Computing Zeros on a Real Interval Through Chebyshev Expansion and Polynomial Rootfinding*

John P. Boyd

Abstract. Robust polynomial rootfinders can be exploited to compute the roots on a real interval of a nonpolynomial function \( f(x) \) by the following: (i) expand \( f \) as a Chebyshev polynomial series, (ii) convert to a polynomial in ordinary, series-of-powers form, and (iii) apply the polynomial rootfinder. (Complex-valued roots and real roots outside the target interval are discarded.) The expansion is most efficiently done by adaptive Chebyshev interpolation with \( N \) equal to a power of two, where \( N \) is the degree of the truncated Chebyshev series. All previous evaluations of \( f \) can then be reused when \( N \) is increased; adaption stops when \( N \) is sufficiently large so that further increases produce no significant change in the interpolant. We describe two conversion strategies. The "convert-to-powers" method uses multiplication by mildly ill-conditioned matrices to create a polynomial of degree \( N \). The "degree-doubling" strategy defines a polynomial of larger degree \( 2N \) but is very well-conditioned. The "convert-to-powers" method, although faster, restricts \( N \) to moderate values; this can always be accomplished by subdividing the target interval. Both these strategies allow simultaneous approximation of many roots on an interval, whether simple or multiple, for nonpolynomial \( f(x) \).

Key words. rootfinding, single transcendental equation, Chebyshev series

AMS subject classifications. 65H05, 42C10, 65E05

PII. S0036142901398325

1. Introduction. One irony of the history of mathematics is that the problem of finding the roots of a polynomial, which taxed the brains of mathematicians from the Babylonians to Diophantus to Omar Khayyam to Cardano and Tartaglia in the Renaissance to Lagrange, Abel, Gauss, and Galois around the turn of the 19th century, is now largely uninteresting. Reliable polynomial rootfinding software, which requires no a priori estimates for the zeros, is now a part of almost all language packages (Matlab, Maple, Mathematica) and Fortran libraries (NAG, IMSL, etc.). The undergraduate who casually executes the one-line Matlab command, \texttt{roots(p)}, where \( p \) is a vector containing the coefficients of the polynomial, is blissfully ignorant of the three centuries of struggle to move from Ferrari's literal solution of the quartic, published in 1545, to Hermite's solution of the quintic 320 years later.

Nevertheless, the problem of finding the roots of a single transcendental equation in a single unknown is still a staple of numerical analysis courses. The reason is that, until recently, there was no black box for computing the zeros of a nonpolynomial \( f(x) \). Bisection and Brent's algorithm will reliably find some roots, but this is not the same as finding all roots on an interval. It is particularly easy to miss zeros that are closely spaced or multiple.

Kavvadias and Vrahatis [10], Kavvadias, Makri, and Vrahatis [11], and Smiley and Chun [13] have developed bisection-like but more sophisticated subdivision strategies for reliable transcendental rootfinding. However, these algorithms are relatively slow. Later, we shall explain how these subdivision strategies can in principle be accelerated...
We show that it is easy to extend the existing library software for polynomial \( f \) to general \( f \) merely by a simple intervention: expanding \( f \) as a Chebyshev series and then converting the Chebyshev approximation to an ordinary polynomial. As seen through the lens of Chebyshev polynomials, there is no such thing as a “transcendental” function: all rootfinding problems are polynomial rootfinding problems.

In his book *Applied Analysis* (1956) [12], Lanczos published the first example of the “Chebyshevization” of rootfinding. At a time when general polynomial solvers did not exist, he collapsed a cubic equation (hard) to a quadratic (easy!) by expanding the cubic as a Chebyshev series and then neglecting the third degree term.

In an earlier paper [5], we extended Lanczos’s strategy to complicated \( f(x) \). However, our earlier work was criticized because it did not provide estimates for the condition number of the conversion-to-powers step. Since Wilkinson’s famous example of a very badly conditioned polynomial (well illustrated on pp. 330–331 of [1]), all right-thinking numerical analysts have cringed at working with a polynomial expressed as a sum of powers of \( x \). In this work, we show that, although there is some ill-conditioning, the convert-to-powers strategy is robust and reliable if the degree of the Chebyshev expansion is restricted to moderate \( N \) (i.e., \( N < 18 \) or so). By subdividing an interval with many roots into subintervals, and applying a separate Chebyshev expansion to each one, the Chebyshev-to-powers strategy can be applied to almost all functions which are analytic on a desired target interval.

Furthermore, there is an alternative strategy, discussed here for the first time, which allows extraction of roots from a polynomial \( h_{2N}(z) \) whose coefficients are simply those of the Chebyshev series. The ill-conditioning is completely eliminated. However, the degree of \( h_{2N}(z) \) is twice that of the truncated Chebyshev series from whence it came.

