Politecnico di Torino

Applied signal processing laboratory

(A.Y. 2024/25)

Assignment 4

Array signal processing

Authors:
Castorina Giovanni (307594)
Fanton Vittoria (308625)
Giordano Andrea (310679)

Contents

1 Introduction

This report presents the main concepts of array signal processing, focusing on both linear and planar configurations.

In the first exercise, we concentrated on linear array processing and beamforming. We began by comparing the beamforming pattern of the array with the one generated by the Dirichlet function. The analysis then continued by examining the effects of varying the number of elements M, while keeping the direction of arrival and inter-element spacing fixed. Subsequently, we investigated the impact of different values of d/λ, identifying the positions of the nulls in the array factor and computing the First Null Beamwidth (FNBW).

The second exercise deals with planar arrays, where the antenna elements are arranged in two dimensions to allow 3D beamforming in both azimuth and elevation. We started by generating the array response for different steering directions, first using isotropic elements and then including a directive pattern to model more realistic antennas. We then explored how changing the number of elements along the two axes affects beamwidth and shape in the 3D radiation pattern.

In the final part of the assignment, we implemented the MVDR beamformer for a Uniform Planar Array. We studied its behavior in scenarios with multiple interferers, observing how the beamformer places spatial nulls to suppress interference while maintaining unity gain in the user direction. Additional simulations tested the limits of MVDR, particularly in cases where interferers are very close to the user or when the noise power is extremely high.

2 Linear array processing and beamforming

In this exercise, the main concepts explored are linear array processing and beamforming. Starting from the following parameters:

- Number of antennas: $M = 8$
- Carrier frequency: $f_c = 900\,\text{MHz}$
- Speed of light: $c = 3 \cdot 10^8\,\text{m/s}$

we can compute the wavelength λ as:

$$\lambda = \frac{c}{f_c} = \frac{3 \cdot 10^8}{9 \cdot 10^8} = 0.\overline{3}\,\text{m}$$

The antenna spacing d is then chosen as:

$$d = \frac{\lambda}{2} = 0.1667\,\text{m}$$

This value satisfies the sampling theorem in antenna array design, which requires: $d \leq \frac{\lambda}{2}$.

Furthermore, the direction of arrival (DoA) of the signal is set to $\theta_1 = 0°$. The spatial frequency Ψ is then computed as:

$$\Psi_k = 2\pi \frac{d}{\lambda} \sin(\theta_k)$$

Given this value of Ψ, the conventional beamforming vector is defined as:

$$\mathbf{a}(\Psi_k) = \begin{bmatrix} 1 \\ e^{-j\Psi_k} \\ e^{-j2\Psi_k} \\ \vdots \\ e^{-j(M-1)\Psi_k} \end{bmatrix} = \left[e^{-jn\Psi_k} \right]_{n=0}^{M-1}$$

The normalized conventional beamformer is given by:

$$\mathbf{w} = \frac{1}{\sqrt{M}} \mathbf{a}(\Psi_1)$$

Subsequently, a vector of angles $\boldsymbol{\theta}$ with $S = 3601$ elements is created using the command `theta = linspace(-90, 90, S)`, where the angles are expressed in degrees. Then, using a single `for` loop, for each angle θ_i, the corresponding steering vector $\mathbf{a}(\theta_i)$ is computed and stored as a column in the $M \times S$ matrix $\mathbf{V}$.

Once all steering vectors are stacked into $\mathbf{V}$, the array pattern is computed as a vector through the matrix product: $\mathbf{w}^H\mathbf{V}$, where $\mathbf{w}^H$ denotes the Hermitian (conjugate transpose) of the beamforming vector $\mathbf{w}$.

This process can be implemented with the following MATLAB commands:

```
for i = 1:S
    a_theta = exp(1j *2*pi*d / lambda*m*sind(theta(i)));
    V(:, i) = a_theta;
end

w_hermitian = w';
pattern = w_hermitian * V;
```

Here, m = *0:M-1* is assumed to be a column vector representing the element indices of the antenna array. Actually, the request was to perform this task with a for loop but it is possible to do it also without the loop as:

```
V = exp(1j*2*pi*d/lambda*m'*sind(theta_val));
w_hermitian = w';
pattern = w_hermitian * V;
```

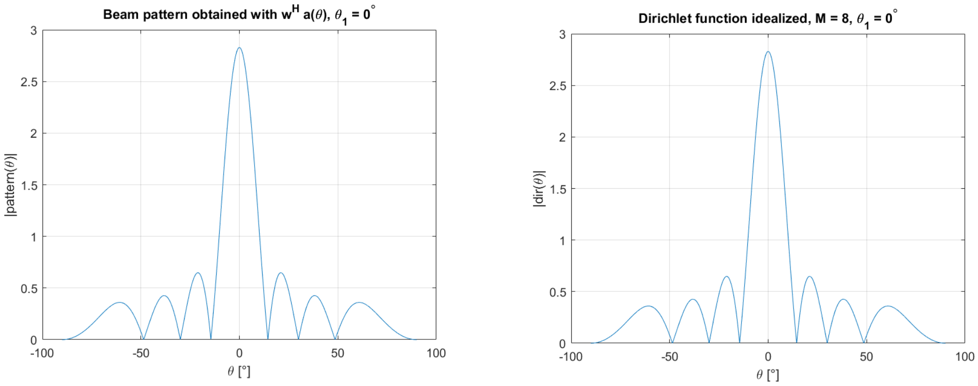
The array factor, which corresponds to the magnitude of the beamforming pattern, is plotted using the standard `plot` function and compared with the output of the `diric` function. The latter takes as arguments the parameter

$$\Psi = 2\pi\frac{d}{\lambda}(\sin(\theta) - \sin(\theta_1))$$

and the number of antennas M, after appropriate normalization. This comparison is used to evaluate the agreement between the analytically derived array response and the theoretical Dirichlet function shape. The implementation is provided below:

```
psi_diric = 2*pi*d/lambda*(sind(theta)-sind(theta_1));
dir = sqrt(M)*diric(psi_diric, M);
```

The comparison between the two plots is shown in Figure 1. As evident from the figure, after appropriate normalization by a factor of $\sqrt{M}$, the two results align perfectly, demonstrating that the computed beamforming pattern matches the ideal theoretical pattern very well.

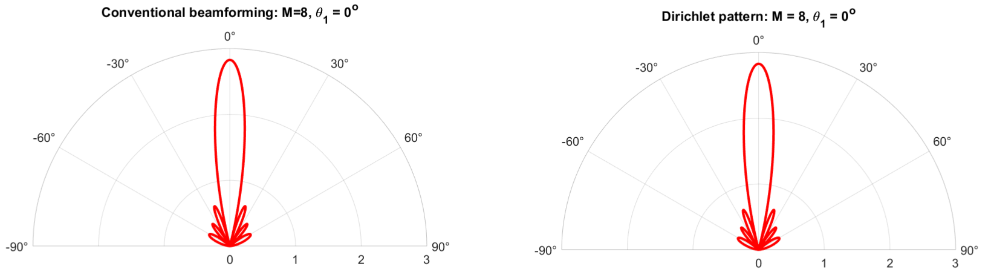


(a) Magnitude of the beamforming pattern computed via $\mathbf{w}^H\mathbf{V}$.

(b) Normalized `diric` function (theoretical) with parameters Ψ and M.

Figure 1: Comparison between the beamforming pattern and the one generated by the `diric` function.

