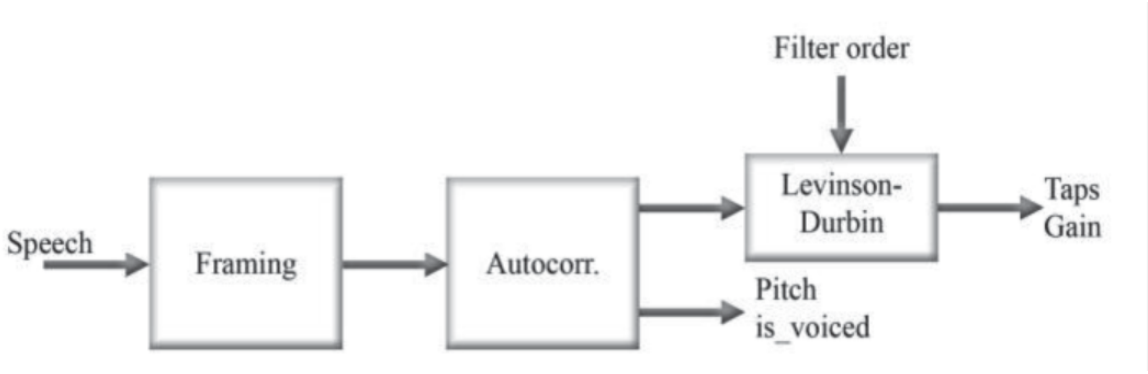
Assignment 4

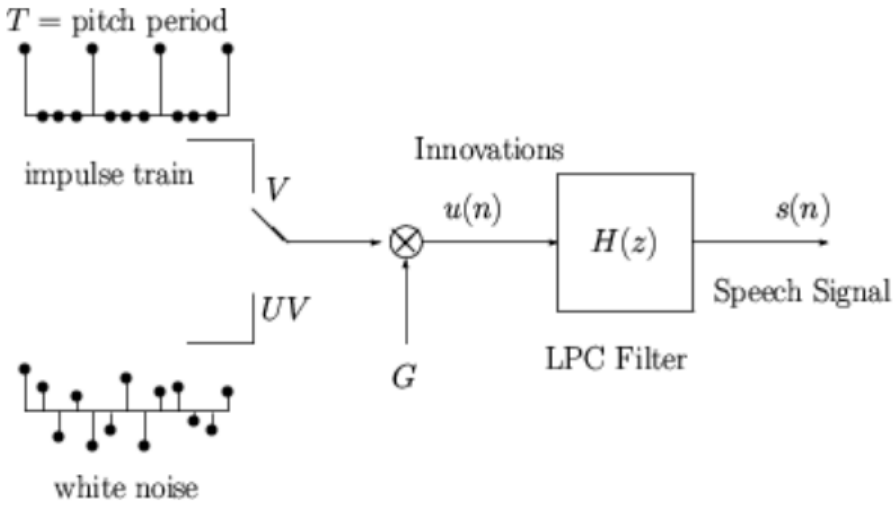
Ayush Bhardwaj
2018134

Implementation design for Encoder



- Encoder function follows the model in the above diagram. All the steps followed are explained briefly in the points below.
- The input speech is preprocessed and pre-emphasis is done in order to boost the higher frequency components.(Plots are attached in the report)
- I calculated some important parameters like Magnitude sum, Zero crossing and pitch period for each frame using their standard formulas from theory.
- For an optimised implementation, I need to send the information for every frame whether it is voiced or not. This information helps in the decoding of the signal more efficiently. In order to get an accurate value, I used 3 types of detector: Magnitude sum, Zero crossing and pitch period. I got 3 boolean arrays by setting an appropriate threshold value for each parameter. I classified a frame as voiced if all the 3 parameters classified it as voiced. (Plots are attached in the report)
- LPC coefficients are calculated using Durbin's method. The order used is 10.
- Finally, I also calculated the gain for each frame.
- All the parameters calculated are quantized and then converted to bits for transmission.
- The **average processing delay** of the encoder() function was calculated through code and its value was found to be 20 ms.
- The encoder function returns bitstream of LPC coefficients, Pitch period, Voiced/Unvoiced, Gain and Avg, Processing delay.

