Complex Polynomial Factoring

Henry G. Baker

July 24, 1986

In this paper, we investigate efficient methods of factoring polynomials with real and complex coefficients into linear and quadratic factors over the real numbers. The *Jenkins-Traub* method has n^2 behavior for a polynomial of degree n, and one wonders if an nlogn method exists for this problem.

We show that the GCD of the polynomial p(x) and its first derivative p'(x) is the key step in all phases of the factoring algorithm. Since GCD(p(x),p'(x)) requires n^2logn operations, its calculation dominates the rest of the algorithm, and only a speedup in the GCD phase would appreciably speed up the factoring of real polynomials.

1. The GCD is used to eliminate multiple roots, which will cause the other steps of the algorithm to fail. If the GCD is not 1, then we also achieve a reduction in degree and size of the polynomial, and hence have made progress.

2. The GCD algorithm produces the continued fraction (CF) expansion of the polynomial divided by its derivative p(x)/p'(x), which can be used as a Sturm chain for root separation.

3. The CF expansion of p(x)/p'(x) is used for the efficient high-accuracy location of a root using Newton's iteration.

4. The contour integral $\int p'/p\, dx$ is used in the "argument principle" in complex analysis. The integral is $\log|p(x)| = \sum\log|x-r_i|$, where r_i is a root of p. Thus, the CF expansion of p/p' is related to the log|p|, which splits the polynomial into its factors.

Eliminate Zero Roots

The zero'th order step eliminates any roots of zero, since they will cause trouble in later steps. Zero roots are easily distinguished by polynomials with zero constant terms. Simply dividing the polynomial p(x) by the smallest power of x which produces a constant term will eliminate all zero roots.

Eliminating Multiple Roots

The first order of business is the elimination of multiple roots, since they will cause trouble in later stages.

The obvious method is to perform the GCD g(x) of p(x) and p'(x), and then to recursively treat g(x) and p(x)/g(x) separately. Since we know that q(x)=p(x)/g(x) has no repeated roots, we know that q(x) and q'(x) will never be simultaneously zero, so we will not cause any trouble in our root finding.

The problem of factoring a real polynomial over the reals seems to involve the process of root-finding, since every factor is either a linear factor—implying a real root—or a quadratic factor—implying two conjugate complex roots. Therefore, one obvious method of factoring real polynomials is in first finding all of the real roots, "deflating" as we go, and then find the conjugate complex roots in pairs.

The process of root finding seems to have two distinct phases: the *separation* of the roots, where we partition the X-axis into segments, each of which has only a single real root, or the real part of a pair of complex conjugate roots. Once the root separation is complete, then we can focus on the finding of each root or conjugate pair, independently from the others.

Copyright (c) 1986 by Henry G. Baker

Finding Real Roots

The next order of business is the discovery and elimination of linear factors (i.e., real roots). The classical answer to this problem is that of Sturm (Knuth, Vol. II, p. 416). Sturm looks at the various polynomials occuring in the sequence produced by the Euclidean GCD calculation of p(x) and p'(x) (we'll also use this GCD calculation later for the continued fraction evaluation of p(x)/p'(x)).

We can determine the Sturm sequence for a polynomial during the evaluation of its continued fraction expansion by counting the number of negative quotients (assuming evaluation from inside out). Thus, the CF expansion of the polynomial is a key to the determination of the Sturm sequence.

The sequence of polynomials generated by the GCD algorithm (called the *Sturm chain*) has some interesting properties. Looked at in reverse order, we start with a constant, then a linear polynomial, then a quadratic polynomial, etc. The constant polynomial is constant throughout the entire interval $-\infty$ to ∞; the linear polynomial undergoes one sign change in the interval $-\infty$ to ∞; the quadratic polynomial undergoes either zero or two sign changes in the interval $-\infty$ to ∞; etc.

What are the properties of the Sturm chain? The major properties are that two neighboring polynomials cannot vanish at the same point, and that the neighbors of a vanishing polynomial must have opposite signs. What are the consequences of these properties?

If $V(-\infty)$ is the number of sign changes at $-\infty$, then as we go from $-\infty$ to $+\infty$, we can only *lose* sign changes; i.e., $V(\infty) \le V(-\infty)$. In other words the number of sign changes goes down monotonically as we traverse from $-\infty$ to $+\infty$. Consider the following graph with the X-axis being the real axis and the Y-axis being the various members of the Sturm chain arranged with p(x) at y=0, the p'(x) at y=1, etc. Then plot "+" everywhere the function is positive, "-" everywhere the function is negative, and "0" everywhere the function is zero. Wherever a "+" appears at (x,n) and a "-" appears at (x,n+1), place a "0" at (x,n+.5). Wherever a "-" appears at (x,n) and a "+" appears at (x,n+1), place a "0" at (x,n+.5). Wherever a "+" appears at (x,n) and a "0" appears at (x,n+1), place "0"'s at all points (x,n+t), for $.5 \le t \le 1$. Wherever a "-" appears at (x,n) and a "0" appears at (x,n+1), place "0"'s at all points (x,n+t), for $.5 \le t \le 1$. When we are done, we should have a graph with a number of "trails" on it, and *none of the trails cross one another.* A trail starts out at $(-\infty,j+.5)$, and then drops to (x,j-1+.5), then to (x',j-2+.5), and perhaps reaching the X-axis, where it exits as a root of p(x).

Consider the rational function p(x)/p'(x). The numerator is a polynomial of degree n and the denominator is a polynomial of degree n-1. Therefore, we can pull out one term such that $p(x)/p'(x)=(x/n+B/n^2)+r(x)/p'(x)$, where B is the coefficient of x^{n-1} in p(x) and both r(x) and p'(x) have degree n-1. Now r(x)/p'(x) can be expanded either as a Taylor series—in positive powers of x—or as a Laurent series—in powers of 1/x. The series expansions of the successive continuants of the continued fraction expansion of r(x)/p'(x) produce closer and closer approximations to the series expansion of r(x)/p'(x).

Polishing Real Roots

Suppose that we have an estimate x for a root of p(x). Then the estimate can be improved using Newton's method:

$$x' = x - p(x)/p'(x)$$

Newton's method gives quadratic convergence for single roots (so long as we are within a radius of convergence of that root and no other in the complex plane), which means that each iteration doubles the precision of the estimate. Until we are within that radius of convergence, we are better off using a bracketing method—perhaps with Sturm sequences—because the rate of convergence outside of the radius is as slow as if the root we were approaching were a multiple root.

Copyright (c) 1986 by Henry G. Baker

(This slow rate of convergence is another reason for polynomial deflation, which improves the convergence by expanding the radius of convergence when interfering roots are removed. The reason why the rate of convergence is slow when we are outside the radius of convergence is that the polynomial acts like a polynomial with higher-order powers. When we are sufficiently close to a single root, the influence of the other roots effectively disappears, and we are solving a linear equation! This effect is like the inverse-square law of physics, where the influence of a near body almost totally obscures the effect of further-away bodies.)

This effect can be made more precise. Consider the *partial fraction* expansion of p'(x)/p(x):

Since $p(x) = \prod_i (x-r_i)$,

$$p'(x) = \sum_i [\prod_j (x-r_j)]/(x-r_i) = \sum_i p(x)/(x-r_i) = p(x) \sum_i 1/(x-r_i),$$

hence $p'(x)/p(x) = \sum_i 1/(x-r_i)$.

