downdating data matrices

$$\hat{\mathbf{Z}}_t = \mathbf{K}^*\hat{\mathbf{Y}}_t\mathbf{V} \quad \text{and} \quad \check{\mathbf{Z}}_t = \mathbf{K}^*\check{\mathbf{Y}}_t\mathbf{V} \tag{27}$$

where $\mathbf{K}$ is determined from (21) or (22), and

[2]The time-updating using (26) requires only about half the number of operations compared to the update in (23).

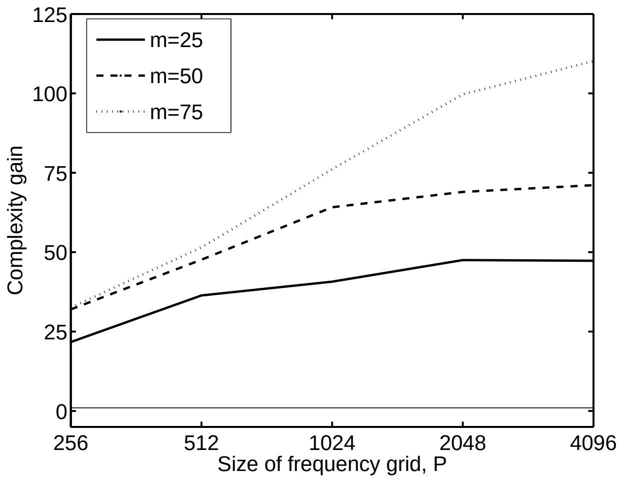

Figure 2: Average complexity gain as a function of the size of the frequency grid, *P*, for varying array sizes, *m*. Here, $d = 3$ and $N = 3m$.

$$\mathbf{V} = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{I} & i\mathbf{I} \\ \mathbf{I} & -i\mathbf{I} \end{bmatrix}. \tag{28}$$

It should be noted that the transforms in (27) imply that $\hat{\mathbf{Z}}_t$ and $\check{\mathbf{Z}}_t$ are real-valued. Here, we are interested in updating $\hat{\mathbf{R}}_y^{-1}(t)$ instead of $\hat{\mathbf{R}}_y(t)$; such an update can be formed using the TVD structure of (26). A time-variant Toeplitz-like $m \times m$ matrix $\mathbf{B}_t$ is said to have a TVD structure if the matrix difference $\nabla\mathbf{B}_t$, defined by [10, 11]

$$\nabla\mathbf{B}_t = \mathbf{B}_t - \mathbf{F}_t\mathbf{B}_{t-\Delta}\mathbf{F}_t^*, \tag{29}$$

has low *rank*, say $r(t)$, where $r(t) \ll m$, for some lower triangular matrix $\mathbf{F}_t$. The TVD rank, $r(t)$, provides a measure of the degree of structure present, with lower rank indicating stronger structure. Thus, if $r(t)$ is close to m, there is little point in pursuing the displacement framework. Combining (29) with (26) implies that

$$\nabla\mathbf{B}_t = \hat{\mathbf{Z}}_t\hat{\mathbf{Z}}_t^T - \check{\mathbf{Z}}_t\check{\mathbf{Z}}_t^T, \tag{30}$$

where $\Delta = 1$, $\mathbf{F}_t = \mathbf{I}$, and the $m \times r(t)$ *generator* matrices $\hat{\mathbf{Z}}_t$ and $\check{\mathbf{Z}}_t$ are each used in turn to update $\mathbf{B}_t$. Further, it can be seen that $r(t) = 2$ for both update and downdating generator matrices. We note that the positive-definite nature of $\hat{\mathbf{R}}_y(t)$ guarantees the existence of a unique lower triangular Cholesky factor, $\mathbf{L}_t$, such that $\mathbf{B}_t = \mathbf{L}_t\mathbf{L}_t^T$, which, exploiting (30), can be expressed in two stages as [11]

$$\begin{bmatrix} \hat{\mathbf{L}}_t & \mathbf{0} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{L}}_t^T \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{L}_{t-1} & \hat{\mathbf{Z}}_t \end{bmatrix} \begin{bmatrix} \mathbf{I}_n & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_m \end{bmatrix} \begin{bmatrix} \mathbf{L}_{t-1}^T \\ \hat{\mathbf{Z}}_t^T \end{bmatrix}$$

