

# ADAPTIVE RE-WEIGHTING HOMOTOPY FOR SPARSE BEAMFORMING

Fernando G. A. Neto, Vítor H. Nascimento

Escola Politécnica, University of São Paulo  
São Paulo, Brazil  
fganeto@lps.usp.br, vitor@lps.usp.br

Yuriy V. Zakharov, Rodrigo C. de Lamare

University of York / CETUC/PUC-Rio  
York, UK / Rio de Janeiro, Brazil  
yury.zakharov@york.ac.uk, delamare@cetuc.puc-rio.br

## ABSTRACT

In this paper, a complex adaptive re-weighting algorithm based on the homotopy technique is developed and used for beamforming. A multi-candidate scheme is also proposed and incorporated into the adaptive re-weighting homotopy algorithm to choose the regularization factor and improve the signal-to-interference plus noise (SINR) performance. The proposed algorithm is used to minimize the degradation caused by sparsity in arrays with faulty sensors, or when the required degrees of freedom to suppress interference is significantly less than the number of sensors. Simulations illustrate the algorithm's performance.

**Index Terms**— Multi-candidate re-weighting homotopy, beamforming, adaptive algorithms

## 1. INTRODUCTION

Adaptive beamformers are used in sensor arrays to enhance the reception of a signal of interest and suppress interference [1]. They implement techniques such as the minimum variance distortionless response (MVDR) beamformer [1] using data collected from sensors, since the second-order statistics required to compute the MVDR, in general, are not available and must be estimated.

Although many fields apply adaptive techniques to beamforming, such as radar, sonar and wireless communications [1], large arrays are difficult to implement with traditional approaches. Techniques such as the least mean-square (LMS), conjugate gradient (CG) and recursive least-squares (RLS) algorithms [1, 2] have their convergence and tracking performances affected either by the size or the eigenvalue spread of the input correlation matrix [2]. Therefore, beamformers with many parameters may require many snapshots to converge, which are incompatible with the requirements of some applications (for instance, space-time adaptive processing for airborne radar [3, 4]). Previous works report techniques that incorporate information about sparsity on the data, or knowledge that only a reduced number of degrees of freedom is required to suppress interference in a sensor array, to enhance their performance. Reduced-rank algorithms [5–7] and sparsity-aware techniques [3, 4] play an important role

in this field. Reduced-rank methods are applied to problems where a reduced number of effective features is required to retain most of the intrinsic information content of the input data. For beamforming, they benefit from the low-rank structure of the interference correlation matrix, reducing the number of parameters to compute. Sparsity-aware algorithms use some form of regularization to include prior knowledge that the signal of interest is sparse. The regularization allows the computation of sparse solutions, providing a considerable reduction of the number of coefficients to adapt in sparse beamformers. As a result, many calculations are avoided and the convergence is accelerated, as reported by [3, 4].

Many sparsity-aware algorithms have been proposed recently (for instance, [8–11]). Among them, the homotopy algorithm [9] is an  $\ell_1$ -norm regularized algorithm used for many purposes, such as recovering of sparse signals from noisy measurements [10] and channel estimation [11]. Motivated by the results of the homotopy algorithms, in this paper we use the complex homotopy (CH) [11] and the adaptive re-weighting homotopy (ARH) [10] techniques to develop two new algorithms for sparse beamforming. We develop the complex ARH (C-ARH) and the multi-candidate C-ARH (MC-C-ARH) algorithms, and show that they achieve better SINR performance than existing adaptive beamformers. The proposed algorithms are useful for interference suppression when the required degrees of freedom is much less than the number of sensors and in the presence of faulty sensors.

**Notation:** Lower case is used for scalar quantities (e.g.:  $a$ ) and bold lower case for column vectors (e.g.:  $\mathbf{b}$ ). Bold capital letters represent matrices (e.g.:  $\mathbf{A}$ ).  $b_k$  stands for the  $k$ -th element of  $\mathbf{b}$ , while  $\mathbf{a}_k$  is the  $k$ -th column of  $\mathbf{A}$ .  $(\cdot)^T$  stands for transposition, while  $(\cdot)^H$  is the Hermitian of a matrix or vector. The operations  $\text{Im}\{\cdot\}$  and  $\text{Re}\{\cdot\}$  take only the imaginary and real parts of a complex number, and  $\text{diag}(\cdot)$  defines a diagonal matrix.  $\|\cdot\|_p$  is the  $\ell_p$ -norm, and  $E\{\cdot\}$  is the expectation operator.  $\mathbf{I}_K$  represents a  $K \times K$  identity matrix, and  $\mathbf{0}_{K \times M}$  represents a  $K \times M$  matrix of zeros.

## 2. SYSTEM MODEL AND PROBLEM STATEMENT

Consider a uniform linear array (ULA) with  $M$  sensors, and assume  $K$  signals amongst which one has a known desired direction of arrival  $\theta_d$  and  $K - 1$  interferences. Define the

This work is funded by CNPq and York-FAPESP grants.