Clearly, if we are close enough to one root r_c, then $(x-r_c)$ will be very small, hence $1/(x-r_c)$ will be very large, and $1/(x-r_c)$ will *dominate* the sum, and the other terms will be negligible. Since $p(x)/p'(x) = 1/(p'(x)/p(x)) \approx 1/(1/(x-r_c)) = (x-r_c)$, Newton's iteration gives us $x'=x-p(x)/p'(x) \approx x-(x-r_c)=r_c$, and we know the first order approximation is in hand.

A second-order method is obtained by noticing that:

$$\sum_i 1/(x-r_i)^2 = [p'(x)/p(x)]^2 - p''(x)/p(x)$$

This sum is dominated by a single term even when x is somewhat further away from a root than using Newton's method, hence we get faster convergence. Assuming that the sum is dominated, we get:

$$\sum_i 1/(x-x_i)^2 \approx 1/(x-r_c)^2 = [p'(x)/p(x)]^2 - p''(x)/p(x), \text{ so}$$

$$1/(x-r_c) = \pm \sqrt{\{[p'(x)/p(x)]^2 - p''(x)/p(x)\}}, \text{ hence}$$

$$x' = x - (x-r_c) \approx x - p(x)/[\pm\sqrt{\{p'(x)^2 - p(x)p''(x)\}}]$$

Now let us examine the Taylor expansion of p(x) about the current estimate of root location, x_e.

$$p(x-x_e) = p(x_e) + p'(x_e)\bullet(x-x_e) + p''(x_e)/2\bullet(x-x_e)^2 + ...$$

If we assume that x is sufficiently close to a root that higher powers of x become insignificant, we can attempt to solve for $p(x-x_e)=0$. The first order approximation we get by keeping only the first two terms of the series:

$$p(x-x_e) \approx p(x_e) + p'(x_e)\bullet(x-x_e)$$

Assuming that $p(x-x_e)=0$ and solving for x, we get:

$$p(x-x_e) = 0 = p(x_e) + p'(x_e)\bullet(x-x_e)$$

$$x = x_e - p(x_e)/p'(x_e)$$

Thus, we have Newton's formula. If we use a second-order approximation to Taylor's series for $p(x-x_e)$, we get:

$$p(x-x_e) \approx p(x_e) + p'(x_e)\bullet(x-x_e) + p''(x_e)/2\bullet(x-x_e)^2 = 0$$

$$x = x_e - p'(x_e)/p''(x_e)\bullet\{1\pm\sqrt{(1-2p''(x_e)p(x_e)/p'(x_e)^2)}\}$$

Laguerre's method makes use of this domination of the sum by the contribution from a single root to make a more robust root-finder than Newton's iteration. Laguerre doesn't try to ignore all other roots, but it does *lump* them together into a *lumped-parameter* approximation to the original polynomial. Thus, Laguerre's method assumes that the n roots fall into two categories: a single root close to the current approximation to a root, and a multiple root of multiplicity n-1 at some distance from the single root. Now this approximation is not very good outside of a small region

Copyright (c) 1986 by Henry G. Baker

surrounding the single root, but it is good enough to get us closer to our single root. Thus, if Newton's method is a first-order approximation method, in the sense that it ignores all roots other than the closest one, then Laguerre's method is a second-order method, in the sense that it acknowledges the existence of the other roots, but lumps them into a single root of multiple order. Laguerre's method has the advantage of providing information about how good its approximation is, so that it can tell us when we are close enough to start using Newton's method.

Once we have located a root, we can use that knowledge to unmask other nearby roots by one of two methods: deflation or "zero suppression". Deflation is the reduction of the degree of the polynomial p(x) by dividing out the root r to get a polynomial q(x)=p(x)/(x-r). Zero suppression is the process of subtracting out the contribution of known roots when computing p'(x)/p(x) as above. Thus, if r_i are *all* the roots of p(x), and s_j are the known roots of p(x), and $q(x)=p(x)/\prod_j(x-s_j)$, then

$$q'(x)/q(x) = [\sum_i 1/(x-r_i)] - [\sum_j 1/(x-s_j)] = p'(x)/p(x) - [\sum_j 1/(x-s_j)]$$

Although deflation and zero suppression are mathematically equivalent, deflation is more efficient, but zero suppression may preserve more accuracy.

Laguerre's method can also be generalized to a "mirrored" method for finding complex conjugate roots without complex arithmetic.

One wonders if it is possible to use a variant of Laguerre's method to produce a *much* more efficient factoring method for polynomials of large degree (n>1000, say). The idea would go something like this: we compute a few (cheap) statistics on the polynomial as a whole to estimate some low order statistics on the general location and orientation of the *cloud* of roots of the polynomial on the complex plane. Then we split the cloud approximately in half using the following method. We *clear* a vertical strip along the y-axis so that there are no roots within a certain distance of the y-axis. This can be done by a process of root finding using zero suppression in the strip. Once the strip has been cleared, then our job is easier, because the exact location of a root in the right half of the complex plane is determined more by other roots near to it than those further away. The roots further away can be characterized by a *lumped-parameter* model, and we can still get acceptable accuracy. In particular, we would like to model the effect of the roots in the left-half of the complex plane on those in the right-half plane with one lumped-parameter model, and the effect of the roots in the right-half plane on those in the left-half plane with another lumped parameter model. Since our assumption is that only the gross characteristics of these lumped-parameter models can affect the relative position of the roots in the other half of the plane (since we have already cleared a wide corridor of potentially interfering roots), we can now independently focus (perhaps using multiple processors) on the roots in each half-plane. Of course, we will recursively subdivide the plane again using the same method until the total number of roots to be located is small enough to tackle with simple brute force.

The major fly in the ointment is how to put the information back together again. Suppose that we have "found" all of the roots in the right-half of the plane using a lumped-parameter model of the left-half plane. We must now convert these roots into true roots of the original polynomial. But we now have a much better approximation to the lumped parameter model in the form of an actual approximation to each root in the other half-plane. We should be able to use this information to "fine-tune" the final location of the roots in both halves. We do not know how to put all of this together into an efficient, guaranteed-to-work algorithm.

We now see that real polynomial factoring is more like solving an N-body problem in the plane, where the "forces" on each of the "bodies" (roots) falls off as 1/d, where d is the distance, rather than as $1/d^2$, as in the 3-dimensional gravitational case. The "force field" in this case is the magnitude and direction of p(x)/p'(x), which acts on a "particle" and draws it into a nearby root. The problem of root finding is then the problem of finding the location of the bodies given various probes into this force field.

Copyright (c) 1986 by Henry G. Baker

Suppose we now consider the problem of finding one root quickly—*any* root—of a polynomial p(x). Probably the easiest root to find would be the one closest to the origin. If p(0)=0, then we are done, because 0 is a root of p(x). If p(0)≠0, then we can transform p(x) into a polynomial q(y)=p(kx)/p(0), where $k=|p(0)|^{1/n}$. Thus, q(0)=±1. Since $q(0)=\prod_i r_i=\pm 1$, we know that either $|r_i|=1$ for every root r_i, or there exists at least one root r_c such that $|r_c|<1$. We would like to find such a root quickly.

