6.094 Introduction to MATLAB®
January (IAP) 2009

Homework #3: Solving Equations and Curve Fitting

What to turn in: Please turn in a document (preferably Word or PDF) of your work including code and plots. Be sure to suppress intermediate output. Assume you should provide final output unless told otherwise. Note that the last two problems are again optional!

Exercise 1. Optimization
Do Problem 13 in Chapter 3 on page 179-180 of Palm. Use `fminbnd`.

Exercise 2. Polynomial Fitting
Do Problem 37 in Chapter 5 on page 349 of Palm. In part a), don't worry about J, S, and r^2. You can run `polyfit` to also get the `S` output (see `help polyfit`), and type `S.r (S.normr` in MATLAB 2008) to get a measure of the error of fit. The lower S.normr, the better the fit.

Exercise 3. Node Equations

Do Problem 9 in Chapter 6 on page 404-405 of Palm. You can either write a script or a function, whichever you prefer.

Exercise 4. Higher-Order ODEs

Do Problem 30 in Chapter 8 on page 537 of Palm. In addition to the time-series for $\theta(t)$, please plot the phase-plane (the phase plane is just a plot of one integrated variable against the other, so plot $\theta(t)$ vs. $\dot{\theta}(t)$).

Exercise 5. Lorenz Attractor

The Lorenz attractor, introduced by Edward Lorenz in 1963, is a non-linear three-dimensional deterministic dynamical system derived from the simplified equations of convection rolls arising in the dynamical equations of the atmosphere. For a certain set of parameters the system exhibits chaotic behavior and displays what is called a strange attractor.

$$\dot{x} = \sigma(y - x) \quad (1)$$
$$\dot{y} = x(r - z) - y \quad (2)$$
$$\dot{z} = xy - bz \quad (3)$$

where a is called the Prandtl number and b is called the Rayleigh number. $\sigma, r, b > 0$, but usually $\sigma = 10$, $b = 8/3$ and r is varied. The system exhibits chaotic behavior for $r = 28$ but displays knotted periodic orbits for other values of r.

1. Use `ode45` to simulate the Lorenz equations with parameters starting from an initial condition of $(x, y, z) = (-6.2, -7, 23)$ at $t = 0$ for $T = 60$ seconds. Plot the time-series for x, y, z on the same figure.
2. Plot the xyz phase-plane using `plot3`. The resulting shape is the famous Lorenz attractor.
3. (Optional) Explore the behavior of the Lorenz equations for different values of r. What kinds of behavior can you find? Periodic? Steady-states? Chaotic?

Exercise 6. Julia Sets (Optional)

In this problem you will generate quadratic Julia Sets. The following description is adapted from Wikipedia at `http://en.wikipedia.org/wiki/Julia_set`. For more information about Julia Sets please read the entire article there.

Given two complex numbers, c and z_0, we define the following recursion:

$$z_{n+1} = z_n^2 + c \tag{4}$$

This is a quadratic map, a dynamical system closely related to the logistic map you explored in a previous problem. Given a specific choice of c and z_0, the above recursion leads to a sequence of complex numbers $z_1, z_2, z_3...$ called the orbit of z_0.

Depending on the exact choice of c and z_0, a large range of orbit patterns are possible. For a given fixed c, most choices of z_0 yield orbits that tend towards infinity. (That is, the modulus $|z_n|$ grows without limit as n increases.) For some values of c certain choices of z_0 yield orbits that eventually go into a periodic loop. Finally, some starting values yield orbits that appear to dance around the complex plane, apparently at random. (This is an example of chaos.) These starting values, z_0, make up the Julia set of the map, denoted $\mathcal{J}_c$. In this problem, you will write a MATLAB script that visualizes a slightly different set, called the filled-in Julia set (or Prisoner Set), denoted $\mathcal{K}_c$, which is the set of all z_0 with yield orbits which do not tend towards infinity. The "normal" Julia set $\mathcal{J}_c$ is the edge of the filled-in Julia set.

Figure 6 illustrates a Julia Set for one particular value of c. You will write MATLAB code that can generate such fractals in this problem.

