Basins of Attraction [Updated: June, 2015]

James F. Epperson

July 8, 2015

[Note: This section requires some familiarity with complex arithmetic.]

We've done enough with root-finding methods to understand that the root we converge to will depend—perhaps very much so—on the choice of initial guess. We can formalize this idea by defining the notion of a *basin of attraction*. Roughly speaking, the basins of attraction for a function, for a given method, are the sets of initial values from which the iteration will converge to each root.

An example will illustrate this, then we will formalize it with a definition. Consider the function

$$f(x) = x^2 + \sin(\pi x)$$

and assume we are using Newton's Method. Using the plotting functions in MATLAB confirms that $x = \zeta_1 = 0$ and $x = \zeta_2 \approx -0.75$ are the two (real) roots of this function. So, the set of all initial values x_0 such that Newton's method applied to f converges to the root $x = \zeta_1$ is the basin of attraction for $x = \zeta_1$. Similarly, the set of all initial values x_0 such that Newton's method applied to f converges to the root $x = \zeta_2$ is the basin of attraction for $x = \zeta_2$. This example is set up for real values, only, but we will want to formalize it for complex variables. Fig. 1 shows a plot of this function.

The formal definition is:

Definition 1 (Basins of Attraction) *Given a function f defined on the complex plane, $\mathbb{C}$, with roots $\zeta_1, \zeta_2, \ldots, \zeta_k \in \mathbb{C}$, and a (convergent) root-finding iteration defined by*

$$z_{n+1} = g(z_n),$$

the Basin of Attraction for the root ζ_k is defined to be

$$\mathcal{B}_{f,g}(\zeta_k) = \{\zeta \in \mathbb{C} | \text{ the iteration } z_{n+1} = g(z_n) \text{ with } z_0 = \zeta \text{ converges to } \zeta_k\}$$

Clearly, if we are close enough to one of the roots, then we should be in the basin of attraction for that root. But we have had enough experience with Newton's method to know that the iteration can jump around a bit before settling in to converge. In Fig. 2 we illustrate this for the two real roots of our example function. For each of $3,000$ points $\zeta \in [-2, 1]$ we have drawn a vertical line varying in color according to the following scheme:

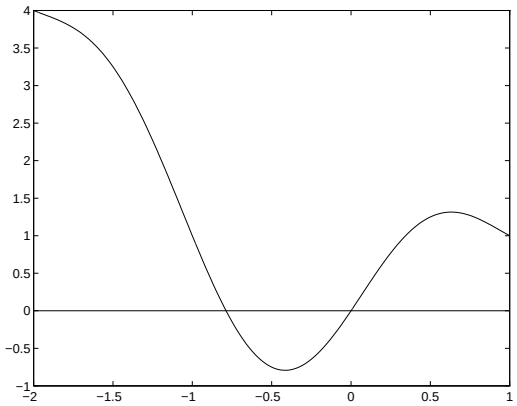

Figure 1: Basins example

- If the iteration with $x_0 = \zeta$ converges to the root $\zeta_1 = 0$ then the line is light blue (cyan).
- If the iteration with $x_0 = \zeta$ converges to the root $\zeta_2 = -0.78723712453434$ then the line is green.
- If the iteration with $x_0 = \zeta$ fails to converge to either ζ_1 or ζ_2, then the line is red.

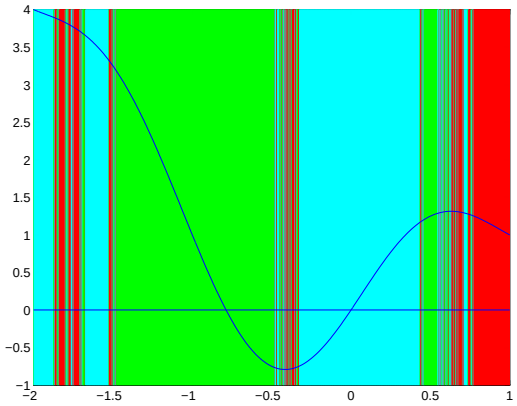

Figure 2: Basins example

Obviously, in the immediate vicinity of each root we have a broad swath of the appropriate color. But in the "transiton regions" we get each color switching back and forth. This is in fact *fractal* in nature.

