Numerical Analysis of Cayley's Problem

Andrea Mecchina

1 Newton's method

Let $f(x)$ be a real function of a real variable x. Let us suppose that exists at least one zero of such function, namely x^*, so that $f(x^*) = 0$. Let x_0 be an initial estimate x^* and define the difference from x^* as $h = x^* - x_0$. It follows that $x^* = x_0 + h$. Now we expand $f(x^*)$ in Taylor series around x_0 and we get

$$0 = f(x^*) = f(x_0 + h) = f(x_0) + hf'(x_0) + \frac{h^2}{2} f''(x_0) + \ldots,$$

where the first equality is due to the fact that x^* is a zero of the function $f(x)$. Assuming that the initial guess x_0 is "close enough" to x^*, we can neglect all higher orders and approximate

$$0 = f(x^*) \approx f(x_0) + hf'(x_0)$$

from which it follows that $h \approx -f(x_0)/f'(x_0)$. In this derivation we are also assuming that $f'(x_0)$ exists and it's non zero. We now get a new estimate of x^*, namely x_1, substituting the approximation for h in its definition, obtaining

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

This method can be iterated, so we obtain a relation for generating a new x_{n+1} from a previous value x_n, given by

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}. \tag{1}$$

It can be proved [1] that the sequence generated in this way, if the initial value is chosen "close enough" to the zero, will eventually converge exponentially fast to x^*.

2 An example: $f(x) = x^2 - a$

Let the function of interest be $f(x) = x^2 - a$, with $a \in \mathbb{R}$ and $a > 0$. If x^* is a zero of such a function, then $(x^*)^2 = a$ and $x^* = \sqrt{a}$. This algorithm then allows us to estimate the square root of any given positive real number a. The derivative of $f(x)$ is $f'(x) = 2x$, so the prescription given by (1) reduces to

$$x_{n+1} = x_n - \frac{x_n^2 - a}{2x_n} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right). \tag{2}$$

This is one of the oldest algorithms known for estimating the square root.

In this particular case it is easy to show the series generated by Newton's method converges. The assumption of a strictly greater than zero is because when we reach $x_n = 0$, the algorithm is not well-defined. We define the relative error on the $n-$th step

$$e_n := \frac{x_n - \sqrt{a}}{\sqrt{a}}$$

from which it follows that $x_n = (1 + e_n)\sqrt{a}$. Substituting this in (2) we get

$$x_{n+1} = \sqrt{a} + \frac{\sqrt{a}}{2}\frac{e_n^2}{e_n + 1}.$$

Moving $\sqrt{a}$ to the left and dividing both members by $\sqrt{a}$ we recognize $e_{n+1} = (x_{n+1} - \sqrt{a})/\sqrt{a}$, for which we get a recursive relation for the relative error given by

$$e_{n+1} = \frac{e_n^2}{2e_n + 2}.$$

Since $2e_n < 2 + 2e_n$, then $1/(2 + 2e_n) < 1/(2e_n)$ and from the (3) we obtain that the relative error at every step is at least the half of the relative error of the previous step, according to

$$e_{n+1} < \frac{e_n}{2}. \tag{3}$$

This result is a proof of the fact that we can reach, with algorithm (2), whatever level of accuracy required, if we take a number of steps large enough.

3 Basins of attraction

The regions of the domain of a function can be characterized by the solution their elements, chosen as starting point for (1), converge to. In the language of dynamical systems, the solution is said to be an attractor, while the regions whose points converge to a certain attractor are said to be the basin of attraction of that attractor.

As an example, we consider the previous function in the specific case $a = 2$, when it reduces to $f(x) = x^2 - 2$. This function has two zeros $\pm\sqrt{2}$ and simulations show that positive starting points lead to $+\sqrt{2}$, while negative starting points lead to $-\sqrt{2}$. There is one point $x_0 = 0$ that does not belong to any basin of attraction, for which the iteration (2) is not well-defined.

Newton's method can be applied even to complex numbers. The problem of determining basins of attraction in the complex case was first proposed by A. Cayley [2]. Accurate studies can now be pursued by the means of computer simulations and numerical analysis. Now we let $z \in \mathbb{C}$ be a complex number and we investigate the roots of the complex function $f(z) = z^3 - 1$.

