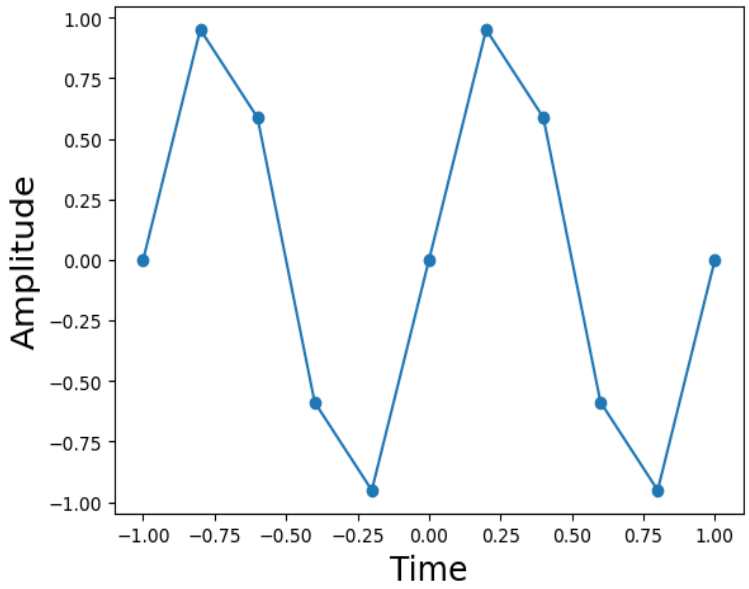
SIGNAL PROCESSING SYSTEM DESIGN LAB

YASH ANAND
(23EE65R22)

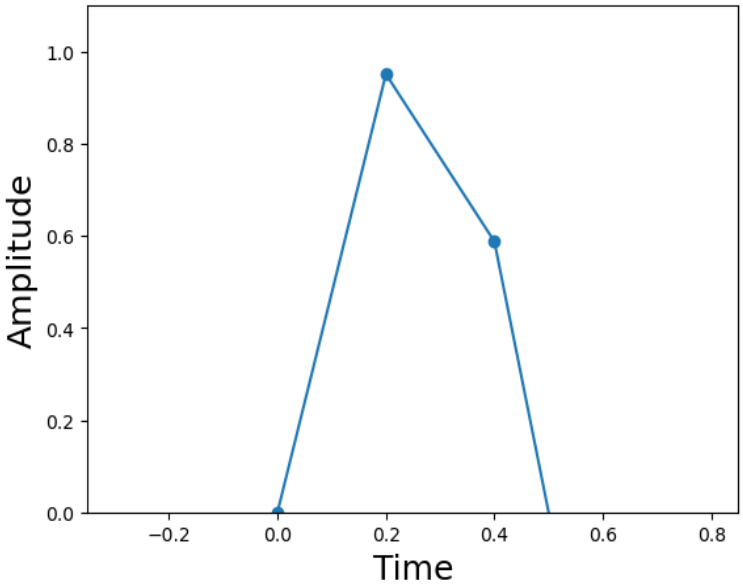
2. Sampling Theorem

Here we have imported the subplots function from the matplotlib.pyplot module which helps in generating the graphs and the new axes.

2.1
Here from the code used here ensures that we use floating-point division instead of the default integer divide,
Establishes the figure and axis bindings using subplots.
The arange function creates a Numpy array of numbers.
Then, we have computed the sine of this array and plot it in the figure we just created. The -o is shorthand for creating a plot with solid lines and with points marked with o symbols.
Attached the plot to the ax variable is the modern convention for Matplotlib that makes it clear where the plot is to be drawn.
Also we have set up the labels for the x-axis and y-axis, respectively, with the specified font size.
OUTPUT-
Text(0, 0.5, 'Amplitude')



2.2
This code creates a time-domain plot of a sine wave.
fig, ax = subplots() creates a new figure and an axis object. The subplots() function takes two arguments: the number of rows and the number of columns in the figure.
ax.plot(t, x, 'o-') plots the data points in the arrays t and x. The plot() function takes three arguments: the x-axis data, the y-axis data, and the line style.
The axis() function takes four arguments: the minimum value of the x-axis, the maximum value of the x-axis, the minimum value of the y-axis, and the maximum value of the y-axis
The set xlabel() function takes two arguments: the label text and the font size.
OUTPUT-
Text(0, 0.5, 'Amplitude')