What we are going to do for the rest of this section is explore the notion of basins of attraction, experimentally, for a very simple polynomial in the complex plane. The results are definitely surprising.

Define the function

$$f(z) = z^4 - 1.$$

This has the four roots $\zeta = 1, -1, i, -i$. We will construct a color plot in the complex plane akin to Fig. 2, showing the basins of attraction for Newton's method applied to this function.

We will consider the square region in the complex plane defined by

$$R = \{z \in \mathbb{C} \,|\, z = x + iy,\ -1.25 \le x \le 1.25,\ -1.25 \le y \le 1.25\}$$

We will associate the roots with colors as follows:

- $\zeta = 1$ is red.
- $\zeta = i$ is green.
- $\zeta = -1$ is yellow.
- $\zeta = -i$ is (dark) blue.

We expect to get wedge-shaped regions of the appropriate color centered at each root, with a fractal region along the diagonal boundaries.

Coding this for a modern MATLAB installation is fairly easy; in fact, the following m-file ought to do the job.

```
xmin = -1.25;
xmax = 1.25;
ymin = - 1.25;
ymax = 1.25;
figure(1)
axis([xmin, xmax, ymin, ymax]);
hold on
n = 100;
N = 5*n;
h = 2.5/N;
h2 = 0.5*h;
for kx = 0:N
     x = kx*h + h2;
     for ky = 0:N
          y = ky*h + h2;
          z = x + y*i
          c = bnewt(z)
          plot(x,y,'.','color',c);
     end
end
```

Note: The function `bnewt` does the actual Newton iteration and returns the color as a single character string.

Running this code can take a long time, and, in fact, my ancient MATLAB on my almost-as-ancient laptop was unable to save the finished figure because of memory issues. I was able to get the figures here by going to the library at the University of Michigan. (To get the code to run *and* save the image, I had to take $n = 17$, which ran very fast but produced a very poor picture.) A black plus sign (+) marks the location of each of the four roots.

This code generates a plot in a square region of the complex plane with corners $z = \pm 1.25 \pm 1.25i$. Plots over larger regions can be produced by fiddling with the parameters. [In fact, each of these plots will probably be soon replaced by a plot over $\pm 2 \pm 2i$—JFE 1/31/2014.]

Note the "lobes" that occur along the diagonal lines separating the main basins for each root. All four colors are present in each lobe, but the larger sub-regions always involve the "other" two roots, i.e., along the diagonal separating $\zeta = 1$ from $\zeta = i$ (red from green) the larger sub-regions are blue and yellow. Note also that the edges of the lobes—as well as the sub-regions within the lobes—themselves appear to be fractal in nature, and the sizes of the lobes seem to be decreasing as we progress up the diagonal.

What about other methods? Do we get different looking basins using things like Halley's method, the super-Halley method, or Chun–Neta? Well, it is fairly easy to modify our m-file to do this: All we have to do is replace the Newton iteration inside the `bnewt` m-file with an alternate root finding method. The results are given in Figs. 4–6.

For the Halley method, the lobes appear to be much smaller than for Newton's method, although the same fractal edges and color tendencies appear to hold. It is unclear if the size of the lobes decreases as we go up the diagonal, since I only computed over a region large enough to get one full large lobe.

