Convex Optimization in Communication: Improving SINR in Adaptive Beamforming

MohammadKia Zamiri-Jafarian, Department of Electrical and Computer Engineering, Queen's University,
Student Number: 20115898

***Abstract*—This project presents the importance of convex optimization and its application in communications especially in adaptive beamforming. In order for a adaptive beamformer to function appropriately the SINR should be maximized. The use of convex optimization in maximizing SINR is portrayed in this work. Two methods of adaptive beamforming which are Sample Matrix Inversion (SMI) and Loaded Sample Matrix Inversion (LSMI) is simulated and the results are compared to an optimal solution. Also, these simulations are considered for two scenarios (i) when the signal steering vector is known and (ii) when the system is mismatched. The simulation results in both scenarios indicate that the LSMI algorithm gives much better performance than the SMI algorithm.**

***Index Terms*—Convex Optimization, Communications, Adaptive Beamforming, SINR Maximization, Sample Matrix Inversion, Loaded Sample Matrix Inversion.**

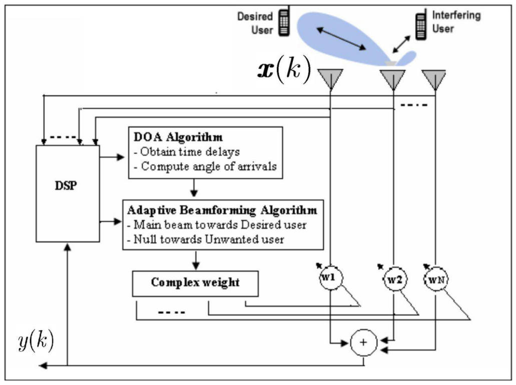
Desired User
Interfering User
$x(k)$
DSP
DOA Algorithm
- Obtain time delays
- Compute angle of arrivals
Adaptive Beamforming Algorithm
- Main beam towards Desired user
- Null towards Unwanted user
Complex weight
w1
w2
wN
+
$y(k)$

Fig. 1: A functional block diagram of a smart antenna system (adaptive beamformer.) [4].

I. Introduction

In the past two decade, we have witnessed a surge of research in optimization. This includes theoretical aspects, as well as algorithmic developments. One of the fields that optimization methods are widely used is Communication and Signal processing. Numerous technical breakthroughs in this field throughout the years has been successful due to the use of convex optimization [1]. In fact, convex optimization has now emerged as a major signal-processing tool because many communication problems can be viewed as convex optimization problems, which greatly facilitate their analytic and numerical solutions [2].

Convex optimization is a subfield of mathematical optimization that demonstrates the problem of minimizing a convex function subject to convex constraints over a convex set [2]. This type of optimization problems are far more general than typical linear programming problems, since it linear functions are convex [1].

There are many advantages in recognizing or reformulating a problem as a convex optimization problem. The most significant one is that the problem can be solved very quickly and efficiently despite having countless constraints and variables. These solutions are reliable enough to be embedded in computer-aided design and analysis tools [3].

In communication and digital signal processing (DSP) where computational efficiency is vital, the usual optimization methods used to be the gradient descent and least square methods. However, both of these approaches are known to offer unreasonably difficulties surrounding the step size selection, algorithm initialization and local minima [3].On the other hand, convex optimization techniques present suitable insights to the structure optimal solutions and provides powerful numerical algorithms. Meanwhile, every local optimum in a convex problem is a global optimum as well, which is found to be an essential feature in engineering applications. Advances in current convex optimization problems (i.e. especially in conic and robust optimization) such as developed interior point optimization schemes and highly efficient software tools have helped to make great achievements in the fields of signal processing and digital communications [2]. One subject that convex optimization have been used is in adaptive beamforming.

Beamforming is a signal processing technique used in sensor arrays for directional signal transmission or rejection. Basically, beamforming means to focus a signal to a desired direction as it is shown in Fig.1. This method assigns certain weights to antenna gains in order to minimize the noise and interference and the simplest way is to maximize SNR (SINR).

This project, first investigates on the mathematical model of a adaptive beamformer and how to formulate it as a optimization problem and afterwards by using convex optimization we will compute the gain weights which gives us the best SINR.

The reminder of this project is organized as follows. In Section II, we discuss briefly on the history of adaptive beamforming and different approaches for implementing it. Section III demonstrates the mathematical model of an adaptive beamformer and the process of formulating it into a convex optimization problem and the results are shown in Section IV. Finally, Section V concludes this project and provides reasoning on why the use of convex optimization benefits us. Note that, MATLAB codes are provided in the Appendix Section.

