ASSIGNMENT- 1

ADYASA MOHANTY - EE18BTECH11048

Download all codes and audio files from

```
https://github.com/adyasa611/EE3250/blob/main/
    Assignment−1_C/Codes
```

Latex-tikz codes from

```
https://github.com/adyasa611/EE3250/blob/main/
    Assignment−1_C/Figures
```

1 Problem

To implement the function to find the Fourier Transform of a signal in C programming Language.

Solution:
Convolution : $y(n) = x(n) * h(n)$,
where x(n) is the audio signal and h(n) refers to the filter.
Taking Fourier Transform, we obtain:
$Y(K) = H(K)X(K)$
To obtain $x(n)$ use the below. The code will generate the .dat file.
$x(n)$ will be generated from Sound_Noise.wav.

```
Codes/generate.py
```

$h(n)$ is generated using the equation:

$$\sum_{m=0}^{M} a(m)\, y(n-m) = \sum_{k=0}^{N} b(k)\, x(n-k) \quad (1.0.0)$$

$$y(n) - 2.52y(n-1) + 2.56y(n-2) - 1.206y(n-3) + 0.22013y(n-4) = 0.00345x(n) + 0.0138x(n-1) + 0.020725x(n-2) + 0.0138x(n-3) + 0.00345x(n-4) \quad (1.0.1)$$

To obtain the Fourier transform in a computationally less expensive manner we break the N point DFT into 2 $N/2$ point DFT recursively.
This reduces the computation rate from $O(n^2)$ to $O(nlogn)$.
The code to find the Fourier Transform can be found here.

```
Codes/EE18BTECH11048.c
```

We generate the plots of FFT of x(n) and h(n).
They look similar to the plots generated from the in built function of FFT.

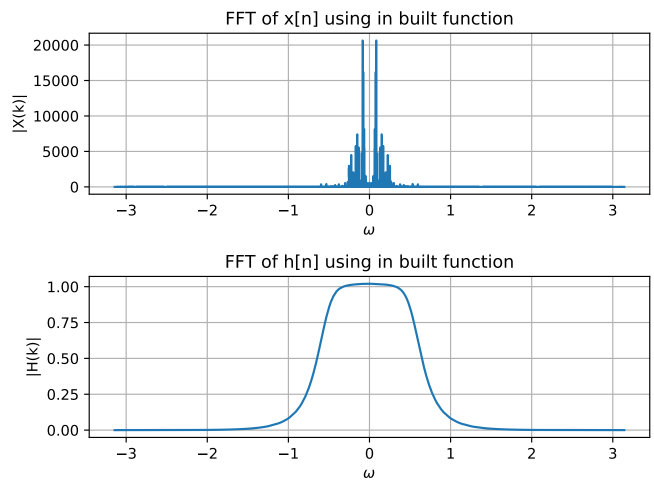

Fig. 0: FFT of x(n) and h(n)

The code to obtain these plots.

```
Codes/X_H_Plot.py
```

Now we obtain the Inverse Fourier Transform of Y(K) to obtain y(n) and the filtered sound.

It looks similar to the plots generated from the in built function of IFFT.

The code to obtain these plots.

```
Codes/y_plot.py
```

The filtered audio file can be obtained here.

```
Codes/Sound_diff_eq_C.wav
```

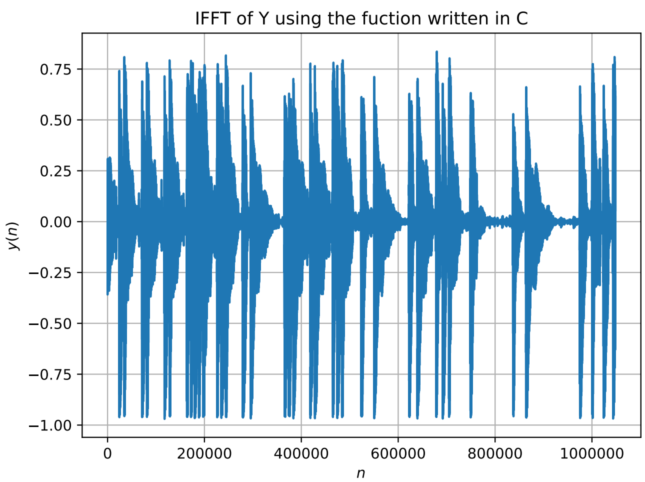

Fig. 0: IFFT of Y(K)