SIAM REVIEW
Vol. 55, No. 2, pp. 375–396

Finding the Zeros of a Univariate Equation: Proxy Rootfinders, Chebyshev Interpolation, and the Companion Matrix*

John P. Boyd†

Abstract. When a function $f(x)$ is holomorphic on an interval $x \in [a, b]$, its roots on the interval can be computed by the following three-step procedure. First, approximate $f(x)$ on $[a, b]$ by a polynomial $f_N(x)$ using adaptive Chebyshev interpolation. Second, form the Chebyshev–Frobenius companion matrix whose elements are trivial functions of the Chebyshev coefficients of the interpolant $f_N(x)$. Third, compute all the eigenvalues of the companion matrix. The eigenvalues λ which lie on the real interval $\lambda \in [a, b]$ are very accurate approximations to the zeros of $f(x)$ on the target interval. (To minimize cost, the adaptive phase can automatically subdivide the interval, applying the Chebyshev rootfinder separately on each subinterval, to keep N bounded or to solve rare "dynamic range" complications.) We also discuss generalizations to compute roots on an infinite interval, zeros of functions singular on the interval $[a, b]$, and slightly complex roots. The underlying ideas are undergraduate-friendly, but link the disparate fields of algebraic geometry, linear algebra, and approximation theory.

Key words. rootfinding, Chebyshev polynomials, transcendental equations

AMS subject classifications. 65H05, 65H05, 34C08

DOI. 10.1137/110838297

1. Introduction.

1.1. Polynomial Roots and Radicals.

Algorithms and explicit formulas for computing the roots of a nonlinear algebraic or transcendental equation have been much studied since the time of the Babylonians, who discovered the formula for the zeros of a quadratic polynomial. The topic is too important for omission from most undergraduate mathematics curricula, but most engineering and applied mathematics majors are exposed only to the traditional remedies: Newton's iteration, the secant iteration, and perhaps the bisection algorithm. All these have the virtue of simplicity combined with the vice of intermittent failure, as explained below.

The Chebyshev-proxy rootfinder (CPR) is more complicated. It is limited to computing the zeros of a smooth function on a real interval (though generalizations that relax these restrictions are described later). Even so, the CPR algorithm offers much better reliability—isn't it embarrassing that undergraduates are taught to worship at the altars of proof and rigor, and then taught rootfinding iterations that will work *most of the time*? *Maybe*? *Frequently*? The CPR algorithm is also educationally

*Received by the editors June 22, 2011; accepted for publication (in revised form) March 19, 2012; published electronically May 8, 2013. This work was supported by the National Science Foundation through grants OCE 0451951, ATM 0723440, and OCE 1059703.
http://www.siam.org/journals/sirev/55-2/83829.html

†Department of Atmospheric, Oceanic & Space Science, University of Michigan, Ann Arbor, MI 48109 (jpboyd@umich.edu, http://www.engin.umich.edu:/∼jpboyd/).

Table 1.1 *Solve-the-proxy methods in one unknown.*

Note: As usually implemented, Brent's method bounds the corrections and shifts to bisection as needed.

Approximation	Name
Piecewise linear interpolation	Make-a-Graph-Stupid (MAGS) algorithm
Linear Taylor series	Newton or Newton–Raphson iteration
Linear interpolant	secant iteration
Quadratic Taylor series	Cauchy's method
Quadratic interpolant	Muller's method
Inverse quadratic interpolant	Brent's algorithm
[1/1] (Linear-over-linear) Padé approximant	Halley's scheme
[2/2] (Quadratic-over-quadratic) Padé approximant	Shafer's method (Shafer [48])
Chebyshev polynomial interpolant	Chebyshev proxy method (Boyd [10])

appealing because it is a blend of ideas from very different branches of mathematics: interpolation theory, Chebyshev polynomials, matrix eigenvalue theory, and algebraic geometry. Furthermore, the necessary concepts are very "undergraduate-accessible."