Figure 1 schematically summarizes our algorithm.

The first issue is, How is the Chebyshev series computed? The answer is that \( f(x) \) must be evaluated at a set of discrete points on the target interval; the Chebyshev coefficients are then given by a matrix-vector multiply where the vector holds the set of grid-point values of \( f(x) \) and the elements of the matrix are trigonometric functions. The complete procedure is described in the appendix.

The second issue is, How does one determine when the truncation \( N \) is large enough? There is a well-established theory for doing this as reviewed in our book [6] and previous article [5]. The most systematic strategy mimics that of the Clenshaw–Curtis quadrature: the number of points is doubled until the approximation ceases to change; all previously used values of \( f(x) \) are reused by finer approximations so that nothing is wasted. We shall briefly review stopping criteria below.

The third issue is, How does one convert a truncated Chebyshev series to an ordinary polynomial? We offer two ways. In the “convert-to-powers” strategy, the coefficients of the powers of \( x \) are the product of an upper triangular matrix with the vector of Chebyshev coefficients; the matrix elements are integers computed by a simple recurrence.

The “degree-doubling” algorithm defines an associated polynomial whose degree is twice that of the truncation of the Chebyshev series. However, the real part of the roots of this polynomial which lie on the unit circle in the complex plane are the roots of \( f(x) \) on the real interval \( x \in [a, b] \).

The fourth issue is, Given that the roots of a polynomial are notoriously sensitive to small perturbations to the coefficients of the powers of \( x \), how ill-conditioned is the
convert-to-powers algorithm? The answer is that if the Chebyshev degree is restricted, the roots of the polynomial on the target interval will be very good approximations to those of the truncated Chebyshev series, whose zeros are in turn very good approximations to those of the original $f(x)$. Small errors can easily be corrected by one or two secant or Newton iterations with $f(x)$. If the interval is large and has many roots, it may be necessary to subdivide the interval into subdomains and compute a different moderate degree Chebyshev series on each. With these precautions of degree restriction and interval subdivision followed by iteration with $f(x)$ itself, the convert-to-powers algorithm can yield roots to full machine precision.
In this article, we offer four novelties (beyond the earlier work of Lanczos and [5]):

1. Condition number estimates for the triangular matrices that convert Chebyshev coefficients to the coefficients of the same polynomial expressed as a sum of powers of \(x\).

2. A second strategy for converting a truncated Chebyshev polynomial to a different polynomial of twice the degree but with (effectively) the same roots; the coefficients of the new polynomial are the same as the Chebyshev coefficients.

3. The Chebyshev-to-polynomial method is extended to unbounded intervals, either infinite or semi-infinite, so long as \(f(x)\) asymptotes to a constant at infinity and has only a finite number of real roots.

4. A different, grid-point-value-based “stopping” criterion for assessing when \(f\) is approximated to sufficient accuracy by a Chebyshev series truncated after the \(N\)th term.

The strength of the algorithm is that no a priori knowledge of the roots is needed. The reliability of existing polynomial solvers is extended to nonpolynomial \(f(x)\).

The sections of the article are as follows:

Sec. 2: first two issues (computing the Chebyshev expansion).

Sec. 3: convert-to-powers and its condition number.

Sec. 4: bounds on errors in roots and their application.

Sec. 5: the degree-doubling theorem.

Sec. 6: numerical example: roots of \(J_0(x)\).

Sec. 7: searching a region in the complex plane instead of a real interval.

Sec. 8: rootfinding on an infinite interval.

Sec. 9: summary and open problems.

2. Adaptive computation of the Chebyshev coefficients. Our Chebyshev approximation is a finite series of Chebyshev polynomials which interpolates \(f(x)\) at a set of \((N+1)\) points known as the Chebyshev–Lobatto grid. This differs little from the truncation of the infinite Chebyshev polynomial series of \(f\) [6]. One must evaluate \(f(x)\), the function whose roots are sought, at \((N+1)\) points on the target interval \(x \in [a, b]\). The Chebyshev coefficients are then the vector which is the product of a square matrix with the column vector of grid-point values of \(f\). (The grid points and the matrix elements are given in the appendix for arbitrary \(N\).)

If \(f(x)\) is expensive to evaluate, the best strategy is to restrict \(N\) to be a power of two. In this case, all previously computed grid-point values of \(f(x)\) can be reused when \(N\) is doubled so that the maximum number of evaluations of \(f\) is never more than the smallest (power-of-two) \(N\) for which the “stopping criteria,” is met. A similar strategy is employed in the adaptive, spectrally accurate Clenshaw–Curtis quadrature scheme [6].