Suppose that |q(0)|=1 and $0\neq|q'(0)|<1$. Then Newton's method gives y'=-q(0)/q'(0) as the first approximation to a root close to zero. But |-q(0)/q'(0)|=1/|q'(0)|>1, so y' is not close to zero. We then consider the "inverse squared" second order method. In this method, |y'|<1 implies $|y'|^2=|y'^2|<1$, so $|y'^2|=|q(0)^2/(q'(0)^2-q(0)q''(0))|=1/|q'(0)^2-q(0)q''(0)|<1$, which implies that $|q'(0)^2-q(0)q''(0)|>1$, and since $|q'(0)^2|<1$, we must have a strong contribution from |q"(0)| (since |q(0)|=1) in order for y' to be small. Thus, we should be able to look at a larger and larger initial segments of the Taylor series of q(y) around y=0 until we find one which has a root |y|<1. However, the larger the initial portion of the Taylor series we need, the more roots we will find at approximately the same distance from the origin. Hence, we will be solving an equation that looks more and more like $y^k\pm 1=0$, which has only roots of unity as its roots.

Thus, in some sense, solving $y^k\pm 1$ is the hardest problem, because it is hard to get even one root (using an approximation method) without getting them all! Consider the action of Newton's method on y^n-1 where |y|<1.

$$y' = y - q(y)/q'(y) = (n-1)y/n + y^{1-n}/n$$

If y is small, say y=1/4, then $y' = (n-1)/4n + (4^{n-1})/n \approx (4^{n-1})/n$, for large n. So Newton's method immediately kicks us out of the middle of the circle of roots to a very large number, so that we can approach the roots from outside the circle. (Of course, inverting the equation so that we exchange 0 and ∞ doesn't affect the equation.)

We now try the "inverse square" method.

$$y' = y - q(y)/\pm\sqrt{(q'(y)^2-q(y)q''(y))} = y - (y^n-1)/\pm\sqrt{((ny^{n-1})^2-(y^n-1)n(n-1)y^{n-2})}$$

$$\approx 4^{n/2}/n, \text{ for } y=1/4$$

So the inverse square method doesn't work very much better. Laguerre's method also gives $y'\approx 4^{n/2}/4$, so it doesn't work, either.

We should be able to break up the monic polynomial p(x) into two pieces using the following trick. We first normalize the polynomial so that p(0)=±1. We then search for the coefficient of largest absolute value. If its value is ±1, then we have problems. However, if its absolute value is greater than 1, then we can split the polynomial there. Why? Because we are looking for a root whose absolute value is <1, and the polynomial (starting from the constant term) gives a convergent series on the root. But since the polynomial evaluated out to this largest coefficient is zero, then the higher order terms cannot possibly affect it very much, because their coefficients are smaller. (Is this really true? What if the signs cancel on the lower order terms, but do not cancel on the higher order terms?) In any case, we need to split the polynomial into two pieces and solve each piece separately.

Thus, the job of *root separation* is clearly defined: partition the complex plane into n+1 disjoint regions, where n of the regions are circles which are centered on a single root, and the final region is the rest of the complex plane in which convergence of Newton's method is slow.

We would like to perform this iteration with as much speed as possible, since we will be doing it for every (real) root. We are also allowed to *precondition* p(x) and p'(x), since we will be evaluating them a lot.

How fast can we evaluate p(x)/p'(x) for x real? Knuth Theorem E (Vol. II, p. 474) tells us that p(x) can be evaluated in about n/2 real multiplications and n real additions (similarly for p'(x)).

Copyright (c) 1986 by Henry G. Baker

Therefore, p(x)/p'(x) can be evaluated in about n real multiplication/division's and 2n real additions. However, this method requires a lot of preconditioning which may not be necessary.

Using a trick of Shaw and Traub (Knuth, Vol. II, p. 470), p(x) and p'(x) can be evaluated together in about n+sqrt(2n) multiplications and 2n additions, but the method is still too complicated. However, Horner's continued fraction expansion (Knuth, Vol. II, p. 479) of p(x)/p'(x) can be evaluated in about n divisions and 2n additions, and it is straight-forward.

Therefore, with approximately n divisions and 2n additions per iteration, we can get double the accuracy on our estimate. It is worth wondering whether an iteration having a higher order of convergence would be worth the additional effort. Newton's method for cubic convergence requires about 3n divisions and 6n additions (i.e., about 3 times as much work), and produces only three times the number of significant digits. Since 3 iterations of Newton's quadratic method takes about the same effort as one iteration of Newton's cubic method, yet produces 8/3=2.7 times as much accuracy, it is easy to see why Newton's quadratic method is preferred.

Polishing Quadratic Factors

In this section, we show how to "polish" the irreducible quadratic factors; i.e., given an approximation to a quadratic factor, how to improve it to a better one. Thus, through a process of iteration, we can eventually produce the coefficients of the quadratic factor to within an acceptable degree of error, and consider the quadratic factor "found".

The obvious method for polishing an irreducible quadratic factor is to use Newton's method in complex arithmetic. But this means evaluating p(x)/p'(x) fast for *complex* x. Knuth Theorem E still allows us to evaluate p(x) using n/2 complex multiplications and n complex additions (similarly for p'(x)); hence, we can evaluate p(x)/p'(x) (x complex) using n complex multiplications and 2n complex additions. If a complex multiplication is 4 real multiplications and 2 real additions, and a complex addition is 2 real additions, then the total for p(x)/p'(x) (x complex) is 4n real multiplications and 6n real additions. If a complex multiplication is 3 real multiplications and 5 real additions, then the total for p(x)/p'(x) (x complex) is 3n real multiplications and 9n real additions. These methods are quite complicated, but provide a performance goal for more straight-forward methods.

Another approach is inspired by Knuth (Vol. II, p. 468) which shows how to evaluate p(x) (x complex) using about 2n real multiplications and 2n real additions (p'(x) similarly); thus, p(x)/p'(x) can be evaluated in 4n real multiplications and 4n real additions—already a savings in real additions over the more complicated method of Theorem E.

Let p(x) be our polynomial of degree n, and p'(x) be the formal derivative of p. Let $f(x)=x^2+Bx+C$ be our initial approximation to an irreducible quadratic factor of p. Finally, let a+b*i*, a-b*i* be the two roots of f(x).

$$B= -2a$$

$$C=a^2+b^2$$

We would like to use Newton's formula to produce a better approximation to one of the complex conjugate roots, but we would like to avoid performing complex arithmetic. There are several reasons for this. The first is that the blind application of complex arithmetic requires more operations than a more careful analysis of the problem to avoid complex arithmetic. The second reason is that with more arithmetic operations, the chances of significant round-off error are increased.

What we would like to do is to *simulate* the Newton computation on complex root estimates through another computation using only real numbers. Our approach is based on the simultaneous Newton iteration on both a+b*i* and its complex conjugate a-b*i*. We know that since all of the coefficients of p(x) are real, all complex roots must occur in conjugate pairs. Therefore, we can investigate the Newton iteration on the conjugate roots:

Copyright (c) 1986 by Henry G. Baker

$$a'+b'i = a+bi - p(a+bi)/p'(a+bi)$$
$$a'-b'i = a-bi - p(a-bi)/p'(a-bi)$$

Since the conjugate iteration is a *mirror image* of the first computation, either calculation can "stand in" for the other. But having both iterations going on simultaneously allows us to pair up the intermediate results, allowing us to solve for the real part and the imaginary part of x':

$$a' = [(a'+b'i)+(a'-b'i)]/2 = [(a+bi - p(a+bi)/p'(a+bi))+(a-bi - p(a-bi)/p'(a-bi))]/2$$
$$= a - [p(a+bi)/p'(a+bi)+p(a-bi)/p'(a-bi)]/2$$
$$b' = [(a'+b'i)-(a'-b'i)]/2 = [(a+bi - p(a+bi)/p'(a+bi))-(a-bi - p(a-bi)/p'(a-bi))]/2$$
$$= b - [p(a+bi)/p'(a+bi)-p(a-bi)/p'(a-bi)]/2$$
$$B' = -2a'$$
$$C' = a'^2+b'^2$$