$M \times K$  matrix  $\mathbf{B}$ , where each column  $\mathbf{b}_k$  corresponds to a steering vector [1] as given by

$$\mathbf{b}_k = [1 \ e^{-j\pi\sin(\theta_k)} \ \dots \ e^{-j\pi(M-1)\sin(\theta_k)}]^T, 1 \leq k \leq K.$$

At snapshot  $n$ , the sensor array data is modeled as

$$\mathbf{u}(n) = \mathbf{B}\mathbf{s}(n) + \boldsymbol{\eta}(n), \quad (1)$$

where  $\mathbf{s}(n)$  contains signals produced by the sources, and  $\boldsymbol{\eta}(n)$  corresponds to zero-mean independent and identically distributed (i.i.d.) Gaussian noise with variance  $\sigma_\eta^2$ . Define  $\theta_d = \theta_1$ . The MVDR coefficients [1] are given by

$$\mathbf{h}_{\text{MVDR}} = \mathbf{x}_{\text{MVDR}}/\mathbf{b}_d^H \mathbf{x}_{\text{MVDR}}, \quad (2)$$

where  $\mathbf{x}_{\text{MVDR}}$  is the solution to  $\mathbf{R}_t \mathbf{x}_{\text{MVDR}} = \mathbf{b}_d$ .  $\mathbf{R}_t$  is the  $M \times M$  correlation matrix [1] and is given by

$$\mathbf{R}_t = E\{\mathbf{u}(n)\mathbf{u}^H(n)\} = \mathbf{R}_d + \mathbf{R}_I + \mathbf{R}_\eta, \quad (3)$$

where we split the contribution of  $\mathbf{u}(n)$  in (1) into the desired direction (subscript d) and the interference (subscript I), i.e.,

$$\mathbf{u}(n) = \mathbf{b}_d s_d(n) + \mathbf{B}_I \mathbf{s}_I(n) + \boldsymbol{\eta}(n), \quad (4)$$

to compute  $\mathbf{R}_d = \sigma_d^2 \mathbf{b}_d \mathbf{b}_d^H$ ,  $\mathbf{R}_I = \mathbf{B}_I E\{\mathbf{s}_I(n)\mathbf{s}_I^H(n)\} \mathbf{B}_I^H$ , and  $\mathbf{R}_\eta = \sigma_\eta^2 \mathbf{I}$ .  $\sigma_d^2$  is the variance of the signal of interest.

Note that the computation of the beamformer requires the solution of a linear system of equations. When the number of sources is much less than the number of sensors,  $\mathbf{R}_t$  can become ill-conditioned, requiring some form of regularization to compute  $\mathbf{x}_{\text{MVDR}}$ . In addition, when the measurements of some sensors are not available, a sparsity-aware algorithm can be applied to compute a low-cost solution.

## 2.1. Small number of interference sources

Consider that  $K \ll M$ , and define  $\mathbf{R}_{\text{dl}} = \mathbf{R}_d + \mathbf{R}_I$ . Assume that the interference sources are uncorrelated among each other, such that  $\text{rank}(E\{\mathbf{s}(n)\mathbf{s}^H(n)\}) = K$ , and that  $\text{rank}(\mathbf{B}) = K$ . Using properties of the rank of matrices [12], one can show that<sup>1</sup>

$$\text{rank}(\mathbf{R}_{\text{dl}}) = \text{rank}(\mathbf{B}E\{\mathbf{s}\mathbf{s}^H\}\mathbf{B}^H) = K. \quad (5)$$

Since  $\text{rank}(\mathbf{R}_{\text{dl}}) = K$ , the eigenvalue decomposition of  $\mathbf{R}_{\text{dl}}$  is given by  $\mathbf{R}_{\text{dl}} = \mathbf{V}\mathbf{D}\mathbf{V}^H$ , where  $\mathbf{V}$  is the eigenvector matrix, and  $\mathbf{D}$  is a diagonal matrix with  $K$  non-zero entries. Using the eigenvalue decomposition of  $\mathbf{R}_{\text{dl}}$  and (3), we obtain

$$\mathbf{R}_t = \mathbf{V} \begin{bmatrix} \mathbf{D}_0 + \sigma_\eta^2 \mathbf{I}_K & \mathbf{0}_{K \times M-K} \\ \mathbf{0}_{M-K \times K} & \sigma_\eta^2 \mathbf{I}_{M-K} \end{bmatrix} \mathbf{V}^H, \quad (6)$$

where  $\mathbf{D}_0$  contains the  $K$  non-zero eigenvalues of  $\mathbf{D}$ . From eq. (6), the condition number [12] of  $\mathbf{R}_t$  is given by

$$\kappa(\mathbf{R}_t) = (d_{\text{MAX}} + \sigma_\eta^2)/\sigma_\eta^2, \quad (7)$$

where  $d_{\text{MAX}}$  stands for the maximum eigenvalue of  $\mathbf{D}_0$ . Equation (7) shows that  $\mathbf{R}_t$  becomes ill-conditioned if the noise power is much smaller than  $d_{\text{MAX}}$ . In this case, a regularization can be added to  $\mathbf{R}_t$  to improve the computation of  $\mathbf{x}_{\text{MVDR}}$ . We show that our sparsity-aware algorithms can regularize  $\mathbf{R}_t$  to improve the SINR performance.

<sup>1</sup>We drop here time coefficients to simplify notation.