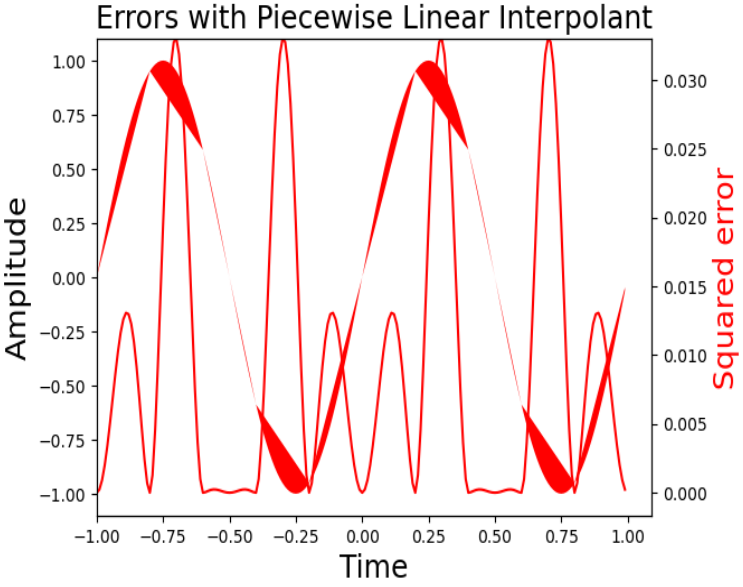


2.3

This Code is for constructing the piecewise linear approximation.
The hstack function packs smaller arrays horizontally into a larger array.
interval = [] creates an empty list to store the piecewise domains.
apprx = [] creates an empty list to store the line on the domains.
The linspace() function takes four arguments: the start point, the end point, the number of points, and a flag indicating whether the endpoint should be included
The piecewise() function takes three arguments: the array of points, the array of boolean arrays, and the array of linear functions.

2.4

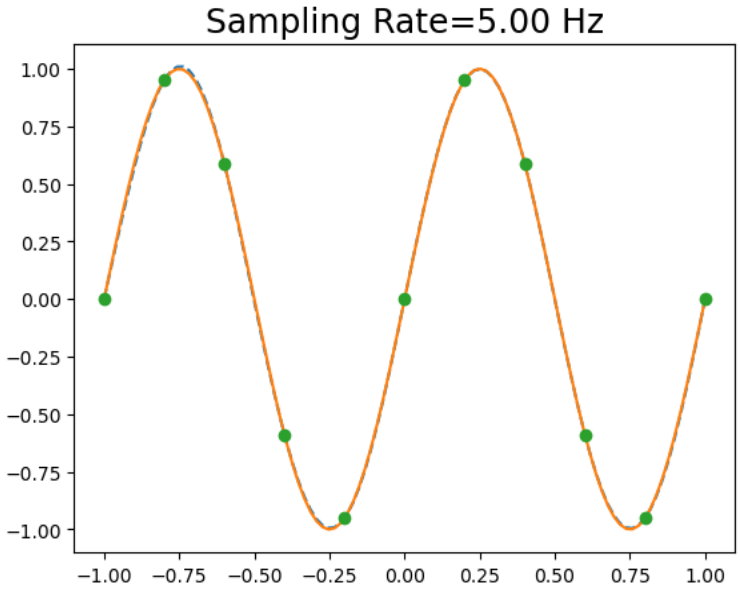
Figure generated by code showing the squared error of the sine function and its corresponding linear interpolant.The piecewise approximation is worse where the sine is curviest and better where the sine is approximately linear. This is indicated by the red-line whose axis is on the right side
OUTPUT-



