Programming Assignment #4

CS 246, Fall 2018

Due Wednesday, October 3

Programming Task #1

Implement a class for a 6 dB/octave band pass filter. I will give you the header file `BandPass.h`, which contains the class declaration:

```
class BandPass : public Filter {
  public:
    BandPass(float f=0, float Q=1, float R=44100);
    void setFrequency(float f);
    void setQuality(float Q);
    float operator()(float x);
  private:
    void reset(void);
    double irate,
           frequency,
           quality,
           a0, b1, b2,
           x1, x2, y1, y2;
};
```

(the `Filter.h` header file has been included). The member functions should do the following.

`BandPass(f,Q,R)` — (constructor) creates a band pass filter with central frequency f, Q–factor Q, and sampling rate R.

`setFrequency(f)` — resets the central frequency of the filter to be f. The Q–factor retains its current value.

`setQuality(Q)` — resets the Q–factor of the filter to the specified value. The central frequency retains its current value.

`operator(x)` — returns the result of applying the band pass filter to the input sample value x.

The band pass filter should be constructed by applying the low pass and high pass filters from the previous assignment in series. Recall that the (improved) low pass filter with cutoff frequency f_L has transfer function

$$H_L(z) = \frac{(1-a_L)(1+z^{-1})}{1-b_L z^{-1}}, \quad \text{with} \quad a_L \doteq \frac{1}{1+\theta_L},\ b_L \doteq \frac{1-\theta_L}{1+\theta_L},\ \theta_L \doteq \tan\left(\frac{\pi f_L}{R}\right)$$

and that the (improved) high pass filter with cutoff frequency f_H has transfer function

$$H_H(z) = \frac{a_H(1 - z^{-1})}{1 - b_H z^{-1}}, \quad \text{with} \quad a_H \doteq \frac{1}{1 + \theta_H},\ b_H \doteq \frac{1 - \theta_H}{1 + \theta_H},\ \theta_H \doteq \tan\left(\frac{\pi f_H}{R}\right)$$

The transfer function of the band pass filter is then

$$H_B(z) = H_L(z)H_H(z)$$

Your implementation of this band pass filter should use a *single* recurrence relation: you may **not** create separate low and high pass filters and apply the two filters in series.

Your submission for this part of the assignment will consist of the implementation file `BandPass.cpp`. You may only include the header files `BandPass.h` (which includes `Filter.h`) and `cmath`. In particular, you may **not** include the `LowPass.h` and `HighPass.h` header files.

Programming Task #2

Implement a function for empirically measuring the gain of a filter at a given frequency. I will give you the header file `FilterGain.h` which contains the following function prototype

```
float filterGain(Filter& F, float f, float R);
```

(the header file `Filter.h` is included). This function returns the measured (linear) gain of filter F at the frequency f, where R is the sampling rate used by the filter.

To measure the gain of filter at a given frequency f, you will have to apply the filter to a sine wave of frequency f, and find the maximum value of the resulting signal. A sine wave of 1 second in duration should suffice. However, you do have to be a little bit careful in computing the maximum. The output signal does not "settle" into a (phase–shifted) sine wave until the initial impulse dies down. So you should disregard the first portion of the filter output in computing the maximum value of the output signal, say from $t = 0$ up to a time $t = t_1$. I'll leave it to you to find a reasonable value for the time t_1.

Your submission for this part of the assignment should consist of the single file, named `FilterGain.cpp`. You may include only `FilterGain.h` (which includes `Filter.h`) and the standard C++ header file `cmath`.