1.2. Three Key Ideas. The first key idea is the replacement of $f(x)$, the function whose root is sought, by a *proxy* $f_N(x)$, a sort of mathematical stunt double for $f(x)$, such that the roots of the proxy are easily found. Many ancient (but effective) rootfinders employ a proxy. In Newton's iteration, the proxy is a two-term (constant-plus-linear) Taylor series. The secant iteration also uses a linear polynomial as the proxy for $f(x)$, but computed by interpolation. The Cauchy and Mueller methods use the quadratic Taylor series and quadratic polynomial interpolant, respectively. Halley's iteration approximates $f(x)$ by a Padé approximant, which is the ratio of two linear polynomials. And so it goes as catalogued in Table 1.1.

The second key idea in the CPR is that, to paraphrase Shakespeare, "all the world's a polynomial," or, more precisely: Every function which is smooth on an interval can be approximated arbitrarily well on that interval by an interpolating polynomial. (This is the Weierstrass approximation theorem [29].) Interpolation on an arbitrary grid may fail due to the Runge phenomenon (see Chapter 4 of [9]), but interpolation at the roots of a Chebyshev polynomial *never* fails for a function $f(x)$ that is analytic on the interpolation interval. This discovery makes it possible to "polynomialize" transcendental equations by replacing transcendental functions by their polynomial approximations.

In the CPR, the proxy $f_N(x)$ is defined to be that polynomial of degree N which exactly matches ("interpolates") $f(x)$ at the points of a Chebyshev–Lobatto grid with $(N+1)$ points [9]. When $f(x)$ is transcendental, it is much easier to find the zeros of the proxy $f_N(x)$ than the roots of $f(x)$ directly.

The third essential idea was the invention of a reliable rootsolver for a polynomial in Chebyshev form, that is, a polynomial written as a series of Chebyshev polynomials. Remarkably, although matrix theory is seemingly far removed from polynomial rootfinding, the desired polynomial *zeros* are simply the *eigenvalues* of a matrix, the so-called Chebyshev–Frobenius companion matrix, whose elements are trivial functions of the Chebyshev coefficients of the polynomial.

The Chebyshev companion matrix was first discovered nearly fifty years ago and then independently rediscovered at least five times since, but it was only after the most recent round of rediscoveries that it became the cornerstone of the Chebyshev-proxy method for solving transcendental equations.

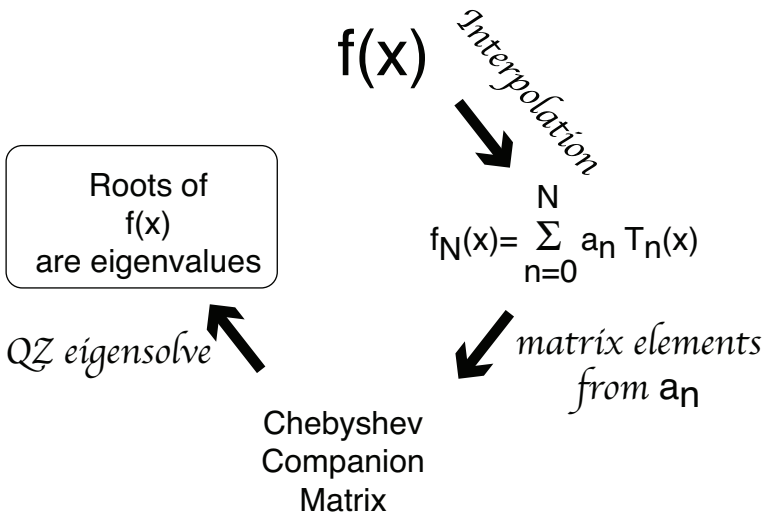

Fig. 1.1 *A schematic of the CPR. The user must specify an interval $x \in [a, b]$ and error tolerance ϵ. The degree of the interpolating polynomial N can be adaptively determined to approximate $f(x)$ on the interval within the user-chosen tolerance ϵ. The elements of the Chebyshev companion matrix, given in Appendix B, are trivial functions of the Chebyshev coefficients a_n. Those eigenvalues of the matrix that lie on $[a, b]$ are highly accurate approximations to roots of $f(x)$ on the interval.*

Figure 1.1 summarizes the algorithm.

