ME526 Homework #5

Fall 2016

Andrew Alferman
Dan Magee

11/30/16

1.(a) **Write a program to compute the steady state solution to the second-order finite difference approximation of the heat equation using the point Jacobi iteration method.**

- A program was written in Python and has been attached to this writeup.

(b) **Now with $N_x = 21$ and $N_y = 21$ apply the Jacobi iteration to the discrete equations until the solution reaches steady state.**

- The attached code reaches steady state using the Jacobi iteration method.

How many iterations are required until the solution at (1, 0.5) *steadily* varies by no more than 0.00005 between the iterations?

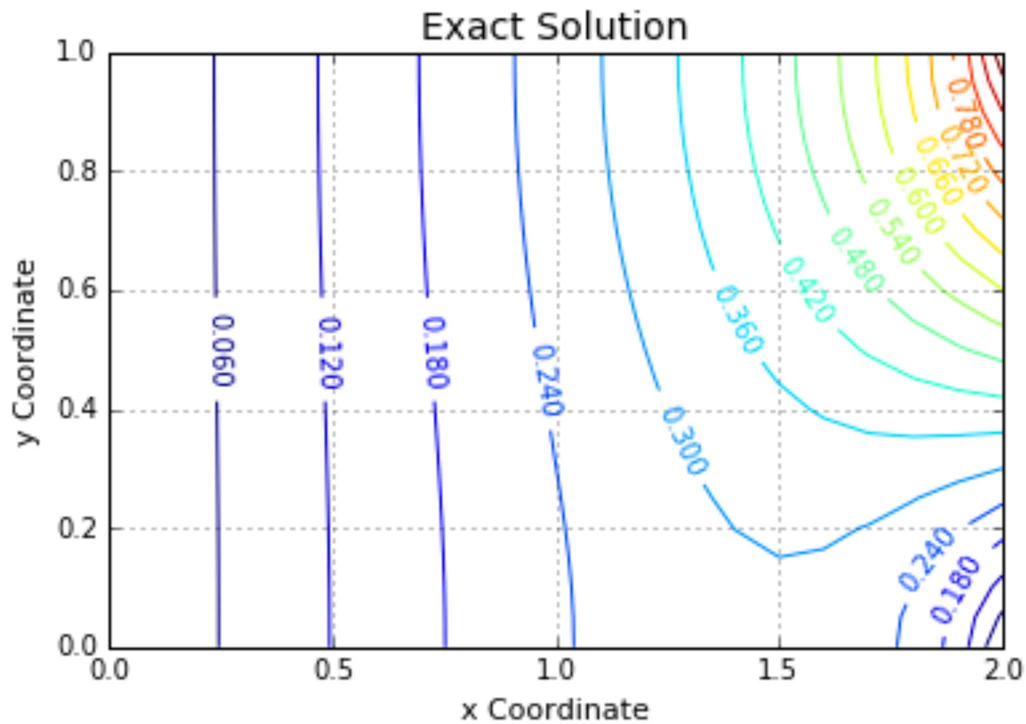
- The Jacobi iteration method uses 1112 iterations before the variation between the iterations becomes less than 0.00005 at the prescribed point, as demonstrated in the attached code.