As explained in [6], Chebyshev series for a function \(f\) which is analytic on the interval \(x \in [a, b]\) converge geometrically fast; that is, the \(j\)th term (and also the absolute value of the \(j\)th coefficient) are bounded by \(\rho^j\) for some \(\rho < 1\). The error in the \((N+1)\)-point interpolation is typically the same order of magnitude as the last computed Chebyshev coefficient \(a_N\) [6]. [5] proposed a cautious “stopping criterion”: increase \(N\) until \(\sum_{j=[(2/3)N]}^{N} |a_j| < \epsilon\), where \([(2/3)N]\) denotes the integer closest to \(2N/3\).

One can also use a grid-point value criterion which is given here for the first time:

\[
\max |f_{2N}(x_j) - f_N(x_j)| < \epsilon,
\]

(1)
where the difference is calculated for all the points on the grid of \((2N + 1)\) points which are not on the coarser grid of \((N+1)\) points. (At points common to both grids, both interpolants equal \(f\) and therefore each other.) Since the error of \(f_N\) tends to be a maximum roughly halfway between the points of the coarse grid, the difference at these intermediate points is likely to be quite close to the true maximum pointwise error of \(f_N\).

Both these stopping criteria are very conservative; \(f_{2N}\) will usually have an error much smaller than \(\epsilon\) with the grid-point criterion, which forces the error of the lower order approximation \(f_N\) to be less than \(\epsilon\). Reliability is built on conservative strategies, however.

2.1. Subdivision. As explained in the next section, conversion-to-powers is a well-conditioned process only if \(N\) is restricted to moderate degree. What if large \(N\) is needed to obtain an accurate Chebyshev approximation?

The answer is that one can divide the interval into subintervals. Our recommendation is to expand \(f\) on the entire interval first, even if this requires using large \(N\). If the maximum degree that allows satisfactory conversion-to-powers is \(N_{\text{max}}\), the asymptotic theory of Chebyshev expansions [6] suggests that one should subdivide into \([N/N_{\text{max}}]\) subintervals where the square brackets denote the integer closest to the ratio of \(N/N_{\text{max}}\). Cautious arithmurgists are encouraged to use a somewhat larger number of subdivisions.

Once the split into subdomains has been made, the algorithm can be applied on each subinterval without modification.

2.2. Scaling. Chebyshev expansions are highly uniform in the sense that the maximum pointwise error (absolute error) oscillates with peaks and troughs of similar amplitude over the entire expansion interval, \(x \in [a, b]\). If \(f(x)\) is itself highly nonuniform, such as \(\exp(10x)\sin(x)\), then the Chebyshev series will have large relative errors where \(f(x)\) is very small.

There are two remedies. The first is to subdivide into subintervals sufficiently small so that \(f(x)\) varies only mildly over each subdomain. The second is to multiply \(f\) by a smooth scaling function that eliminates the large fluctuations in magnitude. For our example, \(\bar{f} \equiv \exp(-10x)f(x) = \sin(x)\) has the same roots as \(f\), but, because it is much more uniform, the Chebyshev expansion of \(\bar{f}\) will have much smaller relative error and yield much more accurate approximations to the roots. Devising a smooth scaling function may be difficult, however, as illustrated in [5].

2.3. Nonanalytic/nonsmooth \(f(x)\). Chebyshev expansions converge, but at a very slow rate, if \(f(x)\) has poles, branch points, discontinuities, or other singularities on the expansion interval, \(x \in [a, b]\), including singularities at the endpoints. We therefore caution the reader that Chebyshev rootfinding methods are useful only when \(f(x)\) is analytic everywhere on the expansion interval including the endpoints. (Singularities off the expansion interval, whether at real or complex locations, however, are powerless to destroy the good properties of the algorithm and are thus largely irrelevant.)

3. Converting Chebyshev series to polynomials, I: Convert-to-powers. The Chebyshev expansion of \(f(x)\) on \(x \in [a, b]\) is

\[
    f_N \equiv \sum_{j=0}^{N} a_j T_j(y) = \sum_{j=0}^{N} b_j y^j, \quad y(x) \in [-1, 1],
\]
ROOTFINDING ON AN INTERVAL VIA CHEBYSHEV EXPANSIONS

where

\[ y \equiv \frac{2x - (b + a)}{b - a}, \quad y \in [-1, 1] \leftrightarrow x \in [a, b], \]

is the stretched-and-translated argument of the Chebyshev polynomials. The cost of transforming from \(\{a_j\}\) to \(\{b_j\}\) can be halved by splitting \(f_n\) into its even and odd parts: let \(\vec{a}_{\text{even}}\) and \(\vec{b}_{\text{even}}\) be vectors containing the even degree coefficients \(\{a_0, a_2, a_4 \ldots\}\) and \(\{b_0, b_2, b_4 \ldots\}\), respectively. Then

\[ \vec{b}_{\text{even}} = \vec{Q}_{\text{even}} \vec{a}_{\text{even}}. \]