## 2.2. Faulty sensors in the array

When a sensor  $j$  is not working properly, the measurements captured by  $j$  must be discarded. This information is incorporated into the model with a modification of eq. (4), where we introduce  $\mathbf{E}$ , i.e.,

$$\mathbf{u}(n) = (\mathbf{E}\mathbf{b}_d) \mathbf{s}_d(n) + (\mathbf{E}\mathbf{B}_I) \mathbf{s}_I(n) + \boldsymbol{\eta}(n) \quad (8)$$

where  $\mathbf{E}$  is an  $M \times M$  diagonal matrix with entries equal to 0 for sensors that do not contribute for beamforming, and 1 otherwise. When  $e_{jj} = 0$ , we zero the  $j$ -th element of all steering vectors, which eliminates the signal produced by sensor  $j$ . With this approach, we assume that the removed sensors do not receive a signal, but they still contribute with noise, and the noise has the same power as before. Using (8) to compute the correlation matrix, we obtain

$$\mathbf{R}_t = \mathbf{E}\mathbf{R}_{\text{dl}}\mathbf{E} + \mathbf{R}_\eta. \quad (9)$$

Assuming that the array has  $L < M$  faulty sensors, and that these sensors are grouped such that the last  $L$  diagonal elements of  $\mathbf{E}$  are zero,  $\mathbf{R}_t$  is given by

$$\mathbf{R}_t = \begin{bmatrix} (\tilde{\mathbf{R}}_{\text{dl}} + \sigma_\eta^2 \mathbf{I}_{M-L}) & \mathbf{0}_{(M-L) \times L} \\ \mathbf{0}_{L \times (M-L)} & \sigma_\eta^2 \mathbf{I}_L \end{bmatrix}, \quad (10)$$

where  $\tilde{\mathbf{R}}_{\text{dl}}$  is a square matrix obtained from the first  $M - L$  columns and the first  $M - L$  rows of  $\mathbf{R}_{\text{dl}}$ . Ideally, if we know matrix  $\mathbf{E}$ , we solve the linear system of equations given by

$$\mathbf{R}_t \mathbf{x}_{\text{MVDR}} = \mathbf{E}\mathbf{b}_d, \quad (11)$$

where only the first  $M - L$  entries of  $\mathbf{E}\mathbf{b}_d$  are non-zero. From (10) and (11), we see that the solution is sparse, and only  $M - L$  entries of  $\mathbf{x}_{\text{MVDR}}$  need to be computed. In general,  $\mathbf{E}$  is unknown and has to be estimated beforehand. In this paper, we use the energy detection method [13] to estimate the faulty sensors. Note that with  $\mathbf{E}$ , one can directly apply size-reduction of (11) to compute  $\mathbf{x}_{\text{MVDR}}$ . However, in our approach we use  $\mathbf{E}$  to explicitly exploit the sparsity in (11) with  $\ell_1$ -norm regularized algorithms. We show that this approach outperforms reduced-rank algorithms in sparse-beamforming scenarios, improving the SINR performance.

## 3. COMPLEX HOMOTOPY ALGORITHMS

The CH algorithm was first proposed in [11], as an extension of real-valued homotopy [9] to the complex field. In both cases, the technique solves the optimization problem

$$\text{minimize}_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{y}\|_2^2/2 + w\|\mathbf{x}\|_1, \quad (12)$$

where  $\mathbf{A}$  is  $M \times N$ ,  $\mathbf{y}$  is  $M \times 1$ ,  $\mathbf{x}$  is an  $N \times 1$  vector that minimizes (12), and  $w$  is a regularization parameter. The technique iteratively solves (12) based on a support set that is updated at every iteration. For each homotopy iteration,  $\mathbf{x}$  must satisfy the following optimality conditions [11]

$$\begin{aligned} \mathbf{a}_i^H(\mathbf{Ax} - \mathbf{y}) &= -wz_i, \text{ for all } i \in \Gamma \\ |\mathbf{a}_i^H(\mathbf{Ax} - \mathbf{y})| &< w, \text{ for all } i \in \Gamma_C \end{aligned}, \quad (13)$$

where  $\Gamma$  is the support set,  $\Gamma_C$  is the complement of  $\Gamma$ , and  $\mathbf{z}$  denotes a vector obtained by applying the sign function elementwise on  $\mathbf{x}$ . The algorithm starts with  $\mathbf{x} = \mathbf{0}$  and computes  $\max_i(|\mathbf{a}_i^H(\mathbf{y} - \mathbf{Ax})|)$  – the maximum initial correlation [11] – to initialize  $w$  and to define the first element of  $\Gamma$ .

At each iteration, one element is added or removed from  $\Gamma$ , which moves  $w$  to  $w - \epsilon$ . To maintain the conditions in (13),  $\mathbf{x}$  moves towards  $\mathbf{x} + \epsilon \partial \mathbf{x}$ , i.e.,<sup>2</sup>

$$\begin{aligned} \mathbf{A}_\Gamma^H(\mathbf{Ax} - \mathbf{y}) + \epsilon \mathbf{A}_\Gamma^H \mathbf{A} \partial \mathbf{x} &= -w \mathbf{z}_\Gamma + \epsilon \mathbf{z}_\Gamma \\ |\mathbf{a}_i^H(\mathbf{Ax} - \mathbf{y}) + \epsilon \mathbf{a}_i^H \mathbf{A} \partial \mathbf{x}| &< w - \epsilon, i \in \Gamma_C \end{aligned} \quad (14)$$

From (14), we arrive at a set of linear equations used to compute  $\partial \mathbf{x}$ . Using  $\partial \mathbf{x}$ , we find  $\epsilon$ , and update  $w$  and  $\Gamma$  in the next step. These steps are repeated until  $w = 0$  or some stopping criterion is met. References [9, 11] present a detailed description of the algorithm.