Implementation design for Decoder



- Decoder function follows the model in the above diagram. All the steps followed are explained briefly in the points below.
- The input received in this function is in the form of bits. These bitstreams are converted into decimal values. The values are further reverse quantified to get the original values.
- The process of decoding is different for both voiced and unvoiced frames. If the current frame is voiced, then u=impulse train otherwise u=white noise.
  (Plots are attached in the report)
- The signal is reconstructed using this formula. G= Gain, H(z) =LPC filter

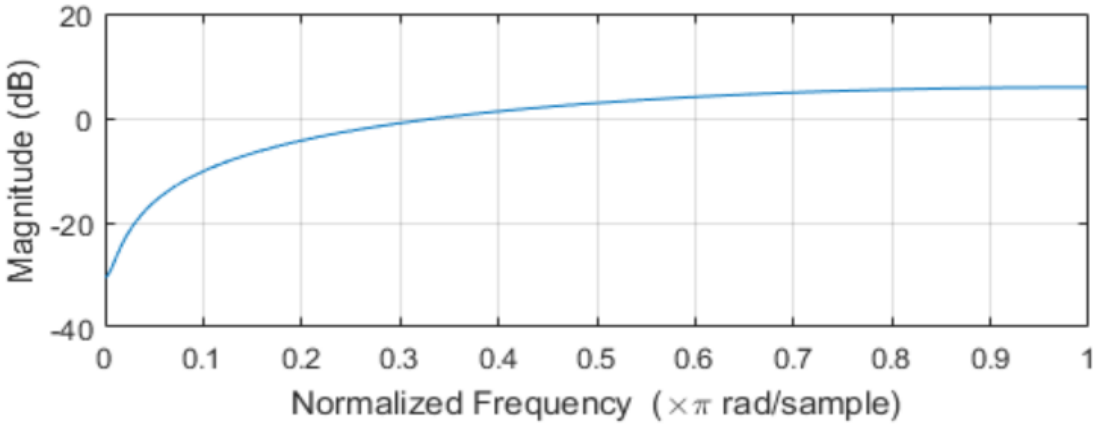
$$H(z) = \frac{S(z)}{GU(z)} = \frac{1}{1 - \sum_{k=1}^{p} a_k z^{-k}}$$

- All the variables are present and we can predict the output signal.

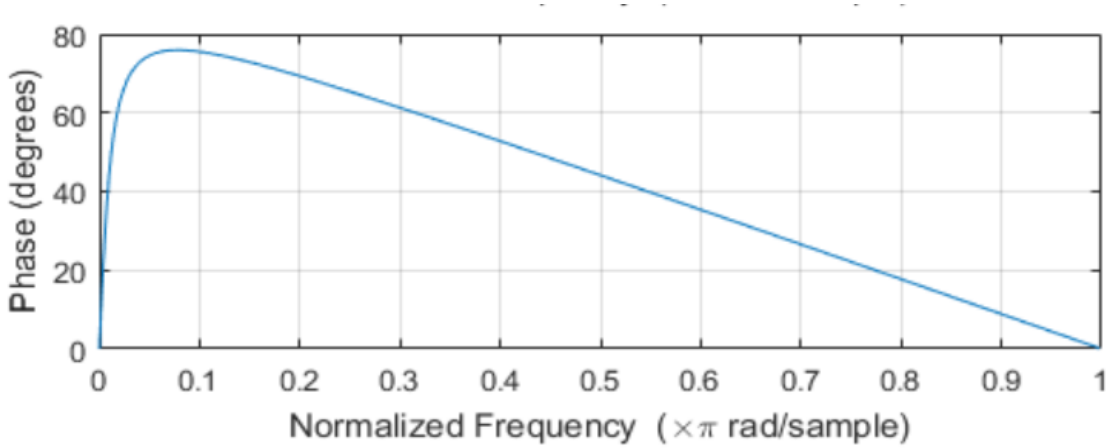
*All the code in this assignment has been written from scratch.

