Mech 510

Programming Assignment 3

Due Date: Monday, November 19, 2018

In this assignment, you will solve the incompressible, laminar, 2D energy equation for a given velocity field. After a few validation cases, you will solve a problem involving the thermal development of flow in a channel (in which the velocity is already fully-developed). Along the way, you will apply approximate factorization to solve the system of equations that arises from an implicit time discretization.

Because this problem is somewhat more challenging from a programming perspective, you should start being systematic about how you debug things — an excellent habit to get into. I strongly encourage you to keep a log describing what tests you ran, what the results were, and what (if anything) you fixed in the code as a result. This is solely for your own benefit, and does not need to be turned in.

The computational domain, shown in the figure, is a rectangular channel of length 5 and height 1. The velocity field is given by the fully-developed profile

$$\begin{aligned} u(x,y) &= 6\bar{u}y(1-y) \\ v(x,y) &= 0 \end{aligned}$$

with $\bar{u} = 3\frac{\text{m}}{\text{sec}}$. The temperature will be fixed at the inflow (see cases below) and the walls (lower wall: $T = 0$, upper wall: $T = 1$), and the temperature profile will be considered fully-developed at the outflow. Finally, the flow conditions are such that $\text{Re} = 50$, $\text{Pr} = 0.7$, and $\text{Ec} = 0.1$.

1. **Code the flux integral and verify correctness.** Write a subroutine that will take the temperature and velocity fields (along with necessary constants) and compute the flux integral for the energy equation. Use second-order centered fluxes throughout; that is, the flux at $\left(i+\frac{1}{2}, j\right)$ should depend exclusively on data at (i, j) and $(i+1, j)$.

   Verify the correctness of your solution by comparing the flux integral computed by your code with the exact flux integral for the following data:

$$\begin{aligned} T(x,y) &= T_0 \cos(\pi x)\sin(\pi y) \\ u(x,y) &= u_0 y \sin(\pi x) \\ v(x,y) &= v_0 x \cos(\pi y) \end{aligned}$$

   For this case, the exact flux integral is given by:

$$\begin{aligned} \int\int \left(u\frac{\partial T}{\partial x} + v\frac{\partial T}{\partial y} - \frac{1}{\text{Re}\cdot\text{Pr}}\nabla^2 T \right) dA &= u_0 T_0 \pi \cos(2\pi x)\, y \sin(\pi y) \\ &\quad + v_0 T_0 \pi x \cos(\pi x)\cos(2\pi y) \\ &\quad + \frac{2T_0\pi^2 \cos(\pi x)\sin(\pi y)}{\text{Re}\cdot\text{Pr}} \end{aligned}$$

To perform this comparison, use a uniform mesh on the square $[0,1] \times [0,1]$. Initialize the control volume average temperature and velocity for both interior and ghost cells using the functions given above (the average value is within second-order of matching the function value evaluated at the cell center, remember). Use your subroutine to compute the flux integral.[1] Compute the L_2 norm of the difference between your flux integral and the exact flux integral for a 10×10 mesh and as many finer meshes as required to demonstrate second-order accuracy convincingly.

Report: Your report should include the error norms for each mesh and an estimate of the order of accuracy.

2. **Code the source term and verify correctness.** This part of the assignment applies the procedure for verifying flux correctness to the source term. As for the fluxes, use second-order centered differences here. Using the same test data, the exact source term is:

$$\frac{1}{A}\int S\,dA \;=\; \frac{\mathrm{Ec}}{\mathrm{Re}}\left(2\left(u_0\pi\cos(\pi x)\,y\right)^2 + 2\left(v_0\pi x\sin(\pi y)\right)^2 + \left(u_0\sin(\pi x) + v_0\cos(\pi y)\right)^2\right)$$

Report: Again, your report should include the error norms for each mesh and an estimate of the order of accuracy.

3. **Code an explicit time advance scheme.** The purpose here is to provide additional confirmation of the correctness of your previous work, in addition to checking that you've assembled things with the proper signs and with proper boundary conditions. Also, later on you'll do an efficiency comparison between your explicit and implicit schemes for steady-state problems. You can choose any explicit time advance scheme you like; be sure to determine in advance the maximum stable time step.

4. **Code implicit time advance scheme.** Implement the implicit Euler time advance scheme, using approximate factorization to decompose the solution of the resulting linear system and the Thomas algorithm to solve the tri-diagonal problems on each mesh line. Code for the Thomas algorithm, in both C and Fortran, is on Connect.

   Unfortunately, this is one piece of the code that is difficult to test in isolation; see Connect for data for a single step of the case in part 5, with $\Delta t = 0.1$. The final data in that file is for the end of the first time step, with the boundary conditions updated and ready to begin the second time step.

5. **Stability and Accuracy Check.** Run your code (with both explicit and implicit time advance) for the 5×1 channel with a 25×10 mesh, an initial temperature distribution of $T(x,y,0) = y$ and the exact fully-developed temperature distribution as an inflow boundary condition:

$$T(0,y,t) = y + \frac{3}{4}\mathrm{Pr}\cdot\mathrm{Ec}\cdot\bar{u}^2\left(1-(1-2y)^4\right)$$

   Your solution should converge almost but not quite exactly to the fully-developed profile everywhere. The difference between your solution and the exact solution is of course due to

[1] Debugging hint: When testing your code, adjust u_0, v_0, and T_0 independently to test the correctness of various terms in your flux integral.

truncation error. Satisfy yourself that your steady-state results are second-order accurate for this case.
Report: Present in your report convincing evidence that your steady-state solution is second-order accurate.

6. **Efficiency Test.** Re-run the previous case using a sequence of meshes, ranging from 25×10 to 200×80, if possible (one or the other time advance scheme may be too slow on the finest mesh). For your explicit code, use a time step that is nearly (80-90%) as large as the maximum stable time step. For your implicit code, use as large a time step as your code will allow, which may vary with mesh size.
Report: For each case, report the time step and the run time.[2] If you found that your implicit scheme had a maximum stable time step, explain why that occurs.

7. **The "Real" Problem.** Change the inflow boundary condition to $T(0,y,t) = y$, and use your code to determine the temperature gradient at the bottom wall. Your goal is to determine quantitatively (a) the fully-developed value for temperature gradient[3] and (b) the thermal development length for the channel. You will need to extend your domain beyond five channel widths for this problem; it is up to you to determine how far the domain must be extended to capture the entire thermal development length.

   You will need to define development length. Keep in mind the following guidelines:

   - Your definition must be quantitative: "It looks fully developed here" isn't good enough.
   - Your definition should provide the same ratio of development length to channel width if the channel is made twice as wide (with the same Reynolds, Prandtl, and Eckert numbers).
   - Likewise, your definition should give the same result for development length if the temperature is doubled everywhere (this could be done, for instance, by doubling the upper wall temperature and the Eckert number).

   Report: Your report should include, at minimum: a plot of the temperature gradient at the bottom wall of the domain as a function of x; the fully-developed value for temperature gradient; and the development length (including a description of how you determined this). Also, you provide evidence that your results are grid converged.

[2] How you get your code to report run time will differ depending on your language, compiler and operating system, but it's generally pretty easy to get the computer to do the timing for you instead of doing it with your wristwatch.

[3] Yes, I know that it's possible to find this analytically for this problem, but your value should be based solely on your numerical results.