Explicitly, the upper left block is given by

\[
\begin{bmatrix}
1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & 1 \\
0 & 2 & -8 & 18 & -32 & 50 & -72 & 98 & -128 \\
0 & 0 & 8 & -48 & 160 & -400 & 840 & -1568 & 2688 \\
0 & 0 & 0 & 32 & -256 & 1120 & -3584 & 9408 & -21504 \\
0 & 0 & 0 & 0 & 128 & -1280 & 6912 & -26880 & 84480 \\
0 & 0 & 0 & 0 & 0 & 512 & -6144 & 39424 & -180224 \\
0 & 0 & 0 & 0 & 0 & 0 & 2048 & -28672 & 212992 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 8192 & -131072 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 32768
\end{bmatrix}
\]

The elements can be computed by recurrence relation:

\[ Q_{11}^{\text{even}} = 1, \quad Q_{jj}^{\text{even}} = 2^{2j-3}, \quad j = 2, 3, \ldots. \]

The recurrence, vertically up the \(j\)-column from the diagonal, is then

\[ Q_{j-K,j}^{\text{even}} = \text{round}\left\{ -\frac{(2j - 2K)(2j - 2K - 1)}{2K(4j - 2K - 4)}Q_{j-K+1,j}^{\text{even}} \right\}, \]

\[ Q_{jj}^{\text{odd}} = 2^{2j-2}, \quad j = 1, 2, 3, \ldots, \]

\[ Q_{j-K,j}^{\text{odd}} = \text{round}\left\{ -\frac{(2j - 2K + 1)(j - K)}{K(4j - 2K - 2)}Q_{j-K+1,j}^{\text{odd}} \right\}. \]
Infinity norms of Chebyshev-to-powers matrices

![Graph showing the comparison between even and odd degree Chebyshev-to-powers matrices.](image)

**FIG. 2.** $L_\infty$ matrix norms of $\tilde Q^{\text{even}}$ and $\tilde Q^{\text{odd}}$ versus the dimension of the matrix $j$.

Because the elements are integers, we can eliminate roundoff error in the recurrences by rounding to the nearest integer.

Figure 2 shows how the norms of the transformed matrices grow exponentially, roughly as

$$||Q^{\text{even}}||_\infty \sim 0.016 (5.8)^j, \quad ||Q^{\text{odd}}||_\infty \sim 0.039 (5.8)^j.$$  

If we wish to avoid sacrificing more than six decimal places of accuracy or, more precisely, to guarantee that the errors in the coefficients of the power form are no more than a million times the floating point and truncation errors in the Chebyshev coefficients, we must restrict the size of $\tilde Q^{\text{even}}$ and $\tilde Q^{\text{odd}}$ to nine or less since

$$||Q^{\text{even}}||_\infty (9 \times 9) = 243,712, \quad ||Q^{\text{odd}}||_\infty (9 \times 9) = 559,104.$$  

If we put the even and odd polynomials together, we obtain a general nonsymmetric polynomial of degree 17.

4. **Condition number of polynomial roots.** Gautschi [8] and Winkler [14] give theorems on the condition number of polynomial roots. To give the flavor of these ideas without undue complexity, we shall state a simpler result applicable only in the limit of an arbitrarily small perturbation. Recall that the argument of the Chebyshev polynomials (and of the sum-of-powers into which it is transformed) is $y \in [-1, 1]$,
which is the image of the interval \( x \in [a, b] \) through the linear change-of-coordinate 
\( y = (2x - (b + a))(b - a) \).

**Theorem 4.1** (sensitivity of polynomial roots). Let \( y_* \) denote a real-valued root of multiplicity \( m \) on the interval \( y \in [-1, 1] \) for a function \( f \) which is written terms of a basis as

\[
f(y) = \sum_{j=0}^{N} a_j \phi_j(y), \quad y \in [-1, 1],
\]

where the basis functions are either Chebyshev polynomials or powers of \( y \) and in either event satisfy

\[
|\phi_j(y)| \leq 1 \quad \forall y \in [-1, 1].
\]

Let \( \tilde{y} \) denote the root of the perturbed function \( \tilde{f} \) which is equal to \( f \) except the \( k \)th coefficient has been altered by an amount \( \epsilon \):

\[
\tilde{f}(y) = \sum_{j=0, j \neq k}^{N} a_j \phi_j(y) + (a_k + \epsilon) \phi_k(y), \quad y \in [-1, 1].
\]