Finally, the beamforming pattern is also plotted in polar coordinates using the command `polarplot(deg2rad(theta), abs(dir), 'r', 'LineWidth', 2)`. Since the `polarplot()` function requires the angular values to be in radians, the angle vector `theta` is converted accordingly using `deg2rad()`. The resulting polar comparison is shown in Figure 2. Again, as in Figure 1, it is possible to affirm that the two results are totally in agreement with each other.



(a) Polar plot of the magnitude of the beamforming pattern computed via $\mathbf{w}^H\mathbf{V}$.

(b) Polar plot of the normalized `diric` function with parameters Ψ and M.

Figure 2: Comparison, in polar form, between the beamforming pattern and the one generated by the `diric` function.

Then, the script is tested under different configurations. Firstly, the initial values of d and θ_1 are kept fixed, while four different values of M are selected within the range [2, 50] to evaluate the effect of the number of antennas on the array pattern. The chosen values are `M = [2, 15, 30, 50]`. For each configuration, the previously described beamforming pattern is computed and visualized using polar plots. All the results are displayed in a 2×2 subplot layout, as shown in Figure 3. As expected, the main lobe's peak increases proportionally to M, confirming that the array factor reaches a maximum of $\sqrt{M}$. According to Figure 3, for $M = 2$, the radiation pattern

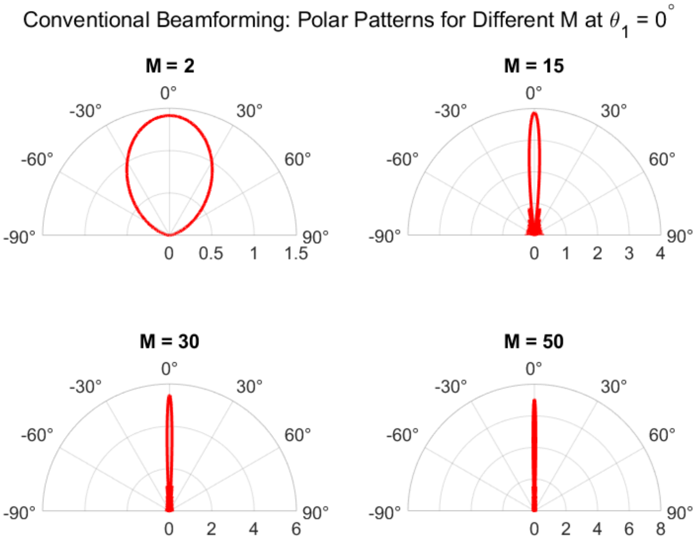


Figure 3: Polar plots of the beamforming pattern for different values of $M \in \{2, 15, 30, 50\}$, with fixed inter-element spacing d and direction of arrival θ_1

consists of a single, very broad main lobe spanning approximately ±40°, with no visible side lobes. A null appears around ±45°. Increasing the number of elements to $M = 15$ results in a much narrower main lobe, extending roughly ±7°, and the emergence of a pair of first-order side lobes at about ±15°, with a peak magnitude of approximately 0.1-0.2 in linear scale (around -13 dB relative to the main lobe). The first nulls occur at approximately ±8°. With $M = 30$, the main lobe becomes even narrower, around ±3°, and multiple side lobes appear, with the first one located near ±6°. For $M = 50$, the main lobe further shrinks to about ±2°, and the first side lobe is observed

at around ±4°.
In general, it is possible to affirm that as M increases, the beam becomes progressively narrower, the number of visible side lobes increases, and the peak of the main lobe increases in magnitude, following a $\sqrt{M}$ trend.

Next, keeping M (M = 8) and the direction of arrival θ_1 fixed, the script is executed for different values of the normalized antenna spacing, specifically $d/\lambda = [0.25, 0.5, 1, 2]$. This analysis allows observing the effect of inter-element spacing on the beamforming pattern, particularly in terms of main lobe width and presence of grating lobes. The resulting polar plots for each configuration are shown in Figure 4.

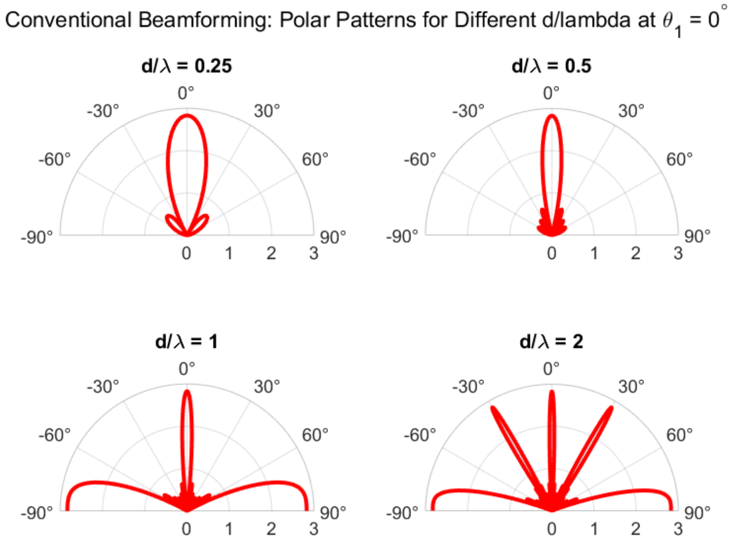


Figure 4: Polar plots of the beamforming pattern for different normalized inter-element spacings $d/\lambda \in \{0.25, 0.5, 1, 2\}$, with fixed number of antennas M and direction of arrival θ_1.

Figure 4 shows how the beamforming pattern changes as the normalized inter-element spacing d/λ varies, while keeping the number of antennas M and the direction of arrival θ_1 constant.

When $d/\lambda = 0.25$, the array elements are closely spaced, resulting in a wide main lobe centered at $\theta = 0°$, with a beamwidth of approximately ±15°. Only two side lobes are visible, and their amplitude is significantly lower than that of the main lobe, making them negligible.

Increasing the spacing to $d/\lambda = 0.5$ causes the main lobe to narrow. However, more side lobes appear, though they remain much smaller in amplitude compared to the main lobe. This behavior is typical as spacing increases: narrower main lobes and more side lobes.
When $d/\lambda = 1$, the pattern changes significantly: additional lobes appear at ±90° with amplitudes comparable to the main lobe. These are grating lobes, which occur when the element spacing reaches or exceeds one wavelength. They are indistinguishable from the main lobe in terms of magnitude and represent undesirable directions where the array radiates with the same strength, thus degrading directional selectivity.
For $d/\lambda = 2$, the array exhibits even more grating lobes distributed symmetrically in the angular domain. All of them have comparable magnitudes, which severely compromises the beamforming performance.

The positions of the main lobes (maxima) of the beamforming pattern can be analytically computed using the following expression:

$$\theta_{\max} = \arcsin\left(\sin(\theta_1) \pm \frac{m}{d/\lambda}\right), \quad m = 0, 1, 2, \ldots$$

The number and position of these maxima depend on the ratio d/λ. The resulting maxima computed for each value of d/λ are shown in Table 1.

Table 1: Angles of maxima for different values of d/λ with $\theta_1 = 0°$

d/λ	$\theta_{\max}$ [degrees]
0.25	0
0.5	0
1	$-90,\ 0,\ 90$
2	$-90,\ -30,\ 0,\ 30,\ 90$

The values in Table 1 perfectly correspond to the one obtained in Figure 4 for different values of $\frac{d}{\lambda}$.

Furthermore, the locations of the nulls in the beamforming pattern can be derived analytically. In particular, we can set the last null to be located at 90°, and solve for the maximum allowed value of d/λ that still ensures null placement at or before that angle.

