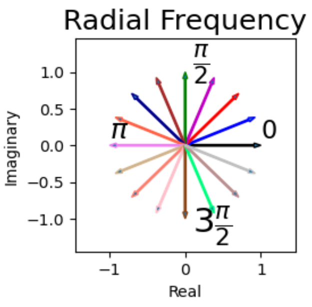
SIGNAL PROCESSING SYSTEM DESIGN LAB

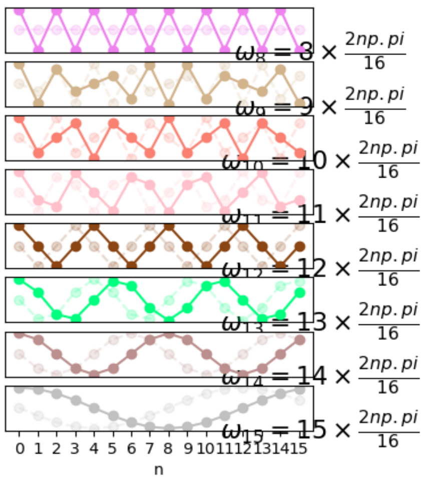
YASH ANAND
(23EE65R22)

3. Discrete-Time Fourier Transform

In this section, we have considered the Discrete-Time Fourier Transform (DTFT) using a matrix/vector approach. We used this approach to develop an intuitive visual vocabulary for the DTFT with respect to high/low frequency and real-valued signals. We used zero-padding to enhance frequency domain signal analysis and examined the consequences of Parseval's theorem.

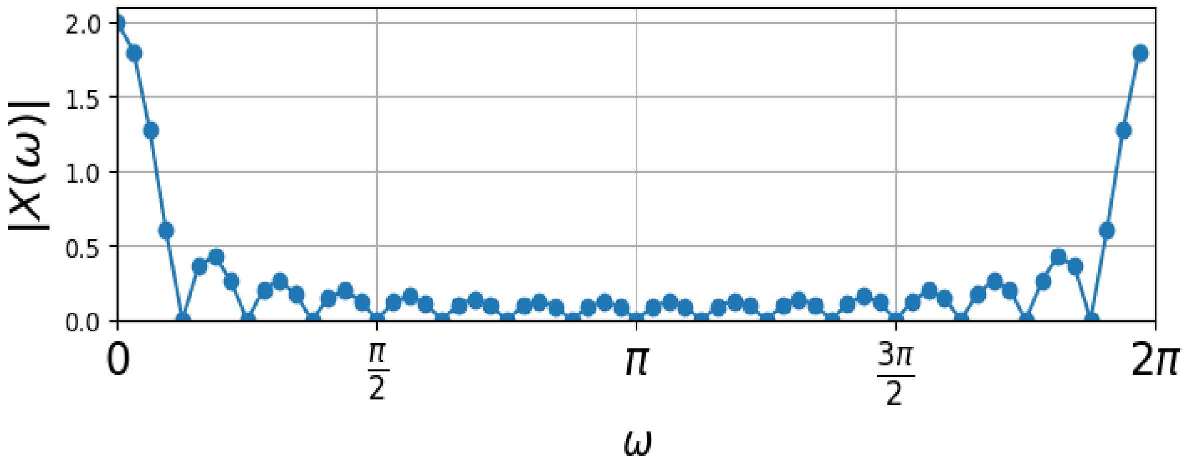
3.1
AIM:
The aim of the code is to visualize the 3D Discrete Fourier Transform (DFT) of a signal. FUNCTION IMPORTED:
FancyArrow from matplotlib.patches is used to create arrows in the 3D plot.
art3d from mpl_toolkits.mplot3d.art3d is used to create 3D plots.
Poly3DCollection from mpl_toolkits.mplot3d.art3d is used to create a collection of 3D polygons.
gridspec from matplotlib.gridspec is used to create a grid of subplots.
sqrt from math is used to calculate the square root of a number.
OBSERVATION:
Here the yaxis.set label position function places the y-label on the right side of the figure instead of on the default left side.
Used set xticks to set the tick labels list to the empty list ([]) which removes all the tick marks on the x-axis.
The set xticklabels functions sets the labels on the tickmarks to the specified list of LATEX formatted strings.
Matplotlib's gridspec is a generalization of subplot that allows more precise control of the layout of nested plots.
The art3d module holds the codes for converting 2D elements into 3D elements that can be added to 3D axes.
The Poly3DCollection object is a container for 3D elements.
OUTPUT-
Text(0.5, 0, 'n')