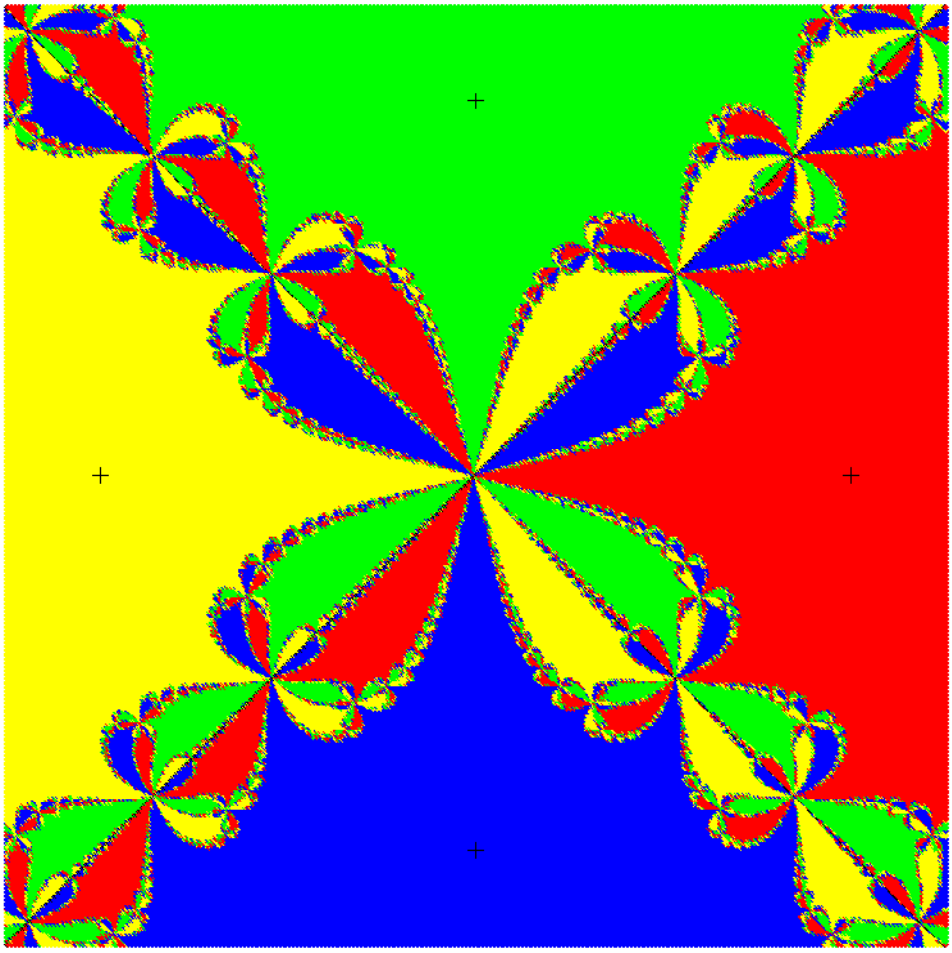

Figure 3: Basins of attraction for Newton's method applied to $f(z) = z^4 - 1$.

For the super-Halley method, we have a mix of the Newton-style lobes and the Halley-style lobes. It is an "eyeball-based" judgment, but I do think the overall size of the lobes for super-Halley is smaller than for Newton (but not as small as for Halley). It might be an interesting experiment to enlarge the region plotted to see what is the nature of the lobes appearing in the corners.