The nulls of the array factor occur at the angles satisfying:

$$\theta_m^{NULL} = \arcsin\left(\sin(\theta_1) \pm \frac{m}{M} \cdot \frac{\lambda}{d}\right), \quad m = 1, 2, \ldots$$

To find the maximum value of d/λ such that the outermost null is exactly at $\theta_{\text{null}} = 90°$, we rearrange the equation to solve for d/λ:

$$\frac{\lambda}{d}_{\max} = \frac{\sin(\theta_{\text{null,max}}) - \sin(\theta_1)}{\pm m/M}$$

and consequently:

$$\frac{d}{\lambda} = \frac{1}{\left(\frac{\sin(\theta_{\text{null,max}}) - \sin(\theta_1)}{\pm m/M}\right)} = \frac{M}{\sin(\theta_{\text{null,max}}) - \sin(\theta_1)} \cdot \frac{1}{\pm m}$$

In MATLAB code, this logic is implemented as follows:

```
lambda_over_d_max_direct = (sind(theta_null_max)...
- sind(theta_1)) ./ (m_pm/M);
valid = lambda_over_d_max_direct > 0;
d_over_lambda = 1 ./ lambda_over_d_max_direct(valid);
```

with `theta_null_max` as the angle of the last null set to 90°. The values of $\frac{d}{\lambda}$ to have maximum directivity and no grating lobes are:

$$[0.1250, 0.25, 0.375, 0.5, 0.625, 0.75]$$

The analysis continues with the computation of the First Null Beam Width (FNBW). The initial configuration is set with $M = 8$, $d/\lambda = 1/2$, and $\theta_1 = 30°$. The FNBW is computed as the angular separation between the first nulls around the main lobe, using the following formulas:

$$\theta_{\text{null}}^{(+1)} = \arcsin\left(\sin(\theta_1) + \frac{1}{(d/\lambda) \cdot M}\right), \quad \theta_{\text{null}}^{(-1)} = \arcsin\left(\sin(\theta_1) - \frac{1}{(d/\lambda) \cdot M}\right)$$

$$\text{FNBW} = \left|\theta_{\text{null}}^{(+1)} - \theta_{\text{null}}^{(-1)}\right|$$

The resulting FNBW is:

$$\text{FNBW} = 34.11°$$

Next, the main lobe direction is changed to $\theta_1 = 60°$, and the goal is to find the minimum number of elements M such that the first null on the right side remains below 90°. This is implemented with the following MATLAB code:

```
theta_1 = 60;
M_options = 1:20;
for M = 1:length(M_options)
    theta1_null = asind(sind(theta_1) + 1/(d_lambda)/M);
    if theta1_null < 90
        fprintf('Minimum M: %d, Theta_null_1: %f \n', M, ...
        theta1_null)
        break
    end
end
```

The script incrementally tests values of M from 1 to 20, computing the position of the first null on the right of the main lobe for each case. The loop terminates as soon as the null position falls below 90°, identifying the minimum number of elements required to satisfy this constraint. The result obtained from the loop is:

$$\text{Minimum } M = 15, \quad \theta_{\text{null}} = 87.95^\circ$$

Afterwards, the same initial parameters as in Step 1 are set, with θ_1 representing the direction of arrival (DoA) of the user of interest (UE). A set of five interferer DoAs is defined as follows: $\theta = [20^\circ, -40^\circ, 60^\circ, -75^\circ, 80^\circ]$. The steering matrix $\mathbf{A} \in C^{M\times 6}$ is then constructed, containing the steering vectors corresponding to both the UE and the interferers.

The spatial covariance matrix $\mathbf{R}_y$ of the signal plus noise and interference is computed as:

$$\mathbf{R}_y = \mathbf{A}(\theta)\mathbf{A}(\theta)^H + \sigma_n\mathbf{I}, \quad \text{with } \sigma_n = 10^5$$

The MVDR beamformer $\mathbf{w}_{\text{MVDR}}$ is computed as:

$$\mathbf{w}_{\text{MVDR}} = \frac{\mathbf{R}_y^{-1}\,\mathbf{a}(\theta_1)}{\mathbf{a}(\theta_1)^{\text{H}}\,\mathbf{R}_y^{-1}\,\mathbf{a}(\theta_1)}$$

The beam pattern is obtained as in the previous steps and plotted using only the `polarplot` function. Additionally, straight dashed lines are drawn at the interferers' DoAs, spanning from the origin to the peak value of the pattern. The resulting plot is shown in Figure 5.

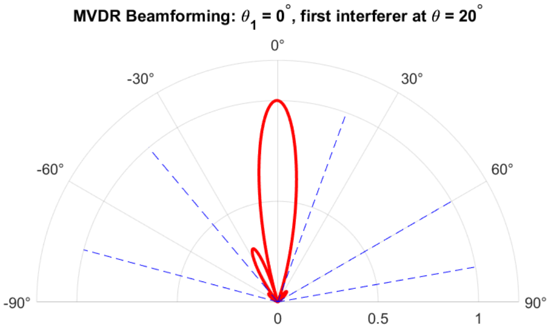


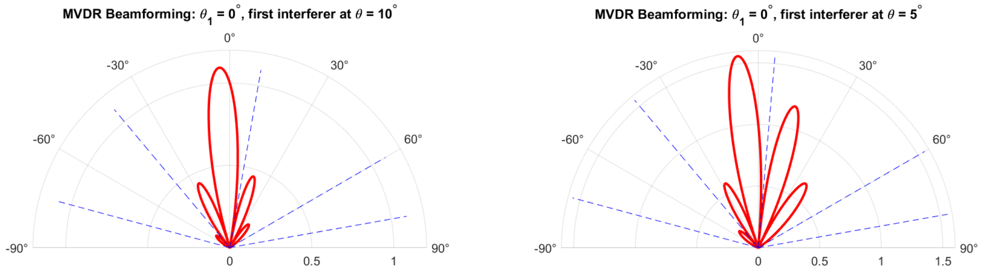
Figure 5: MVDR beam pattern with user of interest at $\theta_1 = 0°$ and interferers at $[20°, -40°, 60°, -75°, 80°]$ (shown by dashed lines).

From Figure 5, it is clear that there is unity gain at the desired direction of $\theta = 0°$. The main lobe is relatively narrow, ensuring good spatial resolution, while several smaller side lobes are present, which actually remain very small in magnitude, around 0.1-0.2 (so in dB scale around -20 dB to -14 dB). Moreover, the beamformer places deep nulls at the interferers' directions, effectively suppressing their contributions.

The analysis is repeated by modifying the DoA of the first interferer. Specifically, two additional cases are considered, with $\theta_{\text{int},1} = 10°$ and $\theta_{\text{int},1} = 5°$, respectively. The corresponding MVDR beam patterns are computed and shown in Figure 6.

Figure 6, in comparison with Figure 5, shows how the performance of the MVDR beamformer degrades as an interferer is placed increasingly close to the desired direction at $\theta = 0°$. When the first interferer is located at 20°, the beamformer generates well-defined nulls in the directions of all interferers, while maintaining a relatively narrow and symmetric main lobe around broadside, even though with reduced peak gain compared to the interference-free case. However, as the interferer moves to 10° and then to 5°, the proximity of the null constraint to the main lobe significantly distorts the beam pattern.

Specifically, the main lobe broadens, from approximately 3°-4° at 20° to 5°-6° at 10°, and even wider when the interferer is at 5°, and rotates slightly



(a) MVDR beam pattern with user of interest at $\theta_1 = 0^\circ$ and interferers at $[10^\circ, -40^\circ, 60^\circ, -75^\circ, 80^\circ]$ (shown by dashed lines).