In the next section, we briefly discuss rootfinders in general. In the rest of the article, we will develop the key ideas of the CPR. Since slow, careful treatments are available in other places (see [9, 34, 31, 51]), the mechanics of Chebyshev interpolation and the formulas for the Chebyshev–Frobenius matrix are condensed into brief appendices.

2. Graphs, Local Iterations, and Failure-to-Find: A Taxonomy of Rootfinders. Rootfinding algorithms are as numerous as the stars in a dwarf galaxy. Comparing every published method with every other would sorely tax even the encyclopedic energies of Didymus of Alexandria (63 BCE–10 CE), who earned his nickname "Chalkenteros"—"Bronze-Guts"—by writing more than 3000 books. Our goal, rather, is to stress ideas and show how the common clay of interpolation and matrix algebra can yield a very powerful algorithm that scores high on all four dimensions of the merit-of-rootfinders classification scheme discussed next.

Algorithms and software for computing zeros can be judged by a four-dimensional classification scheme. The axes of reliability and cost/efficiency are common to all library subroutines and the algorithms that animate them. The other two dimensions are peculiar to rootfinders. One is the axis distinguishing local, find-one-root-at-a-time algorithms from global methods that compute all zeros in a region simultaneously. The fourth axis is the amount of required, user-supplied information. (A fifth dimension, simplicity of programming, is important for many types of software, but since rootfinders are usually library software rather than user-written codes, we shall not discuss ease of programming here except to note that the CPR requires only a few lines of MATLAB.)

The CPR method is good according to all four classifications. It needs no user-supplied information except the search interval. This may be very large; however, automatic adaptation and subdivision, described later, keep the cost modest. The CPR is global, calculating all zeros on the given real search interval. The algorithm has a very low failure rate.

An engineer's crude but effective first line of attack on a transcendental equation is the MAGS algorithm: Make-A-Graph-Stupid. As direct as Alexander the Great untying the Gordian Knot with his sword, MAGS is more practical than a bushel of theorems in identifying the number and approximate location of well-separated roots. Unfortunately, it is not entirely reliable. A computer screen is only a thousand pixels wide. Pairs of closely spaced roots—what an atomic spectroscopist would call a "doublet"—can easily hide between the pixels.

Householder's monograph on solving a single equation in a single unknown [40] is a rich catalogue of iteration schemes for finding roots of a transcendental equation, one at a time. Bisection is the simplest: if $f(a)$ and $f(b)$ are of opposite sign, then there must be a zero on $x \in [a, b]$ according to the "Bolzano Oracle" theorem. Split the interval in two, define the new interval to be whichever half of $[a, b]$ contains a sign change of $f(x)$, and repeat until the root is imprisoned within an acceptably small interval. Newton's iteration is another classic: replace an approximate root $f(x_c)$ by

$$x_* \approx x_c - f(x_c) \Big/ \frac{df}{dx}(x_c). \tag{2.1}$$

The trouble is that local, one-root-at-a-time iterations can be fooled.

Bisection, for example, can be initialized by evaluating $f(x)$ on a very fine grid with points x_j. Every interval $[x_j, x_{j+1}]$ which is such that $f(x_j)$ is opposite in sign to $f(x_{j+1})$ is guaranteed to contain at least one root—but the interval might contain *more*. A "root doublet" is challenging when the roots are very close together. For example,

$$f(x; \delta) \equiv x^2 - \delta^2 \tag{2.2}$$

has two roots at $x = \pm\delta$. Since δ can be arbitrarily small, one needs an arbitrarily fine sampling of $f(x)$ to guarantee the detection of the tiny interval $x \in [-\delta, \delta]$, where $f(x; \delta) < 0$.

