# acoustical_hw3

April 20, 2025

# 1 Acoustical Imaging HW3

Carolina Nygren (5479770)

### 1.0.1 Functions

I wrote a flexible function for directivity that includes options for x and y calculations, as well as optionally incorporating the array factor.

[6]:
```
import numpy as np
import matplotlib.pyplot as plt

"""
=========
Functions
=========
"""

def directivity(H, W, N, f, theta_s=0.0, c=1500, x=True, y=True, range=1000,␣
 ↪kerf=0.0):
    """
    Calculate the directivity of a rectangular aperture.

    Parameters
    ----------
    H : float
        Height of the aperture in m.
    W : float
        Width of the aperture in m.
    N : int
        Number of elements in the array.
    f : float
        Frequency in Hz.
    theta_s : float, optional
        Steering angle in rad.
    c : float
        Speed of sound in m/s. Default is 1500 m/s.
    x : bool, optional
```

```
        If True, calculate directivity in x direction. Default is True.
    y : bool, optional
        If True, calculate directivity in y direction. Default is True.
    range : int
        Number of points in the range of theta and phi. Default is 1000.
    kerf : float, optional
        Space between elements in m.

    Returns
    -------
    directivity : np.ndarray
        Directivity of the aperture.
    """

    theta           = np.linspace(-np.pi/2, np.pi/2, range)
    phi             = np.linspace(-np.pi/2, np.pi/2, range)
    wavelength      = c / f
    k               = 2 * np.pi / wavelength
    d               = kerf + W
    psi             = d * k * np.sin(theta_s)

    x_term          = np.sinc(W * np.sin(theta) / wavelength)
    y_term          = np.sinc(H * np.sin(phi) / wavelength)
    array_factor    = (np.sin(N * 0.5 * (k * d * np.sin(theta) - psi)) /
                        ( N * np.sin(0.5 * (k * d * np.sin(theta) - psi))))


    if x == True and y == True:
        directivity = x_term * y_term * array_factor
    elif x == True and y == False:
        directivity = x_term * array_factor
    elif x == False and y == True:
        directivity = y_term * array_factor
    else:
        print("Error: x and y cannot both be False.")
        return None

    return directivity
```

### 1.0.2 Question 3A

I used my function to produce plots for directivity in the x and y directions, with the set parameters. The plot looks as expected, considering W is smaller than H, and thus the W/lambda factor is smaller.

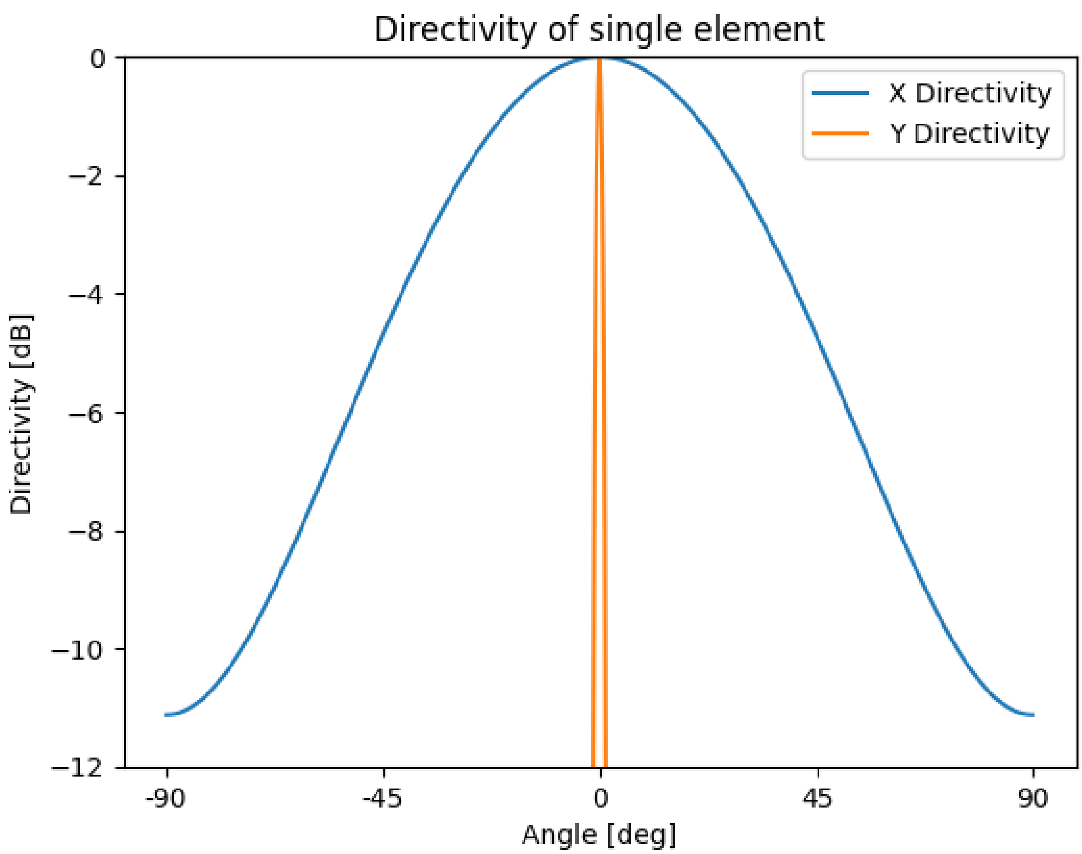
[7]:
```python
"""
==========================================
Question 3A: Fraunhofer directivity plots
==========================================
"""

# Parameters
H = 0.01
W = 0.23e-3
N = 1
f = 5e6

# Directivity in x and y directions
dir_result_x = directivity(H, W, N, f, x=True, y=False)
dir_result_y = directivity(H, W, N, f, x=False, y=True)

# Convert to dB. Clip values to avoid log(0) and set a minimum value to avoid
 ↪-inf
x_directivity_db = 20 * np.log10(np.clip(np.abs(dir_result_x), 1e-12, None))
y_directivity_db = 20 * np.log10(np.clip(np.abs(dir_result_y), 1e-12, None))

# Plotting
plt.plot(x_directivity_db, label='X Directivity')
plt.plot(y_directivity_db, label='Y Directivity')
plt.xticks(np.linspace(0, 1000, 5),
           ['-90', '-45', '0', '45', '90'])
plt.ylim(-12, 0)
plt.title('Directivity of single element')
plt.xlabel('Angle [deg]')
plt.ylabel('Directivity [dB]')
plt.legend()
plt.show()
```




