Pensées 5: Complex Root Finding of Polynomials

0x10af

My Method of Root-Finding

Householder's Method of Degrees 1, 2, and 3

Householder's methods are iterative root-finding algorithms which set x_{n+1} to the zero of a taylor polynomial evaluated at x_n.

We let $f = f(x_n)$ etc.

Newton's method is the assignment of the solution of the linear approximation t to x_{n+1}.

$$t(x_{n+1}) = 0 = f(x_n) + f'(x_n)(x_{n+1} - x_n)$$
$$\Rightarrow x_{n+1} - x_n = -\frac{f(x_n)}{f'(x_n)}$$

Halley's method uses a quadratic approximation to the function instead.

$$t(x_{n+1}) = 0 = f(x_n) + f'(x_n)(x_{n+1} - x_n) + \frac{f''(x_n)}{2}(x_{n+1} - x_n)^2$$
$$-f(x_n) = (x_{n+1} - x_n)\left(f'(x_n) + \frac{f''(x_n)}{2}(x_{n+1} - x_n)\right)$$
$$x_{n+1} - x_n = -\frac{f(x_n)}{f'(x_n) + \frac{f''(x_n)}{2}(x_{n+1} - x_n)}$$
$$x_{n+1} - x_n = -\frac{2f(x_n)}{2f'(x_n) + (f''(x_n))\frac{f(x_n)}{f'(x_n)}}$$
$$x_{n+1} - x_n = -\frac{2f(x_n)f'(x_n)}{2f'(x_n)^2 + f''(x_n)f(x_n)}$$

The derivation follows similarly for higher degrees, substituting the previous degree for $(x_{n+1} - x_n)$.

Halley's method can be reformatted into this form, allowing more efficient symbolic expression and reuse of $f(x_n)/f'(x_n)$.

$$x_{n+1} = x_n - \frac{f}{f' - \frac{f}{f'}\frac{f''}{2}} = x_n - \frac{f}{f'}\left[1 - \frac{f}{f'}\frac{f''}{2f'}\right]^{-1}$$

The general formula for Householder's method is thus:

$$x_{n+1} = x_n + d\frac{\left(\frac{1}{f}\right)^{(d-1)}(x_n)}{\left(\frac{1}{f}\right)^{(d)}(x_n)}$$

In the polynomial case, it is necessarily possible to simplify the expression for any order into the quotient of two polynomials, with the numerator generally having a higher order than the denominator. Thus, there exists a list of coefficients c_n for which, for some range of integers $[a, b]$:

$$x_{n+1} = x_n + d\frac{\left(\frac{1}{f}\right)^{(d-1)}(x_n)}{\left(\frac{1}{f}\right)^{(d)}(x_n)} = \sum_{i=a}^{b} c_i x^i$$

This list of coefficients is deterministically and algorithmically computable, there is an $O(1)$ cost of computing it with respect to the number of iterations.
Thus there is a much slower growth in computation time with respect to the order of convergence of the algorithm, only affecting the $O(1)$ term intially and the coefficents and order of the polynomial which is evaluated at each point.

Example of Symbolic Simplification for Halley's Method

$$f(x) = x^3 - 1$$

$$f'(x) = 3x^2$$

$$f''(x) = 6x$$

$$\frac{f}{f'}\left[1 - \frac{f}{f'}\frac{f''}{2f'}\right]^{-1}$$

$$= \frac{x^3 - 1}{3x^2}\left[1 - \frac{x^3 - 1}{3x^2}\frac{6x}{2x^3 - 2}\right]^{-1} = \frac{x}{3} + \frac{1}{3x} + \frac{1}{3}$$

We can see here the computational utility of symbolic simplification, which, for polynomials, in this case, can trivially be computed algorithmically.