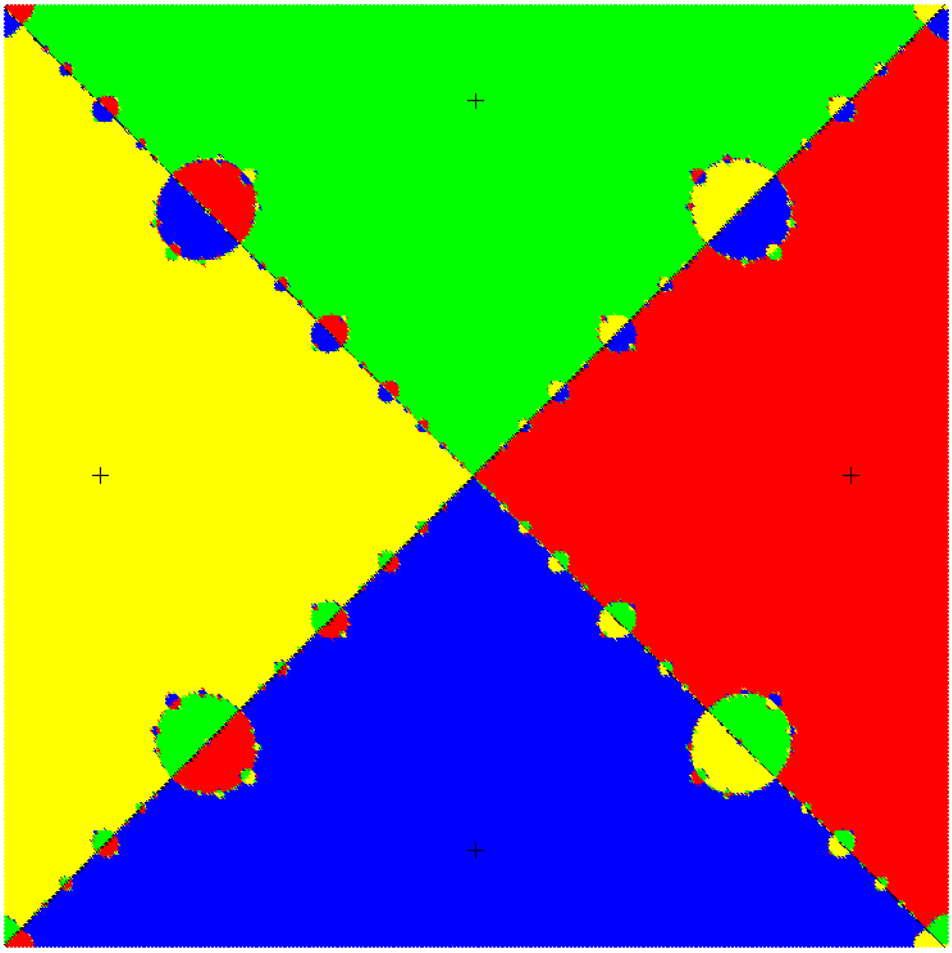

Figure 4: Basins of attraction for Halley's method applied to $f(z) = z^4 - 1$.

The lobe structure for Chun–Neta is more extensive than for Newton, and the lobes look larger. The region around the root where the iteration will *not* jump off and converge to another root appears smallest for Chun–Neta.

So, what do we learn from these figures? It is apparent to me that the Halley method has the "best" basin of attraction, i.e., the regions where you converge to the "wrong" root are the smallest. This suggests to me that the Halley method might be superior to the others. Despite its very high order of convergence the "lobes" in the basin of attraction for Chun–Neta are very large compared to the other methods. In fact, regardless of the efficiency index or the order of convergence, looking only at the basins of attraction for this single example, one is tempted to rank these four methods as follows:

