Newton Raphson Fractals: A Review

Bhumika Gupta

Department Computer Science Engg. G.B.Pant Engg. College
Pauri Garhwal, Uttrakhand, India

Abstract: This paper gives an overview about Fractals, type of fractals then it tell us about Newton raphson method and then explains about Newton raphson fractals.

Keywords: Fractal, Self-similarity, Iterated Function System (IFS), Newton Raphson Method, Newton Raphson Fractals.

1. INTRODUCTION

In recent years many articles and books have focused on the mathematics of fractal sets [1] and on the physical mathematical interpretation of dynamic phenomena in fractal structures. Particularly important is the mathematics of iterated function systems (IFS) [2], which can be considered the basis for a proper definition of fractal sets [3] and multi fractal measures. IFS can be not only successfully applied in image and signal processing [4], Fractal sets are mathematical models of non-integer dimensional sets satisfying certain scaling properties. These may be thought of as objects that are obtained by an infinite recursive or inductive process of successive microscopic refinements. A mathematical fractal looks the same at all scales of magnification. This is an approximation to physical fractals which appear similar to the original object only for a certain range of scales.

Self-similar sets are special class of fractals and there are no objects in nature which have exact structures of self similar sets. These sets are perhaps the simplest and the most basic structures in the theory of fractals which should give us much information on what would happen in the general case of fractals (cf. Kigami [5, p. 1–2, 8]).

2. FRACTALS

2.1. INTRODUCTION

Fractal Geometry is a new science. It was a result to the advances in mathematical representation of equations using computers. It was given its name by the mathematician Benoît B. Mandelbrot of IBM. The name comes from the Latin word fractus which means irregularly broken. Though the Mandelbrot set is not considered as a fractal Benoît B. Mandelbrot is considered the father of fractals. No one can deny the close relation between chaos and fractals, that would be explained later.

2.2. DEFINITION

Fractals are self similar geometric shapes which mean that after magnifying any part of the original shape we get the same shape all over again. Fractals appear both in mathematical equations and in nature. For example in nature they appear in lightning, plants, mountains and most of the rough surfaces. Some of the mathematical examples of fractals are: Cantor set, Koch curve and Julia set. Fractals immeasurably enhance this world-view by providing a description of much around us that is rough and fragmented of objects that have structure on many sizes [6] like mountains, coastlines, rough surfaces, etc. So given that kind of details enough to describe dynamically created objects; fractals are considered the “geometry of chaos” [7]. Fractals are used to describe attractors’ texture and other chaos related geometric objects. Chaotic attractors are defined as attractors with fractal structure or fractal’s geometric property [8]. So we can clearly see the relation between chaos and fractals.” Fractals can help detect chaos” [8].

2.3. TYPES OF FRACTALS

Cantor Set: It’s considered one of the simplest and most famous fractals. It was named after Georg Cantor who invented it. It’s formed by removing the middle third interval of any interval (i.e. [0, 1]) then repeating the same operation over and over again on the remaining elements which gives the sequence:

0, 1, 1/3, 2/3, 1/9, 2/9, 7/9, 8/9, …

In form of numbers it doesn’t really mean anything for the first time. But after plotting its graph representation of it, it’s obvious how The Cantor Set is a fractal as each part of the set is a repetition of the main set. For example the interval [0, 1/3] is a downscaled version of the main interval (proof [7]). The interval [0, 1] has the same cardinality as The Cantor Set. It could be proven as there exists a bijective function that maps elements from The Cantor Set to the interval [0, 1] that function is:

\[
    f \left( \sum_{k=1}^{\infty} a_k 3^{-k} \right) = \sum_{k=1}^{\infty} \left( \frac{a_k}{2} \right) 2^{-k}
\]

Figure 1 Cantor Set
Koch Curve: Its importance lies in that it represents the meaning of a fractal structure and some of the rules made by Mandelbrot to define fractals. It’s constructed by a simple line called the initiator. The curve is constructed by replacing the middle third part of the line by an equilateral triangle without its base then repeating the same operation on the generated lines in the shape. After doing that operation many times we get what might look like a curve but does not have any smooth curves which means that this curve is not differentiable (as no tangent could be placed on the curve).2

Figure 2 Koch Curve

The Pascal Triangle: It’s derived from the famous triangle used to represent the coefficients of the expansion of the polynomial \((1 + x)^n\), \(n\) represents the row number starting from 0 and the number of coefficients are \(n+2\) starting from 0. The value of any cell of the triangle could be obtained from the equation:

\[ b_k = \frac{n!}{k!(n-k)!} \]

where \(n\) is the row number and \(k\) is the number of the element in the row.