II. Background

Beamforming is a type of radio frequency (RF) management in which an access point uses multiple antennas or sensors to transmit or receive out signals. By sending multiple signals and analyzing the feedback from the users (clients), the transmitter then can adjust the signals it sends out. Adjusting the transmitted signals not only helps to determine the best route for the signal to reach out the client but also lowers or completely neglects any interference from other sources with the same frequency. In other words, beamforming characterizes the RF beam as it crosses the physical space [4].

The rise in traffic throughout the electromagnetic spectrum, coupled with the availability of high-speed real-time computational resources made a huge interest in a kind of beamforming in wireless communications called adaptive beamforming. An adaptive beamformer is a system designed to isolate a desired signal from interfering signals that occupy the same spectral band but are spatially different. This technique was initially developed in the 1960s for military purposes such as sonar and radar, but nowadays it is widely used in commercial networks such as LTE.

The basic concepts behind adaptive beamforming is to create higher or lower amplitude wave by assigning different weights to the signal received. The adaptive beamformer will automatically adapts the parameters in order to maximize or minimize other factors such as Signal-to-Interference-plus-Noise ratio (SINR). There are several ways to approach this design but the first two method were maximizing Signal-to-Noise ratio (SNR) and using Least Mean Squares (LMS) error. Both of these approaches are similar to each other and will converge to an optimal solution but have implementation disadvantages. Due to these drawbacks in 1974, a technique known as Sample Matrix Inversion (SMI) was introduced and no longer after Loaded SMI (LSMI) was also announced [5]. We will thoroughly explain these to algorithms and their approach later in this paper. In the next section, we tend to acquire the system mathematical model and then reformulate it into a linear (convex) optimization problem for both SMI and LSMI algorithms.

Notation

In this project, scaler variables are represented by non-bold lowercase letters (e.g. c), the vectors are denoted by bold lowercase letters (e.g. $\mathbf{x}$), matrices and sets of vectors are shown as uppercase bold letters, (e.g. $\mathbf{F}$). Also, the T sign represents the transpose operation on matrices (e.g. $\mathbf{F}^T$) and H sign illustrates the Hermitian transpose of the matrix (e.g. $\mathbf{F}^H$).

III. System Model

This section presents the mathematical formulation of the convex optimization method discussed earlier, in order to improve the performance of an adaptive beamformer system. In this regard, let's revisit Fig. 1 once again, which illustrates a simple block diagram of a smart antenna system. According to Fig. 1, the output of a narrow-band beamformer in discrete time, denoted as $y(k)$, can be expressed as follows [6]

$$y(k) = \mathbf{w}^H \mathbf{x}(k) \qquad 1 \le k \le M \tag{1}$$

where $\mathbf{x}(k)$ and $\mathbf{w}$ represent complex vector of observations and complex vectors of beamformer weights, respectively. In (1), M refers to the number of sensors or antennas presumed for this model. In addition, the observation vector (i.e. training sanpshot), $\mathbf{x}(k)$, is given by

$$\mathbf{x}(k) = \mathbf{s}(k)\mathbf{a} + \mathbf{i}(k) + \mathbf{n}(k) \qquad 1 \le k \le M \tag{2}$$

where $\mathbf{s}(k)$ is the desired signal waveform and $\mathbf{a}$ is the signal steering vector. Here, $\mathbf{i}(k)$ and $\mathbf{n}(k)$ are the interference and the noise signal, respectively. In this project, for simplicity we assume that the interference and noise elements follow zero mean Gaussian distribution. In this regard, for the signal-to-interference-plus-noise ratio (SINR) is defined as [7]

$$\text{SINR} = \frac{\sigma_s^2 |\mathbf{w}^H \mathbf{a}|^2}{\mathbf{w}^H \mathbf{R}_{i+n} \mathbf{w}} \tag{3}$$

where $\mathbf{R}_{i+n}$ is the interference-plus-noise covariance matrix is expressed as

$$\mathbf{R}_{i+n} = \mathbb{E}[(\mathbf{i}(k) + \mathbf{n}(k))(\mathbf{i}(k) + \mathbf{n}(k))^H] \tag{4}$$

and in (3), σ_s^2 is the signal power. Clearly from (3), we can find the weight vector that results the maximization of SINR, by minimizing the output interference-plus-noise power. This will then maintain a distortion-less response for the desired signal. Therefore, the maximization of (3) is equivalent to solving the following convex optimization problem [7]

$$\begin{aligned} \min_{\mathbf{w}} \quad & \mathbf{w}^H \mathbf{R}_{i+n} \mathbf{w} \\ \text{s.t.} \quad & \mathbf{w}^H \mathbf{a} = 1 \end{aligned} \tag{5}$$