### 3.1. Proposed C-ARH algorithm

Based on the CH algorithm, in [10] the ARH algorithm was proposed to solve the  $\ell_1$ -weighted optimization problem

$$\text{minimize}_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{y}\|_2^2 / 2 + \sum_{i=1}^M w_i |x_i|, \quad (15)$$

where  $w_i > 0$  are the entries of the weight vector  $\mathbf{w}$ . The motivation to solve (15) instead of (12) is the possibility to adjust different weights to penalize the solution coefficients, which can be used to enhance the level of sparsity of the solution and improve the performance [10].

The ARH algorithm applies a re-weighting approach to quickly compute  $\mathbf{x}$ , when  $\mathbf{w}$  is replaced by a re-weighting vector  $\tilde{\mathbf{w}}$ . The idea behind the algorithm is that the solution moves to  $\mathbf{x} + \delta \partial \mathbf{x}$  when  $\mathbf{w}$  moves towards  $\tilde{\mathbf{w}}$  along a straight line  $(1 - \delta)\mathbf{w} + \delta \tilde{\mathbf{w}}$ , for  $\delta \in [0, 1]$ . The technique was developed for real-valued signals. Simulation results have shown that ARH yields better performance in terms of computational cost and reconstruction accuracy than  $\ell_1$ -based solvers (see [10]) used for recovering sparse signals from noisy measurements. These results motivated us to develop the C-ARH and the MC-C-ARH algorithms for sparse beamforming.

Note that in [10] the ARH algorithm is presented with  $\delta \in [0, 1]$ . For convenience, when we present the C-ARH, we assume  $\delta \in [0, \tilde{\delta}]$ . In Section 3.2 we use different values of  $\tilde{\delta}$  to construct a diverse set of possible solutions at each homotopy step, thus achieving higher performance.

For eq. (15), the optimality conditions are defined as

$$\begin{aligned} \mathbf{A}_\Gamma^H(\mathbf{Ax} - \mathbf{y}) &= -\mathbf{W} \mathbf{z}_\Gamma \\ |\mathbf{a}_i^H(\mathbf{Ax} - \mathbf{y})| &< w_i z_i \in \Gamma_C \end{aligned} \quad (16)$$

where  $\mathbf{W} = \text{diag}(\mathbf{w}_\Gamma)$ , and  $\mathbf{z}$  contains the signs of the elements of  $\mathbf{x}$  after the entries are applied to the sign function. When  $\mathbf{w}$  moves to  $(1 - \delta)\mathbf{w} + \delta \tilde{\mathbf{w}}$ , (16) changes to

$$\begin{aligned} \mathbf{A}_\Gamma^H(\mathbf{Ax} - \mathbf{y}) + \delta \mathbf{A}_\Gamma^H \mathbf{A} \partial \mathbf{x} &= -\mathbf{W} \mathbf{z}_\Gamma + \delta (\mathbf{W} - \tilde{\mathbf{W}}) \mathbf{z}_\Gamma, \\ |\mathbf{a}_i^H(\mathbf{Ax} - \mathbf{y}) + \delta \mathbf{a}_i^H \mathbf{A} \partial \mathbf{x}| &< w_i + \delta (\tilde{w}_i - w_i), i \in \Gamma_C \end{aligned} \quad (17)$$

and  $\tilde{\mathbf{W}} = \text{diag}(\tilde{\mathbf{w}}_\Gamma)$ . In order to obtain  $\partial \mathbf{x}$ , compute

$$\partial \mathbf{x} = \begin{cases} (\mathbf{A}_\Gamma^H \mathbf{A}_\Gamma)^{-1} (\mathbf{W} - \tilde{\mathbf{W}}) \mathbf{z}_\Gamma & , \\ 0, & \text{in } \Gamma_C \end{cases} \quad (18)$$

where  $z_i = \mathbf{a}_i^H(\mathbf{Ax} - \mathbf{y})/w_i, i \in \Gamma$ . To update  $\Gamma$ , we have to check if a breakpoint occurred. A breakpoint occurs in two situations: when one element of  $\mathbf{x} \in \Gamma$  shrinks to zero,

or when one inequality becomes an equality in (17). When one element shrinks to zero, the direction has to be changed, and this element must be removed from  $\Gamma$ . Recall that  $\mathbf{x}$  is updated as  $\mathbf{x} = \mathbf{x} + \epsilon \partial \mathbf{x}$ . An element of  $\mathbf{x}$  crosses zero when

$$\delta = -x_i/\partial x_i, \text{ for some } i \in \Gamma. \quad (19)$$

Define  $\mathbf{x}_R = \text{Re}\{\mathbf{x}\}$ ,  $\mathbf{x}_I = \text{Im}\{\mathbf{x}\}$ ,  $\mathbf{d}_R = \text{Re}\{\partial \mathbf{x}\}$  and  $\mathbf{d}_I = \text{Im}\{\partial \mathbf{x}\}$ , and recall that  $\delta$  must be a real number in the interval  $[0, \tilde{\delta}]$ . The parameter  $\delta$  is a real value in (19), only if

$$x_{R_i}/d_{R_i} = x_{I_i}/d_{I_i}, i \in \Gamma. \quad (20)$$

We remove an element  $\gamma^-$  from  $\Gamma$  if (20) holds for some  $i$ , and if (19) is in  $[0, \tilde{\delta}]$ , for the same breakpoint. If two or more breakpoints fulfil the restrictions, we choose the smallest one to be removed. In Table 1, we compute