where $\hat{\mathbf{L}}_t$ represents the updated *only* Cholesky factor which

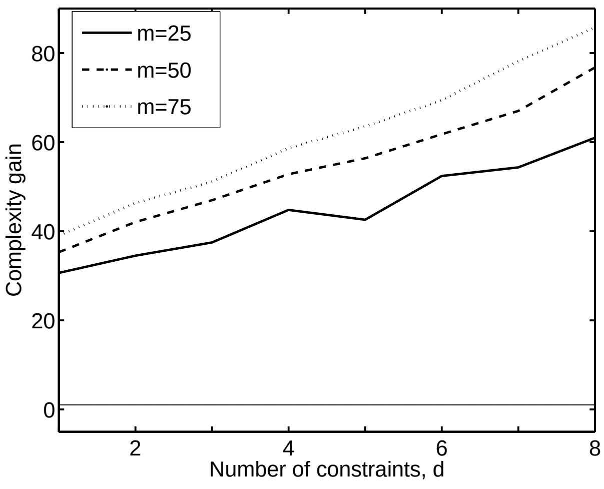

Figure 3: Average complexity gain as a function of the number of constraints, d. Here, the data size is $N = 3m$ and $P = 512$.

is then followed by the downdating process

$$\begin{bmatrix} \mathbf{L}_t & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{L}_t^T \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{L}}_t & \check{\mathbf{Z}}_t \end{bmatrix} \begin{bmatrix} \mathbf{I}_n & \mathbf{0} \\ \mathbf{0} & -\mathbf{I}_m \end{bmatrix} \begin{bmatrix} \hat{\mathbf{L}}_t^T \\ \check{\mathbf{Z}}_t^T \end{bmatrix},$$

in order to effect both the up- and downdating (i.e., to form the sliding window) of the Cholesky factors of the compact form of the FBA covariance estimate, $\mathbf{B}_t$. Hence, it follows that there exists *two* $[\mathbf{I}_n \oplus \mathbf{I}_m]$-unitary *rotation* matrices[3], $\hat{\mathbf{\Gamma}}_t$ and $\check{\mathbf{\Gamma}}_t$, such that [11]

$$\begin{bmatrix} \hat{\mathbf{L}}_t & \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{L}_{t-1} & \hat{\mathbf{Z}}_t \end{bmatrix} \hat{\mathbf{\Gamma}}_t \tag{31}$$

and subsequently

$$\begin{bmatrix} \mathbf{L}_t & \mathbf{0} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{L}}_t & \check{\mathbf{Z}}_t \end{bmatrix} \check{\mathbf{\Gamma}}_t. \tag{32}$$

Note that $\hat{\mathbf{\Gamma}}_t$ and $\check{\mathbf{\Gamma}}_t$ have the effect of rotating the updating generator matrices, $\hat{\mathbf{Z}}_t$ and $\check{\mathbf{Z}}_t$, onto the expressions $\mathbf{L}_{t-1}$ and $\hat{\mathbf{L}}_t$ respectively to produce the up- and down-dated Cholesky factor $\mathbf{L}_t$ and block zero entries in the left-hand sides of both (31) and (32). Both the rotational transforms $\hat{\mathbf{\Gamma}}_t$ and $\check{\mathbf{\Gamma}}_t$ are typically implemented as a sequence of elementary transforms, having the general form, $\mathbf{\Gamma}_t = \mathbf{\Gamma}_t^1 \mathbf{\Gamma}_t^2 \cdots \mathbf{\Gamma}_t^m$, where $\mathbf{\Gamma}_t^k$ annihilates the kth row of a given generator matrix. The rotation matrices $\hat{\mathbf{\Gamma}}_t$ and $\check{\mathbf{\Gamma}}_t$ can be formed in numerous different ways. Generally, however, Givens rotations are used for updating and Householder rotations for down-dating. One should note that in practice it is more efficient for each column of the Cholesky factor to be concatenated with the generator matrix to make an $\{m-k+1\} \times \{r(t)+1\}$ matrix, and as each vector is updated, this process is repeated whilst each row of the generator matrix is annihilated until all the column vectors of the new Cholesky factor are produced. Thus, the appropriate rotation matrices, $\hat{\mathbf{\Gamma}}_t$ and $\check{\mathbf{\Gamma}}_t$ are 3×3 matrices of