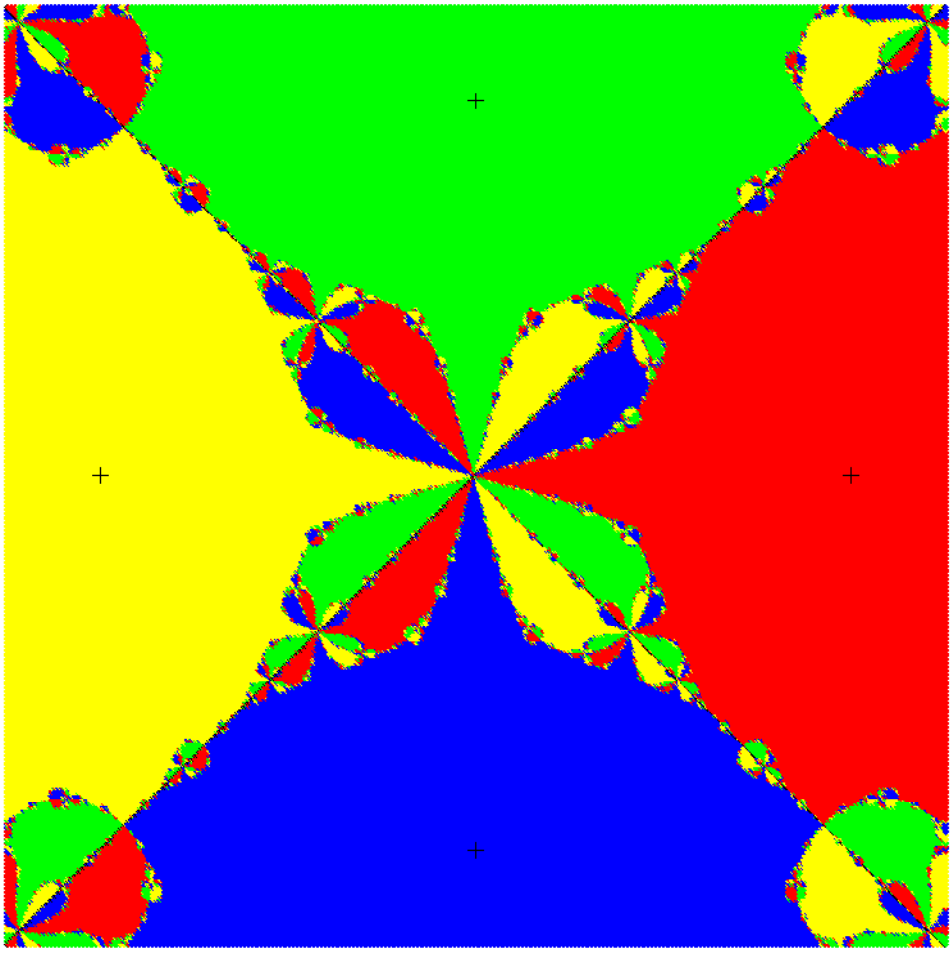

Figure 5: Basins of attraction for the super Halley method applied to $f(z) = z^4 - 1$.

1. Halley
2. Super Halley
3. Newton
4. Chun–Neta

This very crude assessment totally ignores the theoretical order of convergence, however.