So far, we have not been able to eliminate complex arithmetic. However, we can compute both $p(a+bi)$ and $p(a-bi)$ by means of the following trick. If we "synthetically" divide $p(x)$ by $f(x)=x^2+Bx+C$, then we get the quotient $q(x)$ and remainder $r(x)=Dx+E$ as follows:

$$p(x) = q(x)\ (x^2+Bx+C) + Dx + E$$
$$p(a+bi) = q(a+bi)\ 0 + D(a+bi) + E = Da+E + Dbi$$
$$p(a-bi) = q(a-bi)\ 0 + D(a-bi) + E = Da+E - Dbi$$

Similarly, we can compute both $p'(a+bi)$ and $p'(a-bi)$ by means of a similar synthetic division:

$$p'(x) = q'(x)\ (x^2+Bx+C) + Fx + G$$
$$p'(a+bi) = F(a+bi) + G = Fa+G + Fbi$$
$$p'(a-bi) = F(a-bi) + G = Fa+G - Fbi$$

Thus, we can now efficiently compute (using only real arithmetic) the quantities needed to get a', b'.

$$a' = a - [p(a+bi)/p'(a+bi)+p(a-bi)/p'(a-bi)]/2$$
$$= a - [(Da+E+Dbi)/(Fa+G+Fbi) + (Da+E-Dbi)/(Fa+G-Fbi)]/2$$
$$= a - [(Da+E+Dbi)(Fa+G-Fbi)+(Da+E-Dbi)(Fa+G+Fbi)]/(2(Fa+G)^2+2(Fb)^2)$$
$$= a - [(Da+E)(Fa+G)+DbFb]/((Fa+G)^2+(Fb)^2)$$
$$b' = b - [p(a+bi)/p'(a+bi)-p(a-bi)/p'(a-bi)]/2$$
$$= b - [Db(Fa+G)-Fb(Da+E)]/((Fa+G)^2+(Fb)^2)$$

Thus, the operations needed for one Newton's iteration for both conjugate roots simultaneously are:

1. Compute $B=-2a$; $C=a^2+b^2$. (2 •'s; 2 +'s.)
2. Compute D,E by dividing $p(x)$ by x^2+Bx+C. (2(n-1) •'s; 2(n-1) +'s.)
3. Compute F,G by dividing $p'(x)$ by x^2+Bx+C. (2(n-2) •'s; 2(n-2) +'s.)
4. Compute Fa+G, Fb, Da+E, Db, (Da+E)(Fa+G), (Db)(Fb), (Db)(Fa+G), (Fb)(Da+E), $(Fa+G)^2$, $(Fb)^2$, a', b'. (12 •'s; 7 +'s.)

So the total number of real operations per iteration is 4n+8 multiplications and 4n+3 additions.

Since approximately 4n multiplications and 4n additions are in the range of four times as much work as the effort to polish a single real root, one wonders if this can be improved.

Copyright (c) 1986 by Henry G. Baker

Consider (as in the case of real roots) the continued fraction expansion of p(x)/p'(x). We need to evaluate this continued fraction, but in the algebraic field Q(r), where r is a root of the irreducible polynomial x^2+Bx+C. How many (real) divisions and additions are needed for this evaluation?

This depends upon the number of real operations required for division (inversion) in Q(r). Since Q(*i*) is a special case of Q(r), inversion in Q(r) is at least as hard as complex inversion. The number of real multiplication/division's required to invert a+b*i* is 4, so the minimum number to invert a+bx in a general quadratic field is at least 4. Hence, the number of real multiplication/division's to evaluate p(x)/p'(x) over the quadratic field Q(r), where r is a root of x^2+Bx+C is at least 4n (because there are n inversions in the continued fraction p(x)/p'(x)). Therefore, in polishing complex roots, the continued fraction method is no easier than the synthetic division method indicated above.

As an aside, we note that if the complex conjugate root method is used on a polynomial with no complex roots, or where a real root is closer than the complex one, then it will still converge to a real root r, but with the quadratic form $x^2+Bx+C=x^2-2rx+r^2=(x-r)^2$. Thus, this method could actually be used for either complex conjugate roots or real roots, but it is obviously less efficient for real roots than the standard method.

Continued Fractions

Continued fractions for rational functions have the following form (Knuth V.II, p.500):

$$\alpha_1+\beta_1/(x+\alpha_2+\beta_2/(x+...+\beta_n/(x+\alpha_{n+1})...))$$

This form is highly efficient for the evaluation of a rational function, since only n divisions and n additions are required for the evaluation of a rational function of degree n in both the numerator and the denominator. However, there are several problems. In the process of evaluting the continued fraction, one may have a spurious division by zero, which will be cancelled out in a later step. Also, in some domains, a full division might take significantly more work than a full multiplication (e.g. the complex domain).

Both of these problems can be solved by the following method. We first express our continued fraction in a slightly more conventional form:

$$\alpha_1+1/(\gamma_2x+\alpha_2+1/(\gamma_3x+...+1/(x+\alpha_{n+1})...))$$

This form (with the numerator of each fraction always unity) requires only inversions, not full divisions, and may therefore take less work to evaluate than the previous form. But more importantly, we can evaluate the form with n multiplications, n additions and 1 division! We do this by *deferring* divisions and keeping intermediate results in fractional form until the end, when we perform one division.

Thus, we can perform one step of the evaluation with a full multiplication and and a full addition:

$$\alpha+1/(\beta_n/\beta_d) = \alpha+(\beta_d/\beta_n) = (\alpha\beta_n+\beta_d)/\beta_n$$

How efficiently can we evaluate these continued fractions in the quadratic field x^2+Bx+C? Let us first look at continued fractions over the complex numbers.

We can express every complex continued fraction in either of the forms (we need both types):

$$\alpha_1+1/(\gamma_2x+\alpha_2+1/(\gamma_3x+...+1/(x+\alpha_{n+1})...))$$

$$i+\alpha+\beta/(\gamma'i+\alpha') \quad \text{or} \quad \gamma i+1+\beta/(\gamma'i+\alpha')$$

in which case we have either

$$(\beta-\alpha'+\alpha\alpha'+(\gamma'\alpha+\alpha')i)/(\gamma'i+\alpha') \quad \text{or} \quad (\beta+\alpha'-\gamma\gamma'+(\gamma\alpha'+\gamma')i)/(\gamma'i+\alpha')$$

Copyright (c) 1986 by Henry G. Baker

and either case takes 2 real multiplications and 3 real additions per step.

In the case of a real rational function evaluated at a complex point, the steps look like this:

$$1+\beta/(\gamma' i+\alpha') =$$

However, for evaluation in the quadratic field, this form is not the most efficient, since after the first division, the coefficient of x in the denominator will not be 1. Thus, we consider the slightly more general form for the continued fraction:

$$\alpha_1+\beta_1/(\gamma_2 x+\alpha_2+\beta_2/(\gamma_3 x+...+\beta_n/(x+\alpha_{n+1})...))$$

We note that this form is not really more general, since for every level, the factor γ_k can be included in β_{k-1} (after suitably modifying α_k and β_k). Thus, working backwards, we should always be able to arrange the coefficients of x in such a way that all divisions are by divisors of the form $(x+\alpha)$.

Polynomials — General Considerations

In this section, I would like to present some straight-forward, but extremely useful facts about polynomials.