Important plots

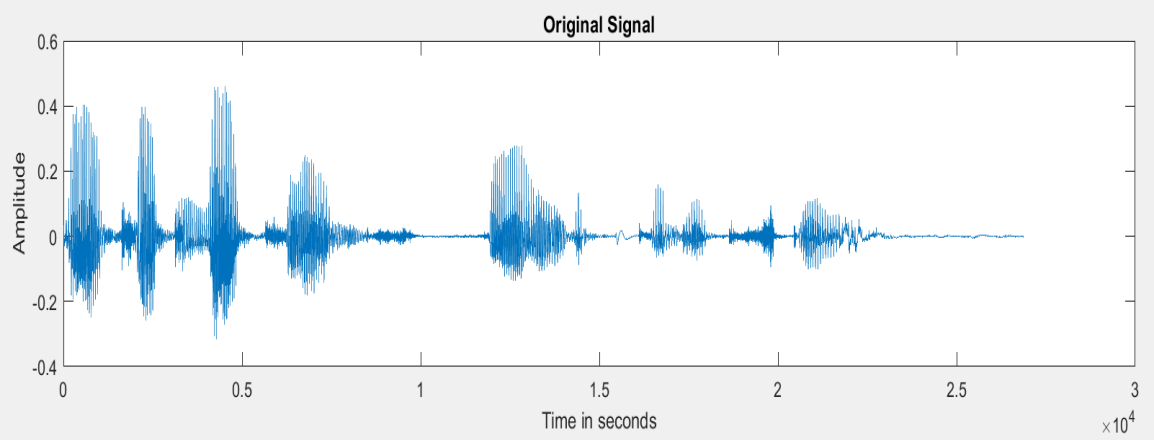
1) Pre- emphasis filter



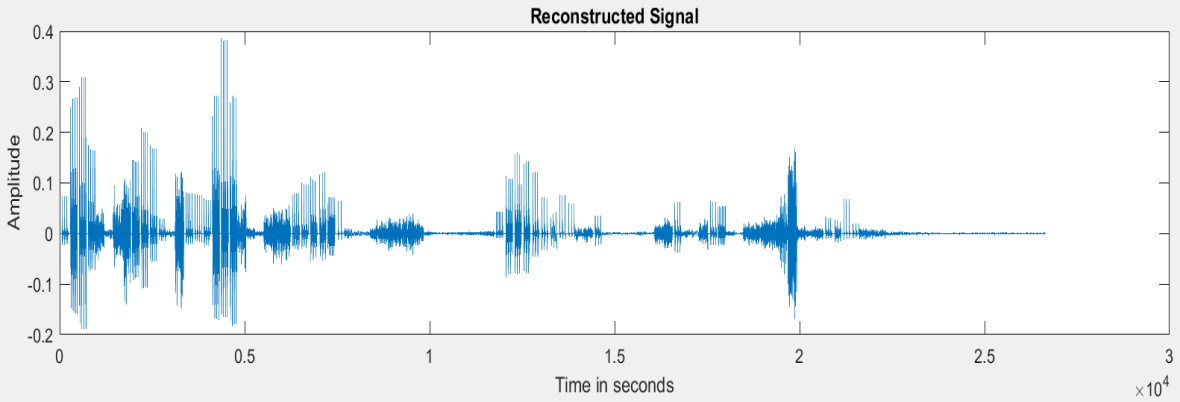
2) Inverse Pre- emphasis filter



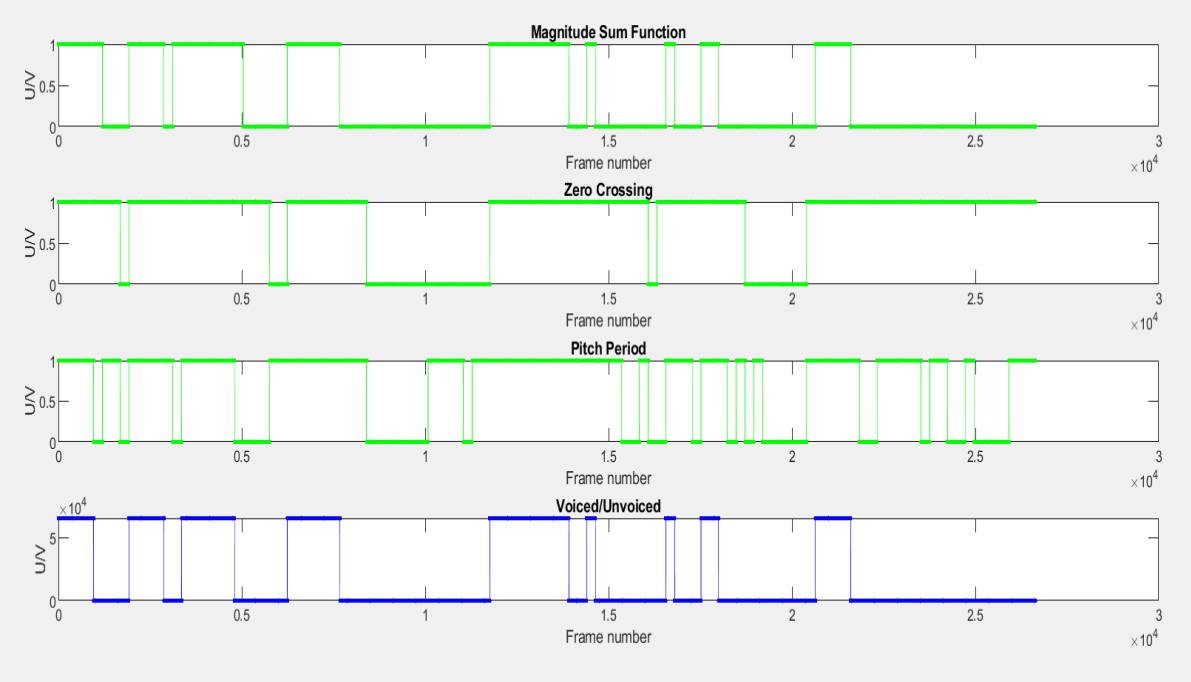
3) Original speech signal



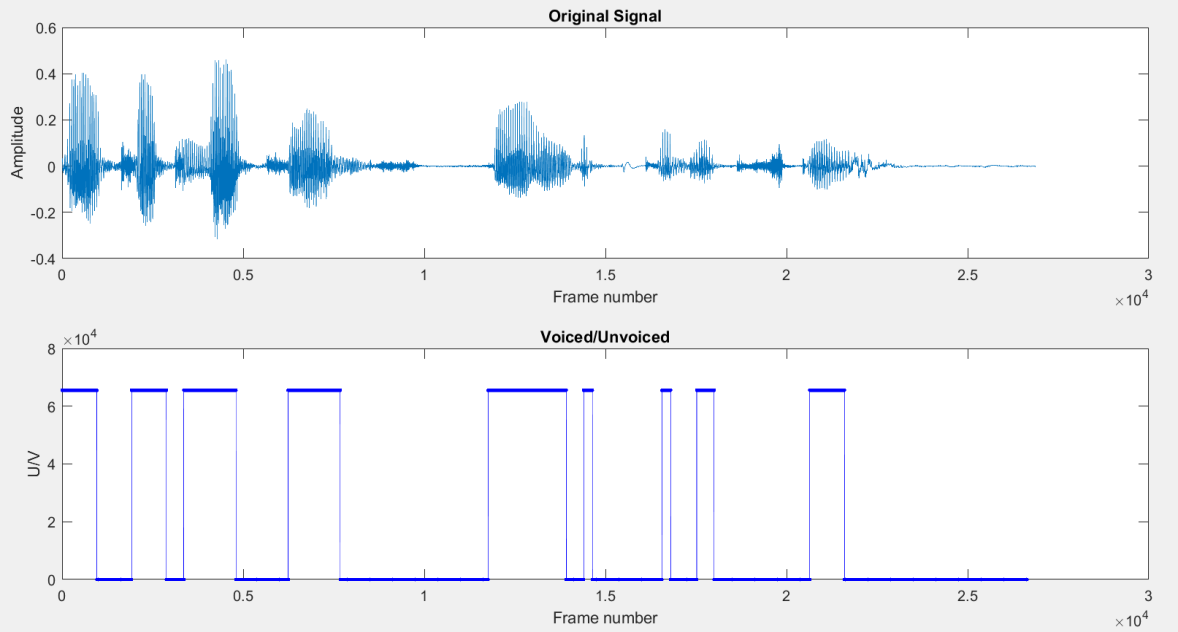
4) Reconstructed speech signal



5) Voiced/ Unvoiced detection plots



6) Original signal and voiced/Unvoiced detection



Quality of the reconstructed signal

The reconstructed signal is very similar to the original signal. We can easily understand the message transmitted. I changed various parameters like frame size, quantization bits, uniform vs nonuniform utilization. The bitrate of our LP coder can be reduced significantly by moderating these parameters. This improved bitrate comes at the expense of lower audio quality. Therefore, We can have a constraint on the audio quality expected and then vary the parameters in order to get the lowest bitrate. At a bitrate of around 1.7~2.1 KHz, the sounds start becoming a bit robotic but the message can be interpreted. For this assignment, I put a constraint on the reconstructed speech that the message could be interpreted easily.

Implementations followed to get the best LP coder