[3]Here, a $\mathbf{J}$-unitary matrix Θ is defined as any matrix Θ such that $\Theta\mathbf{J}\Theta^* = \mathbf{J}$. Further, $\mathbf{a} \oplus \mathbf{b}$ denotes a matrix with the sub-matrices $\mathbf{a}$ $\{n \times n\}$ and $\mathbf{b}$ $\{m \times m\}$ concatenated to produce a matrix of size $\{(m+n) \times (m+n)\}$.

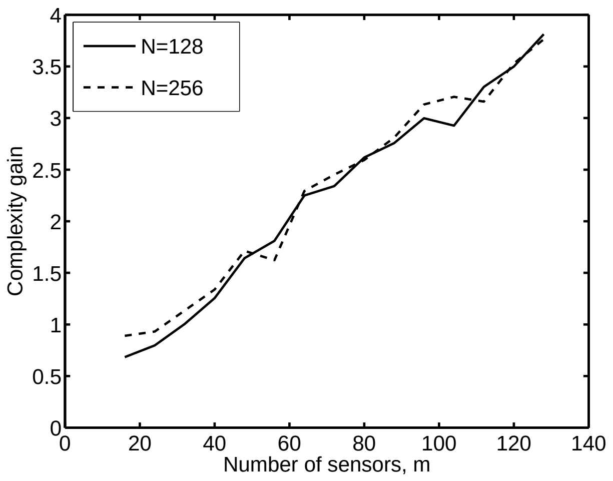

Figure 4: Average complexity gain for time-updating as a function of the number of sensors.

the form

$$\hat{\mathbf{\Gamma}}_t^k = \begin{bmatrix} \frac{l(k,k)}{\beta_k} & \frac{\hat{z}(k,1)}{\alpha_k} & \frac{l(k,k)\hat{z}(k,2)}{\alpha_k\beta_k} \\ \frac{\hat{z}(k,1)}{\beta_k} & -\frac{l(k,k)}{\alpha_k} & \frac{\hat{z}(k,1)\hat{z}(k,2)}{\alpha_k\beta_k} \\ \frac{\hat{z}(k,2)}{\beta_k} & 0 & -\frac{\alpha_k}{\beta_k} \end{bmatrix} \tag{33}$$

$$\check{\mathbf{\Gamma}}_t^k = \begin{bmatrix} \frac{\hat{l}(k,k)}{\delta_k} & -\frac{\check{z}(k,1)}{\gamma_k} & -\frac{\hat{l}(k,k)\check{z}(k,2)}{\gamma_k\delta_k} \\ -\frac{\check{z}(k,1)}{\delta_k} & \frac{\hat{l}(k,k)}{\gamma_k} & \frac{\check{z}(k,1)\check{z}(k,2)}{\gamma_k\delta_k} \\ -\frac{\check{z}(k,2)}{\delta_k} & 0 & \frac{\gamma_k}{\delta_k} \end{bmatrix} \tag{34}$$

where $l(k,\ell)$, $\hat{l}(k,\ell)$, $\hat{z}(k,\ell)$ and $\check{z}(k,\ell)$ denote the (k,ℓ)th element of $\mathbf{L}$, $\hat{\mathbf{L}}$, $\hat{\mathbf{Z}}_t$ and $\check{\mathbf{Z}}_t$, respectively, and

$$\alpha_k = \sqrt{|l(k,k)|^2 + |\hat{z}(k,1)|^2} \quad \beta_k = \sqrt{|\alpha_k|^2 + |\hat{z}(k,2)|^2}$$

$$\gamma_k = \sqrt{|\hat{l}(k,k)|^2 - |\check{z}(k,1)|^2} \quad \delta_k = \sqrt{|\gamma_k|^2 - |\check{z}(k,2)|^2}$$