Degrees of Freedom

A polynomial of degree n has n *degrees of freedom;* i.e., one can independently choose n parameters and get n essentially different polynomials. This fact is incorporated in several representations of a polynomial: 1) the *monic coefficient form*, in which the coefficient of the x^n term is one and the n coefficients of the lower degree terms are independent; 2) the *polynomial value form*, in which the function value of the polynomial at n distinct arguments is chosen; and 3) the *root location form*, or *factored form*, in which the polynomial is specified by the location of its n roots.

By placing various restrictions on polynomials, we can reduce the number of degrees of freedom.

Reducing General Polynomials to Monic Polynomials

The polynomial $p(x)=Ax^n+Bx^{n-1}+Cx^{n-2}+...$ is called *monic* if A=1. We can reduce any polynomial over a field to a monic polynomial by dividing by its leading coefficient; i.e. $q(x)=p(x)/A$. This division does not affect the roots of the polynomial, and monic polynomials will be assumed in the rest of this paper.

Another transformation of p(x) which results in a monic polynomial is the transformation $q(y)=A^{n-1}p(y/A)$; $x=y/A$. While this transformation *does* move the roots, it preserves the integral nature of a polynomial over the integers.

Clearly, the reduction of a polynomial to monic form does *not* change the number of degrees of freedom.

Conjugation of a Polynomial

A transformation on a polynomial with complex coefficients that is also an isomorphism of the underlying field is that of *conjugation*. While not very useful in itself for factoring polynomials, it can be useful in conjunction with other transformations. Since the conjugate of a real polynomial is the same polynomial, this transformation is not very useful on real polynomials at all, but is included here for completeness.

Copyright (c) 1986 by Henry G. Baker

Translation of a Polynomial

A polynomial can be *translated*; i.e., its coefficients altered so that all of its roots are shifted in the complex plane, by means of the transformation x=y-c; q(y)=p(y-c). The root r of q is essentially the same as the root r-c of p. Since translation can be completely specified by the image of one point, it has one degree of freedom.

Probably the most important translation of a polynomial is one which places the origin at the center of mass of all of its roots. Given a polynomial $p(x)=Ax^n+Bx^{n-1}+Cx^{n-2}+...$, we know that the coefficient B is the sum of all the roots. Therefore, if we translate the polynomial so that the origin is at the center of mass of the roots, then the coefficient of x^{n-1} will be zero (i.e., B=0). Since a polynomial of degree n has exactly n roots, the translation q(y)=p(y-B/(An)) will make the coefficient of x^{n-1} zero.

The translation of a polynomial to the center of mass of its roots reduces the number of degrees of freedom by one (the coefficient that now must be zero). We call this translation *centering the polynomial.*

Dilation of a Polynomial

A polynomial can also be *dilated*; i.e., its coefficients altered so that its roots are moved further from or closer to the origin, by means of the transformation x=c•y; q(y)=p(c•y). The root r of q is essentially the same as the root cr of p. Since a dilation can be completely specified by the image of one point (not the origin), it has one degree of freedom. The transformation y=x/A shown above was an example of a dilation.

An important dilation of a polynomial is one which makes the geometric mean of all the roots equal to (plus or minus) one. Given a monic polynomial p(x), the constant term p(0) is $(-1)^n \bullet \prod_i r_i$. Therefore, if $K=\sqrt[n]{|p(0)|}$, then the transformation q(y)=p(Kx)/|p(0)| will give a polynomial which is still monic (because $K^n/|p(0)|=1$) and whose constant term is $p(0)/|p(0)|=\pm 1$. The only exception to this transformation occurs when x=0 is a root of p(x), i.e., p(0)=0, where we would be dividing by 0, so we will apply this procedure only to polynomials with all non-zero roots.

In the case that the degree n of the polynomial q(y) is *odd*, we can force the constant term to be either +1 or -1 as we choose. If q(0)=1, the monic polynomial $q_-(y)=-q(-y)$ will have constant term $q_-(0)=-q(-0)=-1$, while if q(0)=-1, the monic polynomial $q_-(y)=-q(-y)$ will have constant term +1. This trick does not work with polynomials of even degree, because $q_-(-y)$ is no longer monic, but $-q_-(-y)$ is, so the low order term doesn't change sign. The transformation $q_-(y)=-q(-y)$ is essentially a *rotation* of the polynomial q(y) by 180°.

The dilation of a polynomial which makes the product of all its roots ±1 reduces the number of degrees of freedom by one (the constant term must now be 1). We call this transformation *normalizing the polynomial.*

Rotation of a Polynomial

A polynomial can be rotated in the complex plane; i.e., its coefficients altered so that its roots are rotated about the origin, by means of the transformation $x=y\bullet e^{\theta i}$; $q(y)=p(y\bullet e^{\theta i})$. The root r of q is essentially the same as the root $r\bullet e^{\theta i}$ of p. Notice that rotation of a real polynomial by any angle other than π gives a polynomial with complex coefficients. Since a rotation can be completely specified by the image of one point (not the origin), it has one degree of freedom.

Copyright (c) 1986 by Henry G. Baker

Involution of a Polynomial

A polynomial can be *involuted*; i.e., its coefficients altered so that its roots are reflected about the unit circle in the complex plane, by means of the transformation x=1/y; $q(y)=y^n p(1/y)$. The root r of q is the root 1/r of p.

The involution q(y) of a polynomial p(x) is the polynomial with its coefficients written *backwards*. Therefore, the involution of the involution of a polynomial is the original polynomial. Notice that if p(x) is monic and p(0)=1, then the involution of p(x) is also monic. We also call this transformation *inverting the polynomial.*

Möbius Transformations of Polynomials

The translations of a polynomial form a group, with the translations along the real axis as a subgroup. The rotations of a polynomial form another group, which with the translations forms the Euclidean group. The similarity group is the set of transformations x=Ay+B, A,B complex; this group has the Euclidean group as one subgroup and the group of dilations as another subgroup; the similarity group is generated by translations, rotations and dilations.

The group generated by the similarity group plus involution is called the *Möbius group* of transformations, and has the general form x=(Ay+B)/(Cy+D), A,B,C,D all complex, AD-BC≠0. (That this is called a *transform* is the result of the fact that the mapping is 1-1 and onto the complex plane so long as AD-BC≠0; thus, it has a well-defined inverse which is another Möbius transform.) The Möbius group is the group of all rational transformations on a polynomial which do not increase its degree.

An important property of Möbius transformations is that they transform circles in the complex plane into circles in the complex plane (we consider straight lines to be circles through infinity). This can easily be seen in the similarity subgroup; the only nonobvious generator is involution. Thus, we can map a straight line into a circle, or a circle into a straight line, using a Möbius transformation.

Another property of Möbius transformations is that the images of 3 points determine the transformation completely. Thus, the Möbius transformation is like a 3rd degree polynomial in that the value of the transformation can be independently specified at 3 distinct points, which will determine the transformation. In this sense, the Möbius transformation has 3 degrees of freedom. Since Möbius transformations take circles into circles, and since 3 points determine a circle, we can transform any circle onto any other circle. Furthermore, we can transform any three distinct points on a circle into any other three distinct points on the same circle.

A final property of a Möbius transformation is that it leaves at most two distinct points fixed (i.e., have at most two "fixed points"), unless it is the identity.

A Möbius transformation can map the real axis onto the unit circle where 0->1, 1->*i*, -1->-*i*, and "∞"->-1.

$$x \to (i-x)/(i+x)$$

(This transformation also maps the imaginary axis onto the real axis, and the unit circle onto the imaginary axis.)