2.5

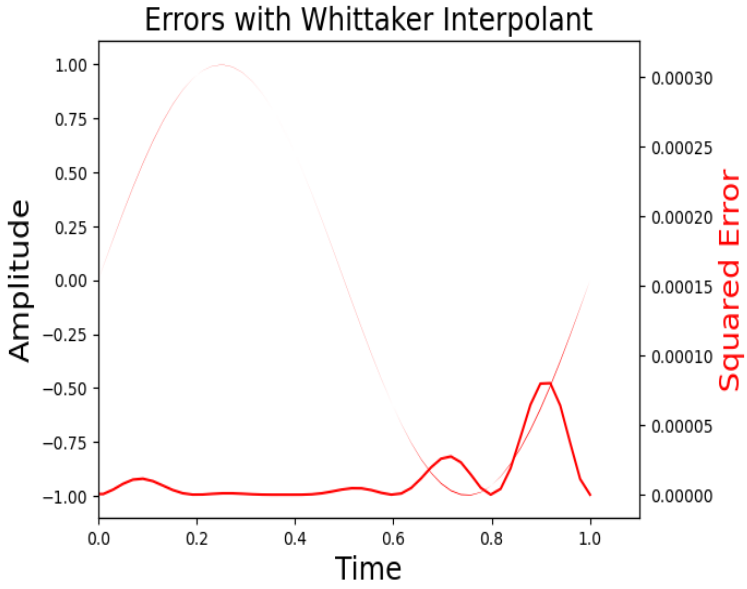
Figure generated by this code showing the fit of sinc-based interpolator.
It tells us that we can draw multiple lines with a single plot function.
The squared-error here is imperceptible in this plot due to improved interpolant.
OUTPUT-

Text(0.5, 1.0, 'Sampling Rate=5.00 Hz')



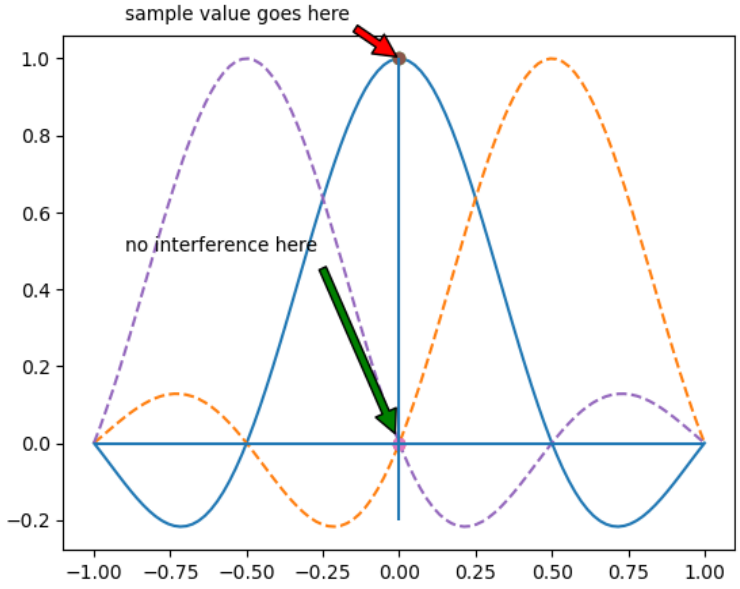
2.6

The plot obtianed shows the massive reduction in squared-error resulting from using the sinc functions which is much better than what we obtained from the linear interpolant.
In this respect,the interpolating sinc functions are called the Whittaker interpolating functions.
OUTPUT-
Text(0.5, 1.0, 'Errors with Whittaker Interpolant')