### 1.0.3 Question 3B

Using the same directivity function, a loop runs through each of the steering angles and plots the results.

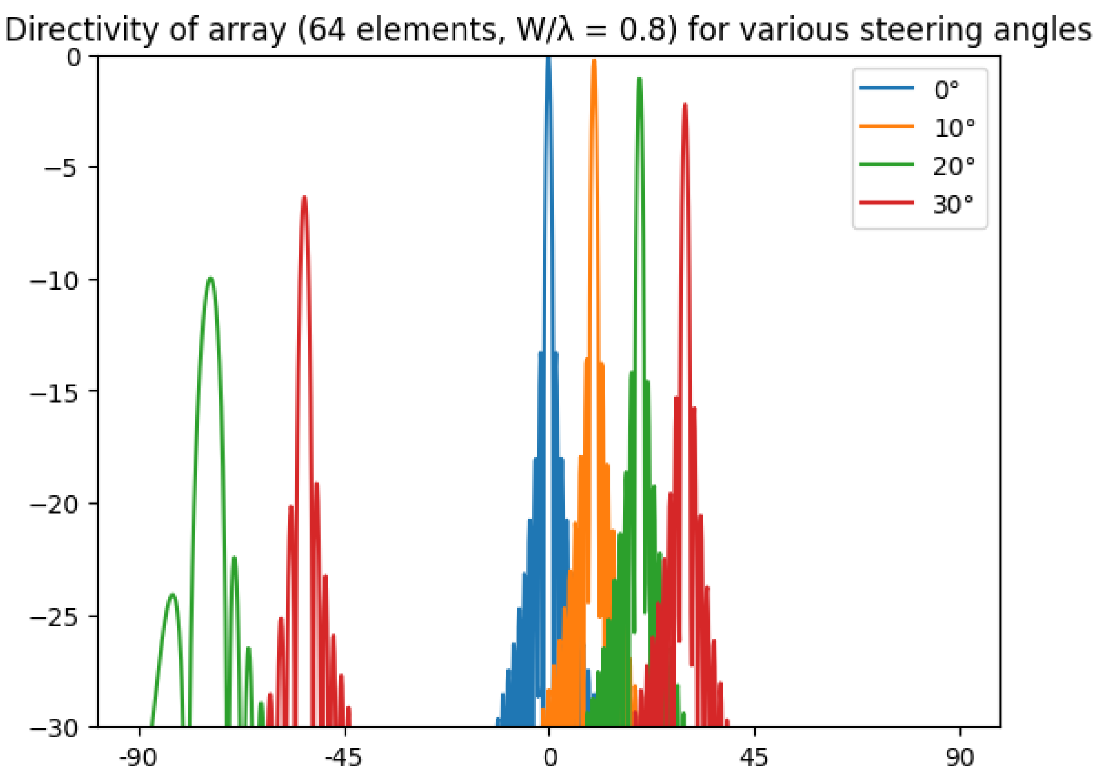
[8]:
```
"""
====================================================
Question 3B: Far-field directivity for entire array
====================================================
"""
# Parameters
H = 0.01
W = 0.23e-3
f = 5e6
N = 64

theta_s_values = np.arange(0, 40, 10)
w_over_lambda = W / (1500 / f)

# Directivity in y (azimuthal) direction at each phase shift
```

```
for i, steering_angle in enumerate(theta_s_values):
    steering_angle_rad = np.deg2rad(steering_angle)
    dir_result = directivity(H, W, N, f, theta_s=steering_angle_rad, x=True,␣
 ↪y=False)
    directivity_db = 20 * np.log10(np.clip(np.abs(dir_result), 1e-12, None))
    plt.plot(directivity_db, label=f'{steering_angle}\u00B0')

# Final plot
plt.title(f'Directivity of array ({N} elements, W/  = {w_over_lambda:.1f}) for␣
 ↪various steering angles')
plt.xticks(np.linspace(0, 1000, 5),
           ['-90', '-45', '0', '45', '90'])
plt.ylim(-30, 0)
plt.legend()
plt.show()
```




### 1.0.4 Question 3C

The maximum value of the main beam directivity becomes less for increased steering angles because, as we introduce a phase difference between the elements in order to have constructive interference in a certain direction, the effective aperture of the array also shrinks. As such, the maximum value reduces accordingly.

The grating lobes occur with increasing steering angle since constructive interference occurs at angles other than the chosen steering angle. With higher element spacing (d > wavelength / 2). The increased steering angle shifts the secondary constructive interference angles into the observation angle.