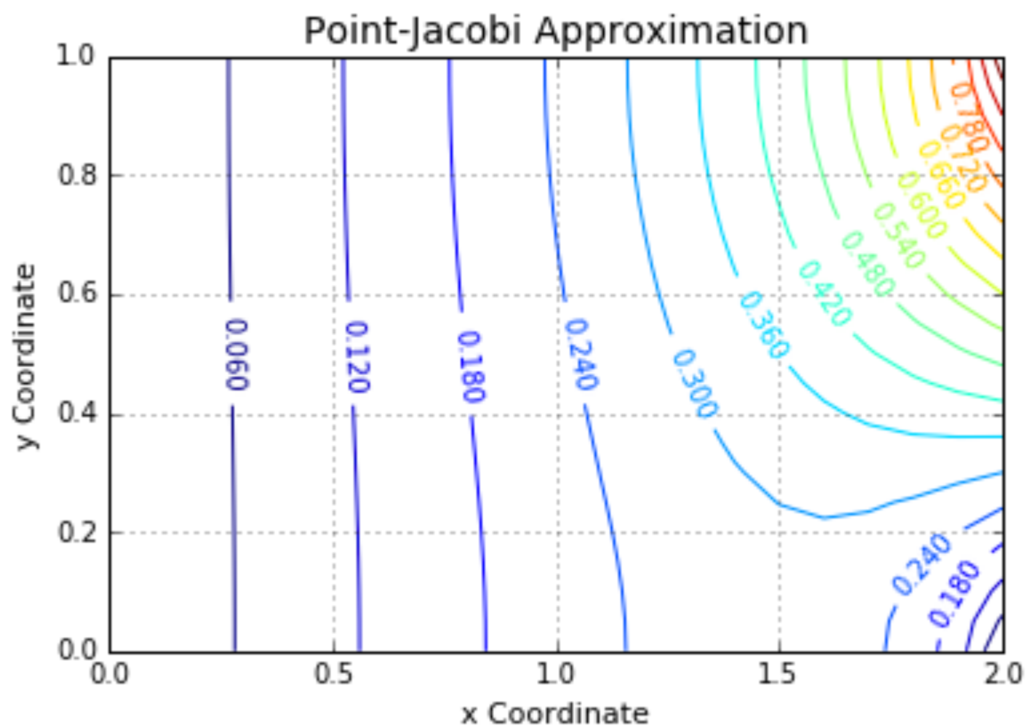
At this point, how does the numerical approximation compare to the analytical solution? What is the absolute error? What is the error in the numerical approximation relative to the analytical solution (percentage error)?

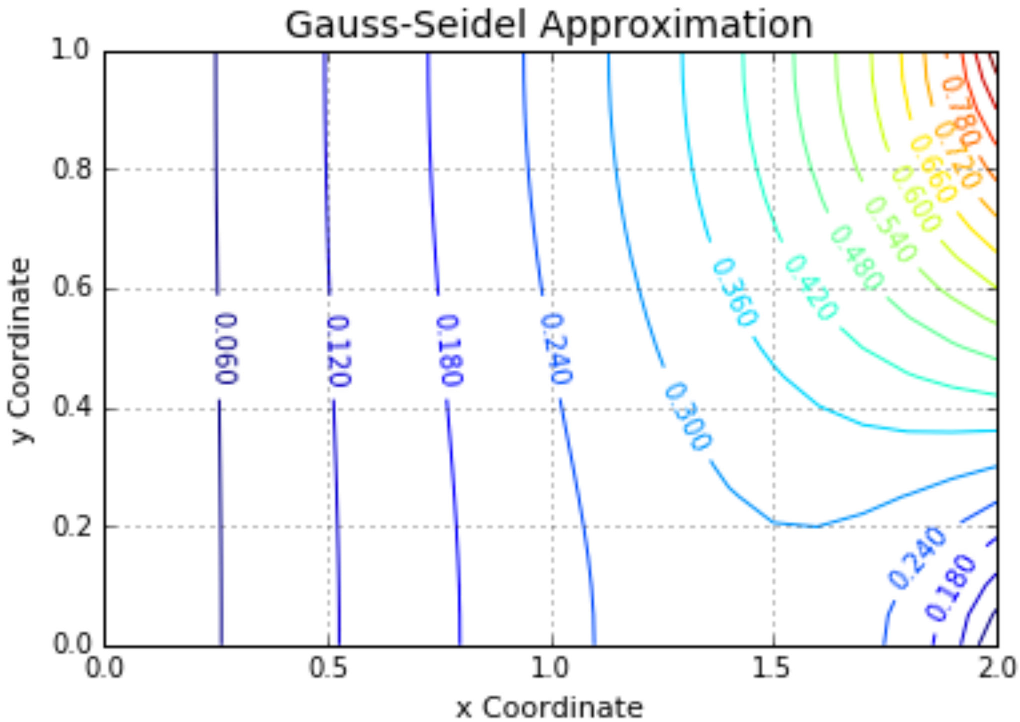
- The Jacobi iteration method produces what would likely be considered to be an acceptable error, depending on how close of an answer is desired. The analytical solution is 0.279, while the point Jacobi method yielded a value of 0.259, which is an absolute error of 0.20 degrees C and a percent error of 7.24%. In general, the Jacobi iteration method tended to underestimate the temperature throughout the plate due to the slow nature of the method.