That equation was derived from the binomial theorem which states that:

\[(x + y)^n = \sum_{k=0}^{n} \frac{n!}{k!(n-k)!}x^{n-k}y^k\]

The fractal is seen by coloring a group of numbers that have the same characteristics like coloring odd numbers in black and even numbers in white, coloring multiples of 3 in black and other numbers in white, etc.

Figure 3 Pascal Triangle

3. APPLICATIONS OF FRACTALS

Natural Events: Fractals are used to represent some natural events. Natural events that happen as bursts that seem unrelated in anyway could be have a fractal representation on the time line. An example is that we consider The Cantor Set’s graph as time lines. The longest is the main timeline and we could assume that some elements as the events that happens through that period. Coastline, maps and borders: On examining a small scale map of the world the continents on the map will appear as curves almost smooth with some bends. On examining a larger scale map those curves turns into other curves which combined form the first one. As the scale is changed, the ability to notice curves of small bays and bends in the coastlines decreases. So when measuring the length of the coastline on deferent scales we will definitely get deferent results. That was a problem that faced Mandelbrot as the length of the border between Spain and Portugal was measured once to be 616 miles and 758 miles. Fractals and their dimensions are used to solve that kind of problems such as measuring maps, rivers and coastlines. That could be mapped to the Koch curve or any similar fractal.

As a note at the end of the talk about fractals, objects obtained from fractals equations and description of fractals should not confused with the fractal itself as fractals are not bounded by size of any kind of limits but graphs expressing them are bounded such as the figures given to describe The Cantor Set and The Koch Curve are not the fractals rather than just a graphical representation.

4. METHODOLOGY

4.1 NEWTON RAPHSON METHOD

In numerical analysis, Newton’s method (also known as the Newton–Raphson method), named after Isaac Newton and Joseph Raphson, is a method for finding successively better approximations to the roots (or zeroes) of a real-valued function.

\[ x : f(x) = 0 \]

The algorithm is first in the class of Householder’s methods, succeeded by Halley’s method. The method can also be extended to complex functions and to systems of equations.

The Newton-Raphson method in one variable is implemented as follows:

Given a function \(f\) defined over the reals \(x\), and its derivative \(f'\), we begin with a first guess \(x_0\) for a root of the function \(f\). Provided the function is reasonably well-behaved a better approximation \(x_1\) is:

\[ x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \]
Geometrically, \((x_1, 0)\) is the intersection with the x-axis of a line tangent to \(f\) at \((x_0, f(x_0))\). The process is repeated as until a sufficiently accurate value is reached.

\[ x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \]

The idea of the method is as follows: one starts with an initial guess which is reasonably close to the true root, then the function is approximated by its tangent line (which can be computed using the tools of calculus), and one computes the x-intercept of this tangent line (which is easily done with elementary algebra). This x-intercept will typically be a better approximation to the function's root than the original guess, and the method can be iterated.

Suppose \(f : [a, b] \rightarrow \mathbb{R}\) is a differentiable function defined on the interval \([a, b]\) with values in the real numbers \(\mathbb{R}\). The formula for converging on the root can be easily derived. Suppose we have some current approximation \(x_n\). Then we can derive the formula for a better approximation, \(x_{n+1}\) by referring to the diagram on the right. We know from the definition of the derivative at a given point that it is the slope of a tangent at that point. That is:

\[ f'(x_n) = \frac{\Delta y}{\Delta x} = \frac{f(x_n) - 0}{x_n - x_{n-1}} \]

\[ f'(x_n) = \frac{\partial x}{\partial y} = \frac{f(x_n) - 0}{x_n - x_{n+1}} \]

Here, \(f'\) denotes the derivative of the function \(f\). Then by simple algebra we can derive

\[ x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \]

We start the process off with some arbitrary initial value \(x_0\). (The closer to the zero, the better. But, in the absence of any intuition about where the zero might lie, a "guess and check" method might narrow the possibilities to a reasonably small interval by appealing to the intermediate value theorem.) The method will usually converge, provided this initial guess is close enough to the unknown zero, and that \(f'(x_0) \neq 0\). Furthermore, for a zero of multiplicity 1, the convergence is at least quadratic (see rate of convergence) in a neighborhood of the zero, which intuitively means that the number of correct digits roughly at least doubles in every step.

