University of Colorado
Department of Computer Science

Numerical Computation CSCI 3656

Spring 2016

Problem Set 12

Issued: 12 April 2016

Due: 21 April 2016

1. *[15 pts]* Use your Simpson's 1/3 rule code from PS11 to compute $\int_{1.0}^{1.8}[e^x + 0.5]dx$ with $h = 0.1$ and $h = 0.05$. Now use one layer of Romberg integration on those results to get a better answer. Finally, comment on how close each of the three numerical answers involved in the calculations above — i.e., with $h = 0.05$, with $h = 0.1$, and with Romberg — is to the correct value that you figured out using calculus in PS11.

2. *[10 pts]* Now compute $\int_{1.0}^{1.8}[e^x+0.5]dx$ using three-term Gaussian quadrature. Comment on the accuracy of the answer and the computational complexity of this approach, compared to the methods that you have used to solve this same problem in this problem set and in the previous one.

3. *[6 pts]* Consider the following system of ordinary differential equations:

$$\begin{aligned} \dot{x}(t) &= 16(y - x) \\ \dot{y}(t) &= 45x - y - xz \\ \dot{z}(t) &= xy - 4z \end{aligned}$$

   Is this system of equations—a famous example from chaos theory called the Lorenz equations—linear? Explain your reasoning: why or why not? Of what *order* is this ODE system?

4. *[24 pts]* In your favorite programming language, implement the forward Euler method for solving that system of differential equations numerically.

   Your program should take the following inputs:

   - an initial condition $[x(t = 0), y(t = 0), z(t = 0)]^T$
   - a time step size h
   - the number of time steps N over which to solve the ODE system

   Generate two 10,000-point-long trajectories using $h = 0.01$ (that is, $N = 10,000$), one from the initial condition $[x, y, z]^T = [1, 1, 1]^T$ and the other from the initial condition $[x, y, z]^T = [1.01, 1.01, 1.01]^T$.

Make time-domain plots of the x coordinates (i.e., x versus t) for both trajectories—preferably on the same set of axes so you can compare them easily. The initial conditions are almost identical, but the time course of the x coordinate of the two trajectories should diverge fairly quickly. That's one of the main attributes of chaos.

Make state-space plots of both trajectories—preferably in 3D (x vs. y vs. z)—or just y vs. x if your environment doesn't support 3D plotting. These two state-space plots should look pretty much identical to one another. That's the other main attribute of chaos.

Please turn in your plots and a copy of your solver code.