Newton's iteration will diverge, converge to a distant zero, or orbit endlessly in a limit cycle if the first guess is insufficiently close to the root, but that is not the big issue. A more serious flaw is that one-root-at-a-time methods like bisection and Newton's iteration can easily miss roots. (A published embarrassment of the author's—omitted zeros—is corrected in [4].) A local iteration is like an antisubmarine homing torpedo: only good after one has already identified a target.

Rootfinders can be promoted to higher states in our classification of merit by combining ideas. The author's procedure for inversion of the incomplete elliptic integral of the second kind requires no information from the user except the arguments of the function, even though the inversion engine is Newton's iteration, because I found (with considerable effort!) an analytic initialization which gives fast convergence throughout all of parameter space [18].

Polynomial equations are exceptional, much easier to solve reliably than transcendental equations. The fundamental theorem of algebra guarantees that a polynomial of degree N has precisely N roots, counting multiplicities, and there is a plethora of bounds, transformations, and reductions that apply only to polynomials. When

a transcendental function can be "polynomialized," that is, well approximated by a polynomial, a hard task is made easy.

3. Chebyshev Interpolation: The Shamrock Principle. Polynomial interpolation is neither a new idea nor a deep one. Every small child who has drawn a "connect-the-dots" picture in a coloring book has performed piecewise linear interpolation, approximating a teddy bear by line segments scrawled in crayon.

The Science Museum in London has a lovely nineteenth century conglomeration of brass pointers and gears called a parabolagraph, a sort of a conjoined pair of compasses: Move the three awl-like arms to three points on a curve, and the fourth arm-with-pencil will draw the interpolating parabola, that is, the unique quadratic polynomial through the three points.

Lagrange showed in the eighteenth century that the unique polynomial interpolant of arbitrary degree N through an arbitrary set of $(N+1)$ points $\{x_k\}$ is

$$f(x) \approx f_N(x) \equiv \sum_{j=0}^{N} f(x_j) \frac{\prod_{k\neq j}^{N+1} (x - x_k)}{\prod_{k\neq j}^{N+1} (x_j - x_k)}. \tag{3.1}$$

There is a very powerful idea embodied in interpolation. To the extent that the interpolant is an accurate approximation, the *continuous* function $f(x)$ can be replaced by the *discrete* set of $(N+1)$ samples, $\{f(x_j)\}$, its "grid point values," in the sense that the *continuous* function can be reconstructed solely from this discrete set of numbers.

Unfortunately, interpolation at arbitrary points, or even at uniformly spaced points, does *not* always yield a good approximation even for very large N. When the grid points are evenly spaced, Runge showed at the turn of the twentieth century that polynomial interpolation of $f(x) = 1/(1+x^2)$ on $x \in [-5, 5]$ *diverges* as $N \to \infty$, the "Runge phenomenon," illustrated by Figure 4.3 on page 84 of [9].

However, the interpolation points do not need to be evenly spaced. In 1938, Cornelius Lanczos showed that the Runge phenomenon could be utterly defeated by interpolating at the roots of the Nth Chebyshev polynomial instead [45].

Lanczos was part of that group of turn-of-the-century Hungarians including John von Neumann, George Pólya, Edward Teller, the physics Nobelists Eugene Wigner and Maria Goeppert Mayer, and others who were so brilliant that it was joked that they were "really aliens disguised as Hungarians." Lanczos was the quiet one in the sense that he had few collaborators, established no school, and spent his last years in respected obscurity in Dublin. He did not win a Nobel Prize in physics. One of his few close colleagues was Einstein, though, and his work in physics was far more influential than that of many Nobelists. The huge symposium held in honor of his centenary [22] sprawls across a vast array of arenas—general relativity, quantum mechanics, matrix methods, interpolation, Fourier theory, etc.—because *he* did, as catalogued in his six-volume collected works [30].