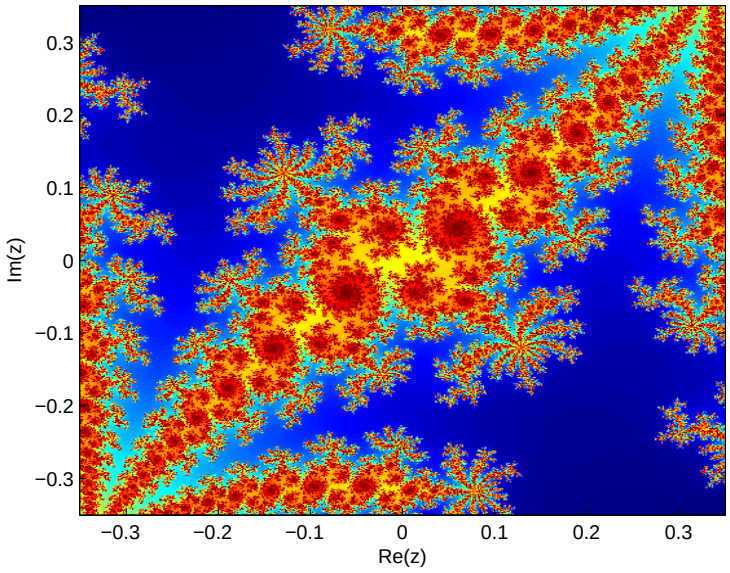

Figure 1: Detail on Julia Set for $c = -0.297491 - 0.641051j$

1. It has been shown, that if the modulus of z_n becomes larger than 2 for some n then it is guaranteed that the orbit will tend to infinity. This test makes it straightforward to plot Julia sets for quadratic maps using a computer.

   Write a function `n=julia_check(z0, c, maxiter)` that takes in an initial value, `z0`, the Julia Set parameter, `c`, and the maximum number of iterations, `maxiter`, to check to see if the quadratic map (4) becomes larger than 2.

   The output parameter, `n`, is the number of iterations it took to become larger than 2, or `maxiter` if the iteration stays bounded after the maximum number of iterations. This is called the 'escape velocity' of a particular point z_0. It is a measure of how fast a particular point is tending toward infinity.

2. Write a script, `julia.m`, that computes the Julia Set, $\mathcal{K}_c$, for a grid of complex numbers. You can generate such a grid using the function `complexmesh` that you wrote in a previous problem. The script should generate a convergence matrix, $\mathbf{M}$, that holds the 'escape velocity' for each point z in the grid of complex numbers. You will need to use the function you wrote in part (a) also.

   By visulazing $\mathbf{M}$ using the function `imagegamma` that we have provided, you can get an image of the Julia Set (to the resolution of the grid). The γ parameter in this plotting function controls a non-linear mapping that gives better visualization of the image.

3. Test your script for $c = -0.75$ and a max iteration count of 50. Find if a 500×500 grid of complex numbers, with with $-2 \leq \mathrm{Re}(z) \leq 2$ and $-2 \leq \mathrm{Im}(z) \leq 2$ is in the Julia Set. Visualize it using `imagegamma` with $\gamma = 0.1$.

   The resulting structure is called the San Marco Fractal. The bands of color represent the different 'escape velocities' of various initial conditions. Note that it may take a second for the script to run, since it is evaluating convergence at 250,000 points.

4. Using a max iteration value of 50 and a 500×500 grid of complex numbers, with with $-1.35 \leq \mathrm{Re}(z) \leq 1.35$ and $-1.05 \leq \mathrm{Im}(z) \leq 1.05$, find the Julia Set for $c = -0.297491 - 0.641051j$. Plot it using $\gamma = 0.007$.

   In a different figure, find the same Julia Set using a max iteration value of 500. Compare the two figures. You will note that the second plot has much finer structure than the first. Why is this?

5. You can use your script to focus in on certain areas and see detailed substructure. Using the same value of c as in (e) and a max iteration value of 500, evaluate the Julia Set on a 500×500 grid of complex numbers, with with $-.35 \leq \mathrm{Re}(z) \leq .35$ and $-.35 \leq \mathrm{Im}(z) \leq .35$. Again, plot it using $\gamma = 0.007$. You will see a more detailed view of the Julia Set in this region.