Then

\[
|\tilde{y} - y_*| \leq \epsilon^{1/m} \left( \frac{1}{m!} \frac{d^m f}{dy^m}(y_*) \right)^{-1/m} + O(\epsilon^{(m+1)/m}), \quad |\epsilon| \ll 1.
\]

**Proof.** Taylor's theorem at \( y = y_* \) is

\[
\tilde{f}(y) \approx \epsilon \phi_k(y) + \frac{1}{m!} \frac{d^m f}{dy^m}(y_*) (y - y_*)^m + O((y - y_*)^{m+1}),
\]

since, at an \( m \)th order zero of \( f \), the function itself and its first \( (m-1) \) derivatives are zero by definition. Solving the Taylor approximation for the root of the perturbed function, \( \tilde{y} \), and then invoking the inequality that all basis functions are bounded in magnitude by one on the interval, is sufficient to prove the theorem.

The theorem implies that Wilkinson's famous example of an ill-conditioned polynomial, which has haunted the dreams of numerical analysts for a generation, is dreaded unduly. When the largest root of interest has magnitude \( y_{\text{max}} \gg 1 \), then \( y^k \) can become as large as \( (y_{\text{max}})^k \) at that root, and thus tiny changes in the coefficient of \( y^k \) can produce huge changes in the largest root. However, when \( |y| \leq 1 \), all the powers of \( y \) are bounded by one, and a simple root will be altered only an \( O(\epsilon) \) amount by an \( O(\epsilon) \) perturbation of the coefficients.

Thus, for our purposes, the powers-of-\( x \) form is not ill-conditioned. The only difficulty is that the Chebyshev-to-powers matrix multiplication greatly magnifies small errors in the Chebyshev coefficients. Thus, when \( N \) is large, an alteration of \( \epsilon \) in the \( k \)th Chebyshev coefficient will produce changes of millions or billions of \( \epsilon \) in the coefficients of the powers-of-\( x \) form. This in turn will produce a comparable change in a simple root.

We conclude that if \( N \) is restricted to moderate values, such as \( N < 18 \), by subdividing the original interval, then the Chebyshev-to-powers algorithm will be reasonably well-conditioned, where "reasonably" means that we lose no more than six decimal places to the ugly condition numbers of the transformation matrices \( \tilde{Q}^{\text{even}} \), \( \tilde{Q}^{\text{odd}} \), and are still able to compute the roots to nine or ten decimal places.
5. Conversion to a polynomial, II: Degree-doubling. An alternative method for deriving a polynomial from a truncated Chebyshev series is given by the following.

**Theorem 5.1 (associated double-degree polynomial).** Let $f_N(x)$ be a polynomial:

\[ f_N(x) \equiv \sum_{j=0}^{N} a_j T_j(x). \]

Define a polynomial $h$ of twice the degree through

\[ h_{2N}(z) \equiv \sum_{j=0}^{2N} b_j z^j. \]

where

\[ b_j = \begin{cases} a_{j-N}, & j > N, \\ 2a_0, & j = N, \\ a_{N-j}, & j < N. \end{cases} \]

Then the roots $x_k$ of $f_N$ on the real interval $x \in [-1, 1]$ are related to the roots $z_k$ of $h_N(z)$ on the unit disk in the complex $z$-plane through

\[ x_k = \Re(z_k). \]

**Proof.** The identity $T_j(x) = \cos(jt)$ when $x = \cos(t)$ plus $\cos(t) \equiv (\exp(it) + \exp(-it))/2$ implies that

\[ f_N(\cos(t)) = \sum_{j=0}^{N} a_j \{\exp(it) + \exp(-it)\}/2. \]

Define

\[ h_{2N}(\exp(it)) = 2 \exp(iNt)f_N(\cos(t)). \]

Because $\exp(iNt)$ never vanishes, the roots of the product are identical with those of $f_N(\cos(t))$. Defining $z \equiv \exp(it)$ and recalling $\exp(ijt) = [\exp(it)]^j$ proves the theorem.

6. Numerical example. Figure 3 shows the success of the Chebyshev algorithm using the two different strategies for polynomial creation: convert-to-powers on the left and degree-doubling on the right.

The left panel shows that, for some $f(x)$ at least, the restriction to $N \leq 17$ for convert-to-powers is very conservative. With $N = 40$, the maximum relative error is less than 1 part in 3800 in all of the first 19 roots of the $J_0$ Bessel function. By increasing $N$, the error can be reduced to $O(10^{-12})$ for some of the roots. However, there are signs of ill-conditioning; the roots at the ends of chosen expansion interval do not converge with increasing $N$.