Note that generally, the solution for the above problem is the optimal weight vector, denoted as $\mathbf{w}_{opt}$, given as

$$\mathbf{w}_{opt} = \alpha \mathbf{R}_{i+n}^{-1} \mathbf{a} \tag{6}$$

where α is a normalized constant that doesn't affect the output SINR and can be omitted. In this regard, by setting $\alpha = 1$ the optimal SINR becomes [7]

$$\mathbf{w}_{opt} = \mathbf{R}_{i+n}^{-1} \mathbf{a} \longrightarrow \text{SINR}_{opt} = \sigma_s^2 \mathbf{a}^H \mathbf{R}_{i+n}^{-1} \mathbf{a} \tag{7}$$

In most applications the interference-plus-noise covariance matrix, $\mathbf{R}_{i+n}$, is unavailable (i.e. although in simulations it is calculated). In such circumstances, an alternative approach is to compute the sample covariance matrix, instead of applying (4). Here, the sample covariance matrix, denoted as $\hat{\mathbf{R}}$, is defined as

$$\hat{\mathbf{R}} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}(n) \mathbf{x}^H(n) \tag{8}$$

where N is the number of observations or training snapshots. By considering the expression in (7), the solution for the sample matrix inversion (SMI) becomes

$$\mathbf{w}_{SMI} = \hat{\mathbf{R}}^{-1} \mathbf{a} \tag{9}$$

where $\mathbf{w}_{SMI}$ corresponds to the beamformer weights when the sample covariance matrix is used.

However, calculating $\hat{\mathbf{R}}^{-1}$ becomes a numerical challenge due to the fact that the eigenvalues of the covariance matrix may be small. To avoid such issue, a simple approach is to diagonally load the covariance matrix. In other words, we substitute $\hat{\mathbf{R}}$ with the following so-called diagonally loaded covariance matrix, denoted as $\hat{\mathbf{R}}_{dl}$,

$$\hat{\mathbf{R}}_{dl} = \zeta\mathbf{I} + \hat{\mathbf{R}} \longrightarrow \mathbf{w}_{LSMI} = \hat{\mathbf{R}}_{\mathbf{dl}}^{-1}\mathbf{a} = (\zeta\mathbf{I} + \hat{\mathbf{R}})^{-1}\mathbf{a} \quad (10)$$

where ζ is a diagonal loading factor and $\mathbf{I}$ is the identity matrix. This scheme is commonly referred to as loaded SMI (LSMI); thus, the beamformer weights are expressed as $\mathbf{w}_{LSMI}$. The main issue in LSMI is to choose the diagonal loading factor, ζ. There are different approaches proposed to select ζ,such as the use of white noise gain [8]. However this method has more computational complexity than the LSMI algorithm. Therefore, it is more convenient to choose the diagonal loading factor as $10\sigma^2$, where σ is the noise power of a single sensor.

Furthermore, to actually implement the convex optimization problem in practical scenarios, the norm of the steering vector distortion, denoted as Δ, must be bounded by a positive constant ϵ; (i.e. $||\Delta|| \leq \epsilon$). In addition, the actual signal steering vector belongs to the set $A(\epsilon) \triangleq \{\mathbf{c}|\mathbf{c} = \mathbf{a} + \mathbf{e}, ||\mathbf{e}|| \leq \epsilon\}$. This indicates that for all the vectors that belong to $A(\epsilon)$, the absolute value of the array response should not be smaller than one. As a result, the formulation of the (5) changes as follows [6]

$$\begin{aligned} \min_{\mathbf{w}} \quad & \mathbf{w}^H\mathbf{R}\mathbf{w} \\ \text{s.t.} \quad & |\mathbf{w}^H\mathbf{c}| \geq 1 \quad \forall \mathbf{c} \in \mathbf{A}(\epsilon) \end{aligned} \quad (11)$$

where $\mathbf{c} = \mathbf{a} + \mathbf{e}$. By applying the triangle and Cauchy–Schwarz inequalities along with the inequality, $||e|| \leq \epsilon$, the following is achieved

$$|\mathbf{w}^H\mathbf{a} + \mathbf{w}^H\mathbf{e}| \geq |\mathbf{w}^H\mathbf{a}| - |\mathbf{w}^H\mathbf{e}| \geq |\mathbf{w}^H\mathbf{a}| - \epsilon||\mathbf{w}|| \quad (12)$$