- **First implementation**- I wanted to get an estimate of the sound quality and bitrate of the transmission when we do not use any dedicated algorithms like LPC 10. I sent the original signal using quantization. The reconstructed speech signal had a very good audio quality.

  Bitrate calculations

Total length of the audio	4.7 seconds
Frame size	60 ms
Samples per frame	480
Quantizing bits	16 bits
Total number of frames	36700/480=76
Total bits sent=Quantizing bits*samples per frame*total number of frame	16*480*76 =583680
Bitrate	583680/4.5~129Kbps

- **Second Implementation**- This is the standard LPC-10 implementation using uniform quantization. The reconstructed speech has a good audio quality. All the data sent is quantized using standard 16 bits.

  Bitrate calculations

Total length of the audio	4.7 seconds
Frame size	22.5 ms
Samples per frame	180
Quantizing bits	16 bits
Total number of frames	36700/180=204
Total bits sent per frame=16*(10 LPC coefficient) + 16*1(voiced/unvoiced) + 16*1(pitch period)+16*1(Gain)	208 bits per frame
Total bits sent=bits per frame*total number of frame	208*204 =42432
Bitrate	42432/4.5~9Kbps

- **Third Implementation**- This is the standard LPC-10 implementation using non-uniform quantization. From the theory of LPC-10, I realised that some coefficients contain more information than the others. Therefore, I used more bits to quantize initial coefficients and less bits for later coefficients. Also unvoiced frames can be transmitted using even less coefficients. The reconstructed speech also has a decent audio quality. The quantization scheme is given below.