$g = \min_+(-x_{R_i}/d_{R_i})$ , for all  $x_{R_i}/d_{R_i} = x_{I_i}/d_{I_i}, i \in \Gamma$  to define the breakpoint that must be removed, where  $\min_+(\cdot)$  returns the smallest positive value in the argument. When  $g$  is empty, no term is removed, and C-ARH chooses an element  $\gamma^+$  that must be added to  $\Gamma$ . In this case,  $\gamma^+$  is chosen by

$$\gamma^+ = \arg \max_{i \in \Gamma_C} |\mathbf{a}_i^H(\mathbf{Ax} - \mathbf{y})|, \quad (21)$$

and  $\delta \leftarrow \tilde{\delta}$ . The last step is the update of  $w_i \in \Gamma_C$ , which is given by  $w_i \leftarrow \max_j |\mathbf{a}_j^H(\mathbf{Ax} - \mathbf{y})|$ , for all  $i \in \Gamma_C$ . The algorithm stops when the maximum  $w_i \in \Gamma$  is smaller than a pre-defined parameter  $\tau$ . Table 1 summarizes the algorithm.

**Re-weighting selection:** For this paper,  $\tilde{\mathbf{w}}$  is given by

$$\tilde{w}_i = \min(\zeta, \zeta/\beta|x_i|), \text{ for all } i \in \Gamma, \quad (22)$$

where  $\zeta = 2\sigma_\eta^2$  and  $\beta = N\|\mathbf{x}\|_2^2/\|\mathbf{x}\|_1^2$ . We emphasize that (22) was used since it provides the best simulation results for our scenario. Other re-weightings can also be applied.

**Computational cost:** Using pre-computation of  $\mathbf{A}^H \mathbf{y}$  and  $\mathbf{A}^H \mathbf{A}$ , and defining  $|\Gamma|$  as the cardinality of  $\Gamma$ , the maximum computational cost of the algorithm corresponds to  $4|\Gamma|N + 6|\Gamma| + N$  multiplications,  $4|\Gamma|N + 4|\Gamma|$  additions and  $3|\Gamma|$  divisions, plus the cost to solve a  $|\Gamma| \times |\Gamma|$  system of equations. The re-weighting presented in (22) uses an additional cost of  $3|\Gamma| + 2$  multiplications,  $3|\Gamma| - 2$  additions, 2 divisions and  $|\Gamma|$  square roots per iteration.

**Application for sparse beamforming:** In this case, C-ARH solves (15) using  $\mathbf{A} = \mathbf{R}$  ( $\mathbf{R}$  is an estimate of  $\mathbf{R}_t$ ),  $\mathbf{y} = \mathbf{b}_d$  and  $\tilde{\delta} = 1$ . The beamformer is computed with  $\mathbf{h} = \mathbf{x}/\mathbf{b}_d^H \mathbf{x}$ .

### 3.2. Proposed MC-C-ARH algorithm

In the C-ARH algorithm, when we choose  $\tilde{\delta} = 1$ ,  $\mathbf{w}$  moves towards  $\tilde{\mathbf{w}}$ . However, we can choose a different  $\tilde{\delta}$  and define a re-weighting that is a linear combination of  $\mathbf{w}$  and  $\tilde{\mathbf{w}}$ . Since in general there is no information about the weighting vector that generates the most accurate  $\mathbf{x}$ , the combination of the two weighting vectors can be a better option than only  $\tilde{\mathbf{w}}$ . In this context, we propose MC-C-ARH to exploit multiple choices.

We start with the definition of a set  $\Lambda$  of  $N_C$  candidates for  $\tilde{\delta}$ . For each  $\tilde{\delta} \in \Lambda$ , the algorithm computes the solution  $\mathbf{x}_i$ . A comparison criterion (e.g., mean-square error (MSE) or SINR) is used to define the best solution computed for each candidate. The candidate with the best figure of merit is selected and the algorithm returns the corresponding solution.

<sup>2</sup>Subscript  $\Gamma$  is used to identify quantities related to the support set.

**Table 1.** C-ARH algorithm

Input: $\mathbf{A}, \mathbf{y}, \tau, \delta$	Output: $\mathbf{x}, \mathbf{w}$
<b>Initialize:</b> $\mathbf{x} = \mathbf{0}, w_i \leftarrow \max_i  \mathbf{a}_i^H \mathbf{y} , \Gamma \leftarrow \arg \max_i  \mathbf{a}_i^H \mathbf{y} $	
<b>Repeat:</b>	
Select $\tilde{\mathbf{w}}$	
For all $i \in \Gamma$ , compute $z_i = \mathbf{a}_i^H (\mathbf{y} - \mathbf{A}\mathbf{x}) / w_i$	
Solve $(\mathbf{A}_\Gamma^H \mathbf{A}_\Gamma) \partial \mathbf{x} = (\mathbf{W} - \tilde{\mathbf{W}}) \mathbf{z}_\Gamma$	
Compute $\mathbf{x}_R = \mathcal{R}e\{\mathbf{x}_\Gamma\}, \mathbf{x}_I = \mathcal{I}m\{\mathbf{x}_\Gamma\}$	
$\mathbf{d}_R = \mathcal{R}e\{\partial \mathbf{x}_\Gamma\}$ and $\mathbf{d}_I = \mathcal{I}m\{\partial \mathbf{x}_\Gamma\}$	
$g = \min_+(-x_{R_i}/d_{R_i})$ , for all $x_{R_i}/d_{R_i} = x_{I_i}/d_{I_i}$	
$\delta = \min(g, \tilde{\delta})$	
$\mathbf{x} = \mathbf{x} + \delta \partial \mathbf{x}$	
$\mathbf{w}_\Gamma = \mathbf{w}_\Gamma + \delta(\tilde{\mathbf{w}}_\Gamma - \mathbf{w}_\Gamma)$	
<b>if</b> $\delta < \tilde{\delta}$	
$\Gamma \leftarrow \Gamma \setminus \gamma^-$ <span style="float: right;">▷ Remove an element from <math>\Gamma</math></span>	
<b>else</b>	
$\gamma^+ = \arg \max_{i \in \Gamma_C}  \mathbf{a}_i^H (\mathbf{A}\mathbf{x} - \mathbf{y}) $	
$\Gamma \leftarrow \Gamma \cup \gamma^+$ <span style="float: right;">▷ Add a new element to <math>\Gamma</math></span>	
<b>end</b>	
$w_i \leftarrow \max_j  \mathbf{a}_j^H (\mathbf{A}\mathbf{x} - \mathbf{y}) $ , for all $i \in \Gamma_C$	
<b>until</b> $\max_i(w_i) \leq \tau$	