It can be verified that for ϵ small enough (i.e. $\epsilon << 1$) we have

$$|\mathbf{w}^H\mathbf{a} + \mathbf{w}^H\mathbf{e}| = |\mathbf{w}^H\mathbf{a}| - \epsilon||\mathbf{w}|| \quad (13)$$

where $\mathbf{e}$ is represented as

$$\mathbf{e} = -\frac{\mathbf{w}}{||\mathbf{w}||}\epsilon e^{j\phi} \quad (14)$$

with $\phi = \angle\{\mathbf{w}^H\mathbf{a}\}$. Then from combining (11) and (13), it is evident that $\min_{\mathbf{c}\in\mathbf{A}(\epsilon)} |\mathbf{w}^H\mathbf{c}| = |\mathbf{w}^H\mathbf{a}| - \epsilon||\mathbf{w}||$. Consequently, the following is obtained

$$\begin{aligned} \min_{\mathbf{w}} \quad & \mathbf{w}^H\hat{\mathbf{R}}\mathbf{w} \\ \text{s.t.} \quad & |\mathbf{w}^H\mathbf{a}| - \epsilon||\mathbf{w}|| \geq 1 \end{aligned} \quad (15)$$

which will lead to

$$\begin{aligned} \min_{\mathbf{w}} \quad & \mathbf{w}^H\hat{\mathbf{R}}\mathbf{w} \\ \text{s.t.} \quad & \mathbf{w}^H\mathbf{a} \geq 1 + \epsilon||\mathbf{w}|| \\ & \text{Im}\{\mathbf{w}^H\mathbf{a}\} = 0. \end{aligned} \quad (16)$$

Note that, to simplify the coding process, in this project we have used the second-order cone (SOC) programming [6]. First, we convert the quadratic objective function into a linear convex optimization problem. In this regard, by defining the sample covariance matrix as follows

$$\hat{\mathbf{R}} = \mathbf{U}^H\mathbf{U} \quad (17)$$

where a Cholesky factorization of $\hat{\mathbf{R}}$ is employed to get lower and upper triangular matrix. Hence, the objective function could be rewritten as

$$\mathbf{w}^H\hat{\mathbf{R}}\mathbf{w} = ||\mathbf{U}\mathbf{w}||^2. \quad (18)$$

Evidently, minimizing $||\mathbf{U}\mathbf{w}||^2$, is equivalent to minimizing the main objective. Moreover, by introducing a new non-negetive scalar τ and a new constraint $||\mathbf{U}\mathbf{w}|| \leq \tau$; we can convert (16) into the following problem [6]:

$$\begin{aligned} \min_{\mathbf{w},\tau} \quad & \tau \\ \text{s.t.} \quad & ||\mathbf{U}\mathbf{w}|| \leq \tau \\ & \epsilon||\mathbf{w}|| \leq \mathbf{w}^H\mathbf{a} - 1 \\ & \text{Im}\{\mathbf{w}^H\mathbf{a}\} = 0. \end{aligned} \quad (19)$$

The above equation is the final formulation that is used to find the optimal gain weights that maximizes the SINR.

IV. Simulation Results

In the simulation, we assume a uniform linear array with M = 10 sensors with half a wavelength spacial apart. For each scenario, 100 simulation runs are used to obtain each simulated point. In both examples, we assume that two interfering sources have the direction of arrival 30° and 50°. The interference-to-noise ratio in a single sensor is equal to 30dB. There are two methods compared in terms of the mean output SINR: the SMI beamformer and the LSMI beamformer. The optimal SINR is also presented to get a better sense of the result. The SeDuMi convex optimization MATLAB toolbox has been used to compute the weight vector and the beamformer has the contant $\epsilon = 3$ and in the case of LSMI we have $\zeta = 10\sigma^2 = 10$. As for the optimal case it is presumed that the loading factor $\zeta = 0.01$. With these assumptions we continue with the two examples given.

A. Example 1: Exactly Known Signal Steering Vector

In this example, we simulate a scenario where the actual signal steering is known and there are no mismatches in other words $\tilde{\mathbf{a}} = \mathbf{a}$. Here the plane-wave signal is assumed to impinge on the array from $\theta_s = 3°$. In Fig. 2 the mean output array SINR of the two approaches are compared the number of snapshots (observations) N for a fixed single-sensor with SNR= $-10dB$. Fig. 3 demonstrates the performance of the two techniques versus SNR for a fixed number of training data $N = 20$.

