CS601: Software Development for Scientific Computing

Programming Assignment 2 - Rectangular Grids and PDEs
Due: 20/9/2021

The objective of this assignment is to gain a hands-on experience with:

1. Implementing rectangular grids using object-oriented concepts and solving partial differential equations (PDEs) on the grid.

1 Problem Statements

1. Design and implement a class `RDomain` as a *sub-class* of the `Domain` class outlined in the lecture. Add methods and attributes to the `RDomain` class that you think are necessary. The class should have the capability of generating simple grids on the domain (Rectangular and 1D). An outline of the `Domain` is given below:

```
1  class Domain{
2  private:
3  //...
4  public:
5  //..
6  virtual void PrintGrid(string outputFileName) const=0;
7  //...
8  };
```

   - Using an object of the `RDomain` class, a user must be able to print the grid to a file via the member function `PrintGrid`. Do NOT change the signature of `PrintGrid` in your implementation. While implementing the `PrintGrid` function, use the File-IO functions `fopen`, `fwrite`, and `fclose` (or the corresponding `ofstream` class) to output the grid coordinates in *binary* format. You should write all the X-coordinates first followed by all the Y-coordinates. Do not write anything else to the file. Refer to the man pages of `fopen` and `fwrite` for details. Example usage of File-IO functions is given below:

```
1  #include<cstdio>
2  FILE *fp;
3  /*initialize the file handle to open the file outfile.bin in binary
       mode for writing.*/
4  fp =fopen("outfile.bin","wb");
5  /*write m double-precision values stored at contiguous memory
       locations starting from x*/
6  fwrite(x,sizeof(double),m,fp);
7  fclose(fp); //close the file handle
```

   - Using an object of the `RDomain` class, a user must be able to specify the step size(s) of the space variables.

2. Design and implement a class `GridFn` for implementing the grid function modeling the 1D heat diffusion problem discussed in class (refer slides). Use the three-point stencil. Also, assume that that the two ends of the metal bar are held at constant temperature of 0 and the initial temperatre distribution is given by $f(x) = x\sqrt{(l-x)^3}$. Note that the time-step δt should neither be part of the grid (`RDomain` class) nor the grid function(`GridFn` class). Instead, time-step should be a higher-level parameter.

3. Design and implement a class `Solution` for implementing the numerical computation of approximating the solving of a PDE over a grid. Specifically, implement your `Solution` class using the classes designed in 1 and 2. For a test case, assume that α, thermal diffusivity, is 1, l(length of the rod)=1.2, δx=0.4, and δt=0.1 as a test case. Design your classes in such a way that the following outline of the `Solution` is possible:

```
1     Create domain
2     Create grid function to operate on a domain
3     Create a solution and prepare to compute the solution
4     Set initial conditions
5     Iterate:
6       compute the grid function till the solution converges or a maximum number
            of steps has been reached.
7       (optional) Set boundary conditions
```

2 What you need to submit

- Create folders called `src` and `inc` to hold your C++ source and header files resp. Also create a folder called `bin` for storing any temporaries generated including your executable. Do not submit any temporary files generated. Your executable should accept three command-line arguments: l, δt, and δx in that order.
- Create a `Makefile` for building your solution. The `Makefile` should reside above the folders you created previously and at least have the following targets:
  1. solution: this target will build all your source code to create an executable for the solution
  2. clean: this target will remove any intermediate files that were created
- A shell script (this must be written in bash) called `runme` that builds your solution using the `Makefile` and executes it using the parameters provided. This script should take in three arguments: first, the input length of the rod, l, then the time-step δt, and space-step δx.

You must tag your source code and submit as you have done previously. The tag name to be used is: `cs601pa2submission`*. All tag names are case-sensitive..*