**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **27/05/2020** | **Name:** | **Varun G Shetty** |
| **Course:** | **Digital signal processing** | **USN:** | **4AL17EC093** |
| **Topic:** | **Fourier transform, fast fourier transform, wavelet transform,**  **ECG signal analysis using MATLAB** | **Semester & Section:** | **6th &‘B’** |
| **GitHub Repository:** | **Varunshetty4** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| Image of session  C:\Users\user\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot (159).png  C:\Users\user\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot (160).png  Fourier transform:  The Fourier transform of a function of time is a [complex-valued function](https://en.wikipedia.org/wiki/Complex-valued_function) of frequency, whose magnitude ([absolute value](https://en.wikipedia.org/wiki/Absolute_value#Complex_numbers)) represents the amount of that frequency present in the original function, and whose [argument](https://en.wikipedia.org/wiki/Argument_(complex_analysis)) is the [phase offset](https://en.wikipedia.org/wiki/Phase_offset) of the basic [sinusoid](https://en.wikipedia.org/wiki/Sine_wave) in that frequency. The Fourier transform is not limited to functions of time, but the [domain](https://en.wikipedia.org/wiki/Domain_of_a_function) of the original function is commonly referred to as the [time domain](https://en.wikipedia.org/wiki/Time_domain). There is also an inverse Fourier transform that mathematically synthesizes the original function from its frequency domain representation, as proven by the [Fourier inversion theorem](https://en.wikipedia.org/wiki/Fourier_inversion_theorem).  Fast fourier transform:  A fast Fourier transform (FFT) is an [algorithm](https://en.wikipedia.org/wiki/Algorithm) that computes the [discrete Fourier transform](https://en.wikipedia.org/wiki/Discrete_Fourier_transform) (DFT) of a sequence, or its inverse (IDFT). [Fourier analysis](https://en.wikipedia.org/wiki/Fourier_analysis) converts a signal from its original domain (often time or space) to a representation in the [frequency domain](https://en.wikipedia.org/wiki/Frequency_domain) and vice versa. The DFT is obtained by decomposing a [sequence](https://en.wikipedia.org/wiki/Sequence) of values into components of different frequencies.[[1]](https://en.wikipedia.org/wiki/Fast_Fourier_transform#cite_note-Heideman_Johnson_Burrus_1984-1) This operation is useful in many fields, but computing it directly from the definition is often too slow to be practical. An FFT rapidly computes such transformations by [factorizing](https://en.wikipedia.org/wiki/Matrix_decomposition) the [DFT matrix](https://en.wikipedia.org/wiki/DFT_matrix) into a product of [sparse](https://en.wikipedia.org/wiki/Sparse_matrix) (mostly zero) factors.[[2]](https://en.wikipedia.org/wiki/Fast_Fourier_transform#cite_note-Loan_1992-2) result, it manages to reduce the [complexity](https://en.wikipedia.org/wiki/Computational_complexity_theory) of computing the DFT from which arises if one simply applies the definition of DFT, to , where is the data size. The difference in speed can be enormous, especially for long data sets where N may be in the thousands or millions  Matlab code:  Fs=1000;  Ts=1/Fs;  dt=0:Ts:2-Ts;  f1=10;  f2=30;  f3=70;  y1=10\*sin(2\*pi\*f1\*dt);  y2=10\*sin(2\*pi\*f2\*dt);  y3=10\*sin(2\*pi\*f3\*dt);  y4=y1+y2+y3;  subplot(4,1,1);  plot(dt,y1, ‘r’);  subplot(4,1,2);  plot(dt,y2, ‘r’);  subplot(4,1,3);  plot(dt,y3, ‘r’);  subplot(4,1,4);  plot(dt,y4, ‘r’);  nfft=length(y4);  nfft2=2^nextpow2(nfft);  ff=fft(y4,nfft2);  plot(abs(ff));  Wavelet transform:  A wavelet transform is a linear transformation in which the basis functions (except the first) are scaled and shifted versions of one function, called the “mother wavelet.” If the wavelet can be selected to resemble components of the image, then a compact representation results  Implementation of signal filtering using WT in matlab:  close all;  clear all;  clc;  [k,Fs]=audioread[‘man\_voice.wav’]  k=k\*0.5/rms(k);  k=awgn(k,12,’measured’);  [c,1]=wavedec(k,3, ‘db4’);  b=wthresh(c, ‘s’ ,0.25);  y=waverec(b,1, ‘db4’);  y=y\*0.5/rms(y));  sound(y,Fs);  ECG signal analysis using matlab:  sig=load(‘ecg.txt’);  plot(sig)  xlabel(‘samples’);  ylabel(‘electrical activity’);  title(‘ECG signal sampled at 100hz’)  hold on  plot(sig,’r0’) |