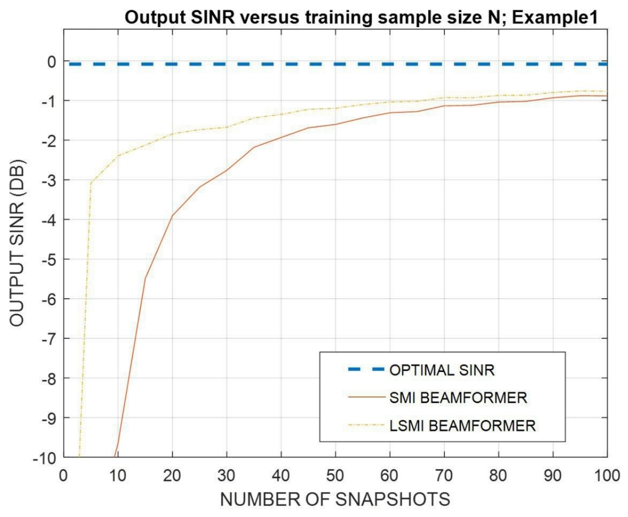

Fig. 2: Comparing the performance of the SINR versus training snapshots with size N for example 1 for three cases of (i) optimal, (ii) SMI algorithm and (iii) LSMI algorithm. The result indicates that LSMI beamformer converges much faster than the SMI beamformer.

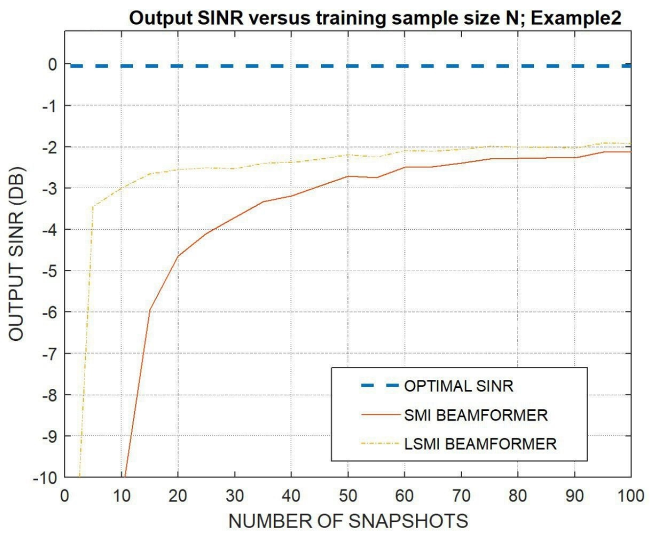

Fig. 4: Comparing the performance of the SINR versus training snapshots with size N for example 2 for three cases of (i) optimal, (ii) SMI algorithm and (iii) LSMI algorithm. The result indicates that LSMI beamformer converges much faster than the SMI beamformer.

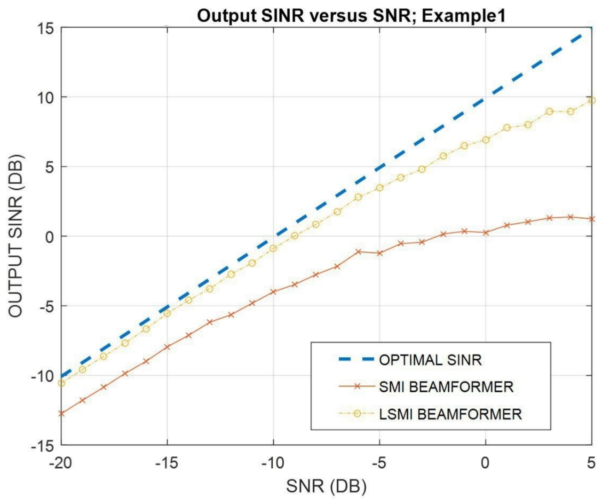

Fig. 3: Comparing the output SINR versus SNR for example 1 for three cases of (i) optimal, (ii) SMI algorithm and (iii) LSMI algorithm. The result shows that the optimal SINR is more compatible with the LSMI beamformer rather than the SMI beamformer.