4 n-th roots of unity

The n-th roots of unity are defined as the zeros of the function $f(z) = z^n - 1$. A complex number z can be written in polar form as

$$z = |z|e^{i\theta}.$$

with θ the polar angle. From Euler's identity we get $(e^{i\theta})^n = e^{in\theta} = \cos(n\theta) + i\sin(n\theta)$, known as the de Moivre's theorem. Clearly it must be $|z| = 1$, so de Moivre's theorem allow us to write the equation for the zeros of $f(z)$ as

$$\cos(n\theta) + i\sin(n\theta) = 1 = \cos(2\pi k) + i\sin(2\pi k)$$

where k is a positive integer. The previous expression forces the polar angle to be $\theta = 2\pi k/n$. For a given value of n, there are n different values of θ. The n distinct roots of unity then can be written as

$$z_k = \exp(i2\pi k/n) \tag{4}$$

where k takes integer values from 0 to $n-1$. The n distinct roots of unity are a set of n point located on the unit circle with phases equally spaced.

We are interested in the complex function $f(z) = z^3 - 1$, so $n = 3$ and we have the three distinct zeros

$$z_0 = 1, \quad z_1 = \cos\left(\frac{2\pi}{3}\right) + i\sin\left(\frac{2\pi}{3}\right), \quad z_2 = \cos\left(\frac{4\pi}{3}\right) + i\sin\left(\frac{4\pi}{3}\right). \tag{5}$$

5 Code implementation

Writing a complex number as $z = (x + iy)$, basins of attraction are studied for starting points with $x \in [-2:2]$ and $y \in [-2:2]$. Note that $z_0, z_1, z_2 \in [-2:2] \times [-2:2]$. Since $f(z) = z^3 - 1$, expression (1) reduces to

$$z_{n+1} = z_n - \frac{z^3 - 1}{3z^2}. \tag{6}$$

All simulation are performed with a program written in Fortran 90 whose listing is reported below.

```fortran
program cayley
  implicit none

  integer    :: i,j,samp,step
  complex*8 :: old,new,start
  real*8     :: thres,real_i,real_j

  open(unit=11,file='input.dat',status='old')
  read(11,*) thres
  read(11,*) samp

  open (unit=12,file='output.dat',status='replace')

  do i=1,samp
    real_i=-2.+(i-1)*4./samp
  do j=1,samp
    real_j=-2.+(j-1)*4./samp

  old    = cmplx(real_i,real_j)
  start  = old
  new     = old-(old**3-1.)/(3*old**2)
  step    = 0

  do while (abs(old-new) > thres)
    old    = new
    new    = old-(old**3-1.)/(3*old**2)
    step   = step+1
  end do

  write(12,*) real(start),imag(start),&
        real(old),imag(old),step

  end do
  end do

end program
```

The iteration stops when the distance between a step and the previous is less than an arbitrary value read from file. The $[-2:2]\times[-2:2]$ grid is sampled in discrete points and the number of samples in a row is read from the input file.

6 Simulation results

The threshold distance is chosen as 10^{-4} and the samples in a row are chosen as 600. All images are generated using GNUPLOT. We assign a color code to any solution, namely

$$\text{yellow} \longleftrightarrow z_0, \quad \text{red} \longleftrightarrow z_1, \quad \text{blue} \longleftrightarrow z_2.$$

so we can plot a map coloring every point in $[-2:2]\times[-2:2]$ according to the root it converges to. A minimal script to generate this image is listed below.

```
reset session
set terminal png
set output '1.png'
unset key
unset colorbox
unset tics
set lmargin 0
set rmargin 0
set tmargin 0
set bmargin 0
set palette negative defined (\
    0 '#D53E4F',\
    1 '#F46D43',\
    2 '#FDAE61',\
    3 '#FEE08B',\
    4 '#E6F598',\
    5 '#ABDDA4',\
    6 '#66C2A5',\
    7 '#3288BD' )
set object circle at 1,0 radius .02\
    fillcolor rgb 'black' fillstyle solid front
set object circle at cos(2*pi/3),sin(2*pi/3) radius .02\
    fillcolor rgb 'black' fillstyle solid front
```

```
set object circle at cos(4*pi/3),sin(4*pi/3) radius .02\
   fillcolor rgb 'black' fillstyle solid front
plot 'output.dat' using 1:2:(arg($3+$4*{0,1})) palette
```

Basins of attraction for the roots of $z^3 - 1$ obtained in this way are shown in figure 1, where we can clearly recognize a fractal structure and self-similarity properties. The black dots are the exact roots (5). We can confirm that Newton's method really is very sensible to the starting point.