As shown in [11], this procedure can easily be extended to also yield the *inverse* Cholesky factor; this is achieved by augmenting (31) and (32) by appending the inverse Cholesky factors according to [11]. By applying $\hat{\mathbf{\Gamma}}_t$ (and subsequently $\check{\mathbf{\Gamma}}_t$), we thus find an efficient time-updating of the inverse Cholesky factor also, yielding one column vector per iteration. Using the updated inverse Cholesky factor, we form the time-updated LCMV estimate in (18), using

$$\hat{\mathbf{R}}_y^{-1}(t) = \left(\mathbf{K}\mathbf{L}_{t-1}^{-1}\right)\left(\mathbf{K}\mathbf{L}_{t-1}^{-1}\right)^*, \tag{35}$$

replacing (12) and (13) with $v_\omega(t) = \left(\tilde{\mathbf{K}}\mathbf{L}_{t-1}^{-1}\right)\mathbf{L}_{t-1}^{-*}\mathbf{a}_{\omega_1}$ and $\mathbf{G}_{\tilde{\omega}}(t) = \left(\tilde{\mathbf{K}}\mathbf{L}_{t-1}^{-1}\right)\left(\tilde{\mathbf{K}}\mathbf{L}_{t-1}^{-1}\right)^*$, where $\tilde{\mathbf{K}} = \mathbf{A}_{\tilde{\omega}}^*\mathbf{K}$; the other components are evaluated accordingly. The time-updated version of (18) is then obtained using the FFT technique described above. Finally, we note that the above proposed sliding-window time-updating can, if desired, be reformulated to form an updating employing an exponential forgetting factor (which then obviates the down-dating and instead removes the influence of old data components by multiplication of the Cholesky factor, $\mathbf{L}_{t-1}^{-1}$, by a suitable factor close to unity).

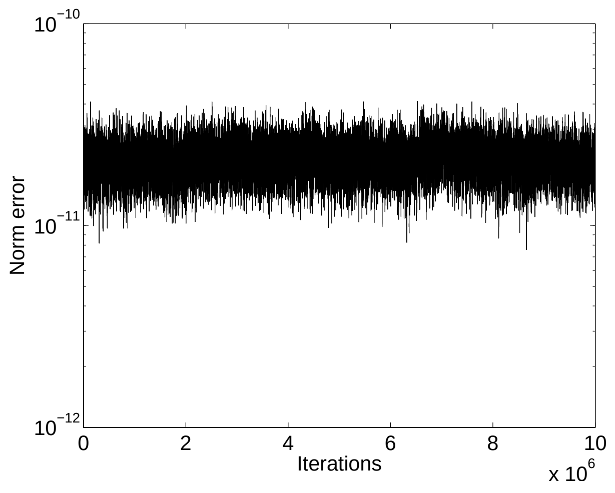

Figure 5: Error propagation of the time-updating method for 10^7 iterations. Here, $m = 32$.

5. NUMERICAL EXAMPLES

In this section, we briefly examine the computational gain of the proposed methods, initially examining the non-recursive implementation introduced in Section 3. Simulation data has been generated to contain a signal of interest impinging on the array from broadside, while $d-1$ known jamming sources, each being ten times stronger than the signal of interest, are evenly spread over the angles $[20^\circ, 50^\circ]$. The measured signal is corrupted by an additive white circularly symmetric Gaussian noise, and the constraint vector, $\mathbf{f}$, has been selected to damp the jamming signals with a factor 0.01. Figure 1 illustrates the complexity gain factor (estimated as the average execution time of 500 iterations using Matlab) of evaluating $\varphi_y(\omega_1)$ using (18), as compared to using the traditional approach in (6), as a function of the array size, m. The data size is selected as $N = 3m$. In the figure, the size of the frequency grid, P, is selected as the next power of two larger than N. Figures 2 and 3 illustrate the computational gain, for varying array sizes, as a function of the size of the frequency grid and the number of constraints, respectively. Here, $P = 512$. In these figures, the cost of evaluating $\hat{\mathbf{R}}_y^{-1}$ has been omitted as both the proposed and the brute-force implementations require this evaluation. As is clear from these Figures, the proposed non-recursive implementation offers a significant computational gain, especially for larger arrays. We stress that the implementation in (18) will yield the same spatial spectral estimate as the brute-force implementation in (6). Proceeding to examine the proposed time-updating, Figure 4 shows the complexity gain of the proposed algorithm as compared to reevaluating the spatial spectrum using the presented non-recursive implementation (including the cost for evaluating $\hat{\mathbf{R}}_y^{-1}$), as a function of the array size, m. As seen from the figure, the proposed updating offers a substantial complexity reduction, especially for larger arrays, even compared to using the efficient implementation to evaluate the spatial spectrum. Finally, Figure 5 illustrates the (norm) error propagation of the sliding-window time-updating as compared to the exact spectrum, clearly indicating the robust nature of the algorithm.