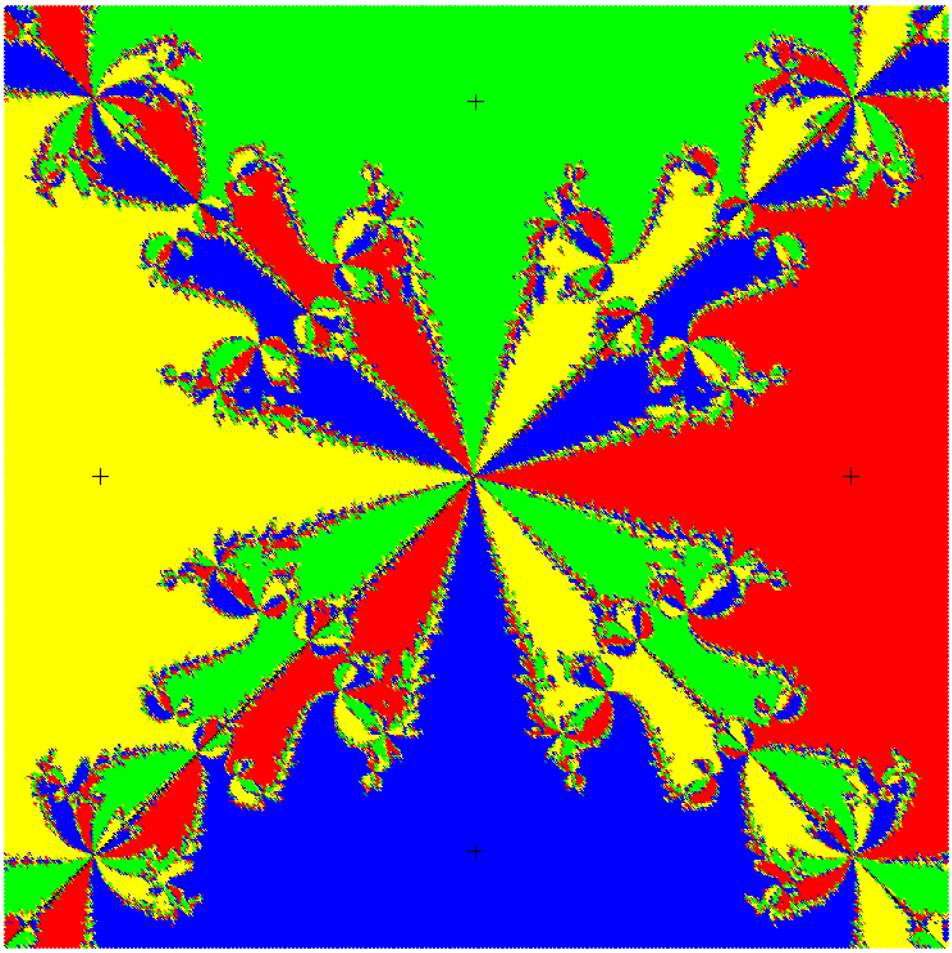

Figure 6: Basins of attraction for the Chun–Neta method applied to $f(z) = z^4 - 1$.

Update: In June of 2015, I read the brief paper [1], which discusses the basins of attraction for the polynomial

$$p(x) = x^3 - 2x + 2.$$

The exact roots are $\zeta = -1.7692924, 0.8846462 \pm 0.5897428i$. Fig. 7 shows a plot of this polynomial near the origin.

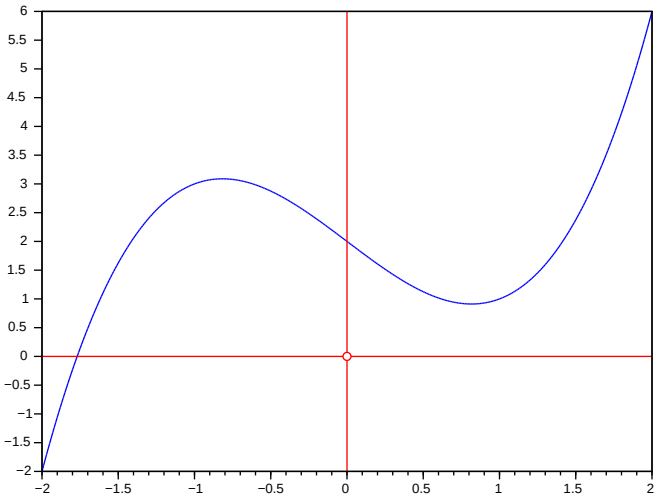

Figure 7: Plot of polynomial $p(x) = x^3 - 2x + 2$.

We can apply Newton's method to this polynomial and construct the corresponding basins of attraction. Since p is a cubic, it only has three roots, so we need only three colors, which we set up as follows:

$$\zeta = -1.7692924 = \text{dark blue}$$
$$\zeta = 0.8846462 + 0.5897428i = \text{yellow}$$
$$\zeta = 0.8846462 - 0.5897428i = \text{green}$$

Fig. 8 shows the plot over the region $-4 \le x \le 2$, and $-3 \le y \le 3$ in the complex plane; Fig 9 shows the plot over the smaller region $-2 \le x \le 1$, and $-1.5 \le y \le 1.5$.

These look much like the previous plots, *except* for the several regions of black. Where did these come from? Well, when I wrote the MATLAB codes, I initialized the color for each point to be black, and this would be changed to blue or yellow or green (or red, in the previous examples) only when the root was identified. So black means that no root was found—the black regions correspond to initial points *for which the iteration fails to converge!*

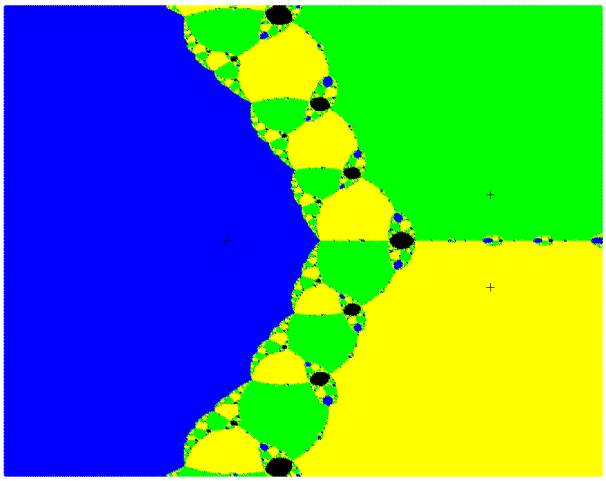

Figure 8: Basins of attraction for $p(x) = x^3 - 2x + 2$, over $-4 \leq x \leq 2$, and $-3 \leq y \leq 3$.