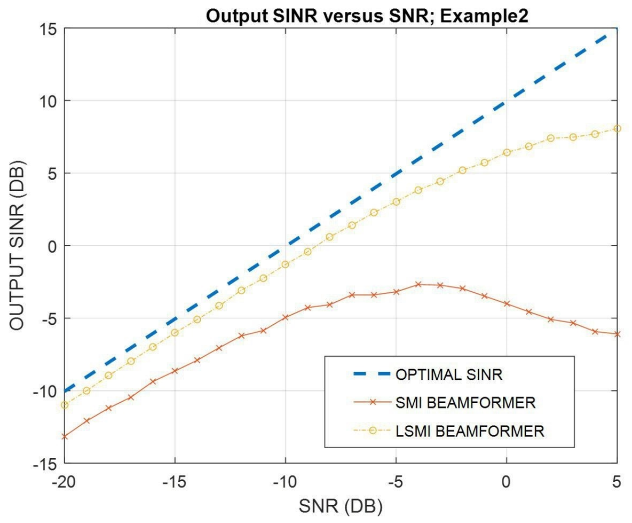

Fig. 5: Comparing the output SINR versus SNR for example 2 for three cases of (i) optimal, (ii) SMI algorithm and (iii) LSMI algorithm. The result shows that the optimal SINR is more compatable with the LSMI beamformer rather than the SMI beamformer. Especially in the case of a high SNR the output SINR for SMI algorithm drops significantly.

B. Example 2: Signal Direction Mismatch

In the second example, we consider a scenario where the signal direction is mismatched (i.e. $\tilde{\mathbf{a}} = \mathbf{a} + \Delta$). We assume that both the presumed and actual signal spatial signatures are plane waves impinging from the direction of arrival 3 and 5, respectively. This results in a two degree mismatch. Fig. 4 shows the two method performances versus the number of training snapshots N with a fixed SNR $= -10dB$. The performance of the two algorithms versus SNR for a fixed number of training data N=20 are presented in Fig. 5.

V. Conclusion

In this project, a brief concept of convex optimization and its application in Communications and Signal Processing has been discussed with a specific example in maximizing SINR in adaptive beamforming. This technique optimizes the performance by minimizing the output interference-plus-noise ratio while maintaining a distortionless response in order to achieve the best SINR. We have presented the results in two separate occasions: 1) signal steering vector is known. 2) mismatch in the signal look direction. In both occasions the LSMI algorithm has given a better result than the SMI algorithm (and both of these algorithms never reach the optimal case.). This deference in algorithm shows itself more in the mismatch example where for a high SNR the output SINR is decreases noticeably in SMI. Although, we have to consider the fact that the computational complexity of the SMI algorithm compared to LSMI algorithms is lesser.

Appendix A MATLAB Codes

This section presents the MATLAB codes in order to determine the simulation results discussed in Section IV. For further simplification of the MATLAB codes, the following definitions are provided [6]

$$\begin{aligned}\hat{\mathbf{w}} &\triangleq [\mathrm{Re}\{\mathbf{w}\}^T, \mathrm{Im}\{\mathbf{w}\}^T]^T\\ \hat{\mathbf{a}} &\triangleq [\mathrm{Re}\{\mathbf{a}\}^T, \mathrm{Im}\{\mathbf{a}\}^T]^T\\ \bar{\mathbf{a}} &\triangleq [\mathrm{Im}\{\mathbf{a}\}^T, -\mathrm{Re}\{\mathbf{a}\}^T]^T\\ \hat{\mathbf{U}} &\triangleq \begin{bmatrix}\mathrm{Re}\{\mathbf{U}\} & -\mathrm{Im}\{\mathbf{U}\}\\ \mathrm{Im}\{\mathbf{U}\} & \mathrm{Re}\{\mathbf{U}\}\end{bmatrix}\end{aligned} \tag{20}$$

we then can rewrite the equation (19) in terms of real-valued vectors and matrices as follows [6]

$$\begin{aligned}\min_{\hat{\mathbf{w}},\tau} \quad & \tau\\ \text{s.t.} \quad & ||\hat{\mathbf{U}}\hat{\mathbf{w}}|| \le \tau\\ & \epsilon||\hat{\mathbf{w}}|| \le \hat{\mathbf{w}}^H\hat{\mathbf{a}} - 1\\ & \hat{\mathbf{w}}^H\bar{\mathbf{a}} = 0.\end{aligned} \tag{21}$$

Also, the following expressions are considered

$$\mathbf{y} \triangleq [\tau, \mathbf{w}^T]^T \in \mathbb{R}^{(2M+1)\times 1} \tag{22}$$

$$\mathbf{F} \triangleq \begin{bmatrix}\mathbf{1} & \mathbf{0}^T\\ \mathbf{0} & \mathbf{U}\\ \mathbf{1} & \mathbf{a}^T\\ \mathbf{0} & \epsilon\mathbf{I}\\ \mathbf{0} & \bar{\mathbf{a}}^T\end{bmatrix} \in \mathbb{R}^{(4M+1)\times(2M+1)} \tag{23}$$

where we only have simplified the notation of (21) in order to make the MATLAB codes easier to follow. Here, $\mathbf{0}$ is the vector of zeros of an equivalent dimension.

