Spectral Analysis in C

Lab #8

Section J

Submitted By:

Wen Zhanghao

Submission Date:

April 9, 2014

Overview

Generally speaking, this lab combine new knowledge that the usage of fftw and pointer to the lab3, music note. The output from lab3 this time play as standard input into this lab8. Lab8 is also another form of lab 7 that using MATLAB to solve the problem.

We were provided by lab3 source code and then asked to pipe that standard input into lab8, reading into an array of types fftw\_complex, which is a two dimensional array. Scan in blocks of n numbers, put them in the real part of the array, and set the complex part to 0. You will then execute your Fourier transform, find the magnitude of each pair of real and imaginary numbers, and then take the index of the maximum and put it into the formula from the last lab that calculated the dominant frequency: 𝑖 ∗ 𝑓𝑆 𝑛 ⁄ where i is the index where the maximum can be found, fs is the sample frequency, and n is the block size.

Analysis

First of all, we need sample rate and block size value that take in from user’s input, known as command line arguments. So argv, argc would be used again. we create two section to store input data and output data by using fftw that actually convert real number and complex number. Use fftw malloc two create two section that with certain memory. Because the value stored is pointer, we can store value into it by just scanf(“”,&inXX). Use transform to execute input data into out data which includes real parts and imaginary parts. Use while loop to start iteration because we don’t know where the input reach the end. And in while loop we used for loop that run based on the size of block value. Then we compute the magnitude and find the index of magnitude value to compute the dominant frequency.

Design

When run the program, first we compile the two program, and then put the command line argument there to define the value of sample rate and sample size and duration time.

./note s 48000 d 2 | ./ spectral s 48000 n 24000

The first part that before the pipe command, is for give command line arguments to the noteGen.c file rather than lab8 file. The second part after pipe is this time’s file. Although there are some similar value between them, but that are two different things. Note file is given. We only need to use argc and argv in lab8 file. So in my program

int main(int argc, char \*argv[]){               //get the sampleSize and sampleRate 9         int i;10         int sampleRate = 48000;11         int sampleSize = 0;12         for (i = 1; i < argc; i++) {13                 if (argv[i][0] == 's') {14                         sampleRate = atoi(argv[i+1]);15                 }16                 if (argv[i][0] == 'n') {17                         sampleSize = atoi(argv[i+1]);   //sampleSize == 'N' in later code18                 }19         }20 I sued a for loop because I know clearly how many input are by argc. And find the character that match ‘s ’ and ‘n’ and then atoi(“next value after ‘s ’ and ‘n’ and changed them into integer ”).

Be sure to include <fftw3.h> library. And then write the followed code.

tw\_complex \*in, \*out;

tw\_plan p;

 = (fftw\_complex\*) fftw\_malloc(sizeof(fftw\_complex) \* sampleSize);        //alloc memory for them

t = (fftw\_complex\*) fftw\_malloc(sizeof(fftw\_complex) \* sampleSize)

 fftw\_plan\_dft\_1d(sampleSize, in, out, FFTW\_FORWARD, FFTW\_ESTIMATE);

This is given in FFTW documentation. To be simple, it generate two empty block and allocated certain memory.

Now we have the space to keep the data and process them. The first two lines from note file is format that match dat file. But we don’t need them as out data. So we fgets them to delete them before we scanf the data we want.

Stdin refers that computer would gets the data from standard rather than another file.

s(trash, 100, stdin);

we don’t know where to reach the end of file, so we use function while(!feof (stdin)) to make sure we scanf data until the end.

Based on the requirement, we cannot take the whole data and then process thme at one time, instead, we should store value partially, and compute them separately and transform them separately to get block index value. How many data we should take in per calculation depends on the value of blocksize.

Therefore, we for loop to scanf the data, transform them and compute them. The iteration times is block size. For in, we want the real part match the second column of the input data, and set the imaginary part all to zero. (for loop use int I to count )In[i][0] is real part and in[i][1] is imaginary parts.

To compute the mag, we need include math.h library. Sqrt , pow(X,x) are used. We comptue is one by one in the for loop of out, and we update max mag if we original mag meet value bigger than its. And we record the index.

The last part is to use fomular from last lab to calculate the dominant frequency. And then print. Once we get the value we want, we can destroy in and out pointer by fftw free() them.

Testing

Segmentation fault is main error when we test out program. The first program I did not write for loop, however, the ‘N’ position occupy my block size, so for example, 48000 sample rate \* 2 duration = 96000 times input data, I just used 24000 ( block size) to compute the index. The 24000 is far smaller than 96000 so it cause segmentation fault.

Comments

Understand what things to be computed, to be written in the code is the primary thing. For hours we don’t know how it worked because we don’t know the whatever the physics or math background knowledge. If don’t understand what to do, what the purpose of this program. This is just wasting time and it is irreverent to programming skills.