For our purposes, though, his masterpiece was the Lanczos Thunderbolt: the 1938 paper, more than a hundred pages long, which is the origin of the vast and widely used family of numerical algorithms known as pseudospectral methods, and much more besides: in T. J. Rivlin's words, "a mighty sequoia, with branches penetrating areas of digital computation from solutions of differential equations to numerical linear algebra to approximation theory and filter design" (page 3-310 of [30]). Lanczos showed that not only could the Runge phenomenon be defeated by Chebyshev interpolation, but also that the interpolant could be written in "Chebyshev form" as a truncated

Chebyshev series:

$$f_N(x) = \sum_{j=0}^{N} a_j\, T_j(x), \tag{3.2}$$

where the Chebyshev coefficients $\{a_j\}$ can be obtained by multiplying the grid point values $f(x_k)$ by a square matrix of size $(N+1)$. (The mechanics and matrix elements are given in Appendix A.) The interpolation points x_j and the matrix elements are cosine functions [9], as follows from Chebyshev's nineteenth century identity:

$$T_n(\cos(t)) \equiv \cos(nt), \tag{3.3}$$

where T_n is the Chebyshev polynomial of degree n. Later, it was discovered that the transform $\{f(x_j)\} \to \{a_j\}$ can be done even faster, in only $O(N \log_2 N)$ operations, using the discrete cosine transform, a variant of the fast Fourier transform (FFT).[1]

The Russian mathematician Sergei Bernstein (1880–1968) had already proved that the Chebyshev series of a function analytic everywhere on an interval $x \in [a, b]$ converges *geometrically* fast [3]. That is, the error in truncating the series after the Nth term falls as $O(\exp(-N\mu))$ for some positive constant μ. (We shall be more precise about μ in a later section, but a full discussion is given in [16], Chapter 2 of [9], and [52].) The difference between the explicit form of a transcendental function and its polynomial approximation is comparable to "machine epsilon" even for rather modest N.

Chebyshev thinking reached an apotheosis of sorts in the MATLAB extension *Chebfun* developed recently by Trefethen and collaborators [53, 2, 47, 32]. MATLAB was originally created by Cleve Moler as a system for manipulating vector and matrices, morphing over the years into a full-featured programming language and visualization system. The *Chebfun* system [2] is a MATLAB extension that handles piecewise *continuous functions* and differential and integral *operators* as if they were vectors and matrices, respectively. That's because in Lanczos World, the realm of Chebyshev interpolation, functions *are* ordinary finite-dimensional vectors, and differential and integral operators are merely matrices operating on those vectors.

Chebfun approximates an arbitrary $f(x)$ on a given interval by a Chebyshev interpolant, adaptively increasing the degree until the polynomial is indistinguishable from the real thing to within machine precision. The function has thus been reduced to a vector of Chebyshev coefficients $\{a_j,\, j = 0, 1, \ldots, N\}$ or, equivalently, to a vector of $(N+1)$ grid point values $\{f(x_k), k = 0, 1, \ldots, N\}$. These vectors can then be manipulated by the usual vector-and-matrix routines that have been part of MATLAB from the beginning.

The underlying idea is that a continuous analytic function is a vector of Chebyshev coefficients in disguise. Indeed, a function wears three interchangeable masks:

- its explicit definition as a formula or the solution to a differential equation,
- its Chebyshev coefficients $\{a_j\}$, and
- its grid point values $\{f(x_j)\}$.

In spectral methods, this threefold unity is the Shamrock Principle [9]: these three representations are one function, just as the three leaves of a shamrock are a single clover as shown schematically in Figure 3.1. Lanczos wrote many essays on

[1] Lanczos and Danielson published the central idea of the FFT in 1942, more than two decades before its independent rediscovery by Cooley and Tukey [24], but even Lanczos did not appreciate how useful the FFT would be when digital computers made it possible to use really large N.