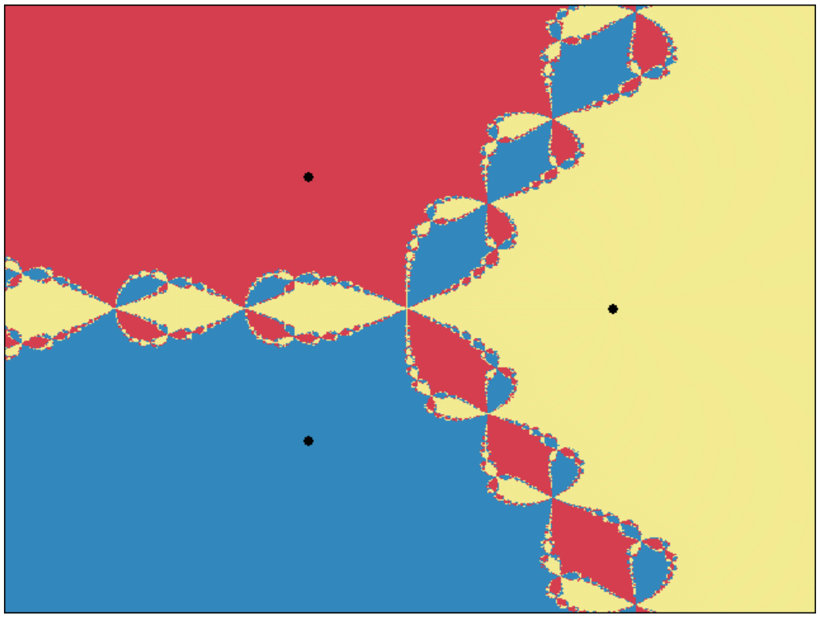

Figure 1: Basins of attraction for $f(z) = z^3 - 1$.

In the context of complex dynamics, a function has two complementary sets associated [3], using a sloppy terminology:

- the Fatou set, made of all points whose dynamic is regular;
- the Julia set: made of points whose dynamic is extremely sensible to the initial conditions.

With even sloppier terminology, points belonging to Julia set are said to exhibit chaotic behavior, but we remind that there isn't anything non-deterministic associated with them.

These regions can be investigated with a color map, relating the shade to the number of iterations needed to converge to any solution: the number of steps increases going from dark to bright colors. The results are shown in figure 2, which can be obtained from the previous script substituting the last line with the following one.

```
plot 'output.dat' using 1:2:5 palette
```

The Julia set should be the border between darker regions, the Fatou sets, but since we are sampling only a discrete number of points, the Julia set is approximated by a set of points for which the iterations needed to converge to a solution are considerably higher than other points.

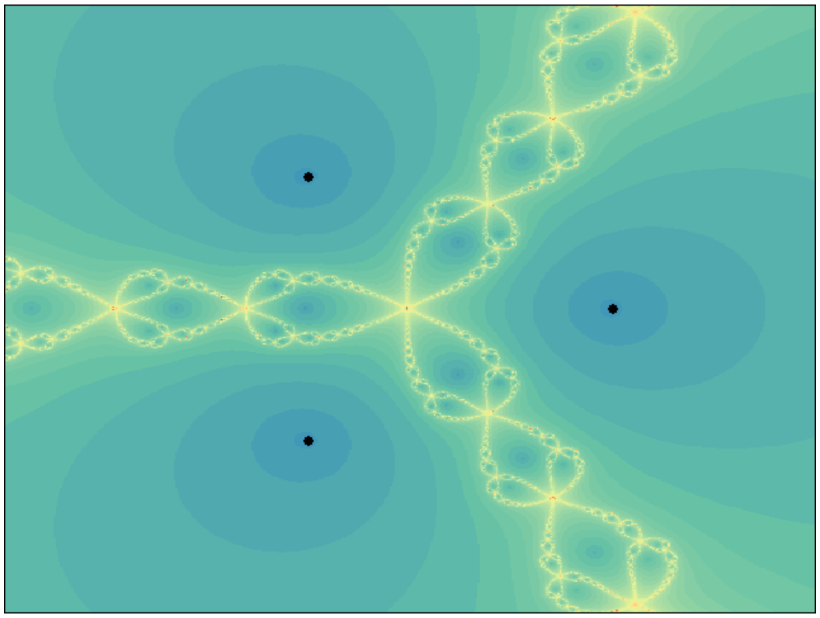

Figure 2: Convergence for $f(z) = z^3 - 1$.

References

[1] S. Sternberg. *Dynamical Systems.* Dover, 2010.

[2] A. Cayley. *The Newton-Fourier Imaginary Problem.* Am. J. Math., 1879.

[3] S. Sutherland. *An Introduction to Julia and Fatou Sets.* Springer Proceedings in Mathematics and Statistics, 2014.