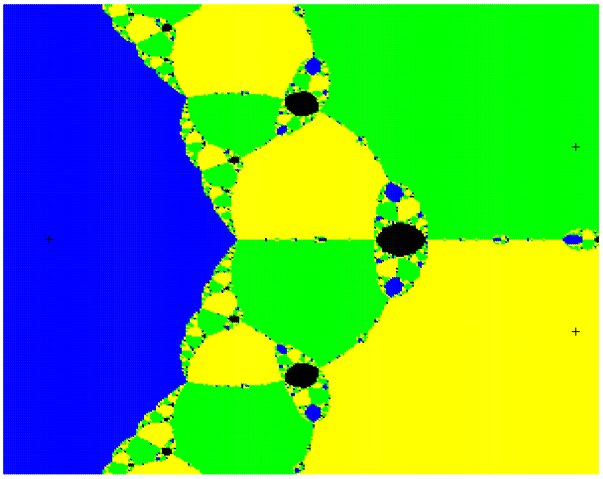

Figure 9: Basins of attraction for $p(x) = x^3 - 2x + 2$, over $-2 \leq x \leq 1$, and $-1.5 \leq y \leq 1.5$.

This in fact was the point of [1] which discusses the work in several other papers ([2], [3]), both of which study iterations like that for our p very theoretically. What happens if you start the iteration in one of the "black" regions? Well, let's give it a try.

Looking carefully at Fig. 9, it is apparent that $z = 0$ is inside the black region along the real axis. (This can be confirmed by plotting a special character in a different color

at that point.) Taking this as our z_0, we get the iterates:

$$z_0 = 0, \quad z_1 = 1, \quad z_2 = 0, \quad z_3 = 1 \ldots$$

Clearly these are not converging. If we start slightly away from the origin, then the iterates converge to the alternating pair (called a *limit cycle*) $(0, 1)$. By doing some experimenting, I was able to determine that using $z_0 = 0.14$ converges to $\zeta_1 = -1.7692924$ in about 85 iterations, but $z_0 = 0.139$ converges to the limit cycle in about 52 iterations. Using complex z_0 gives similar results: using $z_0 = 0.1 + 0.1i$ converges to $\zeta_3 = 0.8846462 - 0.5897428i$ in 12 iterations, but $z_0 = 0.1 + 0.05i$ converges to the limit cycle in 15–25 iterations.

Looking at Fig. 9, it is apparent that there are (very small) black regions continuing along the x-axis. If we "zoom in" we can see that these do indeed exist, and one of them is roughly centered at $x = 1$ (no surprise), while another is at $x = 1.5$. What happens if we take $z_0 = 1.5$ in our iteration? Well, this is an easy computation, and we almost immediately settle in to the $(0, 1)$ limit cycle. Taking $z_0 = 1.6$ results in converging to the limit cycle in about 66 iterations; taking $z_0 = 1.7$ leads to convergence to $\zeta_1 = -1.7692924$ in 20 iterations. Taking the complex initial value $z_0 = 1.55 + 0.01i$ converges to $\zeta = 0.8846462 + 0.5897428i$ in 13 iterations, but using $z_0 = 1.52 + 0.005i$ converges to the limit cycle. As we progress further out the real axis these "regions of non-convergence" will continue to exist, although they do get smaller and smaller.

References

[1] Abate, Marco, "À la Recherche des Racines Perdues (In Search of Lost Roots)," in *Imagine Math 3, Between Culture and Mathematics*, Michele Emmer (ed.), Springer (Milan) 2014.

[2] Aspenberg, M., Bilarev, T., and Schleicher, D., "On the speed of convergence of Newton's Method for complex polynomials, Preprint, arXiv: 1202.2475.

[3] Hubbard, J.H., Schleicher, D., and Sutherland, S., "How to find all roots of complex polynomials by Newton's method, *Invent. Math.*, vol. 146, pp. 1–33 (2001).