(b) MVDR beam pattern with user of interest at $\theta_1 = 0^\circ$ and interferers at $[5^\circ, -40^\circ, 60^\circ, -75^\circ, 80^\circ]$ (shown by dashed lines).

Figure 6: Effect of varying the DoA of the first interferer on the MVDR beam pattern. Dashed lines indicate the locations of the interferers.

away from the interferer. The side lobes also become more pronounced: at 10°, they reach magnitudes of about 0.2 (linear scale), and at 5° one of the side lobes becomes comparable in magnitude to the main lobe. These effects highlight how MVDR, in its attempt to minimize total output power in the presence of closely spaced interferers, sacrifices directivity and increases sidelobe levels, ultimately compromising beam quality and angular resolution.

Finally, an extreme interference scenario is constructed by placing 31 interferers uniformly between -90° and $+90^\circ$, including the user's broadside angle at 0° among them:

$$\theta = \left[0^\circ,\ \text{linspace}(-90, 90, 31)\right].$$

The steering matrix therefore consists of one column for the user's steering vector $a(\theta_1)$ and 31 columns for the interferers. The spatial covariance matrix of the signal plus noise and interference is formed as

$$R_y \;=\; A\,A^H \;+\; \sigma^2 I_M,$$

where the noise variance is $\sigma^2 = 10^{-5}$ and I_M is the $M \times M$ identity. The MVDR weight vector is computed via

$$w_{\text{MVDR}} \;=\; \frac{R_y^{-1}\, a(\theta_1))}{\left\|\, a(\theta_1))^H\, R_y^{-1} \right\|_2},$$

that is, the interference-plus-noise covariance matrix inverse is applied to the user's steering vector and then normalized by its 2-norm. Division by this norm ensures unit gain in the desired direction despite the covariance matrix's bias toward directions occupied by interferers, which includes $\theta = 0°$.

The beam pattern is evaluated over $S = 3601$ sample angles, equally spaced through the `linspace()` function between $-90°$ and $+90°$. First, each steering vector $a(\theta_i)$ is computed as

```
a_theta = exp(1j*2*pi*d/lambda*m*sind(theta_val(i)));
```

where `theta_val` is the vector containing the S sample angles and $m = (0 : M - 1)$ indexes the antenna elements. Each of these vectors will constitute a column of the matrix V.

Each steering vector $a(\theta_i)$ is generated within a simple `for` loop and stored as a column of the matrix V. In code:

```
for k = 1:S
    a_theta = exp(1j*2*pi*d/lambda*m*sind(theta_val(k)));
    V(:,k)  = a_theta;
end
```

Here, `theta_val` is the vector containing the S sample angles and $m = (0 : M - 1)$ indexes the antenna elements.

Finally, it is possible to obtain the beam pattern as the product between the hermitian of the weights vector w, i.e. the complex conjugate transpose, and the matrix V; this is implemented through the following lines of code:

```
w_hermitian = w_MDVR';
pattern = w_hermitian*V;
```

The resulting MVDR beam pattern with 31 interferers is plotted alongside the conventional (fixed-weight) beamformer in Figure 7. The MVDR result appears as a solid red curve, while the conventional pattern is shown in blue dashed lines.

In Figure 7, both MVDR and conventional beamformers produce a narrow main lobe around 0°, with comparable peak levels and similar first sidelobe amplitudes. The conventional method tends to show slightly more energy in the second and third sidelobes, while MVDR offers some improvement in suppressing these. Although MVDR provides marginally better sidelobe

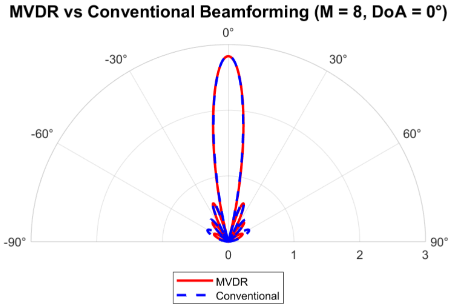


Figure 7: MVDR (red solid) vs conventional beamformer (blue dashed) beam pattern with user of interest at $\theta_1 = 0°$ and 31 interferers uniformly spaced on $[-90°, +90°]$.

control, there is not a significant improvement with respect to traditional beamforming. The resulting beam pattern, characterized by multiple lobes and limited null regions, suggests a reduced ability to distinguish between desired and interfering signals when the array is overloaded, struggling to maintain effective spatial filtering.

3 Beamforming techniques with planar arrays

In this exercise, we explore the same beamforming techniques of above in the planar space (3 dimensions). The general parameters that were left the same in all the simulation were the frequency $f = 6\ GHz$ and, of course the speed of light $c = 3 \cdot 10^8\ m/s$. From this two parameters we were able to determine the wavelength $\lambda = \frac{c}{f} = 0.05\ m = 5\ cm$. Furthermore, in the few first simulations, we set the number of antennas along the y direction to be equal to the number of antennas along the z direction, having $N_y = N_z = 8$, that results in a having a total of $N_{tot} = N_y \cdot N_z = 64$ antennas. Having now the antennas placed along two direction, we need to define the spacing in both those directions:

$$d_y = d_z = \frac{\lambda}{2} = 0.025\ m = 2.5\ cm$$

In our design, the chosen value for the spacing is the maximum value allowed by the sampling theorem in antenna array design, which requires $d \leq \frac{\lambda}{2}$.

To move from a Uniform Linear Array (ULA) to a Uniform Planar Array (UPA) approach, we introduced the directivity function. In this way the Array Factor (AF) can represents more accurately the spatial response of the array in both azimuth (phi, y axis) and elevation planes (theta, z axis). The directivity function formula is reported in Equation 1; we decided to define it in our script as an inline function of the variables `theta phi` for ease of use-ness.

$$d(\theta, \phi) = 0.25 \cdot [1 - cos(2\theta)] \cdot [1cos(\phi)] \tag{1}$$

In our simulation, the Array Factor has to be multiplied with the directivity only in the case that the variable `directive` is set to one, otherwise, the directivity d is set to one (`d = @(theta, phi) ones(size(theta));`.

The beamforming array $a(\theta, \phi)$ is defined as the Kronecher product between the vector $a_z(\theta)$ and $a_y(\theta, \phi)$. In MATLAB, there is the `kron` build-in function, that returns the Kronechen product. The z-axis steering vector is defined equivalently to the one already defined for the linear case:

$$\mathbf{a_y}(\Psi) = \begin{bmatrix} 1 \\ e^{-j\xi_z} \\ e^{-j2\xi_z} \\ \vdots \\ e^{-j(N_z-1)\xi_z} \end{bmatrix}$$

with $\xi_z = 2\pi\frac{d_z}{\lambda}cos(\theta)$. The steering vector along the *y-axis* depends on both theta and phi angles:

$$\mathbf{a_y}(\theta, \phi) = \begin{bmatrix} 1 \\ e^{-j\xi_y} \\ e^{-j2\xi_y} \\ \vdots \\ e^{-j(N_y-1)\xi_y} \end{bmatrix}$$

with $\xi_y = 2\pi\frac{d_y}{\lambda}sin(\theta)sin(\phi)$. Finally, the final array response can be defined as:

$$\mathbf{a}(\theta, \phi) = \begin{bmatrix} 1 \\ e^{-j(\xi_y+\xi_y)} \\ e^{-j2(\xi_y+\xi_y)} \\ \vdots \\ e^{-j(N_y-1)(\xi_y+\xi_y)} \end{bmatrix}$$

The beam steering w is set to compensate for a phase corresponding to the DoA (in the simulation $(\theta_{UE},\ \phi_{UE})$) is defined as:

$$w = \frac{a(theta_{UE},\ \phi_{UE})}{\sqrt{N_y N_z}}$$

Using a nested `for loop`, reported below, we calculated the steering vector for all the possible combination pair of the angle θ and ϕ.

```
for i = 1:length(theta_deg)
    az = a_z(theta_deg(i));
    for j = 1:length(phi_deg)
        ay = a_y(theta_deg(i), phi_deg(j));
        V(:, i, j)  = kron(ay, az);
    end
end
```

where *theta_deg* is a vector containing the angles in degree in the range $[0, 180]$ with step 0.5, thus 361 elements, and similarly *phi_deg* contains all the angle in the range $[-180, 180]$ with the same step of 0.5, for a total of 721 elements. The result are saved in 3-D matrix, of dimension $N_{tot} \times 361 \times 721$.