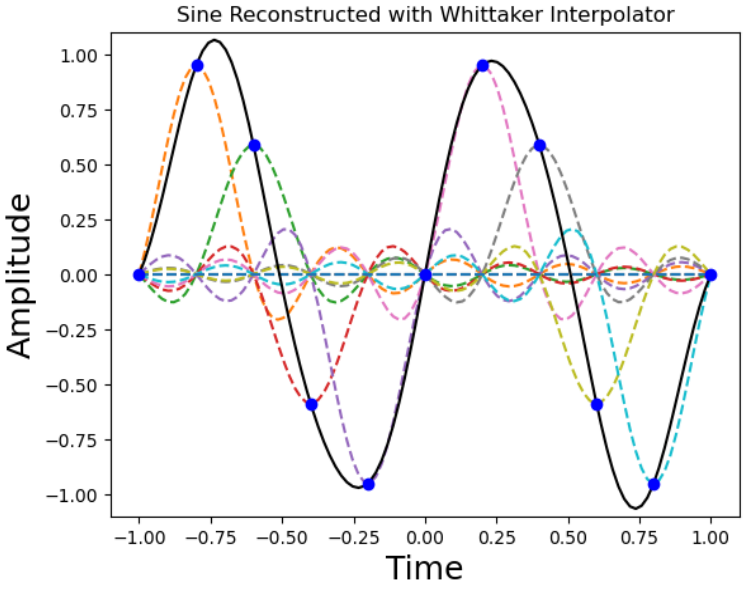
2.7

This plot three neighboring interpolation functions, one for each of three neighboring samples where the peaks and zeros of these functions interleave. Therefore, at each of the sample points, there is no interference from any of the other functions because the others are all zero there.
This is why the interpolated function matches the sample points exactly. OUTPUT-
Text(-0.9, 0.5, 'no interference here')



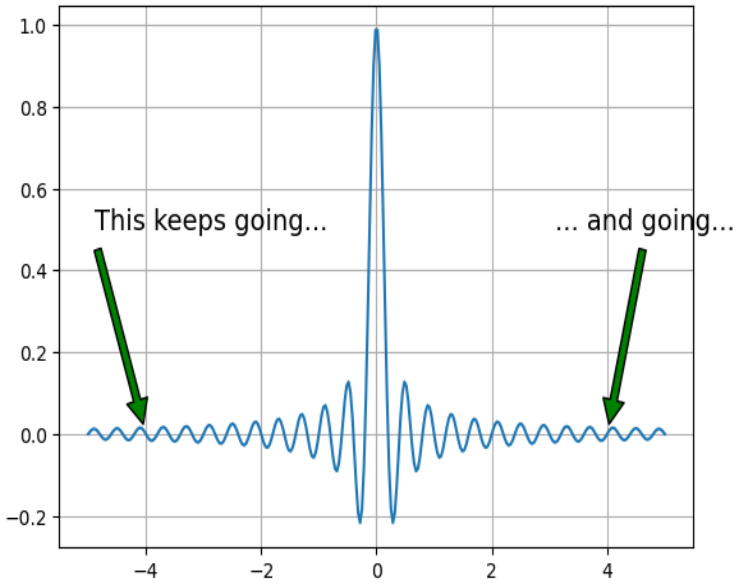
2.8

Plot generated by this code shows how the sampled sine function is reconstructed (solid line) using individual Whittaker interpolators (dashed lines).
Each of the sample points sits on the peak of a sinc function
It uses Numpy broadcasting to create an implicit grid for evaluating the interpolating functions. OUTPUT-



2.9

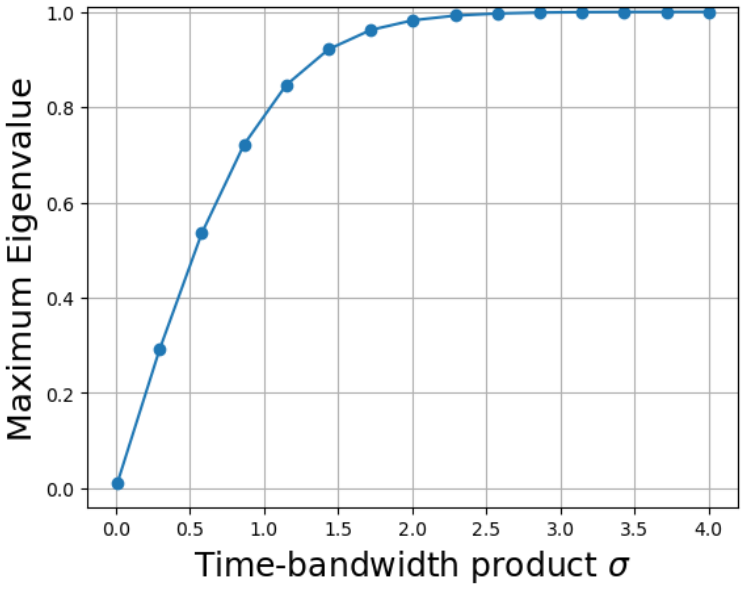
It plots the sinc function and annotates the plot with two arrows.
The annotations in the plot indicate that the sinc function extends infinitely to the left and right. This is because the sinc function is aperiodic, meaning that it does not have a period.
OUTPUT-



2.10

Here we have we made a connection between the quality of our reconstruction via theWhittaker interpolant and the time-bandwidth product. Here the eigvalsh function comes from the LAPACK compiled library.