The degree-doubling method requires more computation because the “black box” polynomial rootfinder is asked to solve a polynomial of degree $2N$ instead of $N$. (The cost of the triangular matrix multiplications of the convert-to-powers scheme is only $(N^2/2)$ multiplications and the same number of additions.) However, the right panel shows that the degree-doubling method is completely free of the ill-conditioning that dogs the convert-to-powers method for large $N$. 
Nevertheless, the convert-to-powers strategy is very effective even for $N$ far larger than the restrictions suggested in earlier sections. Why is the ill-conditioning completely not ruinous for $N = 80$? Figure 4 (left) shows that the coefficients asymptote to a plateau of $O(10^{-16})$ for $N > 60$, which is a magnitude controlled by roundoff error. The very small size of the high degree Chebyshev coefficients keeps these coefficients from causing major problems when the series is converted to an ordinary polynomial of degree 80. It is important to note from (5) and (6) that the size of the matrix elements increases rapidly with the column so that the elements that dominate the condition number of these matrices are the multipliers of very tiny coefficients in the Bessel–Chebyshev series.

Even with the mild ill-conditioning, it is remarkable that 19 roots can be captured so accurately with no more than two to four evaluations of $f(x)$ per root. Another robust interval rootfinding such as bisection would surely require far more evaluations of $f$.

When $N$ is restricted to smaller values (and the expansion interval proportionately reduced), the ill-conditioning disappears as predicted. Figure 5 shows that, for a smaller interval, the errors are nearly uniform over the interval and decrease uniformly as $N$ increases. (No comparison with the degree-doubling scheme is shown because for this case, where $N$ is small and the convert-to-powers method is well-conditioned, there is no graphically discernible difference between the two algorithms.) With $N =$
FIG. 4. Left: absolute values of the Chebyshev coefficients of $J_0(x)$ for the interval $x \in [0, 60]$. Right: A plot of $J_0$ on this same interval, which contains 19 roots of $J_0$.

28, somewhat larger than our ultraconservative recommended maximum of $N = 17$, the first six roots of $J_0$ are all approximated to within a relative error of less than 1 part in a hundred billion!

7. Generalization and alternative: Interpolation in powers around a circle in the complex plane. To generalize our method to the complex plane, the crucial fact is that a power series is optimal for interpolation in a disk in the complex plane in the same way that Chebyshev polynomials are optimal for interpolation on a real interval [9]. Interpolation by a series of Chebyshev polynomials on a real interval is replaced by interpolation of a series of powers of $z$ on a circle in the complex $z$-plane. By applying interpolation-on-a-circle to many overlapping circles, polynomial root solvers can thus be applied to find roots of nontranscendental functions in arbitrary regions of the complex plane. Although they employ a different strategy to find roots within a circle, Delves and Lyness give a good discussion of such regional root finding methods [7].

8. Root finding on the whole real axis. If a function $f$ has an infinite numbers of roots on the real axis, it is obviously impractical to find them all numerically. However, it is often possible to find an asymptotic approximation to the roots of large $|x|$ and then numerically compute the finite number of roots for which $|x|$ is too small for the asymptotic formula to be accurate. For the $J_0$ Bessel function used as an
example, for instance, the kth root, conventionally denoted by \( j_{0,k} \), is asymptotically

\[
(24) \quad j_{0,k} \sim (k - 1/4)\pi + \frac{1}{8(k - 1/4)\pi} - \frac{31}{6(4k - 1)^3\pi^3} + O(k^{-5}).
\]

This approximation has an absolute error of only 0.0018 even for the first root and an error of just \( 2.7 \times 10^{-10} \) for the 20th root.

When \( f \) has only a finite number of roots on an unbounded interval, it is possible to find them directly by using a change-of-coordinate that maps the infinite interval into the canonical interval for Chebyshev polynomials, \( x \in [-1, 1] \). One can then apply the Chebyshev-to-polynomial algorithms, either convert-to-powers or degree-doubling, without modification.

If the coordinate on the infinite interval is denoted by \( y \), then a good mapping [2] is

\[
(25) \quad y = \frac{Lx}{\sqrt{1 - x^2}}; \quad x = \frac{y}{\sqrt{L^2 + y^2}}, \quad x \in [-1, 1], \ y \in [-\infty, \infty],
\]

where \( L \) is a constant, the user-choosable map parameter. Although the optimum \( L \) is problem-dependent [2, 6], the Chebyshev rate of convergence is not very sensitive to \( L \), and \( L = 1 \) is a good choice in most applications. Other mappings can work, too, as discussed in [4] and [6]; a similar transformation for the semi-infinite interval is explained in [3] and [6].
Infinite interval expansion: Errors in positive, finite root