REFERENCES

[1] H. Krim and M. Viberg, “Two Decades of Array Signal Processing Research,” *IEEE Signal Processing Magazine*, pp. 67–94, July 1996.

[2] H. L. Van Trees, *Detection, Estimation, and Modulation Theory, Part IV, Optimum Array Processing,* John Wiley and Sons, Inc., 2002.

[3] P. Stoica and R. Moses, *Spectral Analysis of Signals*, Prentice Hall, Upper Saddle River, N.J., 2005.

[4] O. L. Frost, III, “An Algorithm for Linearly Constrained Adaptive Array Processing,” *Proc. IEEE*, vol. 60, no. 8, pp. 926–935, August 1972.

[5] T.-C. Liu and B. van Veen, “A Modular Structure for Implementation of Linearly Constraint Minimum Variance Beamformers,” *IEEE Trans. Signal Processing*, vol. 39, no. 10, pp. 2343–2346, October 1991.

[6] L. S. Resende, J. M. T. Romano, and M. G. Bellanger, “A Fast Least-Squares Algorithm for Linearly Constrained Adaptive Filtering,” *IEEE Trans. Signal Processing*, vol. 44, no. 5, pp. 1168–1174, October 1996.

[7] J. A. Apolinário, Jr., M. L. R. de Campos, and C. P. Bernal O., “The Constrained Conjugate Gradient Algorithm,” *IEEE Signal Processing Letters*, vol. 7, no. 12, pp. 351–354, December 2000.

[8] S. Werner, J. A. Apolinário, Jr., and M. L. R. de Campos, “On the Equivalence of RLS Implementations of LCMV and GSC Processors,” *IEEE Signal Processing Letters*, vol. 10, no. 12, pp. 356–359, December 2003.

[9] W. Liu, C. L. Koh, and S. Weiss, ““Constrained Adaptive Broadband Beamforming Algorithm in Frequency Domain”,” in *IEEE Sensor Array and Multichannel Signal Processing Workshop*, 2004, pp. 94–98.

[10] T. Kailath and A. H. Sayed, *Fast Reliable Algorithms for Matrices with Structure*, SIAM, Philadelphia, USA, 1999.

[11] A. H. Sayed, H. Lev-Ari, and T. Kailath, “Time-variant Displacement Structure and Triangular Arrays,” *IEEE Trans. Signal Processing*, vol. 42, pp. 1052–1062, May 1994.

[12] M. Jansson and P. Stoica, “Forward-Only and Forward-Backward Sample Covariances – A Comparative Study,” *Signal Processing*, vol. 77, no. 3, pp. 235–245, 1999.

[13] H. Li, P. Stoica, and J. Li, “Capon Estimation of Covariance Sequences,” *Circuits, Systems, and Signal Processing*, vol. 17, no. 1, pp. 29–49, 1998.

[14] T. T. Wang, “The Segmented Chirp Z-transform and its Application in Spectrum Analysis,” *IEEE Trans. Instrum. Meas.*, vol. 39, no. 2, pp. 318–323, April 1990.

[15] H. V. Sorensen and C. S. Burrus, “Efficient Computation of the DFT with Only a Subset of Input or Output Points,” *IEEE Trans. Signal Processing*, vol. 41, no. 3, pp. 1184–1200, March 1993.

[16] D. A. Linebarger, R. D. DeGroat, and E. M. Dowling, “Efficient Direction-Finding Methods Employing Forward/Backward Averaging,” *IEEE Trans. Signal Processing*, vol. 42, pp. 2136–2145, August 1994.