Another `for loop` was then employed the beamforming vector w to all the steering vector. Since all the steering vector are contained in one 3D matrix, the command `squeeze` was used to rewrite a $N_{tot} \times 1 \times 721$ matrix into a $N_{tot} \times 721$ matrix, which allow standard matrix multiplication; following the MATLAB code that generated the UPA pattern.

```
for i = 1:length(theta)
    Pattern(i, :) = w' * squeeze(V(:, i, :));
end
```

To obtain the final UPA pattern, the result computed using the last `for` loop must be multiplied by the directivity function d. The angles used to evaluate

d are meshgrid matrices, representing the full range of azimuth and elevation angles over which the pattern is computed. These matrices were obtained using the following line of code:

```
angles_1 = 361;
theta = linspace(0, pi, angles_1);
angles_2 = 721;
phi = linspace(-pi, pi, angles_2);
[PHI, THETA] = meshgrid(phi, theta);
```

All our analysis has been carried out using spherical coordinates, since on MATLAB there is not a function that allows to plot graphs with the elements saved as polar coordinates, the results are all converted in Cartesian coordinates, using the general formulas reported below:

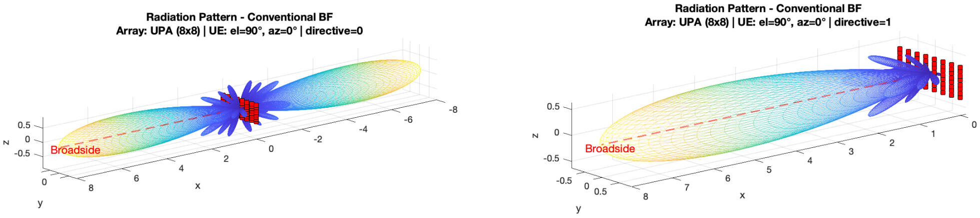
$$\begin{aligned} x &= r \cdot \sin(\theta)\cos(\phi) \\ y &= r \cdot \sin(\theta)\sin(\phi) \\ z &= r \cdot \cos(\theta) \end{aligned}$$

with $r = \sqrt{G}$, where G is the array gain, defined as the absolute value squared of the pattern array. This conversion is also used to derive the broadside direction, that in spherical coordinates is $\theta = 90,\ \phi = 0$, which correspond to $x = r,\ y = 0,\ z = 0$. The broadside direction is needed to see the effect of the directive antenna.

Our simulation was tested with different angles, leading to different results. Here a more detailed analysis.

3.1 Case 1: $\theta_{UE} = 90,\ \phi_{UE} = 0$

In the first case, we set the direction of arrival, referred to in the simulation as the User Equipment (UE) to the broadside direction. Figure 8 illustrates the resulting radiation patterns for both the isotropic antenna (on the left) and the directive antenna. As expected, the isotropic antenna radiates uniformly in all directions, producing a symmetric pattern with respect to the origin. In contrast, the directive antenna concentrates its radiation along the positive half of the x-axis, significantly reducing emissions in other directions. Essentially, the directive pattern can be seen as a filtered version of the isotropic one, emphasizing transmission in a preferred direction while suppressing it elsewhere. Other than the main lobe in the broadside direction, many smaller lobes are present in the various directions. It is important to note that, since a relatively high number of antenna elements is used ($N_{\text{tot}} = 64$), the array does not behave as an ideal point-like isotropic source. Even in the isotropic case, the resulting pattern is shaped by the array configuration, resulting in a main lobe along the broadside direction rather than a perfect spherical radiation pattern.



(a) Isotropic antenna along broadside direction

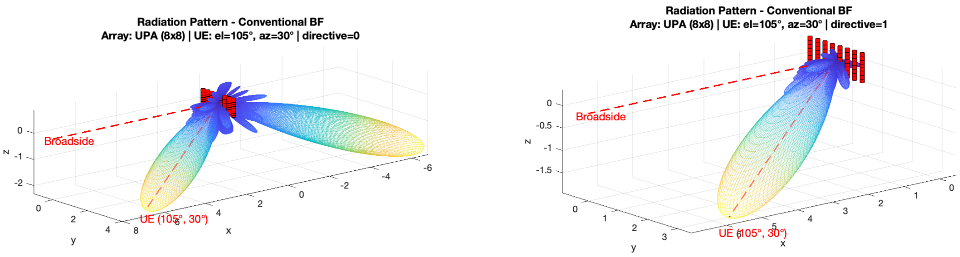
(b) Directive antenna along broadside direction

Figure 8

3.2 Case 2: $\theta_{UE} = 105, \ \phi_{UE} = 30$

After understanding how the different antennas behaves, we change the direction of transmission, to see whether the antenna behaves similarly or other patterns exists. In this first case the elevation angle (vertical angle between the direction of interest and the horizontal plane) $\theta_{UE} = 105$, while the azimuth angle (horizontal angle measured clockwise from the *x-axis*) $\phi = 30$.

Figure 9 shows the radiation pattern of both antenna when the elevation and azimuth angles are changed. We notice that the behaviour of the isotropic and of the directive antenna is the same as before, what has changed is the direction of the array pattern: before it was along the broadside direction, now it is aligned with the specified input angle (highlighted by a dotted line in the plot). As before, the pattern consists of a main lobe, and many smaller lobe around it; the main lobe is aligned with the desired direction, while the side lobes points in various casual directions.



(a) Isotropic antenna along ($\theta = 105,\ \phi = 30$) direction

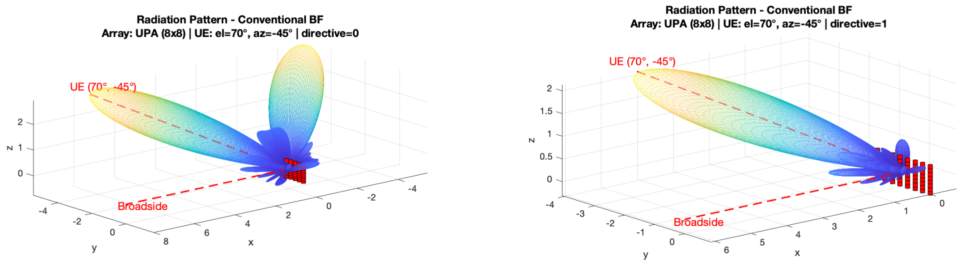
(b) Directive antenna along ($\theta = 105,\ \phi = 30$) direction

Figure 9

3.3 Case 3: $\theta_{UE} = 70,\ \phi_{UE} = -45$

This simulation differs from the previous one only by the angle pair that define the direction of propagation. For this simulation we set the elevation angle $\theta = 70$ and the azimuth angle $\phi = -45$.