Listing 1: MATLAB codes for comparing the performance of SINR vs SNR

```
clear all;
N = 20; %number of observations
M = 10; %number of antennas
tI= [30, 50]; % impinging angles of interfering
    sources
J  = 2 ; % Number of interfering sources
aI = exp(1i*2.0*[0:M-1]'*pi*0.5*sin(tI*pi/180.0))
    ; % interfering steering matrix
INR  = 1000; % interference noise ratio
SNR_dB = -20:1:5;

for i = 1:length(SNR_dB)
    SNR(i) = 10 ^ (SNR_dB(i)/10);
end

tN = 3; % impinging angle
tA = 5; % actual impinging angle(In Example 2)
RI_n=INR*aI*aI'+eye(M,M); % ideal interference-
    plus-noise covariance matrix for optimal SINR

SINR_SMI=zeros(1,length(SNR));
SINR_LSMI=zeros(1,length(SNR));

for loop = 1:10
     a = exp(1i*2.0*[0:M-1]'*pi*0.5*sin(tN*pi
         /180.0)); %generation of actual signal
         steering vector a_tilde (Example 1)
     a_t = a; %Example 1
     %a_tilde = exp(1i*2.0*[0:M-1]'*pi*0.5*sin(tA
         *pi/180.0)); %Example 2
     a_t = sqrt(10)*a_t/norm(a_t)';
     Ncount=0;
     for p=SNR
         Ncount = Ncount + 1;
         R_h = zeros(M);
         for i = 1:N
          s = (randn(1,1) + 1i*randn(1,1))*sqrt(
              ps)*sqrt(0.5);  % signal waveform
          Is = (randn(J,1) + 1i*randn(J,1)).*sqrt
              (INR)'*sqrt(0.5); % interference
              waveform
          n = (randn(M,1) + 1i*randn(M,1))*sqrt
              (0.5); % noise
          x = aI*Is + s*a_t + n; % snapshot with
              signal
          R_h = R_h + x*x';
         end
         R_h = R_h/N; %norm of covariance matrix

         R_dl_h = R_h + eye(M,M)*10; %loaded
             sample matrix inversion (LSMI)
         w_LSMI = inv(R_dl_h)*a;

         w_SMI = inv(R_h)*a; %sample matrix
             inversion (SMI)

         R_h = R_h + eye(M,M)*0.01; %diagonal
             loading where the loading factor for
              an optimal system is 0.01
         U=chol(R_h);  % Cholesky factorization
         U_h=[real(U),-imag(U);imag(U),real(U)];
         M2 = 20;
         a_h=[real(a);imag(a)];
         a_b = [imag(a);-real(a)];
         FT = [1,zeros(1,M2);zeros(M2,1),U_h;0,
             a_h';zeros(M2,1),3*eye(M2,M2);0,a_b
             '];
         cvx_begin
```

```
            variable y(M2+1,1)
            minimize (y(1))
            c = FT*y;
            subject to
              c(1) >= norm(c(2:M2+1));
              c(M2+2)-1 >= norm(c(M2+3:2*M2+2));
              c(2*M2+3) == 0;
         cvx_end
         SINR_SMI_v(Ncount) = p * (abs(w_SMI'*a_t
             ) * abs(w_SMI'*a_t)) / abs(w_SMI'*
             RI_n*w_SMI);  % SINR
         SINR_LSMI_v(Ncount) = p * (abs(w_LSMI'*
             a_t) * abs(w_LSMI'*a_t)) / abs(
             w_LSMI'*RI_n*w_LSMI);
         SINR_opt(Ncount) = p * a_t' * inv(RI_n)
             * a_t;
     end
     SINR_SMI     = ((loop-1)/loop)*SINR_SMI +
         (1/loop)*SINR_SMI_v; % just average
     SINR_LSMI    = ((loop-1)/loop)*SINR_LSMI +
         (1/loop)*SINR_LSMI_v;
end
```