Gauss-Lucas Theorem

The Gauss-Lucas theorem states:
"If P is a (nonconstant) polynomial with complex coefficients, all zeros of P' belong to the convex hull of the set of zeros of P."
The theorem is true if $P'(z) = P(z) = 0$
If not, the roots being a_i, the leading coefficient α, and n being its degree:

$$p(z) = \alpha \prod_{i=1}^{n} (z - a_i)$$

$$p'(z) = 0 \wedge p(z) \neq 0 \Rightarrow \frac{\partial}{\partial z} \ln(p(z)) = \frac{p'(z)}{p(z)} = 0$$

$$\frac{1}{z} = \frac{\bar{z}}{|z|^2}$$

$$\frac{\partial}{\partial z} \ln(p(z)) = \frac{\partial}{\partial z} \prod_{i=1}^{n} (z - a_i) = \sum_{i=1}^{n} \frac{1}{z - a_i}$$

$$= \sum_{i=1}^{n} \frac{\bar{z} - \bar{a}_i}{|z - a_i|^2}$$

$$\bar{z} \sum_{i=1}^{n} |z - a_i|^{-2} = \sum_{i=1}^{n} \frac{\bar{a}_i}{|z - a_i|^2}$$

$$z \sum_{i=1}^{n} |z - a_i|^{-2} = \sum_{i=1}^{n} \frac{a_i}{|z - a_i|^2}$$

$$z = \sum_{i=1}^{n} \frac{a_i}{|z - a_i|^2} \sum_{j=1}^{n} |z - a_j|^{-2} = \sum_{i=1}^{n} \frac{|z - a_i|^{-2}}{\sum_{j=1}^{n} |z - a_j|^{-2}} a_i$$

Which is a convex sum of the roots of p.

Method of Sampling

Since the roots of the derivative of a function lie inside the convex hull of its roots, sampling radii around its roots and then using an iterative root-finding algorithm will likely converge to all or most of its roots. This leads to a recursive algorithm where there is an eventual derivative of $p(z)$ which can be trivially solved with the quadratic formula (or you could do one more derivative and use $b + ax = 0 \Rightarrow x = -\frac{b}{a}$).

$$p(z) = c_0 + \ldots + c_n x^n$$

$$\frac{\partial^a}{\partial z^a} \gamma z^k = \gamma \frac{k!}{(k-a)!} z^{k-a}$$

$$\frac{\partial^a}{\partial z^a} \sum_{i=0}^{n} \gamma_i z^i = \sum_{i=a}^{n} \gamma_i \frac{i!}{(i-a)!} z^{i-a}$$

$$\frac{\partial^{n-2}}{\partial z^{n-2}} p(z) = (n-2)! c_{n-2} + (n-1)! c_{n-1} x + \frac{n!}{2} c_n x^2$$

Then for a degree n polynomial with radii coefficient (user-defined) μ:

$$j \in [0, \mu n - 1] \subset \mathbb{Z}$$
$$s_j(z) = z + \exp\left(\frac{j}{r}\tau\right)$$

The points s_j form the intial guesses for the iterative root-finding on the next order of the polynomial.
In addition, it is extremely important to filter the roots found to ones for which $|a - b| > \varepsilon$ for performance.

Empirically, $\mu = 2$ almost always converges to all the roots.

Newton and Halley Fractals
The choice of this radial sampling method is motivated by the commonalities in structure of different orders of Newton fractals. An order n polynomial generally has n basins of attraction to different roots at arguments of the nth roots of unity.
This structure is preserved with higher-order methods like Halley's method, but with a more simple fractal structure, these basins being larger, and faster convergence, thus the choice of this sampling method and Halley's method is logical.

Aberth-Ehrlich Method

$$x_{n+1} = x_n - \frac{p(x_n)}{p'(x_n)}\left[1 - \frac{p(x_n)}{p'(x_n)} \sum_{j \neq k} \left(z_k - z_j\right)^{-1}\right]^{-1}$$

"Conceptually, this method uses an electrostatic analogy, modeling the approximated zeros as movable negative point charges, which converge toward the true zeros, represented by fixed positive point charges. A direct application of Newton's method to each approximated zero will often cause multiple starting points to incorrectly converge to the same root. The Aberth method avoids this by also modeling the repulsive effect the movable charges have on each other. In this way, when a movable charge has converged on a zero, their charges will cancel out, so that other movable charges are no longer attracted to that location, encouraging them to converge to other "unoccupied" zeros."