The behavior of \(N-R\) method of expanding a small area into a large one, is exactly the sort of behaviour we expect to give rise to self-similar fractals. So if we were to start a Newton-Raphson iteration at each point on the real line, run each iteration until it converged to within a given tolerance level of a root, and then color the starting point according to which root it ended up at, we might well expect to see fractal shapes. Fractals on one line are not very interesting, however; so let's work in the complex plane. The Newton-Raphson iteration still works perfectly well there, so we will be generating some fractals.

**4.2 Newton Raphson fractals**

In this paper, we present a simple geometric generation principal for the fractals that is obtained when applying Newton method to find, the solution of non linear eq[10]. Recently computer graphics has become an important tool for studying this and similar algorithm[11] and many useful and beautiful fractal images have resulted[12,13].

The behavior of \(N-R\) method of expanding a small area into a large one, is exactly the sort of behaviour we expect to give rise to self-similar fractals. So if we were to start a Newton-Raphson iteration at each point on the real line, run each iteration until it converged to within a given tolerance level of a root, and then color the starting point according to which root it ended up at, we might well expect to see fractal shapes. Fractals on one line are not very interesting, however; so let's work in the complex plane. The Newton-Raphson iteration still works perfectly well there, so we will be generating some fractals.

**5. ILLUSTRATION**

Example 1-

Here's an example fractal, generated from the polynomial \(z^4 + 1\) - so the four roots of the function are at \(-1, +1, -i\) and \(+i\).

In this image, we see a large boring area surrounding each root of the function - as we would expect, since any point near a root will converge rapidly to that root and do
nothing interesting. But between the areas of boring well-behaved convergence, we see some beautiful fractal shapes. Let's zoom in on one of those boundary areas:

Just as we predicted - each of the heart-shaped blobs making up the boundary line is itself composed of boundary lines made up of further heart-shaped blobs. This pattern is a true fractal.

Example 2-

Here's a rather different example. This time the function being used is $(z-3)(z-2)(z-1)z(z+1)(z+2)(z+3)$, so it has seven roots strung out in a long line:

In this case, the fractal shapes are much smaller compared to the overall structure of the image. But they're not completely absent. If we zoom in on a couple of the little blobs on the boundary lines, we see this:

Each blob is divided up into coloured areas similar to those covering the whole plane, and on each dividing line we see more blobs looking much the same as the larger blobs.

Example 3-

Here's a rather different example. This time the function being used is $z-(z^8-17z^4+16)$

If we zoom in on a couple of the little blobs on the boundary lines, we see this:

Decoration (fractal art)

These images are reasonably pretty, but we can make them better. One obvious thing we can do, by noticing which root of the function the iteration ended up at, is to count how many iterations it took to get there. We can then colour each pixel a different shade of the colour assigned to that root depending on the number of iterations. So, using the obvious approach of setting the pixel shade to the number of iterations modulo the number of available shades (so that each colour cycles through those shades), we see something like this:

This is not only prettier, but it also shows us exactly where each root of the function is - instead of just
knowing the roots are somewhere in the large coloured areas, we can now positively identify each root as the centre of the bright spot in each area.

The cyclic behaviour is not quite optimal, though; it works well enough if the number of available colours is limited, but it means there are sudden edges (like the ones at the very centre of each region) where a dark colour suddenly becomes a bright colour again. Perhaps if we have true colour available, it would be better to have the pixel shading be monotonic - always getting darker the more iterations are needed, but fading out by less and less and never actually reaching blackness.

Now that's starting to look much nicer, I think. But it would be even better if the visible boundaries between different shades of the same colour could be removed. I suspect that doing this rigorously requires some really horrible maths and a lot of special cases, but I've found that a good ad-hoc approximation is obtained simply by looking at the last iteration, in which the point first comes within the specified distance of a root. We look at the distance $D_0$ from the previous point to the root and the distance $D_1$ from the new point to the root, and we know that the threshold distance $T$ is somewhere in between the two. Looking at $(\log T-\log D_0)/(\log D_1-\log D_0)$, in other words whether the log of the threshold radius was near to the start or the end of the inward distance travelled by the point (on a logarithmic scale), produces a perfectly acceptable result which we can use to smooth out those boundaries:

References