Thus, Möbius transformations can do a reasonable amount of (smooth) violence to the complex plane. Since it can map 3 distinct points into 3 arbitrary distinct points, a Möbius transformation is capable of mapping the 3 roots of a cubic equation onto the 3 roots of unity, thus showing that all cubics are solvable, at least in theory. We will show later how to obtain the particular Möbius transformation which performs this mapping.

The low-order coefficients of the Möbius transform of a polynomial can be recovered using Taylor's Series:

Copyright (c) 1986 by Henry G. Baker

$$q(y) = q(0) + q'(0)\bullet y + q''(0)/2\bullet y^2 + ...$$

$$q(y) = p(x)\bullet(Cy+D)^n = p((Ay+B)/(Cy+D))\bullet(Cy+D)^n$$

$$q'(y) = p'(x)\bullet x'\bullet(Cy+D)^n + p(x)\bullet nC\bullet(Cy+D)^{n-1}$$

$$q'(y) = [(AD-BC)\bullet p'(x)+nC\bullet p(x)\bullet(Cy+D)]\bullet(Cy+D)^{n-2}$$

$$q(0) = p(B/D)\bullet D^n$$

$$q'(0) = [p'(B/D)\bullet(AD-BC) + p(B/D)\bullet nCD]\bullet D^{n-2}$$

$$q(y) = p(B/D)\bullet D^n + [p'(B/D)\bullet(AD-BC) + p(B/D)\bullet nCD]\bullet D^{n-2}\bullet y + q''(0)/2\bullet y^2 + ...$$

$$q(y) = p(s) + [p'(s)\bullet(r-s) + p(s)\bullet n]\bullet CD^{n-1}\bullet y + q''(0)/2\bullet y^2 + ... \text{ [r=A/C; s=B/D]}$$

The high-order coefficients of the Möbius transform of a polynomial can be recovered using Taylor's Series on the involution of the transformed polynomial

(if $q(y)=p((Ay+B)/(Cy+D))\bullet(Cy+D)^n$,

then $r(z)=z^n q(1/z)=z^n\bullet p((A/z+B)/(C/z+D))\bullet(C/z+D)^n=p((A+Bz)/(C+Dz))\bullet(C+Dz)^n$.)

$$r(z) = r(0) + r'(0)\bullet z + r''(0)/2\bullet z^2 + ...$$

$$r(0) = p(A/C)\bullet C^n = p(r)\bullet C^n$$

$$r'(0) = p'(A/C)\bullet(BC-AD)\bullet C^{n-2} + p(A/C)\bullet nD\bullet C^{n-1} = [p'(r)\bullet(s-r) + p(r)\bullet n]\bullet C^{n-1}\bullet D$$

$$q(y) = r(0)\bullet y^n + r'(0)\bullet y^{n-1} + ...$$

If $p(r)\bullet C^n=\pm 1$, (i.e., $C=p(r)^{1/n}$), then $q(y)$ will be a monic polynomial. If $p(s)\bullet D^n=\pm 1$, (i.e., $D=p(s)^{1/n}$), then $q(0)=\pm 1$. If $(C/D)^n=\pm(p(s)/p(r))$, (i.e., $C/D=\sqrt[n]{|p(s)/p(r)|}$), then $q(y)$ will be *normalized.* Thus, the Möbius transformation $x=(ry+st)/(y+t)$, where $t=\sqrt[n]{|p(r)/p(s)|}$, retains two degrees of freedom—r,s—but still produces a normalized result. We can use these two degrees of freedom to force the coefficients of x^{n-1} and x to zero to further normalize the polynomial.

Since the Möbius transform incorporates translation and dilation (plus more), we can use this powerful operation to transform any polynomial into the following form: the coefficient of x^n is 1, the coefficient of x^{n-1} is 0, and the coefficient of x is 0. After this transformation, all polynomials up to a cubic are in the form $x^n\pm 1$, and the quartic is in the form $x^4+C'x^2\pm 1$, which is solvable in x^2 as a quadratic.

Since q'(0) is the coefficient of x and r'(0) is the coefficient of x^{n-1}, we set them to zero.

$$q'(0) = 0 = p'(s)\bullet(r-s) + p(s)\bullet n$$

$$r'(0) = 0 = p'(r)\bullet(s-r) + p(r)\bullet n$$

Subtracting one equation from the other, we get:

$$p'(r)\bullet(s-r) + n\bullet p(r) - p'(s)\bullet(r-s) - n\bullet p(s)$$

$$= n\bullet(p(r)-p(s)) - (r-s)\bullet(p'(r)+p'(s))$$

$$= (p(r)-p(s))/(r-s) - (p'(r)+p'(s))/n$$

But this is an equation of degree n-1.

$$r^{n-2}\bullet[p'(s)\bullet(r-s)+p(s)\bullet n] - s^{n-2}\bullet[p'(r)\bullet(s-r)+p(r)\bullet n] = 0$$

$$r^{n-2}\bullet[n\bullet p(s)-s\bullet p'(s)] - s^{n-2}\bullet[n\bullet p(r)-r\bullet p'(r)] + r^{n-1}\bullet p'(s) - s^{n-1}\bullet p'(r) = 0$$

Solving these equations for reasonble values of r and s will produce the form sought after.

Copyright (c) 1986 by Henry G. Baker

Linear Polynomials

Linear polynomials have the form $p(x)=Ax+B$; changing to monic form gives $p(x)/A=x+B/A$; translating to the center of mass gives $q(y)=p(y-B/A)/A=(y-B/A)+B/A=y$, which cannot be further reduced. $q(y)=y$ has the single root $y=0$, so we have zero degrees of freedom.

Quadratic Polynomials

Quadratic polynomials have the form $p(x)=Ax^2+Bx+C$; changing to monic form gives $p(x)/A=x^2+(B/A)x+(C/A)$; translating to the center of mass gives

$$q(y)=p(y-B/(2A))/A=y^2+C/A-B^2/(4A^2);$$

dilating so that the product of the roots =1 gives $r(z)=q(\text{sqrt}(-D)\bullet y)/(-D)=z^2+1$, where $D=(B^2-4AC)/(4A^2)$, so we have zero degrees of freedom.

Cubic Polynomials

Cubic polynomials have the form $p0(x)=Ax^3+Bx^2+Cx+D$; changing to monic form gives the form $p1(y)=y^3+B'y^2+C'y+D'$; translating to the center of mass gives the form $q(z)=z^3+Pz+Q$. We could dilate to make $Q=1$, but since that change will not solve the equation, we go directly to the big gun: the Möbius transformation $w=(rz+st)/(z+t)$.

As above, we get:

$$(q(r)-q(s))/(r-s) - (q'(r)+q'(s))/3 = 0$$

$$[r^3-s^3+P(r-s)]/(r-s) - (3\bullet(r^2+s^2)+2P)/3$$

$$= (r^2+rs+s^2)+P - (r^2+s^2)-2P/3$$

$$= rs + P/3 = 0$$

$$r[3\bullet q(s)-s\bullet q'(s)] - s[3\bullet q(r)-r\bullet q'(r)] + r^2\bullet q'(s) - s^2\bullet q'(r) = 0$$

$$3Q + P(r+s) = 0$$

So, r and s are the two roots of the quadratic polynomial $x^2+3(Q/P)x-P/3$. Using this transformation, we get the equation $w^3-1=0$, which is easily solved by the extraction of a cube root. We note that the discriminant of this quadratic indicates whether r,s will be real or complex. If r,s are real, then the Möbius transformation is real, and can only squeeze (or stretch) the real axis, but cannot "bend" it into a circle. In this case, the Möbius transformation repositions the complex roots of the cubic as the base of an equilateral triangle with the real root at its apex—preparing for the cube root extraction, which can handle only such an equilateral case. However, if r,s are complex, then the Möbius transformation bends the real axis into a circle—in this case a circle centered on the origin, and can therefore position 3 real roots of a cubic at the three points of an equilateral triangle so that the polynomial can be solved using a root extraction.

