Signals and Systems Laboratory
Indian Institute of Technology Jammu
Experiment No.-5

Objective-

Record your own voice, x(n), on Raspberry Pi. Add noise (for example, car noise) with your own voice, x(n) at SNR (signal to noise ratio) 10 dB. Perform DFT and IDFT on your recorded voice, x(n) and noisy voice.

Apparatus- python+matplotlib, MS-Excel, LibreOffice-Calc, Matlab, Scilab, etc

Theory-

Discrete Fourier Transform:

Discrete Fourier Transform (DFT) is used for transforming discrete-time sequence x(n) of finite length into discrete-frequency sequence X(k) of finite length. It is very powerful tool for frequency analysis of discrete-time signals.

Mathematically, the DFT of discrete-time sequence x(n) is denoted by X(k), given by,

$$X(k) = \sum_{n=0}^{N-1} x(n).e^{-j2\pi kn/N}$$

here, k = 0, 1, 2,.....,N-1. Since, this summation is taken for N points; it is called as N-point DFT.

Inverse Discrete Fourier Transform:

We can obtain discrete sequence x(n) from its DFT. It is called as inverse discrete fourier transform (IDFT). Mathematically, the IDFT is given by:

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k).e^{j2\pi kn/N}$$

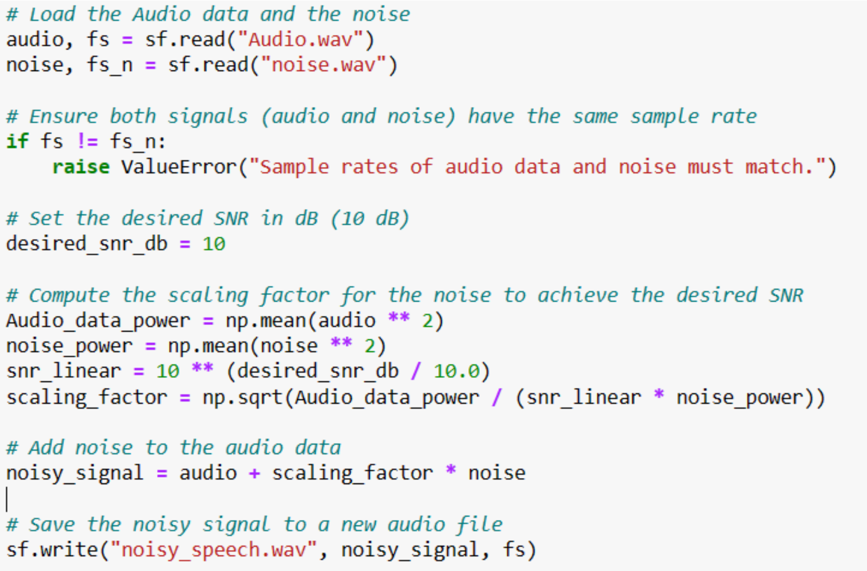
here, n = 0, 1, 2,.....,N-1. This is called as N-point IDFT.

For example, x(n) = Your own voice,

w(n) = Noise,

s(n) = Noisy speech = x(n) + w(n)

Hints:



```python
# Load the Audio data and the noise
audio, fs = sf.read("Audio.wav")
noise, fs_n = sf.read("noise.wav")

# Ensure both signals (audio and noise) have the same sample rate
if fs != fs_n:
    raise ValueError("Sample rates of audio data and noise must match.")

# Set the desired SNR in dB (10 dB)
desired_snr_db = 10

# Compute the scaling factor for the noise to achieve the desired SNR
Audio_data_power = np.mean(audio ** 2)
noise_power = np.mean(noise ** 2)
snr_linear = 10 ** (desired_snr_db / 10.0)
scaling_factor = np.sqrt(Audio_data_power / (snr_linear * noise_power))

# Add noise to the audio data
noisy_signal = audio + scaling_factor * noise

# Save the noisy signal to a new audio file
sf.write("noisy_speech.wav", noisy_signal, fs)
```

Algorithm-

a. Start the process.

b. Record your own voice, x(n).

c. Add noise, w(n) with your voice, x(n) to make noisy voice, s(n) = x(n) + w(n).

d. First, perform DFT and IDFT on your voice, x(n).

e. Second, perform DFT and IDFT on noisy voice, s(n).

f. Execute the program, display the result and verify theoritically.

g. Plot the graph for the DFT and IDFT.

h. Stop the process.

Observations-
Result- Performed DFT and IDFT on recorded voice.
Precautions:-

- Program must be written carefully to avoid errors.
- Program can never be saved as standard function name.
- Commands must be written in proper format.