In Table 2 we summarize the algorithm, which is proposed for sparse beamforming. For the  $i$ -th candidate, C-ARH solves eq. (15), where  $\mathbf{A} = \mathbf{R}$  ( $\mathbf{R}$  is an estimate of  $\mathbf{R}_t$ ),  $\mathbf{y} = \mathbf{b}_d$  and  $\tilde{\delta} = \lambda_i \in \Lambda$ . The beamformer  $\mathbf{h}_i$  is given by  $\mathbf{h}_i = \mathbf{x}_i / \mathbf{b}_d^H \mathbf{x}_i$ .

In general,  $\mathbf{R}_d$ ,  $\mathbf{R}_l$  and  $\mathbf{R}_{\eta}$  are not available, and an indirect method is required to select the candidate which provides the highest SINR. Define  $\mathbf{R}_{l\eta} = \mathbf{R}_l + \mathbf{R}_{\eta}$  and recall that  $\mathbf{R}_d = \sigma_d^2 \mathbf{b}_d \mathbf{b}_d^H$  and  $\mathbf{b}_d^H \mathbf{h}_i = 1$ . The SINR is given by

$$\text{SINR}_i = \mathbf{h}_i^H \mathbf{R}_d \mathbf{h}_i / \mathbf{h}_i^H \mathbf{R}_{l\eta} \mathbf{h}_i = \sigma_d^2 / \mathbf{h}_i^H \mathbf{R}_{l\eta} \mathbf{h}_i, \quad (23)$$

and it is maximized when  $\mathbf{h}_i^H \mathbf{R}_{l\eta} \mathbf{h}_i$  is minimum. Since  $\mathbf{R}_{l\eta}$  is unknown, (23) cannot be directly minimized. As an alternative, we note that the minimization of

$$\mathbf{h}_i^H \mathbf{R}_l \mathbf{h}_i = \sigma_d^2 + \mathbf{h}_i^H \mathbf{R}_{l\eta} \mathbf{h}_i, \quad (24)$$

also maximizes the SINR, and an estimate  $\mathbf{R}$  of  $\mathbf{R}_t$  can be used to compute (24). However, the computation of (24) is costly, proportional to  $O(N^2)$ . To reduce the number of computations, we propose a simpler method, with cost  $O(N)$ .

Defining  $\mathbf{R}_{l\eta} = \mathbf{D}_{l\eta} + \Delta$ , where  $\mathbf{D}_{l\eta} = \sigma_{l\eta}^2 \mathbf{I}$  has the diagonal entries of  $\mathbf{R}_{l\eta}$ , and  $\Delta$  contains the other elements, we can write  $\text{SINR}_i = 1/[(\sigma_{l\eta}^2/\sigma_d^2) ||\mathbf{h}_i||_2^2 + (1/\sigma_d^2) \mathbf{h}_i^H \Delta \mathbf{h}_i]$ . If we consider only two candidates, and that  $\text{SINR}_1 > \text{SINR}_2$ , then we obtain

$$||\mathbf{h}_1||_2^2 < ||\mathbf{h}_2||_2^2 + (\mathbf{h}_2^H \Delta \mathbf{h}_2 - \mathbf{h}_1^H \Delta \mathbf{h}_1) / \sigma_{l\eta}^2. \quad (25)$$

We omit some mathematical manipulation here, but one can show that  $\sigma_{l\eta}^2 = \sum_{i=2}^K \sigma_{s_i}^2 + \sigma_{\eta}^2$ , where  $\sigma_{s_i}^2$  are the power of the interferers. If we assume that  $\Delta$  is small compared to  $\sigma_{l\eta}^2 \mathbf{I}$ , then the second term in the right-hand side of (25) can be neglected. Extending the idea to  $N_c$  candidates, we obtain the proposed selection algorithm

$\mathbf{h}_{\text{MAX}} = \mathbf{h}_k$  when  $k = \arg \min_i (||\mathbf{h}_i||_2^2)$ ,  $i = 1, 2, \dots, N_c$ . Our simulations show that the MC-C-ARH improves the convergence and steady-state SINR, when compared to C-ARH.

**Computational cost:** The algorithm's complexity is  $N_c$  times the complexity of C-ARH, plus the cost to obtain  $\mathbf{h}_i$  and  $||\mathbf{h}_i||_2^2$ , which requires  $10N + 2$  multiplications,  $8N - 2$  additions and 2 divisions per candidate.