Thus, we have shown how to construct the Möbius transformation (which we knew had to exist) to map the 3 roots of a cubic polynomial onto 3 regularly spaced points around a circle centered on the origin.

Since all cubic polynomials can be transformed into either $x^3=0$ or x^3-Px-1 by using the transformations given above, it is useful to study the location of the roots of $p(x)=x^3-Px-1$ as we vary P. Since

$$x^3-Px-1 = 0$$

$$P = (x^3-1)/x = x^2-1/x$$

Copyright (c) 1986 by Henry G. Baker

We can now consider this equation under different conditions. As x->+∞, the 1/x term becomes negligible, so P≈x^2, hence x≈±√P. However, as x->0, the term x^2 becomes negligible, so P≈-1/x, hence x≈-1/P. Rephrasing in terms of P, we can piece together the following behavior:

$$x \approx -1/P \text{ as } P\text{->}\infty$$

$$x = 1 \text{ as } P\text{->}0$$

$$x \approx \pm\sqrt{P} \text{ as } P\text{->}+\infty$$

Summarizing, as P goes from -∞ to 0, x is single-valued and moves smoothly from 0 to 1. That branch continues smoothly upward and becomes asymptotic to +√P. However, at $P=3\cdot2^{-2/3}$, another two branches come in starting with $x=-2^{-1/3}$ and one goes smoothly upward to become asymptotic to x=0 while the other falls smoothly downward to become asymptotic to -√P.

Hence, we can construct two useful functions of x as a function of P: the first, x(P) is continuous and differentiable and follows the upper branch from P=-∞ to P=+∞; the second, $x_{min}(P)$ follows the positive branch from P≈-∞ to $P=3\cdot2^{-2/3}\approx1.889882$, where x drops from $x=2^{2/3}\approx1.587401$ down to $x=-2^{-1/3}\approx-0.793701$, and follows the x≈-1/P branch from there to x≈+∞. $x_{min}(P)$ produces the real root of minimum absolute value, while x(P) produces the positive root. For the purposes of factoring the polynomial, $x_{min}(P)$ is to be preferred, since it produces the real root of minimum absolute value more accurately than if it has to be recovered from the deflated polynomial later, while for many other applications, x(P) is to be preferred, since it is a continuous, infinitely differentiable function of P.

(The actual accurate calculation of $x_{min}(P)$ is somewhat involved. In the range $3\cdot2^{-2/3}\le P<2$, the two negative real roots are close together and not readily resolved using Newton's method on the third degree equation. However, if we first locate the positive root using x(P), and then mask it (effectively giving us a quadratic equation), then we can focus in on separating the two negative real roots.

The problem of numerical instability on the lower two branches is obvious when one looks at the graph. The slope of the second and third branches approaches ∞ as $P\text{->}3\cdot2^{-2/3}$. How is the computational engineer to locate the two roots as accurately as possible in this situation? Clearly, the accurate location of the positive real root followed by polynomial deflation, since deflation in this instance is not stable—e.g., the deflated quadratic polynomial could have imaginary roots, even though $P>3\cdot2^{-2/3}$, a situation which is clearly undesirable. Reorigining the polynomial at $x=-2^{-1/3}$ through polynomial substitution is also unstable. Probably the most accurate method for estimating the exact location of the two sensitive real roots is using a Taylor series expansion around $x=-2^{-1/3}$:

$$p(x+2^{-1/3}) = p(2^{-1/3}) + p'(2^{-1/3})\cdot x + p''(2^{-1/3})/2\cdot x^2+x^3$$

$$p(2^{-1/3}) = (2^{-1/3})^3-P\cdot(2^{-1/3})-1 = -2^{-1/3}P-1/2$$

$$p'(2^{-1/3}) = 3\cdot(2^{-1/3})^2-P = 3\cdot2^{-2/3}-P$$

$$p''(2^{-1/3})/2 = 3\cdot2^{-1/3}$$

However, if $P= 3\cdot2^{-2/3}+\Delta$, then

$$p(2^{-1/3}) = -2^{-1/3}(3\cdot2^{-2/3}+\Delta)-1/2 = -3/2-2^{-1/3}\cdot\Delta-1/2 = -2^{-1/3}\cdot\Delta-2$$

$$p'(2^{-1/3}) = 3\cdot2^{-2/3}-(3\cdot2^{-2/3}+\Delta) = -\Delta$$

$$p''(2^{-1/3})/2 = 3\cdot2^{-1/3}$$

so

$$p(x+2^{-1/3}) = -(2^{-1/3}\cdot\Delta+2) - \Delta\cdot x + 3\cdot2^{-1/3}\cdot x^2 + x^3$$

Copyright (c) 1986 by Henry G. Baker

The polynomial is now origined so that the (old) positive root is so far away that it has little influence on our Newton methods, so that if we can get good starting approximations for our two close-in real roots, we can converge rapidly. Since $dP/dx|_x=-2^{-1/3} = 0$ and $d^2P/dx|_x=-2^{-1/3} = 6$, a good initial guess for the two real roots is $x=-2^{-1/3}\pm 3^{-1/2}\bullet\sqrt{\Delta}$.).

Quartic Polynomials

Quartic polynomials have the form $p(x)=Ax^4+Bx^3+Cx^2+Dx+E$, which can be reduced to the form $q(y)=y^4+Py^2+Qy+R$, by means of the translation given above, so we have three degrees of freedom left. We would like to also make $Q=0$ and $R=\pm 1$ using a Möbius transformation: $z=(ry+st)/(y+t)$. (If $Q=0$ already, then we can skip this step.)

$$(q(r)-q(s))/(r-s) - (q'(r)+q'(s))/4 = 0$$

$$2rs(r+s) + P(r+s) + Q = 0$$

$$r^2[4\bullet q(s) - s\bullet q'(s)] - s^2[4\bullet q(r) - r\bullet q'(r)] + r^3\bullet q'(s) - s^3\bullet q'(r) = 0$$

$$2Prs(r+s) + Q(r+s)^2 + 2Qrs + 4R(r+s) = 0$$

Letting $u=rs$, $v=r+s$, we get:

$$2uv + Pv + Q = 0 \text{ and}$$

$$2Puv + Qv^2 + 2Qu + 4Rv = 0$$

$$Qv^3 + (4R-P^2)v^2 - 2PQv - Q^2 = 0$$

$$\text{or} \quad v^3 + (4R-P^2)/Q\, v^2 - 2Pv - Q = 0 \quad (\text{since } Q\neq 0)$$

$$u = -P/2 - Q/(2v)$$

Thus, by solving a 3rd degree equation, we can get u,v; hence, we can get r,s (r,s are the roots of x^2-vx+u); and then transform our equation into the form $r(z)=x^4+P'x^2\pm 1$, which can be solved as a quadratic equation in x^2.

Thus, by solving quadratics and cubics, one can eventually solve a quartic.

However, following the recipe above may result in using complex arithmetic, even when all of the coefficients of the polynomial are real. However, if we are more careful, we can use only real arithmetic for our Möbius transformations, and at least factor our quartic polynomial over the reals, even when it has quadratic factors which are irreducible over the reals.

