PROJECT 1
AUDIO COMPRESSION USING THE FFT

Due: May 10, 2000

EE 6360 - Digital Signal Processing I

Accompanying the explosive growth of the Internet is the growing need for audio compression, or data compression in general. The major goal in audio compression is to compress the audio signal, either for reducing transmission bandwidth requirements or for reducing memory storage requirements, without sacrificing quality. Digital cellular phones, for example, use some type of compression algorithm to compress in real-time the voice signal over general switched telephone networks. Audio compression can also used off-line to reduce storage requirements for mail forwarding of voice messages or for storing voice messages in a digital answering machine. Due to the increasing demand for better speech compression algorithms, several standards were developed, including MPEG, GSM (in Europe), VSELP, CELP, etc.

As the implementation of the existing audio compression standards is generally complex (and beyond the scope of this project), we will investigate in this project a much simpler approach to audio compression using the Fast Fourier Transform (FFT). An efficient method for transform-based data (which is audio in our case) compression is to use a subset of transform components for signal synthesis. This method works as follows: the data set to be compressed is segmented into N-point segments (or frames) using a sliding data window (Fig. 1). Each segment is then transformed using an N-point FFT, Fig. 2. Since the data is real, only $1+N/2$ components are independent (symmetry property of the DFT). Of the $1+N/2$ independent components only *D independent dominant (maximum)* components are retained for data reconstruction ($D<1+N/2$). Symmetry in the transform components is maintained to reconstruct a real signal. In this project, you are asked to implement and simulate this audio compression scheme. Simulations are required for different size (N's) FFT and different D's. You will use two performance measures to evaluate your simulations, namely the overall signal-to-noise ratio (SNROV) and the average segmental signal-to-noise ratio (AVSNRSG), defined as follows:

$$SNROV = 10\log_{10}\left(\frac{\sum_{n=0}^{L-1} s^2(n)}{\sum_{n=0}^{L-1}(s(n)-\hat{s}(n))^2}\right) \text{ dB}$$

$$SNRSEG(m) = 10\log_{10}\left(\frac{\sum_{n=0}^{N-1} s^2(mN+n)}{\sum_{n=0}^{N-1}(s(mN+n)-\hat{s}(mN+n))^2}\right) \quad \text{dB}$$

where m=0,1, .. ,J-1 and J is the total number of frames, i.e., $J = \lfloor L/N \rfloor$.

$$AVSNRSG = \frac{1}{J}\sum_{m=0}^{J-1} SNRSEG(m) \quad \text{dB}$$

Note that s(n), n=0,2,…,L-1 is the original speech signal, which you can download from http://www.utdallas.edu/~loizou/ee6360_info.htm and $\hat{s}(n)$ is the reconstructed signal. The signal s(n) was sampled at 20,161 Hz with 16 bit A/D, and has a 512-byte header [see MATLAB code below on how to read the speech signal s(n)].

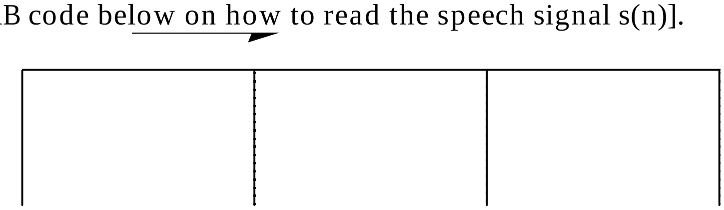

Fig. 1 Sliding rectangular window.

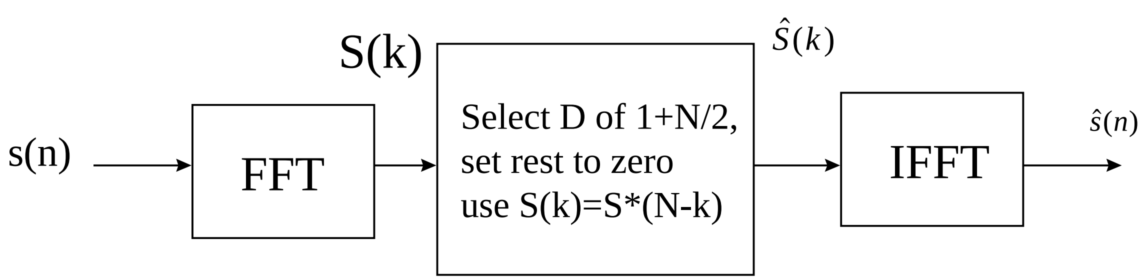

Fig. 2 FFT data compression scheme.

Tabulate your results as follows:

N	D	SNROV	AVSNRSG
64	2		
64	4		
64	8		
64	16		
128	4		
128	8		
128	16		
128	32		
256	8		
256	16		
256	32		
256	64		

Prepare a typed report that contains your MATLAB program, results, and answers to the following questions:

1. What is the effect of the parameter D (for fixed N) on the performance measure? Explain.
2. Express (D / N) as a percentage. Plot SNROV vs. (D / N)% and AVSNRSG vs. (D / N)% for the different values of N(64, 128, 256). From these plots describe the effect of N on the performance.
3. Modify your program such that the <u>first</u> D FFT components are selected instead of the D <u>dominant</u> (maximum) components. Obtain results and fill in the table shown above. Compare the new results to the previous ones and give comments.

REPORT

Submit a report answering all the questions above, including figures and tables. Your report needs to be TYPED. The report needs to be coherent, clearly written and concise. The report will be graded based on the technical content (90% of the grade) as well as on the writing (10% of the grade), i.e., in terms of presentation, grammar, syntax, etc.

<u>MATLAB CODE</u>

Use the following MATLAB code to read the signal s(n) from the binary file 'sent.ils':

```
fp=fopen('sent.ils','r');
hdr=fread(fp,512,'char');  % skip the 512-byte header
s=fread(fp,inf,'short') ; % read the whole speech signal into the array s(n)
fclose(fp);
```

If your PC has loudspeakers, you can listen to the compressed speech signal using the MATLAB command:

```
>> soundsc(shat,20161)
```

where **shat** is the reconstructed speech signal $\hat{s}(n)$, and 20,161 is the sampling frequency.