For example, the function

\begin{equation}
(26) \quad f(y) \equiv (2y^2 - 1) \exp(-[1/2]y^2)
\end{equation}

has only two roots on the infinite interval, \( y_* = \pm1/\sqrt{2} \). Under the mapping, this becomes

\begin{equation}
(27) \quad f(y) = \left( 2\frac{L^2x^2}{1-x^2} - 1 \right) \exp \left( -\frac{1}{2} \frac{L^2x^2}{1-x^2} \right).
\end{equation}

Because this is symmetric with respect to the origin, we can improve efficiency by computing a Chebyshev expansion only on \( x \in [0, 1] \). The function has only a single finite root on the whole positive real axis at \( y = 1/\sqrt{2} \), which is equivalent to \( x = 1/\sqrt{3} \) in the transformed coordinate.

Figure 6 shows that the error decreases exponentially with \( N \), the degree of the Chebyshev approximation, with oscillations in \( N \) superimposed. At \( N = 70 \), the roundoff error in the Chebyshev-to-powers conversion finally asserts itself so that further increases in \( N \) worsen the error compared to \( N = 70 \) where the root is approximated to a relative error of about 1 part in six million. This roundoff problem
can be suppressed, as in earlier examples, by using the "degree-doubling" conversion instead.

The accuracy is not as good for the finite interval example, which is hardly surprising. Nevertheless, with the added device of a mapping of the infinite interval into a finite interval, the Chebyshev-to-powers rootfinding method is successful.

9. Chebyshev methods and other "black box" rootfinders. The rootfinders of Kavvadias and Vrahatis [10], Kavvadias, Makri, and Vrahatis [11], and Smiley and Chun [13] both claim to be "black boxes," that is, to find the roots of a transcendental \( f(x) \) without (i) requiring user input except for a subroutine to evaluate \( f(x) \) and (ii) requiring the user to understand their algorithms—unnecessary because of their reliability. Both methods are a kind of "bisection-with-macho"; after the interval is subdivided, tests are applied to exclude subintervals which are root-free. Subdivision-and-test is then recursively applied until all the roots are isolated in sufficiently narrow intervals. Kavvadias and Vrahatis test for roots by numerically evaluating the Kronecker–Picard integral, whose value is the number of roots; Smiley and Chun use the cheaper but less precise criterion that if the Lipschitz constant \( L \), defined by \( L = \max \{|f(x) - f(y)|/|x - y|\} \) for all \( x, y \) on the interval is such that \( |f(a)|, |f(b)| > L|b - a| \), then the interval \( x \in [a, b] \) must be root-free. Refinements such as Newton–Ralphson iteration in the "end game" (when a root has been isolated within a small interval) and local Lipschitz constants are used to accelerate convergence for both methods.

We shall not attempt detailed comparisons between our algorithm and theirs. The efficiency of both Kronecker–Picard and Lipschitz-test algorithms strongly depend upon subdivision strategies, numerical quadrature tactics, Lipschitz constant approximation schemes, and so on. These black boxes will significantly improve as further experience allows better "tuning."

Instead, we will merely note that these algorithms require a large number of evaluations of \( f(x) \) because of the repeated subdivisions and also the numerical quadratures or Lipschitz constant approximations. In principle, the cost of these evaluations could be dramatically reduced by replacing \( f(x) \) by its Chebyshev interpolant. It follows that our ideas are perhaps complementary rather than competitive with subdivide-and-test methods.

However, our algorithm is completely self-contained. Kronecker–Picard and Lipschitz-test algorithms are useful in an \( f \to \text{Chebyshev} \) approach only if these algorithms are superior to polynomial rootfinders. Are they? Alternatively, subdivide-and-test methods will fail to benefit from replacement of \( f \) by its Chebyshev proxy only for problems where \( f \) is cheap to evaluate and the subdivide-and-test methods converge faster than polynomial rootfinders. Such comparisons are highly problem-dependent and also implementation-dependent. We must leave these as open research questions.

10. Summary and open problems. Our main conclusion is that by "Chebyshevizing" a function \( f(x) \), that is, by replacing \( f(x) \) by its Chebyshev interpolant, the availability of robust polynomial rootfinders can be leveraged into reliable software for finding the roots of a smooth, analytic but otherwise arbitrary function \( f(x) \) on a given real interval. Our Matlab subfunction that computes the roots using the convert-to-powers method has only 45 executable statements, and the degree-doubling algorithm is even shorter. Automation of subdivision would require a few additional lines, but the overall algorithms are commendably simple: all the complexity is in the polynomial rootfinder, which the user borrows from a software library.
By using an unbounded-interval-to-finite-interval mapping, the method easily generalizes to finding the real-valued roots of a function over the entire real axis if these roots are finite in number and the function is sufficiently smooth as $|x| \to \infty$ so that the transformed function is $C^\infty$.