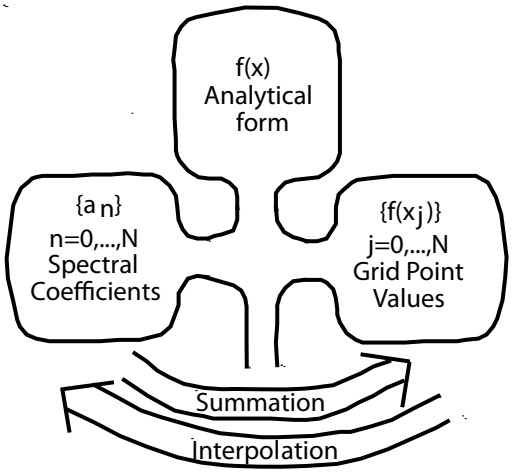

Fig. 3.1 *A visualization of the Shamrock Principle, or the trinity of spectral methods. The first N Chebyshev coefficients, $\{a_n\}$, and the set of N values on the interpolating grid, $\{f(x_j)\}$, are* equivalent *to the analytical definition of $f(x)$, represented by the top leaf, in the sense that either of these two sets allows one to compute $f(x)$ to arbitrarily high precision for sufficiently large $N \to \infty$.*

his own Jewish culture, history, and religion, but he spent his last quarter century among the Irish. He would not have been disconcerted to find his thinking visualized by the small clover that St. Patrick used to explain the concept of the Trinity—Three Persons but One God—to the people of Ireland.

To a pure mathematician, of course, this equating of the discrete with the continuous is anathema; to continue the religious metaphor, most mathematicians are Unitarians. However, the error in approximating $f(x)$ by a Chebyshev interpolant falls as $\exp(-\mu N)$ for some $f(x)$-dependent constant $\mu > 0$, as long as $f(x)$ is analytic on the interval $x \in [-1, 1]$. In the quantized world of the microchip, where reality is measured not continuously but in units of "machine precision," roughly 2×10^{-16} in MATLAB and IEEE standard double precision, the world view is decidedly Trinitarian.

4. Adaptive Chebyshev Interpolation. Lanczos' 1938 paper is lacking only in one respect: a systematic procedure for increasing the degree N until the error $|f(x) - f_N(x)|$ is everywhere less than some desired tolerance ϵ. For special cases, this is not an issue. The Kepler equation of celestial mechanics and fluid mechanics, for example, is $x - \epsilon \sin(x) - \lambda = 0$. Because the coefficients of the Chebyshev series of $\sin(x)$ are Bessel functions and the error in a truncated Chebyshev series is bounded by the sum of the absolute values of all neglected coefficients [34, 9], it is easy to show that if the Kepler equation is "polynomialized" by replacing the sine function by its Chebyshev approximation of degree fifteen,

$$\begin{aligned} -\lambda + \pi y - \epsilon \big\{ & 3.14y - 5.16771277519855y^3 + 2.55016394839721y^5 \\ & - 0.59926386322604y^7 + 0.08214347708860\,y^9 - 0.00736564609504y^{11} \\ & + 0.00046097562573\,y^{13} - 0.00001877013878y^{15} \big\} = 0, \end{aligned}$$

where $y \equiv x/\pi$, the single real root of this polynomial equation on $y \in [-1, 1]$ approximates the root of the Kepler equation to better than ten decimal places for all

$\epsilon \in [0,1]$, $\lambda \in [0,\pi]$ [15]. But the infinite Chebyshev coefficients of $f(x)$ are known only in special cases: how can we check the accuracy of the "polynomialization" of general transcendental functions? Our recommended procedure exploits an observation of Clenshaw and Curtis [23]: all points on the $(2M+1)$-point Chebyshev–Lobatto grid are also points on the $(M+1)$-point grid. Thus, if the number of grid points is increased with $N = M+1, 2M+1, 4M+1, \ldots$ for some positive integer M, all samples of $f(x)$ on coarser grids can be *reused* on the finer grid, which is very economical.