OUTPUT-

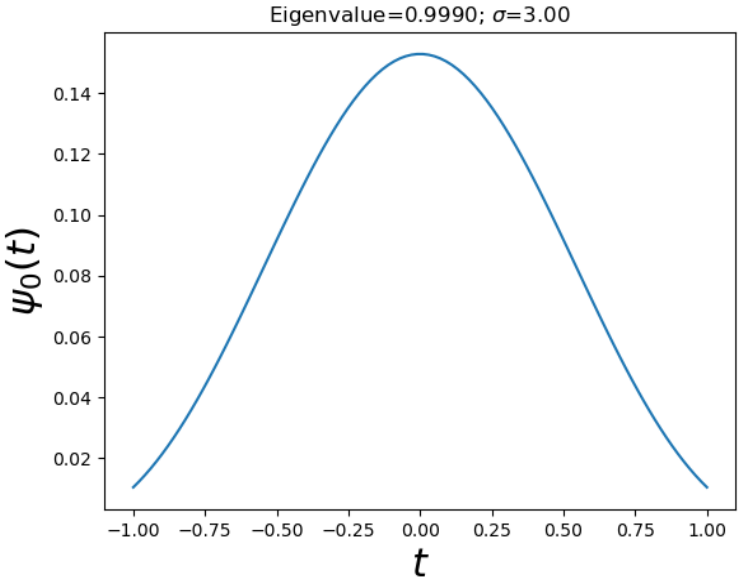


2.11

Here the argmax function will find the array index corresponding to the array maximum.

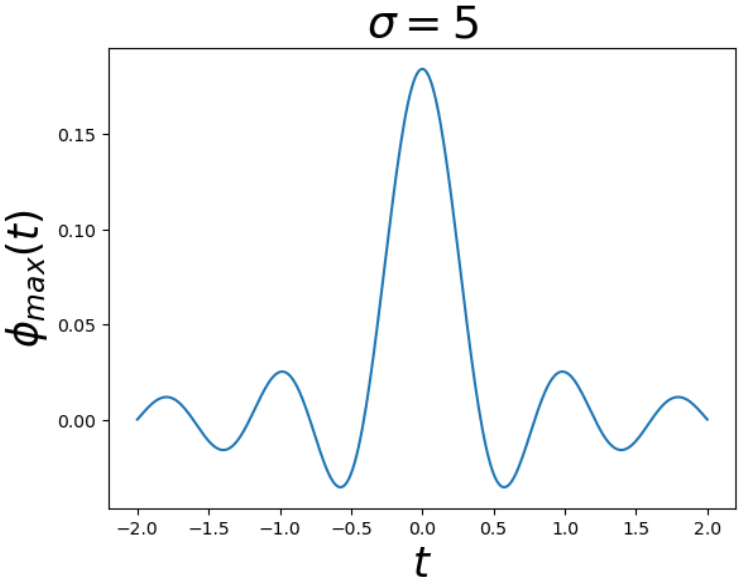
Here the plot looks similar to sinc function. In fact, in the limit as sigma tends to infinity, the eigenfunctions devolve into time-shifted versions of the sinc function. These are the same functions used in the Whittaker interpolant.

OUTPUT-



2.12

Here the sign function computes the sign of the argument.
It will computes and plots the eigenfunctions of the diffusion equation with a Gaussian kernel.
The parameter sigma controls the width of the Gaussian kernel. A larger value of sigma will result in a wider kernel and a smoother eigenfunction.
OUTPUT-



2.13

Here the code I shared draws the 2D and 3D representations of a discrete Fourier transform (DFT).
This is the Setup code for gyphs.
The add patch function places graphics primitives ("patches") like rectangles and arrows onto the plot.
The FancyArrow is one of the graphics primitives for an arrow.
The gridspec is a more powerful tool than subplots for controlling placement of plots on a grid.
The facet filled() function takes three arguments: the array x, the opacity alpha, and the color color. The function returns a polygon collection that fills the area between the points in the array x.
The drawDFTView() function takes two arguments: the array X and the axis object ax. The function first calculates the angle spectrum of the array X.