There are several points in the solution of a quartic where complex numbers may enter. The first place is in the solution of the cubic in v. But since a cubic with real coefficients always has at least one real root, we can always be guaranteed of a real solution to that equation. The next place that complex numbers can enter is in the computation of r,s from the quadratic r^2-vr+u. *There should be some way to ensure that real r,s can be generated.* The final place that complex numbers can enter is in the solution of the quadratic $z^4+P'z^2\pm 1$, if the discriminant of the quadratic in z^2 is less than zero. However, in this case, we can use another Möbius transformation $(1+x)/(1-x)$ and be guaranteed that the resulting quadratic has a positive discriminant. Since compositions of Möbius transformations are also Möbius transformations, we can now use the composite Möbius transformation to produce a quadratic $z^4+P''z^2\pm 1$ which is guaranteed to have real roots.

To see this, consider the polynomial z^4+Pz^2+1 (the polynomial z^4+Pz^2-1 has quadratic discriminant P^2+4, which is always positive, hence the solutions z^2 are always real). Suppose that $P^2-4<0$, i.e., $P^2<4$, i.e., $|P|<2$. Consider the result of the Möbius transformation $z=(1+x)/(1-x)$ on this equation.

$$z^4 + Pz^2 + 1 = (2+P)\, x^4 + (12-2P)\, x^2 + (2+P) = 0$$

$$x^4 + 2(6-P)/(2+P)\, x^2 + 1 = 0$$

Copyright (c) 1986 by Henry G. Baker

But this equation has quadratic discriminant $4(6-P)^2/(2+P)^2-4$, which is positive when $(6-P)^2/(2+P)^2>1$, i.e., when $(6-P)^2 > (2+P)^2$. But this holds so long as $P<4$, which is certainly true because $P^2<4$. Thus, our transformation guarantees the existence of real roots.

Thus, quartics with real coefficients can always be solved using only real arithmetic, assuming that cubics can be solved using only real arithmetic.

Iterated Transformations on Polynomials

Consider the following series of transformations on a monic polynomial: *center, normalize, invert.* This cycle attempts to make both the coefficient of x^{n-1} and x vanish at the same time, while keeping the constant coefficient and the high-order coefficients ±1. On linear polynomials, this cycle stabilizes after one cycle to $x\pm1$. On quadratic polynomials, this cycle stabilizes after one cycle to $x^2\pm1$. On cubic polynomials, this cycle converges to $x^3\pm1$. However, on quartic polynomials, this cycle does not converge. This cycle (and similar cycles) should be investigated as to whether they ever converge, and why.

Polynomial Transformation which Increase the Degree

There are some interesting transformations on polynomials which increase the indicated degree of the polynomial without increasing its degrees of freedom.

Given a polynomial p(x), if we compute $q(x)=p(x)\bullet p(-x)$, we are guaranteed to produce a polynomial $r(y)=r(x^2)=q(x)$; i.e., q(x) has non-zero coefficients only for even powers of x. This is most easily seen in the polynomial's factored form:

$$\begin{aligned} p(x)\bullet p(-x) &= (x-r_1)(x-r_2)...(x-r_n)\bullet(x+r_1)(x+r_2)...(x+r_n) \\ &= (x-r_1)(x+r_1)(x-r_2)(x+r_2)...(x-r_n)(x+r_n) \\ &= (x^2-r_1^2)(x^2-r_2^2)...(x^2-r_n^2) \\ &= r(x^2) = r(y) \end{aligned}$$

Thus, r(y) still has degree n (the degree of p(x)), but all of p's roots are squared! Thus, if p(x) has only real roots, then r(y) has only positive real roots.

Using this transformation, we can actually compute the second moment of the roots without actually solving for the roots of the polynomial! (We first move the origin to the center of mass of the roots using the transformations given earlier. We then transform to the polynomial of the squared roots, and find the center of gravity of that. It is the second moment of the roots of the original polynomial.)

We can use the same trick with other roots of unity. If ω is an m'th root of unity, then the product $p(x)\bullet p(\omega x)\bullet p(\omega^2 x)\bullet...\bullet p(\omega^{m-1}x)=q(x)=r(x^m)=r(y)$ is a polynomial whose only nonzero coefficients are those in front of powers of x^m, and hence is a polynomial in x^m. Thus, r(y) is a polynomial of degree n.

<< Things to mention: Loos's method for sum or product of roots of different polynomials. >>

Characteristic Polynomials

It is useful in this context to summarize a number of facts about polynomials arising from a special source—as the characteristic polynomials of a square matrix.

1. The characteristic polynomial of a matrix can be translated by adding a constant to each of the diagonal elements of the matrix. Thus, if the characteristic polynomial of the matrix A is p(x), then the characteristic polynomial of the matrix A+cI is p(x-c).

Copyright (c) 1986 by Henry G. Baker

2. The characteristic polynomial of a matrix can be dilated by multiplying every element of the matrix by a constant. Thus, if the characteristic polynomial of a matrix A is p(x), then the characteristic polynomial of the matrix cA is p(x/c).

3. The characteristic polynomial of a matrix can be reversed by taking the inverse of the matrix. Thus, if the characteristic polynomial of a matrix A is p(x), then the characteristic polynomial of the matrix A^{-1} is $p_r(x)$.

4. The roots of the characteristic polynomial of a Hermitian matrix (or symmetric matrix, if the entries are real) are all real.

Localizing the roots of a polynomial

Sturm sequences provide for the localization of real roots of a polynomial, but do nothing to localize complex roots. However, there are slightly more general methods, involving the polynomial remainder sequences of p(x) with polynomials other than p'(x) which give information about the location of the roots of p(x) with respect to circles or straight lines in the complex plane. In particular, the method of Routh-Hurwitz provides a count of roots in a half-plane, while the method of Schur-Cohn provides a count of roots in a circle. (Since a Möbius transformation can be used to transform any line into a circle and vice versa, the two methods are in some sense equivalent, although the direct formulation is somewhat more economical.)

The Routh-Hurwitz criterion involves the continued fraction expansion of Re(p(x))/Im(p(x)) to count zeros in the upper and lower halves of the complex plane, while the Schur-Cohn criterion involves the continued fraction expansion of $p(x)/p_r^*(x)$ (where $p_r^*(x)$ is the complex conjugate of the reverse of p(x)). Obviously, counting zeros in any other half-plane or circle can be achieved through the translation, rotation or dilation of the polynomial to fit these cases.

Root-finding using the Q-D algorithm

An intriguing algorithm for finding the zeros of a polynomial with real coefficients was given in the ACM (8,9) by Henrici and Watkins, which is a form of Aitkin's method, which is based on the fact that ratio of adjacent coefficients in the power series expansion of 1/p(x) eventually converges to the root of largest modulus (absolute value), unless there are several roots of equal modulus.

Root-finding using Graeffe's method

Graeffe's method involves the root-squaring trick indicated above. Since continually squaring all the roots will eventually separate roots of differing moduli, we can then read off the moduli as the ratios of successive coefficients of the multiply-squared polynomial. Since we can square a polynomial using FFT's with only nlogn operations, we should be able to separate the roots very quickly using this method.

While going from the 2^n'th power of the roots to the roots themselves might be a bit of a problem for complex roots, it should be simple for real roots, since there are only two choices of values with a given modulus.

Reciprocal Equations

$p(x)=p_r(x)$ or $p(x)=-p_r(x)$. Can be reduced to equation of 1/2 size.

Descartes' Rule of Signs and Budan's Theorem

Not as good as Sturm sequences, but a hell of a lot cheaper.

Copyright (c) 1986 by Henry G. Baker