GLG490/598 Numerical methods
Homework #10
Due 11:59pm, 4/8/2021

(100 points)

Question: Use finite difference method to solve the following 2nd order ordinary differential equation:

$$\begin{cases} y'' = 3x + 4y \\ y(x = 0.0) = 0.0 \\ y(x = 1.0) = 1.0 \end{cases}$$

Review:

The ODE with the following format:

$$y'' + p(x)y' + q(x)y = r(x)$$

where $p(x), q(x), r(x)$ are functions of only x, can be approximated using finite difference method into:

$$a_i y_{i-1} + b_i y_i + c_i y_{i+1} = d_i$$

where

$$a_i = 1$$

$$b_i = -2 - hp(x_i) + h^2 q(x_i)$$

$$c_i = 1 + hp(x_i)$$

$$d_i = h^2 r(x_i)$$

Jacobi method:

$$y_i^{(k+1)} = \frac{d_i - a_i y_{i-1}^{(k)} - c_i y_{i+1}^{(k)}}{b_i}$$

Gauss-Seidel method:

$$y_i^{(k+1)} = \frac{d_i - a_i y_{i-1}^{(k+1)} - c_i y_{i+1}^{(k)}}{b_i}$$

It is important to note that, i is the index for the nodal points, and i ranges from 0 to n. k is the index for the iteration, and k ranges from 0 to any value that you want until you get the precise solution. You may want to check the notes 'Finite-difference-for-BVP-ODE.pdf' under the 'notes' folder of the class materials.

Requirements:

- You need to divide the interval of $x \in [0.0,1.0]$ into n=10 equal size sub-intervals.
- For the initial iteration, you need to provide a guess to all the unknows points (y_i for $1 \leq i \leq n-1$). The guess should be the average of the boundary conditions (a common thing to do). In

this case, the average of boundary conditions is 0.5. Therefore, the initial values of y should be y_i=0.5.

- Your iterative solver should have the following convergence criteria:

$$MAXIMUM\left(\left|y_1^{k+1} - y_1^k\right|, \left|y_2^{k+1} - y_2^k\right|, \ldots, \left|y_{n-1}^{k+1} - y_{n-1}^k\right| \right) < 1e-6$$

  After each iteration, compare the new y values with the previous y values. If all absolute values of the difference between any of these pairs of values are less than 1e-6, then stop the iteration.

- Your code must produce output that states which solver is being used and the number of iterations that were required for convergence. See example in the next bullet points.
- When I enter ‘./a.exe 1’ on the command line, my output is as follows:

```
$ ./a.exe 1
Jacobi method
143 iterations used
x[ 0]=0.00 y[ 0]=0.000
x[ 1]=0.10 y[ 1]=0.022
x[ 2]=0.20 y[ 2]=0.049
x[ 3]=0.30 y[ 3]=0.083
x[ 4]=0.40 y[ 4]=0.129
x[ 5]=0.50 y[ 5]=0.193
x[ 6]=0.60 y[ 6]=0.279
x[ 7]=0.70 y[ 7]=0.395
x[ 8]=0.80 y[ 8]=0.547
x[ 9]=0.90 y[ 9]=0.745
x[10]=1.00 y[10]=1.000
```

- When I enter ‘./a.exe 2’ on the command line, my output is as follows:

```
$ ./a.exe 2
Gauss_Seidel method
74 iterations used
x[ 0]=0.00 y[ 0]=0.000
x[ 1]=0.10 y[ 1]=0.022
x[ 2]=0.20 y[ 2]=0.049
x[ 3]=0.30 y[ 3]=0.083
x[ 4]=0.40 y[ 4]=0.129
x[ 5]=0.50 y[ 5]=0.193
x[ 6]=0.60 y[ 6]=0.279
x[ 7]=0.70 y[ 7]=0.395
x[ 8]=0.80 y[ 8]=0.547
x[ 9]=0.90 y[ 9]=0.745
x[10]=1.00 y[10]=1.000
```

- You can use EX12.c as a start.

- The answers provided by both solution methods are identical, as expected. However, notice that Gauss-Seidel is almost twice as fast (meaning less iterations) as Jacobi!

How to submit your homework

1. Change the name of your C code as 'FirstName-LastName-HW10.c'.
2. Send your code file to Mingming.Li@asu.edu and enter the email subject title as "Numerical Methods Homework 10".

Important comments

Note that when using methods that involve solving a set of equations, such as finite difference methods, there are 2 separate and independent issues with respect to how well you are reproducing the "real problem." As we saw here, the more we iterate (using the iterative method), the closer we get to accurately solving the set of equations. However, don't forget that the set of equations itself is an approximation to the real problem, so we should also perform an additional convergence test that compares the resultant answer as we increase the number of the intervals. Here, we used 10 intervals to represent the domain, but you could easily discover that this is not enough. If you modify your code to use 100 intervals, you will see a slightly different answer. It would be more appropriate if your code had a larger, outer loop that keeps doubling the number of intervals until the answer converges within some tolerance (like we did with HW08 and HW09).