The patterns shown in Figure 10 completely met our expectations: the directive antenna pattern is equivalent to the left side of the isotropic antenna pattern (which is the side along positive *x-axis* component). As before, the patterns present the main lobe, aligned toward the user specified direction and some smaller lobes.



(a) Isotropic antenna along ($\theta = 70,\ \phi = -45$) direction

(b) Directive antenna along ($\theta = 70,\ \phi = -45$) direction

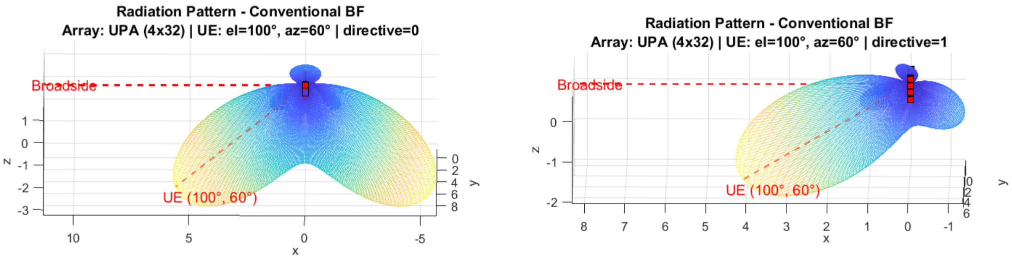
Figure 10

3.4 Case 4: $\theta_{UE} = 70$, $\phi_{UE} = -45$ and $N_y = 32$, $N_z = 4$

For this last simulation, other than varying the angle of transmission, $\theta = 100$, $\phi = 60$, we also changed the number of element along both the *y-axis* and the *z-axis*. Before, we had $N_y = N_z = 8$, $N_{tot} = 64$, now $N_y = 32 N_z = 4$, $N_{tot} = 128$. This modification affects the beamforming capabilities of the array. Since more elements are distributed along the y-axis, our expectation it to have the pattern highly directive in the azimuthal plane (along the y-axis), while, the reduced number of elements along the z-axis, results in a broader, less directive pattern along the elevation plane.

The result reported in Figure 11 shows exactly this behavior: along the azimuthal plane the direction is really precise, along the elevation plane we obtain large lobes.

As in all the previous case, the pattern of the directive antenna is half of the pattern for the isotropic one; side lobes are present.



(a) Isotropic antenna along ($\theta = 100$, $\phi = 60$) direction with $N_y = 32$, $N_z = 4$

(b) Directive antenna along ($\theta = 100$, $\phi = 60$) direction with $N_y = 32$, $N_z = 4$

Figure 11

3.5 MVDR Beamforming for a Uniform Planar Array

In this part of the exercise, the objective is to implement and analyze the MVDR beamformer for a 16×16 uniform planar array (UPA) operating at $f = 6\,\text{GHz}$ ($\lambda = 0.05\,\text{m}$). As above, the array elements are arranged on a rectangular grid with half-wavelength spacing in both the y and z directions:

$$d_y \;=\; d_z \;=\; \frac{\lambda}{2} \;=\; 0.025 \text{ m}$$

so that $N_y = N_z = 16$ and $N_{\text{tot}} = 256$. A set of five interferers is placed at elevation angles

$$\theta_{\text{int,deg}} = [\,86^\circ,\ 85^\circ,\ 80^\circ,\ 100^\circ,\ 105^\circ\,], \quad \phi_{\text{int,deg}} = [\,4^\circ,\ 20^\circ,\ 5^\circ,\ -15^\circ,\ 15^\circ\,]$$

each converted to radians for computation. The noise variance is chosen as $\sigma_n^2 = 10^{-5}$. The user (UE) direction is specified by two angles $\theta_1 = 90^\circ$ (elevation) and $\phi_1 = 0^\circ$ (azimuth) in degrees; these are converted to radians before constructing the steering vector.

Construction of the MVDR Weight Vector

The steering vectors along the z- and y-axes are set as $\mathbf{a}_z(\theta)$ and $\mathbf{a}_y(\theta, \phi)$, as defined above. By taking the Kronecker product of these two subvectors, the steering vector toward the user at (θ_1, ϕ_1) is

$$\mathbf{a}(\theta_1, \phi_1) \;=\; \mathbf{a}_z(\theta_1) \;\otimes\; \mathbf{a}_y(\theta, \phi_1) \tag{2}$$

Next, define the element directivity function as

$$d(\theta, \phi) \;=\; 0.25\,\big[\,1 - \cos(2\theta)\big]\;\big[\,1 + \cos\phi\big], \tag{3}$$

which applies only if `directive=1`; otherwise, $d(\theta, \phi) = 1$. For each interferer indexed by $k = 1, \ldots, 5$, with angles $\big(\theta_{\text{int},k},\ \phi_{\text{int},k}\big)$, the scaled steering vector is

$$\mathbf{a}_{\text{int},k} \;=\; d\big(\theta_{\text{int},k},\ \phi_{\text{int},k}\big)\,\mathbf{a}\big(\theta_{\text{int},k},\ \phi_{\text{int},k}\big).$$

Hence, the (256×6) matrix of source steering vectors is assembled as

$$\mathbf{A} \;=\; \big[\,\mathbf{a}(\theta_1, \phi_1)\ \big|\ \mathbf{a}_{\text{int},1}\ \big|\ \cdots\ \big|\ \mathbf{a}_{\text{int},5}\big].$$

The interference-plus-noise covariance matrix is then formed by

$$\mathbf{R}_y \;=\; \mathbf{A}\,\mathbf{P}\,\mathbf{A}^H \;+\; \sigma_n^2\,\mathbf{I}_{256}, \quad \sigma_n^2 = 10^{-5} \text{ and } \mathbf{P} = diag\{P_1, ..., P_k\}, \forall\ P_k = 1 \tag{4}$$

Finally, the MVDR weight vector is given by

$$\mathbf{w}_{\text{MVDR}} \;=\; \frac{\mathbf{R}_y^{-1}\,\mathbf{a}(\theta_1, \phi_1)}{\big\|\,\mathbf{R}_y^{-1}\,\mathbf{a}(\theta_1, \phi_1)\big\|_2}. \tag{5}$$

In MATLAB, this computation is executed by forming $\mathbf{A}$ column-by-column, computing `R_y = A * A' + sigma_n2 * eye(Ntot);` and then `w_MVDR = (R_y a_1) / norm(R_y  a_1, 2);` so that $\mathbf{w}_{\text{MVDR}}$ always yields unit gain in the user's direction.

