AMATH 301 - Fall 2016 Due 4:30pm
Homework 8 Wednesday, 30 November, 2016

Exercise 1: A Chaotic ODE

The following three-dimensional autonomous chaotic ODE was proposed and analyzed in a recent paper, ***Analysis of a novel three-dimensional chaotic system*** (Li et al., 2013):

$$\begin{cases} \dot{x} &= -ax + fyz \\ \dot{y} &= cy - dxz \\ \dot{z} &= -bz + ey^2 \end{cases}$$

Where x, y, z are the state variables, and a, b, c, d, e, f are positive constant parameters. You will now model this system, reproducing a few of their results. The paper is avaliable at

`http://www.sciencedirect.com/science/article/pii/S0030402612002823?np=y.`,

and you should check it out.

Much of this work will parallel that done in video lecture 23.

(a) You will begin by creating a cube of initial values. The cube should be oriented in the natural manner; that is, with edges parallel the x, y, and z axes. Each edge should be .4 long. The spacing between grid points should be .05. The center of the cube is at $(x, y, z) = (.3, 2, 12)$. Using `meshgrid()` create three three-dimensional arrays containing the x-coordinates, y-coordinates, and z-coordinates, respectively. To avoid dealing with high dimensional arrays (as is done in the online video lectures), rescale each of these matrices as a vector (a matrix, `A` , can be rescaled as a vector by `V = A(:)`). Store the resultant vectors as the rows in a three row matrix with the x, y, and z coordinate vectors in the 1st, 2nd, and 3rd rows, respectively. Save the matrix as `A1.dat`. To check you work, try the command `plot3(x,y,z,'.')` and verify that you have a cube of points spaced as described above.

(b) Let $a = 16$, $b = 5$, $c = 10$, $d = 6$, $e = 18$, $f = .05$. Model the above system with Runge-Kutta 4 (you can reuse some of your code from Homework 7), from time 0 to time 7, using a time step $\Delta t = .01$, and with the initial condition cube created in (a). Structure your solution in the same way as your initial condition, however you should now have a 3-dimensional array (size = `701, 3, 729`). If your output is called

`A`, then `A(1,:,:)` will be your initial condition cube, `A(2,:,:)` will correspond to the positions of those initial points after Δt seconds, and so on. Be sure to properly vectorize your code, or it will run extremely slowly (and will probably terminate prior to completion on scorelator). Once completed, use the `squeeze()` function to make the 3-D array of solutions into 3 2-D matrices containing the x, y, or z coordinates at the different time steps as the rows of these matrices. Type `help squeeze` to learn how to use this command.
Save the matrix of x-coordinates as `A2.dat`, the matrix of y-coordinates as `A3.dat`, and the matrix of z coordinates as `A4.dat`. The size of each of these should be 701×729 (the x, y, and z coordinates of the 729 points from time 0 to time 7 in steps of 0.01).

(c) You will not submit anything more for this problem, but you should construct some cool plots! Plot your solutions at the last time using the code:
`plot3(A2(end,:),A3(end,:),A4(end,:),'.');` where `A2`, `A3`, and `A4` are what you created in part b). Verify that your solution looks similar to the plots of the system found in the paper referenced above (this is also a good way to help you troubleshoot if things are not working out). Another interesting plot you should make is of the tracks of a few of your initial positions:
`plot3(A2(:,1:5),A3(:,1:5),A4(:,1:5));`
You can increase the number of tracks you plot (you have one for each of the 9^3 starting positions), but note that this will quickly become quite taxing on your computer.
Finally, try watching the full evolution of the system by typing:

```
for i=1:length(t)
    plot3(A2(i,:),A3(i,:),A4(i,:),'.');
    axis([-.5,.5,-10,10,0,50])
    drawnow
    pause(dt)
end
```

Make sure to comment out your plots before submitting the assignment.

Exercise 2: A Chaotic Map

A map is a system which progresses in discrete steps according to a function f, such that we have $x_{n+1} = f(x_n)$ (this is opposed to the ODEs we have studied, which advance continuously according to a differential equation). In this class, we have seen maps in the context of solving ODE's, where we have approximated the solution to the continuous systems with similarly constructed maps (such as the forward Euler or Runge Kutta methods). But maps are worth studying in their own right. In this problem we will be looking at the famous logistic map:

$$x_{n+1} = r x_n (1 - x_n)$$

Despite its simplicity, it exhibits complex and chaotic behavior.

a) With $x_0 = .7$ calculate 30 iterations of the logistic map where r takes on the values 2.5, 3.2, 3.52, and 4. Store these as the columns of a 31×4 matrix (x_0 is included, so the first row of this matrix should be equal to 0.7 in every entry). Each column of the matrix corresponds to one of the r values in the order given.
Save this matrix as `A5.dat`
Try plotting each of these solutions (make sure to comment this out before submitting). For $r = 2.5$, 3.2, and 3.52, you should notice that the solutions eventually begin to repeat themselves (in the language of dynamical systems, they "converge to a periodic orbit"). We will call the "period" of an orbit the number of iterations it takes to return to where it was before. For example, the sequence "1,2,1,2,1,..." would have period two, while the sequence "1,2,6,4,5,1,2,6,4,5,..." would have period five. Define P_R as the period of the orbit the sequence of the logistic map approaches when $r = R$. Looking at the plots of your results, identify the periods when $r = 2.5$ (that is, $P_{2.5}$), when $r = 3.2$, and when $r = 3.52$.

   Save these in a column vector $[P_{2.5}; P_{3.2}; P_{3.52}]$ as `A6.dat`. Notice that with $r = 4$, the solution does not appear to have any regular behavior.

b) Now find the first 500 iterations of the map with $x_0 = .7$ under a whole range of r values: `r = 2.5:.001:4`. These should be stored in a 501×1501 matrix, in which each column represents the sequence from x_0 to x_{500} for a different r. Make sure your code is properly vectorized (there should only be a single `for` loop).

Save the last 100 iterations (a 100×1501 matrix) in `A7.dat`.
To see the nature of the long-term solutions with respect to r, plot these last hundred iterations using this code:
`plot(R,A7,'k.','Markersize',1);`
You have made what is known as the bifurcation diagram of the logistic map, a celebrated image from the history of chaos theory. Notice how the map first becomes fractal-like, repeatedly splitting, before eventually giving rise to chaos as r increases. Make sure to comment out the plot before submitting.