MATLAB Computer Project 1: **Exploratory Analysis of Newton Fractal Networks**

Maya Ramde

*Description: This document details the written calculations and fractal images produced from Netwon’s Method and Modified Newton’s Method in MATLAB. The overall project analyzes the behavior of Newton’s Method and the fractal behavior of the convergence of polynomial functions on the complex plane. At the end of the document, you can view the code.*

(1) Find the roots of the function f in the form x + iy.

From the first term,

🡪 Root 1

Convert to polar coordinates

Substitute polar values in complex number formula

Let

According to De Moivre’s Theorem, if , then

Since we are evaluating z3 = -1, we need 3 roots where r3 = -1 and *.*

In this case, r is a real number, so the only solution to r3 = -1 is r = -1

The solutions for are

Therefore, the roots of z3 = -1 are

1. z = -1
   1. Written in complex number form: z = -1
2. 1. Written in complex number form:
3. 1. Written in complex number form:

(2) Implement Newton’s method in the computer for solving this equation. Once you are convinced the algorithm is working, execute the code with a large number of starting points x0+i y0 in the window where −2 ≤ x0 ≤ 2 and −2 ≤ y0 ≤ 2.

Color each starting point in the grid with one of four colors indicating to which root the method converged or if it did not converge within some maximum number of iterations. This should give you a nice pattern.

A picture containing chart

Description automatically generated

(3) Around each solution you should see a region in which, once an approximate solution is inside this region, the convergence to the solution is quadratic. What do these regions look like and how big are they?

Newton’s method has quadratic convergence because the difference between the root and the approximation is squared. In the case of z3 + 1, there are three roots and each of the three colors in the graph represent the function converging to one of the roots in a given iteration.

The complex pattern that borders each solution consists of teardrop-shaped blobs that are outlined by intricate, smaller-scaled versions of more teardrop-shaped blobs. Each teardrop radius appears to be ½ of the radius of the preceding larger teardrops. Since Newton’s method converges quadratically, we can generalize that the number of teardrop-shaped regions increases quadratically around each solution.

(4) Zoom in on a region where the colors change and run Newton’s method again. Can you see some fractal behavior?

Chart

Description automatically generated

I zoomed in on a region where the colors changed in the image. The zoomed image demonstrated the property of self-similarity, as the image followed the same continuous, universal pattern though dimensions were re-scaled. The graph is scale-invariant as the pattern’s features persist. Therefore, the fractal behavior is independent of scale and geometric space. The intricate arrangement of colors also shows the sensitivity of Newton’s method based on the initial conditions.

(5) When two very close points converge to different solutions, this may be because the initial steps of Newton’s method are too large. Consider the modified Newton’s method

zk+1 = zk − tk \* f(zk) / f ’(zk) ,

where tk is chosen so that |zk+1 − zk| < δ for some selected δ.

Does the fractal image you created change?

Chart

Description automatically generated

After implementing the modified Newton’s method into my program, the new product is a graph that demonstrates fractal behavior. I edited my code to evaluate zk+1 at each iteration by comparing the absolute difference from zk+1 (when calculated in regular Newton’s method) and the zk. If the difference is greater than the specified δ, a coefficient tk is then computed to decrease the step size and produce a modified zk+1. In this case, I chose δ = 0.25. The re-use of the old derivative in tk yields a more appropriate value of the derivative in the corrector step.

By incorporating a corrector step to modify function’s step size, Newton’s method’s sensitivity to initial points that are set too far apart significantly decreases and ensures that the method converged quadratically. The solutions converge at a faster rate since each iterate is set closer to the last. In turn, the modified method requires less iterations to converge to a root. As a result, the teardrop-shaped fractals which surround the large solution sectors appear smaller and contain less variance in color.

***MATLAB CODE***

Text, letter

Description automatically generated

Graphical user interface, text, application

Description automatically generated