Generation of the 3-D MVDR Pattern

The next step is to compute the beamformer output over a dense (361×721) grid of elevation and azimuth angles, i.e.

$$\theta_{\text{grid}} \;=\; \text{linspace}(0, \pi, 361), \quad \phi_{\text{grid}} \;=\; \text{linspace}(-\pi, \pi, 721).$$

The element-wise directivity factor is computed on the same grid by

```
directivity = d(THETA, PHI);
directivity_3D = repmat(directivity, [1,1,Ntot]);
directivity_3D = permute(directivity_3D, [3,1,2]);
```

where $N_{\text{tot}} = N_y\,N_z = 256$. Reordering via `permute`$([3, 1, 2])$ yields a $[256 \times 361 \times 721]$ array so that `directivity_3D`$(k, i, j) = d(\theta_i, \phi_j)$ for antenna element index k. Next, all UPA steering coefficients are computed at once without the need of a double for loop by vectorizing the angle pairs generated with `meshgrid()`, in code:

```
vectorized_THETA = THETA(:).';
vectorized_PHI = PHI(:).';
```

Then, the inline functions `a_z` and `a_y` to compute the steering vectors, previously defined, functions are called on the row vectors `vectorized_THETA`, `vectorized_PHI` to yield

```
az = a_z(vectorized_THETA);
ay = a_y(vectorized_THETA, vectorized_PHI);
```

Because the defined functions implicitly multiply $n = [0 : 15]'$ or $m = [0 : 15]'$ with the $1 \times K$ angle arrays, the results are $[16 \times K]$ matrices. Each column k corresponds to a pair (θ_i, ϕ_j) in increasing angles order. To form the full UPA steering vector $\mathbf{a}(\theta_i, \phi_j) = a_z(\theta_i) \otimes a_y(\theta_i, \phi_j)$ for all (i, j), one reshapes these results into three dimensions and multiplies elementwise:

```
az_kron = reshape(az, [Nz, 1, K]);
ay_kron = reshape(ay, [1, Ny, K]);
```

```
V_temp = az_kron .* ay_kron;
V = reshape(V_temp, [Ntot, N_theta, N_phi]);
```

The array $\texttt{V}(k, i, j)$ now equals the k-th entry of $\mathbf{a}(\theta_i, \phi_j)$. Multiplying by $\texttt{directivity\_3D}(k, i, j)$ produces the effective steering array:

$$\texttt{pattern\_dir}(k, i, j) \;=\; d(\theta_i, \phi_j)\,\big[\mathbf{a}(\theta_i, \phi_j)\big]_k,$$

computed in MATLAB through an element-wise multiplication between the corresponding arrays.

Once pattern_dir is available, the complex beamformer response at each (i, j) is given by

$$AF_{MVDR}(i, j) \;=\; \mathbf{w}_{\mathrm{MVDR}}^H\,\mathbf{s}_{i,j}, \quad \mathbf{s}_{i,j} = \texttt{pattern\_dir}(:, i, j),$$

where $\mathbf{w}_{\mathrm{MVDR}}$ is the 256×1 weight vector. In code, each slice $\texttt{pattern\_dir}(:, i, :)$ is first squeezed to size $[256 \times 721]$, then multiplied by $\mathbf{w}_{\mathrm{MVDR}}^H$ to form a 1×721 row. Iterating over $i = 1 \ldots 361$ yields the entire complex response matrix:

```
TotalPattern_MVDR = zeros(N_theta, N_phi);
    for i = 1:N_theta
        V_slice = squeeze(pattern_dir(:, i, :));
        TotalPattern_MVDR(i, :) = w_MVDR' * V_slice;
    end
Gain_MDVR = abs(TotalPattern_MVDR).^2;
```

Finally, the array gain Gain_MDVR is also computed as the square root of the absolute value of the final pattern array. To visualize this as a 3-D surface, one lets the radius at each direction be

$$r(i, j) \;=\; \sqrt{G_{\mathrm{MVDR}}(i, j)}$$

where G_{MVDR} is the array gain Gain_MDVR and applies the standard spherical-to-Cartesian transform

$$X(i, j) \;=\; r(i, j)\,\sin\!\big(\theta_i\big)\,\cos\!\big(\phi_j\big), \quad Y(i, j) \;=\; r(i, j)\,\sin\!\big(\theta_i\big)\,\sin\!\big(\phi_j\big), \quad (6)$$

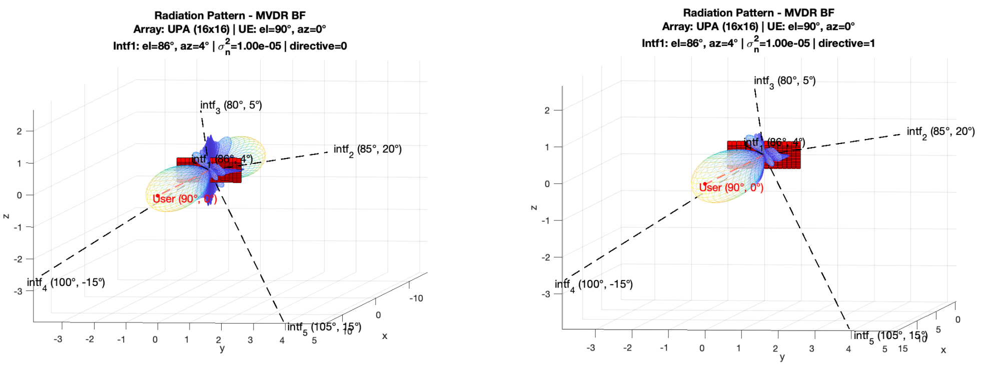
$$Z(i, j) \;=\; r(i, j)\,\cos\!\big(\theta_i\big)$$

In MATLAB, these arrays are computed via elementwise operations on $Pattern_{\mathrm{MVDR}}$ and the meshgrid arrays `THETA, PHI`. Finally, a 3-D mesh is plotted using:

```
mesh(X,Y,Z,r_C,'FaceAlpha',0.5,'EdgeAlpha',0.5)
```

where `r_C` = $\sqrt{X^2 + Y^2 + Z^2}$ serves as the color mapping. The antenna element positions are overlaid as red markers in the y-z plane.

The beamforming pattern obtained with the first interferer located at (86°, 4°) is shown in Figure 12. The plots display both the isotropic and directive array responses. From the figure, it is evident that the main lobe is correctly steered toward the desired user's direction, maximizing the array gain in that direction. Moreover, the presence of interferers is clearly indicated by dashed lines, marking their respective azimuth and elevation angles. It is clear that the beamforming is able to maintain high directivity in the user direction of interest while minimizing the gain toward the interfering sources.



(a) Isotropic array pattern.

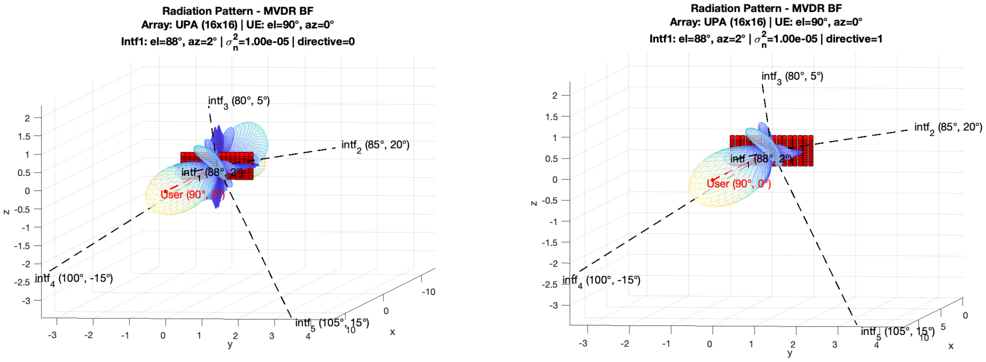
(b) Directive array pattern.

Figure 12: Beamforming pattern for one interferer located at (86°, 4°). Both the isotropic and directive responses are shown. Dashed lines indicate the directions of the interferers.

Interferer Close to the User: $(\theta, \phi) = (88°, 2°)$

To investigate the effect of an interferer that lies very near the user's direction, the first interferer is relocated from $(86°, 4°)$ to $(88°, 2°)$. All other parameters remain unchanged. In practice, this forces the MVDR algorithm to place a deep null only 2° away from broadside both in azimuth and elevation. After reconstructing the matrix $\mathbf{A}$ and recomputing $\mathbf{R}_y$, the MVDR weights are obtained as in (5), and the 3-D pattern is regenerated.