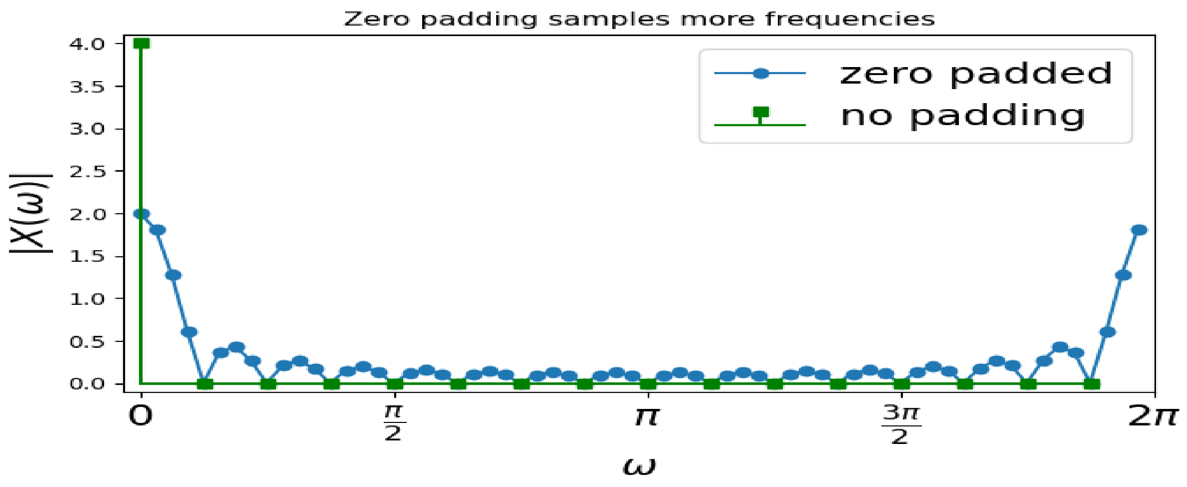


3.2
AIM:
The aim of the code is to plot the magnitude of the discrete Fourier transform (DFT) of a rectangular signal.
FUNCTION IMPORTED:
subplots from matplotlib.pyplot is used to create a subplot.
set_aspect from matplotlib.pyplot is used to set the aspect ratio of the subplot.
grid from matplotlib.pyplot is used to add grid lines to the subplot.

plot from matplotlib.pyplot is used to plot the magnitude of the DFT.
xticks from matplotlib.pyplot is used to set the ticks on the x-axis.
xlabel from matplotlib.pyplot is used to set the label of the x-axis.
axis from matplotlib.pyplot is used to set the limits of the axes.
set_xticklabels from matplotlib.pyplot is used to set the labels of the x-ticks.
OBSERVATION:
This plot shows the magnitude of the DFT of $x = 1 \in R^{16}$ with Ns = 16 samples with DFT-length, N = 64 The subtle point here is that the DFT matrix has dimensions 64 * 16.
OUTPUT-



3.3
AIM:
The aim of the code is to compare the DFT of a signal with and without zero padding.
FUNCTION IMPORTED:
DFTmatrix() from numpy.fft is used to create a DFT matrix.
H from numpy.linalg is used to take the conjugate transpose of a matrix.
abs() from numpy is used to calculate the absolute value of a vector.
plot() from matplotlib.pyplot is used to plot the magnitude of the DFT.
stem() from matplotlib.pyplot is used to plot a stem plot.
legend() from matplotlib.pyplot is used to add a legend to the plot.
OBSERVATION:
From the plot of this code we are shwoing that without zero-padding, x is the 0th column of the 16-point DFT matrix and so all the coefficients except for the 0th column are zero due to orthonormality (shown by the green squares). But, the zero-padded 64-element-long x vector is definitely not a column of the 64-point DFT matrix so we would not expect all the other terms to be zero. In fact, the other terms account for the 63 other discrete frequencies that are plotted here.
OUTPUT-



3.4

AIM:

The aim of the code is to visualize the in-phase and quadrature components of a signal at a specific frequency.