**Table 2.** MC-C-ARH algorithm

Input: $\mathbf{R}, \mathbf{b}_d, \tau, \Lambda$	Output: $\mathbf{h}_{\text{MAX}}$
<b>for all</b> $\lambda_i \in \Lambda$ :	
Use $\tilde{\delta} = \lambda_i$ in C-ARH to compute $\mathbf{x}_i$ (see Table 1)	
Compute $\mathbf{h}_i = \mathbf{x}_i / \mathbf{b}_d^H \mathbf{x}_i$	
<b>end for</b>	
Find $k = \arg \min_i \{  \mathbf{h}_i  _2^2\}$ to obtain $\mathbf{h}_{\text{MAX}} = \mathbf{h}_k$	

#### 4. SIMULATIONS

In our simulations, we use a 64-sensor ULA and assume one direction of interest with an angle of  $20^\circ$ , and 4 interferences with angles  $30^\circ, 45^\circ, 53^\circ$  and  $60^\circ$ . We perform two simulations. In the first one, 48 sensors are randomly chosen to have zero-input, creating a sparse scenario. In this case, we assume that after 300 snapshots one additional sensor is faulty, and that 400 snapshots later, the sensor starts to work properly again. In the second simulation, we assume that there are no faulty sensors, and we compare the algorithms' performance when C-ARH and MC-C-ARH are used to regularize the correlation matrix. We compare the SINR performance of RLS [1],  $\ell_1$ -constrained RLS ( $\ell_1$ -RLS, that we have adapted from [8] for beamforming), CH [11], C-ARH, MC-C-ARH, and the theoretical MVDR beamformer. In the sparse scenario, we also present the reduced-rank RLS (RR-RLS), which is computed with  $\mathbf{E}$ , and oracle C-ARH and oracle MVDR, which use the theoretical  $\mathbf{E}$ .

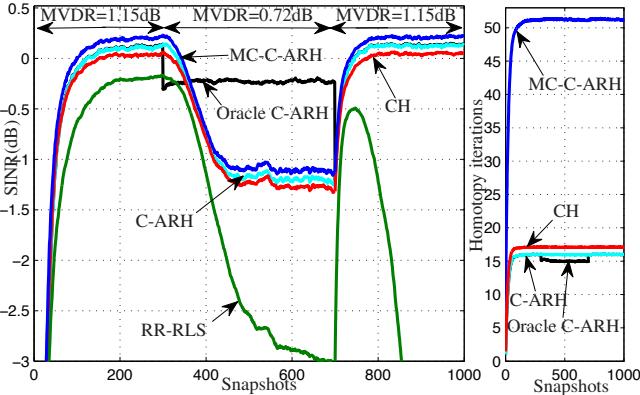
The power of each interferer is 10 times the power of the signal of interest,  $\sigma_d^2 = 1$ . The SNR is 8dB, and the noise is a zero-mean Gaussian i.i.d. sequence. The signals produced by the sources are zero-mean binary sequences of  $-1$  and  $1$ , and we use an iterative estimate of the  $\mathbf{R}_t$ , i.e.,

$$\mathbf{R}(n+1) = \nu \mathbf{R}(n) + \mathbf{u}(n) \mathbf{u}^H(n), \quad (26)$$

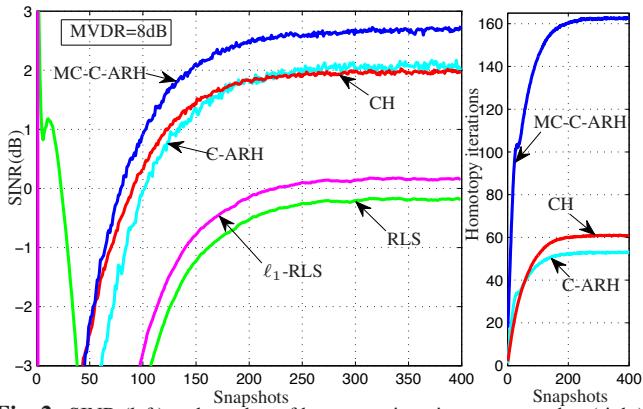
where  $\nu = 0.98$  and  $\mathbf{R}(0) = 10^{-3} \mathbf{I}$ .

The estimation of  $\mathbf{E}$  is performed with the diagonal entries of  $\mathbf{R}(n)$ . We assume that once the  $j$ -th sensor is faulty,  $r_{jj}(n)$  is proportional only to the noise power. Otherwise, when the  $j$ -th sensor works properly,  $r_{jj}(n)$  contains the power contribution of the interferences and the noise. In this case, one must expect the diagonal entries of  $\mathbf{R}(n)$  related to faulty sensors to have smaller values. To define the faulty sensors, we compute the threshold  $\text{Thr} = 10^{-0.6} \max_j(r_{jj}(n))$ , and compare it to all  $r_{jj}(n)$ . If  $r_{jj}(n)$  is smaller than  $\text{Thr}$  for some  $j$ , then the  $j$ -th sensor is faulty, and  $e_{jj}$  is set to 0. Otherwise, we set  $e_{jj} = 1$ . Note that using this threshold, we assume that all the diagonal entries which are at least 6dB smaller than the maximum  $r_{jj}$  come from faulty sensors. In our simulations,  $\mathbf{E}$  is estimated only during the first 100 snapshots. Then, we use the last estimated matrix for the remaining snapshots.

