Physics 427 – Introduction to Computational Physics

James B. Mertens

Exam 2 – April 2, 2021

Read the following. It contains important information about this exam.

- You must submit your solutions to Canvas **by 4:00 pm**! Please plan accordingly. This gives you approximately 1.5 hours for this exam.
- There is both a written and computational part to this exam. You will need to download both from Canvas, and submit solutions for both to Canvas.
- You are allowed to use any reference materials you like for this exam, except each other.
- You should make an effort to answer all questions on this exam. Clearly identify your final answers, and clearly explain your solutions. This will be graded according to the syllabus rubric, so effort will be heavily weighted.

Problem 1: In the computational portion of this exam, we will obtain solutions to the Lane-Emden equation, which describes a self-gravitating, spherically symmetric, polytropic fluid. It is often used as a (rather) simple model of a star. The differential equation describing this system is given by

$$\frac{1}{\xi^2}\frac{\mathrm{d}}{\mathrm{d}\xi}\left(\xi^2\frac{\mathrm{d}}{\mathrm{d}\xi}\theta(\xi)\right) + \theta(\xi)^n = 0.$$

for a coordinate ξ and function $\theta(\xi)$. The variable n is a polytropic index which, for our purposes, is just a constant.

a) Write a solution to this equation when $n = 0$, subject to the conditions $\theta(0) = 1$ and $d\theta/d\xi = 0$ when $\xi = 0$.

Hint: You can directly integrate this system twice, or use a polynomial function as a guess.

b) For solving this system numerically in Python, we need to reduce this equation to a set of first-order equations we can supply to `solve_ivp`. Provide such a set of equations; you will need these for the computational portion of the exam.

Problem 2: Clearly explain your answers to the following questions.

a) For integrating PDEs and ODEs, we saw a number of methods. What condition did we require in order for these methods to be stable?

b) We were able to develop numerical schemes that were accurate to higher order in a resolution Δx using Richardson Extrapolation. Describe how this is accomplished.

You may support your description with an example.

c) We saw certain partial differential equations could be classified according to their hyperbolicity. Describe each of these classifications qualitatively, and name one example of a system in each case, 1) elliptic, 2) parabolic, and 3) hyperbolic.

There are also limitations to this classification system. What types of systems might you encounter that are not (readily) classified as one of these systems?

d) One example of an integration method we encountered was the midpoint method. While we explored this in the context of ODEs, this method is also a valid integration scheme for PDEs.

The following code integrates the 1+1-dimensional diffusion equation by taking an Eulerian timestep. Provide pseudocode, or clearly describe, what modifications you could make to this code in order to take a step of the midpoint method.

[Note: This problem is intended to be more challenging. You do not have to provide working code, although feel free to provide code as a solution if you are inclined.]

```
import numpy as np
import matplotlib.pyplot as plt

def diffusive_step(f, dx, dt) :
    """Parameters:
    f: 1-d array containing function values
    dx, dt: point spacing, timestep
    """
    f_new = np.zeros_like(f)

    # Laplacian of f
    f_lap = (f[2:] - 2*f[1:-1] + f[:-2]) / dx**2

    # Forward-Euler time step
    f_new[1:-1] = f[1:-1] + dt*f_lap

    # Dirichlet boundary conditions
    f_new[0] = 0
    f_new[-1] = 0

    return f_new

xs = np.linspace(-100, 100, 201) # Choose a grid of x-points
dx = xs[1] - xs[0] # Determine spacing between grid points
dt = dx/10. # Choose a timestep
n_steps = 1000 # Number of timesteps to take

f_ini = np.exp(-xs**2/20**2) # Initial Gaussian profile
f = f_ini
for n in range(n_steps) : # Perform integration
    f = diffusive_step(f, dx, dt)

```

```
33  plt.plot(xs, f_ini)
34  plt.plot(xs, f)
```