FUNCTION IMPORTED:

DFTmatrix() from numpy.fft is used to create a DFT matrix.

H from numpy.linalg is used to take the conjugate transpose of a matrix.

real() from numpy is used to extract the real part of a vector.

imag() from numpy is used to extract the imaginary part of a vector.

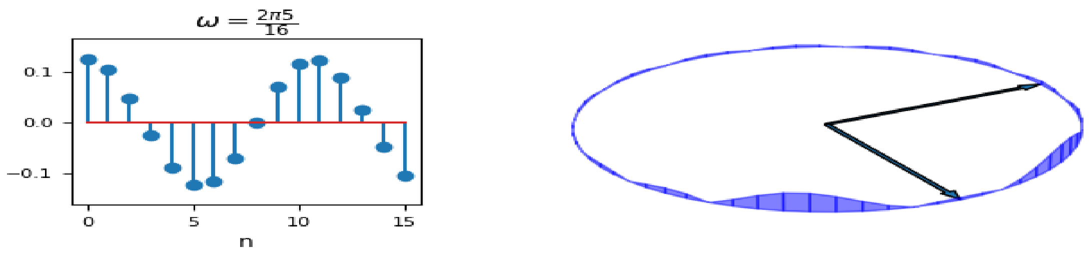
plot() from matplotlib.pyplot is used to plot the in-phase and quadrature components of the DFT.

OBSERVATION:

Here we have plotted the 64-point DFT on the face of a three-dimensional cylinder to emphasize the periodicity of the discrete frequencies.It shows the same magnitude of the $X(\omega_k)$ 64-point DFT as in case of 3.2, but now plotted on the face of a cylinder, we can really see the periodic discrete frequencies.

The two arrows in the xy-plane show the discrete frequencies zero and π ,respectively for reference.

OUTPUT-



3.5

AIM:

The aim of the code is to visualize the in-phase and quadrature components of a signal at the highest and lowest frequencies.

FUNCTION IMPORTED:

DFTmatrix() from numpy.fft is used to create a DFT matrix.

H from numpy.linalg is used to take the conjugate transpose of a matrix.

real() from numpy is used to extract the real part of a vector.

imag() from numpy is used to extract the imaginary part of a vector.

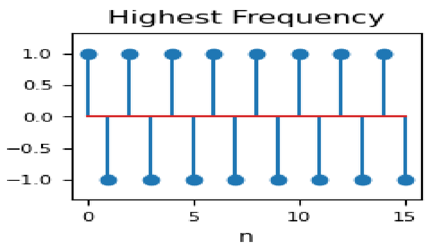
plot() from matplotlib.pyplot is used to plot the in-phase and quadrature components of the DFT

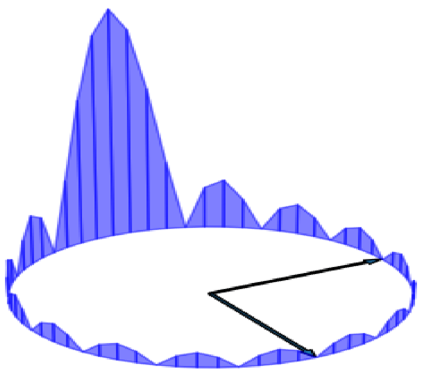
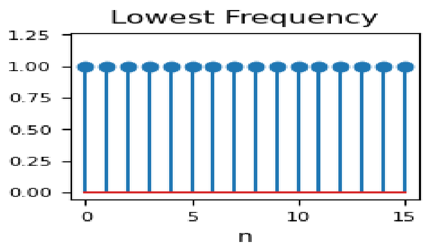
OBSERVATION:

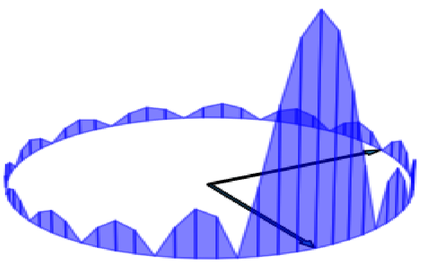
This code plots the symmetric lobes of the DFT of a real signal.