Eigenvalues of Companion Matrix of Monic Polynomial
The companion matrix for a monic polynomial of degree n is constructed (degree 2 example):

$$c_0 + c_1 x + c_2 x^1 + x^2$$
$$C = \begin{bmatrix} 0 & 0 & -c_0 \\ 1 & 0 & -c_1 \\ 0 & 1 & -c_2 \end{bmatrix}$$

Whose characteristic polynomial is:

$$\det(\lambda I - C) = 0$$
$$\det(\lambda I - C) = \det \begin{bmatrix} \lambda & 0 & c_0 \\ -1 & \lambda & c_1 \\ 0 & -1 & c_2 + \lambda \end{bmatrix} = \lambda(\lambda(c_2 + \lambda) + c_1) + c_0 = \lambda^3 + c_2\lambda^2 + c_1\lambda + c_0 = 0$$

Thus, since finding the eigenvalues is equivalent to solving the characteristic polynomial, the eigenvalues of the companion matrix are the zeros of the monic polynomial. This method is the easiest, fast, and accurate, although it may have problems if the leading coefficent is very large or small, like, for example, in Taylor polynomials.
The Jenkins-Traub algorithm solves an equivalent problem in a different method.

The Multinomial Theorem and Generalized Liebnitz Rule

$$\left(\sum_{i=1}^{m} a_i\right)^n = \sum_{\sum_{i=1}^{m} k_i = n} \binom{n}{k_1, \ldots, k_m} \prod_{t=1}^{m} a_t^{k_t}$$

$$\frac{\partial^n}{\partial x^n} \prod_{i=1}^{m} a_i = \sum_{\sum_{i=1}^{m} k_i = n} \binom{n}{k_1, \ldots, k_m} \prod_{t=1}^{m} \frac{\partial^{k_t}}{\partial x^{k_t}} a_t$$

$$\frac{\partial^b}{\partial x^b} (a_0 + \ldots + a_m)^n = \sum_{\sum_{i=1}^{m} k_i = n} \binom{n}{k_1, \ldots, k_m} \frac{\partial^b}{\partial x^b} \prod_{t=1}^{m} a_t^{k_t}$$

$$= \sum_{\sum_{i=1}^{m} k_i = n} \binom{n}{k_1, \ldots, k_m} \sum_{\sum_{i=1}^{m} l_i = n} \binom{b}{l_1, \ldots, l_m} \prod_{t=1}^{m} \frac{\partial^{l_t}}{\partial x^{l_t}} a_t$$

$$\frac{\partial^n}{\partial x^n} f(x)^{-1} = \frac{1+n}{f(x)} \sum_{k=0}^{n} \frac{(-1)^k}{1+k} \binom{n}{k} f(x)^{-k} \frac{\partial^n}{\partial x^n} f(x)^k$$

This could be used to find an explicit expression to substitute into Householder's methods for higher orders but it would be extremely slow and complex.
Assuming we already know the coefficients it's somewhat simplified, this would only really be useful for generating fractals though.

$$p(z) = \alpha \prod_{i=1}^{m} (z - a_i)$$

$$p(z)^{-1} = \alpha^{-1} \prod_{i=1}^{m} (z - a_i)^{-1}$$

$$\frac{\partial^a}{\partial z^a} p(z)^{-1} = \alpha^{-1} \frac{\partial^a}{\partial z^a} \prod_{i=1}^{m} (z - a_i)^{-1}$$

$$= \alpha^{-1} \sum_{\sum_{i=1}^{m} k_i = a} \binom{a}{k_1, \ldots, k_m} \prod_{t=1}^{m} \frac{\partial^{k_t}}{\partial z^{k_t}} (z - a_i)^{-1}$$

$$= \alpha^{-1} \sum_{\sum_{i=1}^{m} k_i = a} \binom{a}{k_1, \ldots, k_m} \prod_{t=1}^{m} (-1)^{k_t} k_t! (z - a_i)^{-k_t - 1}$$