	Voiced	Unvoiced
Pitch	7	7
Gain	5	5
p_1	5	5
p_2	5	5
p_3	5	5
p_4	5	5
p_5	4	-
p_6	4	-
p_7	4	-
p_8	4	-
p_9	3	-
p_{10}	2	-
Synchronisation	1	1

Bitrate calculations

Total length of the audio	4.7 seconds
Frame size	22.5 ms
Samples per frame	180
Total number of frames	36700/180=204
Total bits sent per frame	54 bits per frame
Total bits sent=bits per frame*total number of frame	54*204 =11016
Bitrate	11016/4.7~2.34Kbps

- **Final Implementation**- This is the standard LPC-10 implementation using uniform quantization but less number of bits. Pitch was given 7 bits, gain was given 5, U/UV was given 1 and coefficients were given 7 bits. I also increased the frame length in order to further improve the bitrate. The audio quality was decent and we could interpret simple messages. The audio sounded very robotic and difficult to interpret.

  Bitrate calculations

Total length of the audio	4.7 seconds
Frame size	60 ms
Samples per frame	480
Total number of frames	36700/480=76
Total bits sent per frame	7+5+10*7+1= 83bits per frame
Total bits sent=bits per frame*total number of frame	83*76 =6308
Bitrate	6308/4.7~1.4Kbps

Note- We can modify parameters in the code and can run all the implementations.