Listing 2: MATLAB codes for comparing the performance of SINR vs the number of training snapshots

```
clear all;

M=10;  % No. of antennas
SNR_dB = -10;
p = 10^(SNR_dB/10);

t_I= [30, 50]; % impinging angles of interfering
    sources
J  = 2; % No. of interfering sources

INR  = 1000; % interference noise ratio
aI = exp(1i*2.0*[0:M-1]'*pi*0.5*sin(t_I*pi/180.0)
    ); % interfering steering matrix
RI_n=INR*aI*aI'+eye(M,M); % ideal interference-
    plus-noise covariance matrix for computing
    optimal SINR

N=[1 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75
    80 85 90 95 100];
SMI=zeros(1,length(N));
LSMI=zeros(1,length(N));

tN=3; % impinging angle
tA=5; % actual impinging angle(In Example 2)

for loop = 1:100;
    a = exp(1i*2.0*[0:M-1]'*pi*0.5*sin(tN*pi
        /180.0));
    %a_t = a; %Example 1
    a_t = exp(1i*2.0*[0:M-1]'*pi*0.5*sin(tA*pi
        /180.0)); %Example 2
    a_t = sqrt(10)*a_t/norm(a_t)';
```

```
  Ncount=0;
for N= [1 5 10 15 20 25 30 35 40 45 50 55 60 65
     70 75 80 85 90 95 100];
    Ncount=Ncount+1;
    R_h = zeros(M);
    for l=1:N
        sw=(randn(1,1) + 1i*randn(1,1))*sqrt(ps
            )*sqrt(0.5);  % signal waveform
        Is = (randn(J,1) + 1i*randn(J,1)).*sqrt
            (INR)'*sqrt(0.5); % interference
            waveform
        n = (randn(M,1) + 1i*randn(M,1))*sqrt
            (0.5); % noise
        x = aI*Is + sw*a_t + n; % snapshot with
             signal
        R_h = R_h + x*x';
    end
    R_h=R_h/length(N);

    w_SMI=inv(R_h)*a;

    R_dl_h = R_h + eye(M,M)*10;
    w_LSMI=inv(R_dl_h)*a;

    R_h = R_h + eye(M,M)*0.01; % diagonal
        loading where the loading factor for an
         optimal system is 0.01

    M2 = 20;
    a_h=[real(a);imag(a)];
    a_b = [imag(a);-real(a)];
    U=chol(R_h);  % Cholesky factorization
    U_h=[real(U),-imag(U);imag(U),real(U)];
    FT = [1,zeros(1,M2);zeros(M2,1),U_h;0,a_h';
        zeros(M2,1),3*eye(M2,M2);0,a_b'];

    cvx_begin
        variable y(M2+1,1)
        minimize (y(1))
        c = FT*y;
        subject to
            c(1) >= norm(c(2:M2+1));
            c(M2+2)-1 >= norm(c(M2+3:2*M2+2));
            c(2*M2+3) == 0;
    cvx_end
    SMI_v(Ncount) = p * (abs(w_SMI'*a_t) * abs(
        w_SMI'*a_t)) / abs(w_SMI'*RI_n*w_SMI);
         % SINR
    LSMI_v(Ncount) = p * (abs(w_LSMI'*a_t) *
        abs(w_LSMI'*a_t)) / abs(w_LSMI'*RI_n*
        w_LSMI);
    SINR_opt1(Ncount) = p * a_t' * inv(RI_n) *
        a_t;
end
SMI     = ((loop-1)/loop)*SMI + (1/loop)*SMI_v;
     % just average
LSMI    = ((loop-1)/loop)*LSMI + (1/loop)*
    LSMI_v;
```

```
end
```

References

[1] D. P. Palomar and Y. C. Eldar, *Convex optimization in signal processing and communications*. Cambridge university press, 2010.

[2] Z.-Q. Luo and W. Yu, "An introduction to convex optimization for communications and signal processing," *IEEE Journal on selected areas in communications*, vol. 24, no. 8, pp. 1426–1438, 2006.

[3] Z.-Q. Luo, "Applications of convex optimization in signal processing and digital communication," *Mathematical programming*, vol. 97, no. 1-2, pp. 177–207, 2003.

[4] R. Shubair and A. Merri, "A convergence study of adaptive beamforming algorithms used in smart antenna systems," in *11th International Symposium on Antenna Technology and Applied Electromagnetics [ANTEM 2005]*, pp. 1–5, IEEE, 2005.

[5] J. S. Blogh, J. Blogh, and L. L. Hanzo, *Third-generation systems and intelligent wireless networking: smart antennas and adaptive modulation*. John Wiley & Sons, 2002.

[6] S. A. Vorobyov, A. B. Gershman, and Z.-Q. Luo, "Robust adaptive beamforming using worst-case performance optimization: A solution to the signal mismatch problem," *IEEE transactions on signal processing*, vol. 51, no. 2, pp. 313–324, 2003.

[7] R. Monzingo and T. Miller, "Introduction to adaptive arrays, john willey & son," *New York*, 1980.

[8] H. Cox, R. Zeskind, and M. Owen, "Robust adaptive beamforming," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 10, pp. 1365–1376, 1987.