The plot on the left is the signal in the sampled time-domain and the plot on the right is its DFTmagnitude glyph.

Because the input signal is real, the DFT is symmetric. OUTPUT-

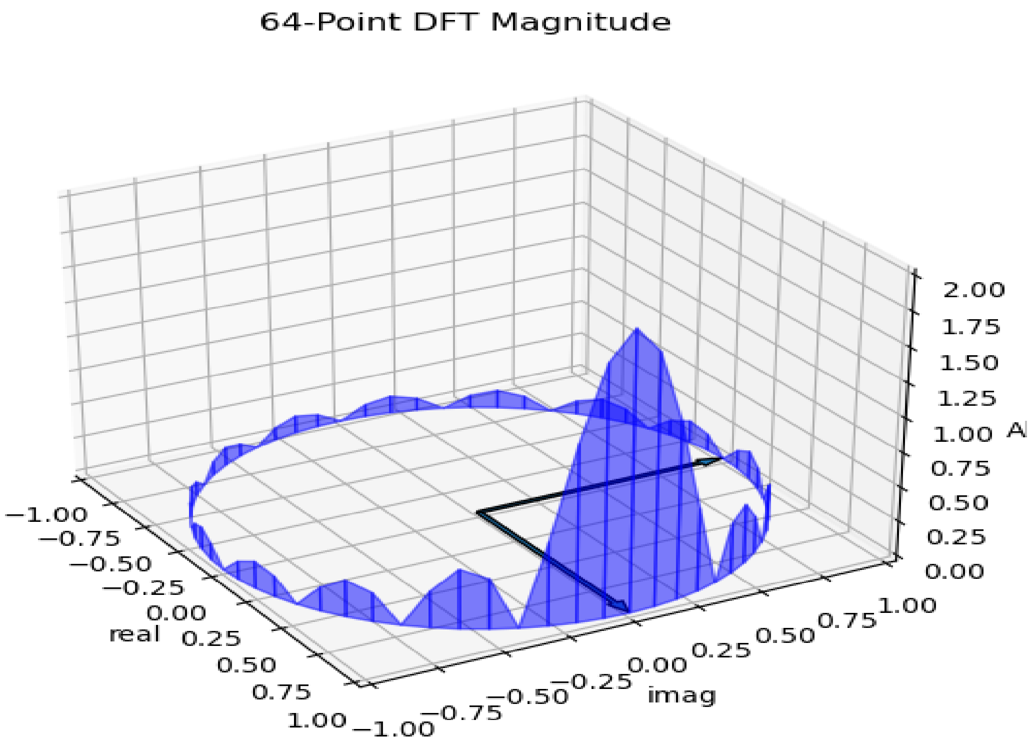




3.6
AIM:
The aim of the code is to visualize the 64-point DFT magnitude in 3D.
FUNCTION IMPORTED:
np.vstack() from numpy is used to stack two vectors together.
ax.set_zlim() from matplotlib.pyplot is used to set the z-axis limits of the subplot.
ax.view_init() from matplotlib.pyplot is used to set the viewing angle of the subplot.
facet_filled() from my_functions is a custom function that is used to create a filled polygon in 3D.
mpl_toolkits.mplot3d.art3d is used to add the arrows to the 3D plot.

OBSERVATION:
Here with the given code we have shwon that signal toggles back and forth positive and negative.
Here the amplitudes of this toggling are not important, it is the rate of toggling that defines the high frequency signal.
This is the the highest frequency signal (i.e.$\omega\_N/2) = \pi$) in the sampled time-domain.
OUTPUT-



64-Point DFT Magnitude
real
imag
A
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00