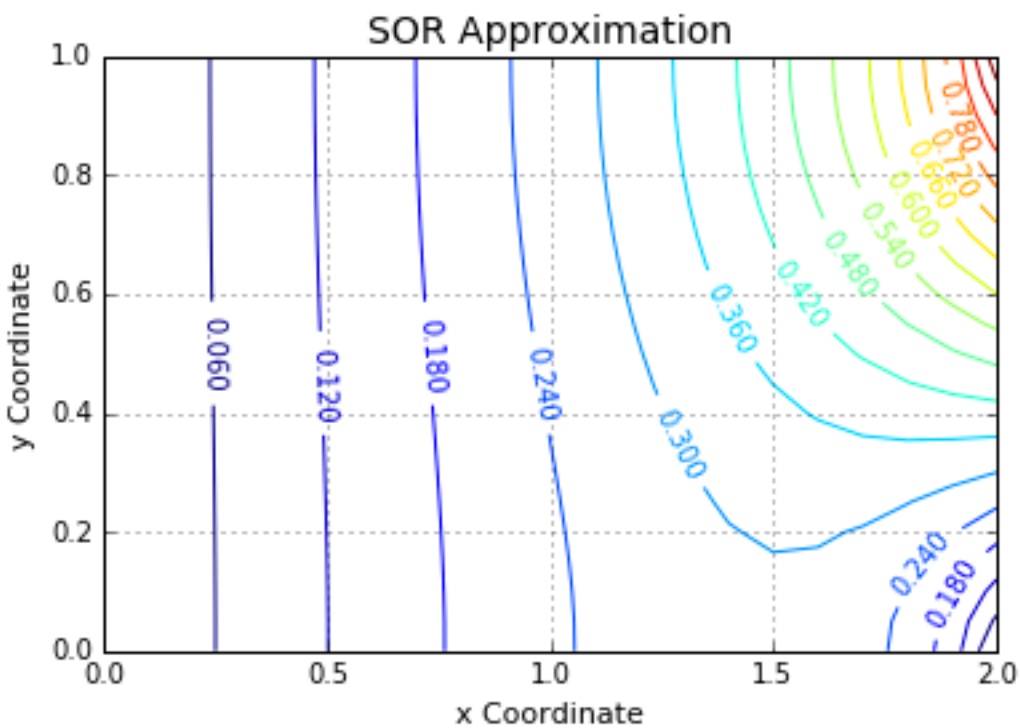
Plot isotherms of the numerical and exact temperature distributions (say, 16 isotherms).

- Isotherms are as follows for the analytical solution, point Jacobi, Gauss Seidel, and SOR:


Exact Solution
y Coordinate
x Coordinate
0.060
0.120
0.180
0.240
0.300
0.360
0.420
0.480
0.540
0.600
0.660
0.720
0.780
0.240
0.180


Point-Jacobi Approximation
y Coordinate
x Coordinate
0.060
0.120
0.180
0.240
0.300
0.360
0.420
0.480
0.540
0.600
0.660
0.720
0.780
0.240
0.180


Gauss-Seidel Approximation
y Coordinate
x Coordinate
0.0
0.2
0.4
0.6
0.8
1.0
0.5
1.5
2.0
0.060
0.120
0.180
0.240
0.300
0.360
0.420
0.480
0.540
0.600
0.660
0.720
0.780


SOR Approximation
y Coordinate
x Coordinate
0.0
0.2
0.4
0.6
0.8
1.0
0.5
1.5
2.0
0.060
0.120
0.180
0.240
0.300
0.360
0.420
0.480
0.540
0.600
0.660
0.720
0.780

(c) Repeat the above using Gauss Seidel and SOR. Compare the performance of the methods (number of iterations needed to reach the same accuracy).

- The attached code completed all of the same steps as the point Jacobi method. The Gauss Seidel approximation needed 702 iterations to converge to the same tolerance and achieved a percent error of 3.61% in that time. The SOR method needed only 203 iterations and received a percent error of 0.62% in that time. This demonstrates that Gauss Seidel and SOR both produce greater accuracy and faster computation that point Jacobi.

2. Consider the convection-diffusion equation in 1D.

(a) i. Solve the problem for 0<=t<=8 using

A. Explicit Euler for time advancement and second-order central for spatial derivative.

B. Leapfrog for time advancement and second order central for special derivative.

Plot the solution for 0,4,8. Use at least 51 points in the x direction.

- The problem was solved as directed. Plots can be found in the appendix to this writeup.