The “degree-doubling” method converts the truncated Chebyshev series of degree $N$ into an ordinary polynomial of degree $2N$ whose coefficients are the same as the Chebyshev coefficients. This completely eliminates the ill-conditioning problem. However, because the degree is doubled, the computational cost is greater than for the convert-to-powers algorithm.

The convert-to-powers method has the flaw that it is mildly ill-conditioned. This difficulty can be cured by restricting $N$ to moderate degree (less than 18) and subdividing the original target interval into as many subintervals as needed so that $f(x)$ is well-approximated by a Chebyshev series of restricted degree on each subinterval. However, degree restriction and subdivision are annoying complications. The reward is that the polynomial rootfinder is only asked to solve a polynomial of degree $N$ rather than $2N$.

The numerical examples show that the convert-to-powers method is not as ill-conditioned as the norms of the conversion matrices would indicate. The reason is that, for a given problem, the true condition number depends upon the rate of convergence of the Chebyshev series as well as upon the matrix norms. The most extreme example is when $f(x)$ is a polynomial of finite degree $k$ so that all Chebyshev coefficients $a_j$ are zero for $j > k$. In this case, only the upper left $(k+1)/2 \times (k+1)/2$ blocks of the transformation matrices have anything to operate on. The effective condition number is determined by these blocks and not by the actual size of $N$, which may be much larger. For the Bessel function example, the Chebyshev series of $J_0$ is not identically zero for large degree, but the exponentially fast decrease of the Chebyshev coefficients does drastically reduce the effective condition number. An open problem is to develop a refined $f$-dependent condition number that takes the rate of Chebyshev convergence into account.

Because of the competing virtues and flaws of well-conditioned versus mildly ill-conditioned, fast versus slow, it is not possible to anoint either the degree-doubling or convert-to-powers algorithm as the “best” choice. What can be said is that both work well.

A minor unsolved problem, discussed at length in [5] but not here, is to find a good multiplicative scaling function when $f(x)$ varies by many orders of magnitude on the search interval $x \in [a, b]$. Because Chebyshev expansions are highly uniform in absolute error, there may be annoyingly large relative errors when $f(x)$ is badly scaled in the sense of having huge maxima on some parts of the interval but only tiny peaks and valleys on other subintervals. In theory, this difficulty can always be solved by subdividing the interval into subintervals and applying the algorithm on each subdomain.

The major unsolved problem is to find a good direct way to find all the roots of a polynomial on a real interval when the polynomial is defined by its Chebyshev coefficients without prior conversion to powers-of-$x$ form. If such an algorithm could be found, then both the convert-to-powers and degree-doubling procedures become unnecessary.

Seen through the lens of Chebyshev polynomial series, there is no such thing as a nonpolynomial function. Every $f(x)$ is a truncated Chebyshev series in disguise. From the Chebyshev perspective, it is as easy to simultaneously find all roots of a
function \( f(x) \) on a real interval, whether simple zeros or multiple roots, as it is for a polynomial.

**Appendix A. Chebyshev interpolation of a function \( f(x) \).**

**Goal.** Compute a Chebyshev series, including terms up to and including \( T_N \), on the interval \( x \in [a, b] \).

**Step 1.** Create the interpolation points (Lobatto grid):

\[
x_k = \frac{b - a}{2} \cos \left( \pi \frac{k}{N} \right) + \frac{b + a}{2}, \quad k = 0, 1, 2, \ldots, N.
\]

**Step 2.** Compute the elements of the \((N + 1) \times (N + 1)\) interpolation matrix. Define \( p_j = 2 \) if \( j = 0 \) or \( j = N \) and \( p_j = 1, j \in [1, N - 1] \). Then the elements of the interpolation matrix are

\[
I_{jk} = \frac{2}{p_j p_k} \cos \left( j \pi \frac{k}{N} \right).
\]

**Step 3.** Compute the grid-point values of \( f(x) \), the function to be approximated:

\[
f_k = f(x_k), \quad k = 0, 1, \ldots, N.
\]

**Step 4.** Compute the coefficients through a vector-matrix multiply:

\[
a_j = \sum_{k=0}^{N} I_{jk} f_k, \quad j = 0, 1, 2, \ldots, N.
\]

The approximation is

\[
f \approx \sum_{j=0}^{N} a_j T_j \left( \frac{2x - (b + a)}{b - a} \right) = \sum_{j=0}^{N} a_j \cos \left\{ j \arccos \left( \frac{2x - (b + a)}{b - a} \right) \right\}.
\]

**REFERENCES**


