Coding Assignment

Background Subtraction

Consider a mechanical system that is being driven by a sinusoidal signal $A\sin(\omega t)$, where $\omega = 2\pi f$ is the angular frequency, A is the amplitude and t is time. The system will have some periodic response that is measured as a signal $s(t)$.

The system can either be in a "free" state or in an "interacting" state. The response signals will be different in both cases, but the free response will always be periodic with the same period as the driving signal.

Background model

A background model is some kind of expected response of the free system during each cycle. It should be derived from the measured response of the free system.

The interacting signal shows additional features that one would like to examine after subtracting the background model.

Data

The provided data files (`background.txt` and `signal.txt`) contain times and signal values for 50 cycles each, where each cycle consists of 200 samples.

Task

Write code implementing the following specification. You can use the classes given below. (You will need to provide a minimal implementation of the class `TimeSeriesData`). You are also free to extend the interface of the given classes.

Tasks 1 and 2 require you to write code that will later be used in runnable applications for tasks 3 and 4.

1. Provide a method to create an instance of `TimeSeriesData` from one of the provided data files.
2. Implement a class representing an averaged, discretized background model (with $i = 0 \ldots N$ samples within the cycle) that can be refined by providing it new cycles of background signal data (out of $j = 0 \ldots M$ cycles):
   - The model should be configurable with a smoothing parameter ν between 0 and 1.
   - The model $b_{i,j}$ after refining it with the jth cycle should have the following value for sample number i within its modeled cycle:
     $b_{i,j} = \nu s_i + (1 - \nu) b_{i,j-1}$,
     where s_i is the ith sample of the input signal.
   - When adding the first cycle, since there is no previous cycles to compare to, take the smoothing factor to be $\nu = 1$ as a special case.
3. Write a command-line application that receives a file name (e.g. `signal.txt`) on the command line. It should read time-series data from the given file and use the given mock implementation `MockFittedBackgroundModel` to create a subtracted signal $s_i - b_i$. The subtracted signal should be written to a new file.
4. Write a second command-line application that receives a value for ν and two file names (e.g., `background.txt` and `signal.txt`) on the command line. The application should divide the data from the first file into cycles and use it to create a background model using your implementation from task 2, subtract that model from the data in the second file, and then write the subtracted signal to a new file.

Classes

In the following, `AbstractBackgroundModel` is a base class for different background models (the averaged model you should implement and the mock version given here). These models are intended to be used as follows:

1. In a first phase, the free signal (represented by the data in `background.txt`) is being fed to the model cycle by cycle (using the `addCycle` method). The model uses this data to refine itself.
2. In a second phase, the model is being used to create a periodic signal (consisting of the refined model for one cycle, repeated as often as necessary) that can then be compared to a different signal (represented by the data in `signal.txt`).

AbstractBackgroundModel

```
class AbstractBackgroundModel {
public:
    /**
     * @returns the i-th sample of the modeled background signal
     */
    virtual double evaluateBackground(int i) const = 0;

    /**
     * Refines the model using a new cycle of the background signal
     * @param sampleData a time series representing one new cycle of a signal
     */
    virtual void addCycle(TimeSeriesData sampleData) = 0;

    /**
     * Generate a modeled background signal
     * @param nCycles number of cycles that the generated signal should contain
     * @returns a time series consisting of the modelled background cycle
     *          repeated nCycles times.
     */
    TimeSeriesData createData(int nCycles);
};
```

MockFittedBackgroundModel

```
const int SAMPLES_PER_CYCLE = 200;
class MockFittedBackgroundModel : public AbstractBackgroundModel {
public:
    MockFittedBackgroundModel(
        double A = 0.05, double f = 1.0e3, double offset = 0.45, double phase = 5.0
        ) : A(A), f(f), offset(offset), phase(phase) {}

    double evaluateBackground(int i) const override {
        double dt = 1.0 / SAMPLES_PER_CYCLE / f;
        return A * sin(2.0 * M_PI * f * i * dt + phase) + offset;
    }

    void addCycle(TimeSeriesData sampleData) override {}

private:
    const double A;
    const double f;
    const double offset;
    const double phase;
};
```

Criteria

In order of decreasing importance:

1. The code must compile and run
2. The code should be well-structured, object-oriented, and reasonably documented.
3. Robust handling of user input is desirable.
4. The code should come with a *Makefile* or *cmake* file.
5. The expected time for completion is 1/2 - 1 working day.