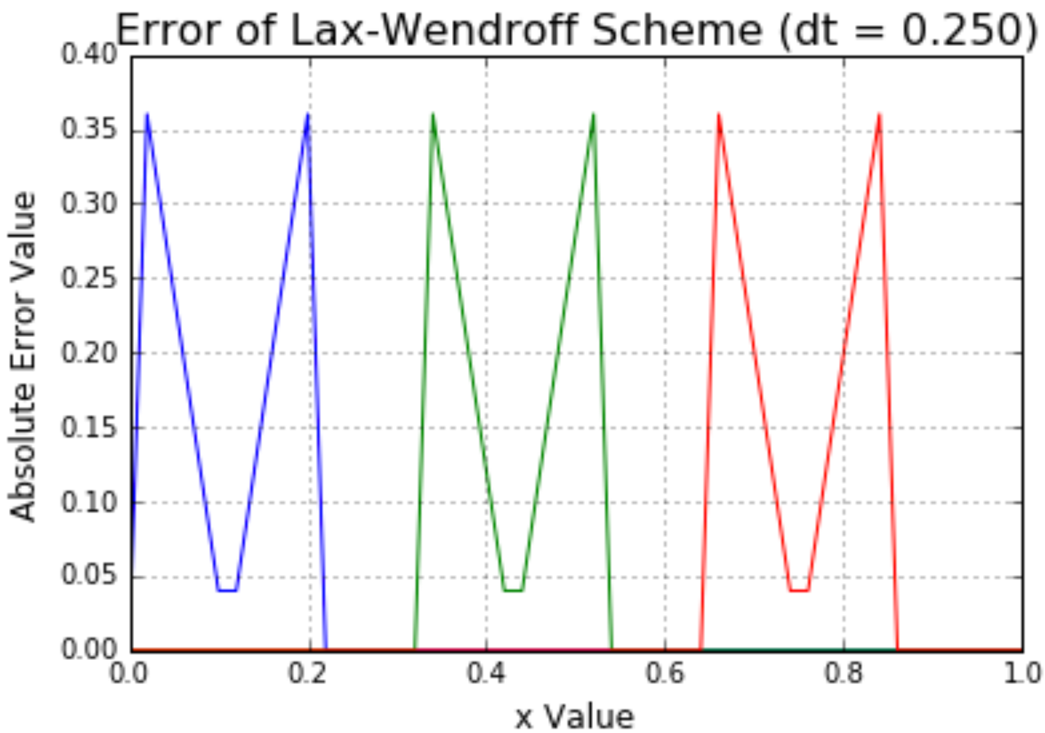
Discuss your solution in terms of the stability and accuracy of these schemes.

- The first scheme with the explicit Euler was generally stable throughout the entire range, though instability began to manifest for t = 8 seconds as the solution progressed towards the "wave". As the timestep was increased, the solution became less and less stable, and by the time u*delta t/ delta x had reached 0.8, both the solutions for t = 4 and t = 8 had completely blown up. Even at u*delta t/ delta x = 0.2, the accuracy at t=8 had been completely compromised before x=0.6. Accuracy and stability both increased as delta t decreased.

The second scheme with the leapfrog time advancement exhibited greater stability and accuracy than the first method, though many of the trends exhibited by the first method also applied to the second method. No solutions blew up completely while solving using this approach, though the accuracy began to be compromised as u*delta t/delta x approached 1.

ii. With the results in (i)A as motivation, the Lax-Wendroff scheme is suggested. What are the accuracy and stability of this scheme? Solve the problem using this scheme for gamma = 0.8, 1, and 1.1 and plot and compare with i.A.

- Plots of this scheme are included in the appendix. The accuracy and stability of the scheme improved as gamma increased to 1. The approximation was very good compared to the previous two schemes. However, as soon as gamma increased above 1, the scheme blew up. A plot of the error of this scheme for gamma = 1 is as follows:



(b) **Repeat parts i.A and i.B with the addition of second-order differencing for the diffusion term. Discuss your results and choices of time-steps.**

- The code was modified to include a term for alpha. If alpha were equal to one, the solution would be exactly the same as the answer to the previous parts. Time-steps were chosen as exactly the same as the ones used for the previous steps in order to provide the best comparison possible.

The first scheme provided an excellent (smooth) approximation of the equation with diffusion included. The solution only began to diverge in accuracy and stability as gamma increased above 0.8. At gamma = 1, the solution blew up for t = 4 and t = 8, thus the presence of diffusion improved the behavior of the numerical solution.

The second scheme became unconditionally unstable as a result of the presence of diffusion. For t = 4 and t = 8 across the entire range of gamma values, the solution blew up.