The result obtained with the first interferer at (88°, 2°) is shown in Figure 13. As evident by comparing Figure 13 to Figure 12, when the interferer



(a) Isotropic array pattern.

(b) Directive array pattern.

Figure 13: Beamforming pattern for one interferer located at (88°, 2°). Both the isotropic and directive responses are shown. Dashed lines indicate the directions of the interferers.

is very close to the user's direction, in this case at $(88°, 2°)$, the MVDR beamformer cannot keep the main lobe well-shaped and symmetric. In this case, the main lobe becomes noticeably wider both in elevation and azimuth, and the maximum of the pattern is no longer toward the desired direction, but shifted, by little, away from the first interferer. Right to the direction of the first interferer, a side-lobe with non negligible amplitude is present. This highlight how the MVDR beamforming method loses precision as the position of the interferer is placed near the direction of interest.

In the directive case, even though the array pattern provides some attenuation at $(88°, 2°)$, since $d(88°, 2°) = 0.25[1 - \cos(176°)][1 + \cos(2°)]$, the interferer still lies in a region where the single-element gain is relatively high. Because of that, the MVDR beamformer needs to "spend" most of its degrees of freedom to create a deep null in that direction, which causes the main lobe to widen and shift slightly off center.

We observed, comparing the results shown in Figure 12 with the one of Figure 13 that the gain of the array pattern in the latter case is slightly lower. We interpret this as a consequence of the first interferer being located very close to the desired user direction, requiring additional beamforming effort to suppress the adjacent interference. This effort likely leads to power being redirected into the first side-lobe, slightly compromising the overall gain toward the user.

High Noise Variance: $\sigma_n^2 = 10^3$

As the final step, the noise variance is increased from 10^{-5} to 10^3. Since

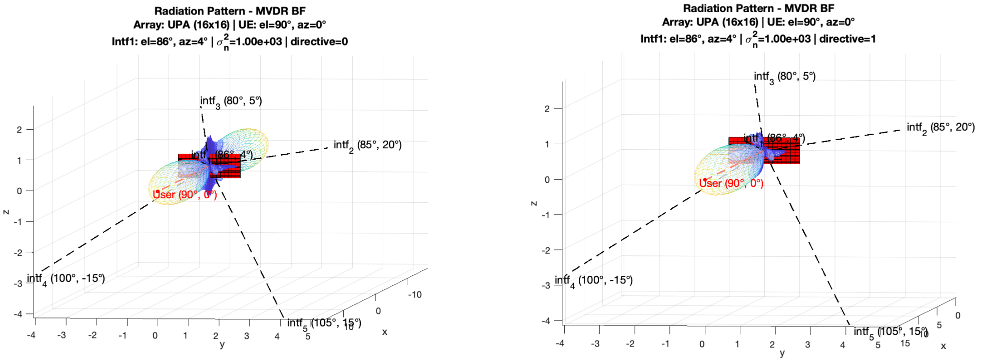
$$\mathbf{R}_y = \mathbf{A}\,\mathbf{A}^H \; + \; 10^3\,\mathbf{I},$$

the covariance matrix becomes dominated by the large diagonal term. In this high-noise regime, the inverse of $\mathbf{R}_y$ approaches $(1/10^3)\,\mathbf{I}$, and the MVDR weights simplify to

$$\mathbf{w}_{\text{MVDR}} \;\approx\; \frac{(10^{-3}\,\mathbf{I})\,\mathbf{a}(\theta_1,\phi_1)}{\big\|\,(10^{-3}\,\mathbf{I})\,\mathbf{a}(\theta_1,\phi_1)\big\|_2} \;=\; \frac{\mathbf{a}(\theta_1,\phi_1)}{\sqrt{256}} \;=\; \mathbf{w}_{\text{conv}}$$

which is precisely the conventional (equal-weight) beamformer. Consequently, when $\sigma_n^2 = 10^3$, the computed 3-D mesh of $\mathit{Pattern}_{\text{MVDR}}$ becomes indistinguishable from the UPA pattern obtained by simply steering toward (θ_1, ϕ_1) with uniform weights. In particular, all deep nulls at the five interferer angles vanish, and the side-lobe structure reverts to the conventional beamforming. This confirms that MVDR provides no additional suppression in the presence of too high noise; it reverts to maximizing array gain toward the user.

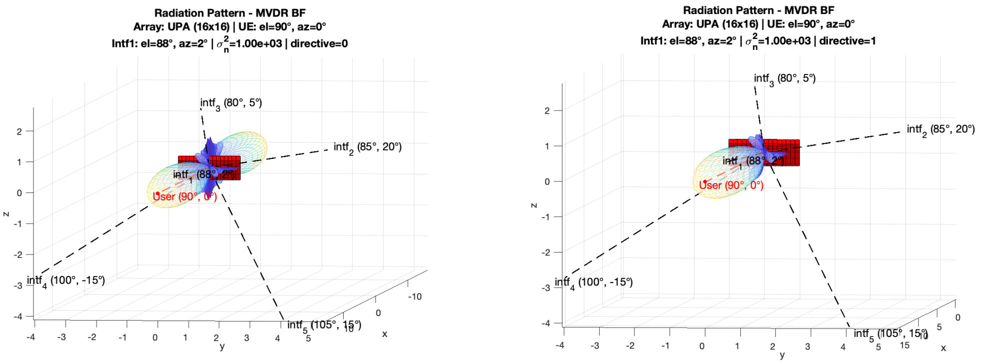
The result obtained with the first interferer at (86°, 4°) is shown in Figure 14, while Figure 15 illustrates the radiation pattern with the first interferer at (88°, 2°), therefore closer to the user.



(a) Isotropic element pattern with high noise.

(b) Directive element pattern with high noise.

Figure 14: MVDR beam pattern with interferer at (86°, 4°) and noise variance $\sigma_n^2 = 10^3$.



(a) Isotropic element pattern with high noise.

(b) Directive element pattern with high noise.

Figure 15: MVDR beam pattern with interferer at (88°, 2°) and noise variance $\sigma_n^2 = 10^3$.

By comparing Figures 14 and 15 with Figure 8, which shows the traditional beamforming pattern, it is easy to see that the MVDR pattern under high noise looks almost the same. All the effects seen before, like the widening of the main lobe, its shift, or the deep nulls in the directions of the interferers, completely disappear. The array just focuses on the user direction, as if it were using traditional beamsteering. This confirms that when the noise is too strong, MVDR can no longer adapt to suppress interference and behaves like a traditional beamformer.

4 Conclusion

Through this analysis, we observed how different parameters influence the behavior of both linear and planar antenna arrays. In particular, it was found that increasing the number of antenna elements improves the array's ability to focus energy in a specific direction, resulting in higher directivity and a narrower main lobe. On the other hand, increasing the inter-element spacing beyond certain limits can lead to the formation of grating lobes, which severely degrade the array's performance by introducing unwanted directions of radiation.

When applying MVDR, we observed a significant improvement in interference suppression. The MVDR beamformer was able to place deep nulls exactly in the directions of the interferers. This makes it particularly effective in scenarios with multiple interferers and limited angular separation.

However, when an interferer is located too close to the direction of the desired user, the MVDR algorithm is forced to compromise between maintaining high gain in the user direction and placing a deep null toward the interferer. This often results in distorted main lobes and increased side lobes. Furthermore, in the presence of very high noise levels, the covariance matrix becomes dominated by noise, and MVDR loses its ability to adapt to the environment. In these conditions, it behaves essentially like a conventional beamformer, simply steering toward the desired direction without null placement.

Overall, these results provide a clear understanding of the trade-offs involved in array signal processing.