The "grid point representation," $\{f(x_j)\}$, generates through interpolation the coefficients $\{a_n\}$ of the "spectral" representation. Summation of the Chebyshev series transforms $\{a_n\} \to \{f(x_j)\}$.

The second point is that because $|T_n(x)| \leq 1$ for all $x \in [-1,1]$, as is obvious from its trigonometric definition, $T_n(\cos(t)) = \cos(nt)$, the magnitude of a truncated Chebyshev series $f_N(x)$ is bounded by the sum of the absolute values of its coefficients:

$$\max_{x\in[-1,1]} |f_N(x)| \leq \sum_{n=0}^{N} |a_n|. \tag{4.1}$$

Together, these ideas allow the following adaptive procedure.

First, specify an error tolerance ϵ, choose an initial polynomial degree N, and then calculate the polynomial interpolant $f_N(x)$ that approximates the target $f(x)$ using the procedure of Appendix A. The second step is to calculate $f_{2N}(x)$ and the correction polynomial

$$\delta_{2N}(x) = f_N(x) - f_{2N}(x) = \sum_{n=0}^{2N} d_n T_n(x). \tag{4.2}$$

If

$$\sum_{n=0}^{2N} |d_n| \leq \epsilon, \tag{4.3}$$

then stop. If not, double the degree and repeat until the difference between two successive interpolants (more precisely, the upper bound on the difference) is less than the desired tolerance.

Why is this stopping criterion reasonable? Because a geometric rate of convergence is typical for Chebyshev series of analytic functions, it follows that if $|f_N(x) - f_{2N}(x)|$ is less than a small number ϵ, then $|f(x) - f_{2N}(x)|$ will be tiny compared to ϵ.

It is not possible to guarantee that adaptation will succeed for all $f(x)$; one could, for example, construct a function whose first million Chebyshev coefficients are zero. The nonzero high degree coefficients would be aliased to low degree coefficients, but differently for different N, forcing the adaptive scheme to keep increasing N until $N >$ 1,000,000. However, one could in principle adjust the coefficients so that f_N and $f_{2N}(x)$ are the same, triggering a premature stop. However, this is very artificial and unlikely in applications.

In practice, a variety of simpler N-is-enough criteria have also been applied successfully [2, 41, 23].

To reduce the cost of later steps in CPR, a final refinement is to impose an upper limit N_{max} on N such that the search interval is automatically split in two whenever adaptation requests $N > N_{max}$. This is easily programmed and is standard in *Chebfun*.

The crucial point is that adaptive Chebyshev interpolation, reliable for all but weird, contrived $f(x)$, is now a reality. One can approximate a smooth $f(x)$ by a polynomial with close to machine precision.

5. The Radical Failure of Radicals: Frobenius' Great Idea. The Babylonians knew how to solve a quadratic equation by taking square roots. In the 1500s, Tartaglia and Ferrari solved the cubic and quadratic equations by radicals, but there was no further progress for a quarter of a millennium. Finally, Ruffini and Abel at the beginning of the nineteenth century proved that the general quintic could not be solved by radicals. Ruffini, though, was bitterly disappointed when he sent several reprints to Carl Gauss and never received a reply. The Prince of Mathematicians, it seems, felt that the solution-by-radicals of the quintic would be too complicated to be of any practical use, even if one could be found. The cubic is much more complicated than the quadratic, and the solution to the quartic is a page full of nested radicals.

Not long after Gauss, Hermite solved the general quintic using elliptic modular functions. R. Bruce King's full-length monograph on the quintic shows the beauty and interrelationships of the relevant mathematics. However, the sheer length of his work also demonstrates that the quintic solution is not a *formula*, but a *book*, confirming Gauss's intuition [42, 43, 44]. To find the roots of polynomials of degree higher than four by a practical algorithm, it is necessary to take a completely different route.

In the last four decades, a number of reliable "black box" polynomial rootfinders have been developed and incorporated into popular software libraries. Without casting aspersions on the Traub–Jenkins method and other good schemes, we will concentrate on the "companion matrix" strategy, because this shows the deep connections between seemingly distant branches of mathematics, is the default solver in MATLAB, and, most important, has been generalized to the Chebyshev coefficients of a polynomial.

