ECEN 636 Phased Array
Homework 4

October 2, 2022

By: Jacob Ruff

Electrical Engineering Department
Texas A & M University
United States of America

Contents

1 Problem 4

1.1 Matlab Code Method

The Matlab code for this homework problem was broken up into multiple functions and scripts so that they could easily be used in future problems. There were 3 Matlab scripts used for this analysis: homework4_q4.m, pattern_generator.m, uniformarraypattern.m, and directivity.m .

1.1.1 homework4_q4.m

homework4_q4.m is the main script for this problem. It uses the other helper scripts to produce the final graphs and verification of the results. The script starts by defining the element spacings d, total number of elements N, and phase shift $\beta = beta$ to be plotted. It also defines the angle step sizes for pattern simulation ($dtheta$ and $dphi$).

Then for each of the distances and phase shifts, it calls the pattern_generator script with the uniformarraypattern function handle. The produced patterns are then passed to the directivity script which numerically integrates to calculate the directivity. For specific phase shifts, the numerically integrated results are compared to the expected value produced by the directivity formula.

1.1.2 pattern_generator.m

The pattern generator script takes in angle step sizes, a function handle, and varargin (any other necessary parameters for the function handle). The function passed through its function handle (this is achieved using @function_name in the script calling pattern_generator) is the desired pattern function to be evaluated.. The passed function is evaluated over all of θ and ϕ and an array of the pattern and angles is returned to the calling function.

1.1.3 directivity.m

The directivity function takes in θ, ϕ and a corresponding pattern and then numerically integrates to calculate the directivity of that pattern.

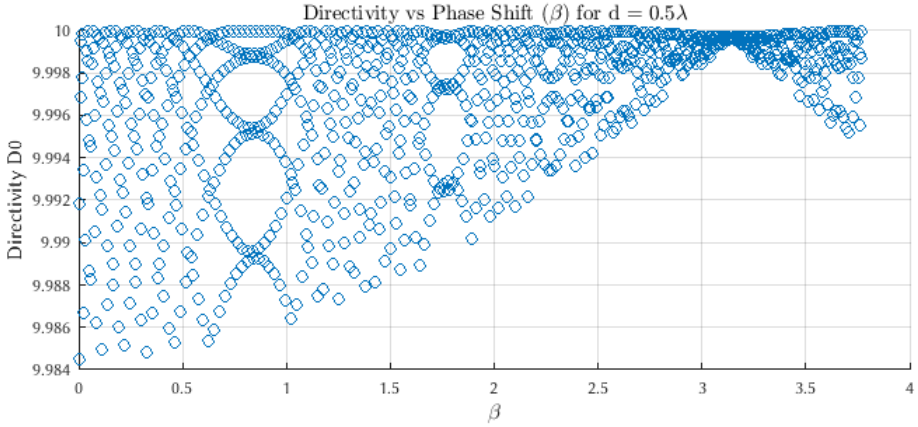
1.1.4 unifrormarraypattern.m

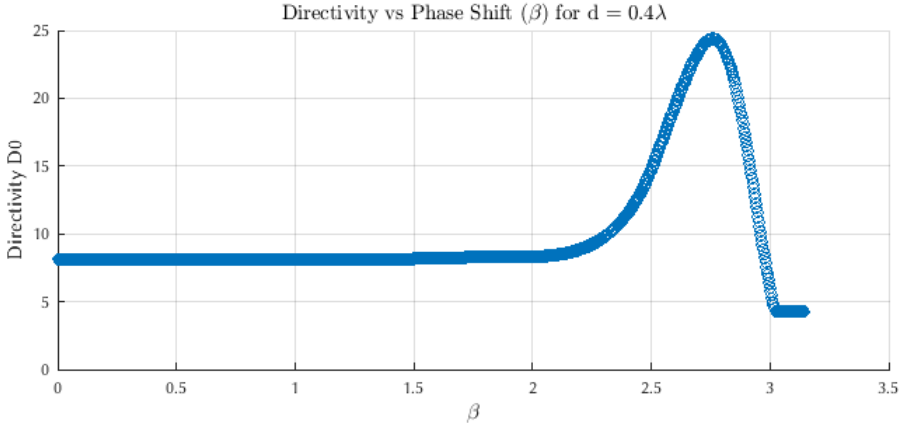
This script returns the array factor for a given uniform array described by the number of elements, element spacing, phase shift, θ, and ϕ.

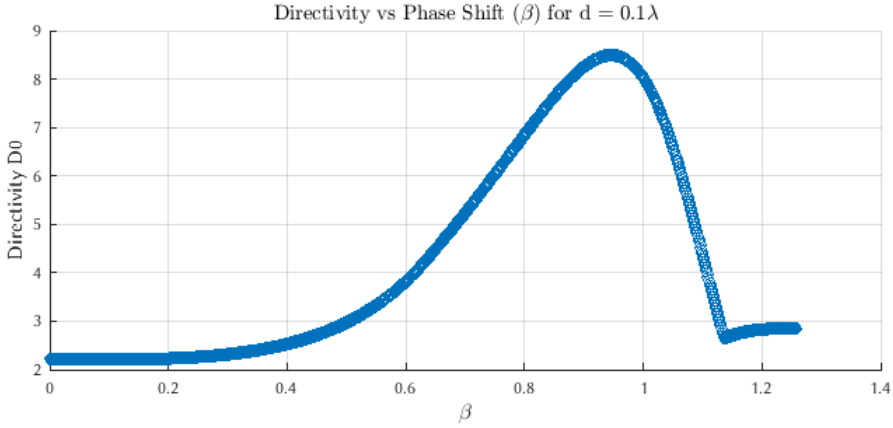
1.2 Results: Directivity vs Phase Shift

The following graphs display the directivity vs phase shift for a 10 element uniform array at element spacings of [0.5 0.4 0.1] λ . They were produced using the numerical integration Matlab script I wrote mentioned above. For these 3 spacings, the directivity for 1000 different phase shifts was calculated.

The one oddity I saw in my results was that for d = 0.5 λ there initially appeared to be a lot of variability from the expected result of a constant 10. Upon a closer look, the max difference was less than 0.015 and the variability could be explained as just a small limitation caused by using a limited number of points for the numerical integration.


Directivity vs Phase Shift (β) for d = 0.5λ
Directivity D0
β


Directivity vs Phase Shift (β) for d = 0.4λ
Directivity D0
β


Directivity vs Phase Shift (β) for d = 0.1λ
Directivity D0
β

1.3 Verification

To verify my results, I programmed a directivity formula based off of solid angles. $D_0 = \frac{4*\pi}{\Omega}$.

$$D = \frac{(norm)}{\frac{1}{N} + \frac{1}{N^2} * \sum_{m=1}^{N-1} \frac{N-m}{mkd} * sin(mkd) * cos(m\beta)} \tag{1}$$

For $\delta = [\beta - k * d, \beta + k * d]$

$$norm = max(\left| \frac{sin(N\delta/2)}{N * sin(\delta/2)} \right|^2) \tag{2}$$

I compared my numerical integration results to those produced by the formula. They align very well with only minor errors. Thus, I feel confident that my numerical integration is working properly.

```
D0_formula =

   10.0000   10.0000   10.0000
    8.0831   15.3371   22.6656
    2.2071    4.1807    8.4912


D0_numericalIntegration =

    9.9845    9.9997    9.9995
    8.0751   15.3368   22.6613
    2.2069    4.1807    8.4907
```