The algorithms are adjusted to achieve the maximum SINR in the steady-state. In the sparse scenario, C-ARH and MC-C-ARH use  $\tau = 0.36$ , and we set  $\tilde{\delta} = 1$  for C-ARH. MC-C-ARH uses three candidates,  $\Lambda = [0.4 \ 0.8 \ 1]$ . The CH algorithm is adjusted to stop when  $|\beta| < 1.8$ , where  $\beta$  corresponds to the residue. The  $\ell_1$ -RLS regularization factor is set to 0.1. For the second simulation, MC-C-ARH and C-ARH use  $\tau = 0.32$ ,  $|\beta| < 2$  in CH and the  $\ell_1$ -RLS regular-



**Fig. 1.** SINR (left) and number of homotopy iterations per snapshot (right) when there are 48 faulty sensors. Mean of 500 realizations. RLS and  $\ell_1$ -RLS had inferior performance (maximum performance of  $SINR_{RLS} = -6.8$ dB and  $SINR_{\ell_1-RLS} = -6.4$ dB, respectively), and are not shown. Oracle MVDR is computed for the 3 intervals, assuming that the faulty sensors are known (presented in the figure just as MVDR).



**Fig. 2.** SINR (left) and number of homotopy iterations per snapshot (right) when all sensors are active. Mean of 500 realizations.

ization factor is 0.05. MC-C-ARH uses the same candidates as before, and  $\tilde{\delta} = 1$  in the C-ARH algorithm.

From Fig. 1 and 2, we see that the homotopy algorithms outperform RLS,  $\ell_1$ -RLS and RR-RLS, but C-ARH and MC-C-ARH have superior performance. For both scenarios, C-ARH provides better SINR steady-state. MC-C-ARH outperforms C-ARH, improving both convergence and steady-state, with the extra cost of additional iterations introduced by the candidates. In the sparse scenario, we note that the estimated  $\mathbf{E}$  allows C-ARH to have almost the same performance as oracle C-ARH. Between snapshots 300 and 700, when there is an additional faulty sensor, C-ARH is worse than oracle C-ARH, since the algorithm is not aware of this failure. However, C-ARH is better than the other algorithms and is only outperformed by MC-C-ARH. RR-RLS is worse than the homotopy techniques presented in Fig. 1, for all snapshots. The performance difference is even more noticeable between 300 and 700 snapshots (when the support is wrongly estimated), and after 750 snapshots, when RR-RLS diverges. In this case, we notice that C-ARH and MC-C-ARH using the estimated  $\mathbf{E}$  to explicitly exploit the sparsity in  $\mathbf{x}_{MVDR}$  perform better than the RR-RLS algorithm used to solve a reduced-size system of equations, which justifies our approach.

## 5. CONCLUSIONS

In this paper, we extend the real ARH algorithm to the complex domain and apply it to sparse beamforming and to the regularization of non-sparse arrays. For the proposed scenarios, we showed that C-ARH outperforms traditional algorithms of the literature, both in convergence and in steady-state. The MC-C-ARH algorithm presents superior performance, and outperforms C-ARH, with the cost of additional iterations introduced by the candidates.

## REFERENCES

- [1] H.L. Van Trees, *Optimum Array Processing: Part IV of Detection, Estimation and Modulation Theory*, Wiley, 2002.
- [2] S.S. Haykin, *Adaptive Filter Theory*, 4ed., Prentice Hall, 2002.
- [3] Z. Yang, R.C. de Lamare, and X. Li, “L1-regularized STAP algorithms with a generalized sidelobe canceler architecture for airborne radar,” *IEEE Trans. Signal Processing*, vol. 60, no. 2, pp. 674–686, 2012.
- [4] Z. Yang, R.C. de Lamare, and X. Li, “Sparsity-aware space-time adaptive processing algorithms with L1-norm regularisation for airborne radar,” *IET Signal Processing*, vol. 6, no. 5, pp. 413–423, 2012.
- [5] M.L. Honig and J.S. Goldstein, “Adaptive reduced-rank interference suppression based on the multistage Wiener filter,” *IEEE Trans. Commun.*, vol. 50, no. 6, pp. 986–994, 2002.
- [6] R.C. de Lamare, “Adaptive reduced-rank LCMV beamforming algorithms based on joint iterative optimisation of filters,” *Electronics Letters*, vol. 44, no. 9, pp. 565–566, 2008.
- [7] R.C. de Lamare and R. Sampaio-Neto, “Adaptive reduced-rank processing based on joint and iterative interpolation, decimation, and filtering,” *IEEE Trans. Signal Processing*, vol. 57, no. 7, pp. 2503–2514, 2009.
- [8] D. Angelosante and G.B. Giannakis, “RLS-weighted lasso for adaptive estimation of sparse signals,” in *IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, 2009, pp. 3245–3248.
- [9] D.L. Donoho and Y. Tsaig, “Fast solution of L1-norm minimization problems when the solution may be sparse,” *IEEE Trans. Inform. Theory*, vol. 54, no. 11, pp. 4789–4812, 2008.
- [10] M.S. Asif and J. Romberg, “Fast and accurate algorithms for re-weighted L1-norm minimization,” *IEEE Trans. Signal Processing*, vol. 61, no. 23, pp. 5905–5916, Dec 2013.
- [11] C. Qi, L. Wu, and X. Wang, “Underwater acoustic channel estimation via complex homotopy,” in *IEEE Int. Conf. on Commun. (ICC)*, 2012, pp. 3821–3825.
- [12] C.D. Meyer, *Matrix Analysis and Applied Linear Algebra*, SIAM, Philadelphia, USA, 2000.
- [13] S.M. Kay, *Fundamentals of Statistical Signal Processing: Detection Theory*, Prentice Hall, 1998.