   Use your script to further explore the Julia Set in (e), focusing in on regions that you think may have particularly nice structures. Print out some of the images you get from your explorations. Note that as you focus into certain regions, you will need to up the iteration count to see the detailed structure there. In addition, be warned that if you attempt to evaluate the Julia Set at too many points, MATLAB may take forever to run.

6. Explore the structure of Julia Sets for various different c. Some particularly nice values of c are $c = i$ which generates the dendrite fractal, $c = -0.123 + 0.745i$ which generates Douady's rabbit fractal, and $c = -0.726895347709114071439 + 0.188887129043845954792j$ which just looks cool. You can find additional c to try from the Wikipedia article mentioned above.

   When visualizing the Julia Sets using `imagegamma` you will want to tweak the parameter γ to get a good looking image. It will be different for each fractal. Set it to suit your aesthetic tastes.

   Print out some of the images from your explorations. If you find a particularly nice image you may want to save your convergence matrix, $\mathbf{M}$, using `save`.

Exercise 7. Mandelbrot Set (Optional)

This problem is a quick and easy extension of the previous one on Julia sets. We have included it here because the Mandelbrot Set is a famous picture and we're 90% of the way there from doing the previous problem. The following description of the Mandelbrot Set is adapted from Wikipedia at `http://en.wikipedia.org/wiki/Mandelbrot_set`.
In mathematics, the Mandelbrot set is a fractal that is defined as the set of points c in the complex plane for which the iteratively defined sequence

$$z_0 = c \tag{5}$$
$$z_{n+1} = {z_n}^2 + c \tag{6}$$

does not tend to infinity. Figure 7 illustrates the Mandelbrot set.

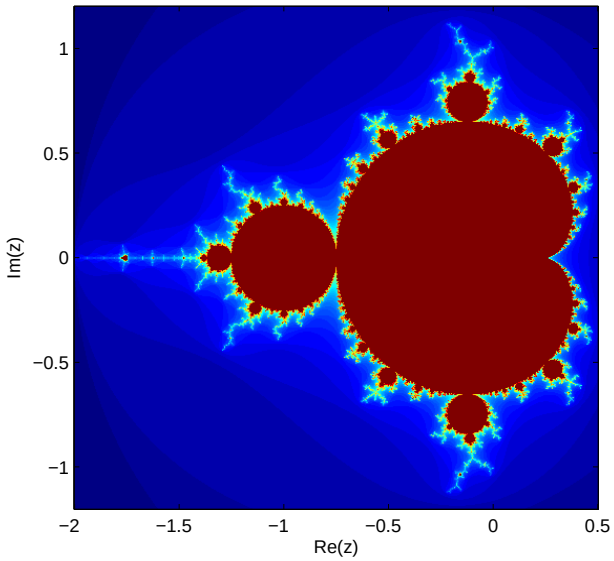

Figure 2: The Mandelbrot Set

The Mandelbrot set was created by Benoit Mandelbrot as an index to the Julia sets: each point in the complex plane corresponds to a different Julia set. The points within the Mandelbrot set correspond precisely to the connected Julia sets, and the points outside correspond to disconnected ones.
Intuitively, the "interesting" Julia sets correspond to points near the boundary of the Mandelbrot set; those far inside tend to be simple geometric shapes, while those well outside look like dust surrounded by blobs of color.

1. Write a script `mandelbrot.m` that checks if a grid of points is in the Mandelbrot set. The script should generate a convergence matrix, **M**, that holds the 'escape velocity' for each point z in the grid of complex numbers. The structure of this script will be almost identical to `julia.m`. You will literally have to change one line.

   Pick limits and resolution so that you can see the entire set and not too much 'dead space'. Visualize the set using `complexplotgamma`. Pick a reasonable value of γ that shows the details and is aesthetically pleasing. Print out your plot of the Mandelbrot set.

2. Focus in on certain portions of the set by changing the limits. Note that as you focus into certain regions, you will need to up the iteration count to see the detailed structure there. You may also want to change the γ to see the detail better. Print out a few images from your explorations.