Matrix theory did not really begin until the latter half of the nineteenth century, after Gauss, Galois, Ruffini, and Abel were all dead. Late in the century, Georg Frobenius discovered something remarkable. For any polynomial, one could find a matrix with the amazing property that the *eigenvalues* of the *matrix* are also the *zeros* of the polynomial. The elements of this "Frobenius companion matrix" are mostly zeros or ones with a final row composed of the ratios of the lower degree coefficients to that of the highest power. The problem of nonlinear rootfinding is really a problem in linear algebra.

A polynomial in the form of a truncated Chebyshev series can always be converted into the usual "monomial" or "power" form by a matrix multiplication acting on the vector of Chebyshev coefficients. Unfortunately, Gautschi [35] showed that error in this conversion grows as $(1+\sqrt{2})^N$, where N is the degree of the polynomial [8]. This means that the conversion is acceptable for N less than twenty, say, but hopelessly inaccurate in floating point arithmetic for $N > 60$.

This ill-conditioning is not a peculiarity of the conversion alone, but rather reflects an unpleasant reality: the familiar "power" or "monomial" form, beloved of high school algebra teachers and pure mathematicians,

$$f_N = \sum_{j=0}^{N} b_j \, x^j, \tag{5.1}$$

is a very ill-conditioned way of representing a polynomial. In rational arithmetic in a high school textbook or a computer symbolic manipulation system like Maple,

algebra is, well, *exact*, and thus the student is shielded from the distasteful truth that in floating point arithmetic, roundoff errors for the power form grow exponentially with N.

In contrast, the Chebyshev form is always well-conditioned, and the MATLAB extension *Chebfun* (discussed earlier) routinely and accurately manipulates a truncated Chebyshev series whose degree is in the thousands. But how can one find the roots of a polynomial in Chebyshev form?

The answer is to find the eigenvalues of a matrix, the "Chebyshev–Frobenius" matrix, whose elements are simple functions of the Chebyshev coefficients a_j as given explicitly in Appendix B. The Chebyshev–Frobenius matrix was independently discovered by Specht [49, 50] in 1957, Good [39] (who called it the "colleague matrix") in 1961, Barnett [1] and Gol'berg and Malozemov [38] in the mid-1970s, and again by Day and Romero [31] and Stetter [51]. As in Frobenius's original idea, the companion matrix is a sparse matrix whose few nonzero elements are problem-independent fractions or the ratios of Chebyshev coefficients. Barnett and Day and Romero used recurrence relation ideas from the general theory of orthogonal polynomials; Stetter employed the algebra of a quotient ring in algebraic geometry to derive the same matrix. A brief derivation using undergraduate-level means is provided as Appendix C.

The Chebyshev–Frobenius matrix method is a very accurate and well-conditioned rootfinder [31, 19, 13]. However, the cost is about $10N^3$ floating point operations. Boyd [8, 10] suggested subdividing the interval into subintervals and approximating $f(x)$ by separate but lower degree Chebyshev interpolants on each subdomain as mentioned previously. Boyd [11, 12] provided theorems and experiments that put subdivision on a solid footing.

6. The Chebyshev-Proxy Algorithm for Computing the Zeros of a Transcendental Equation: A Recapitulation. For an algorithm four millennia in the making, the final procedure is remarkably simple. To compute the real-valued roots of $f(x)$ on an interval $x \in [a, b]$:

1. Adaptively approximate $f(x)$ by Chebyshev interpolants of increasingly high degree N until $|f(x) - f_N(x)|$ is less than a preset tolerance ϵ everywhere on $x \in [a, b]$.
2. Compute the roots of $f_N(x)$ as the eigenvalues of the Chebyshev–Frobenius companion matrix (if N is small or moderate); if N is large, subdivide the interval and apply the Chebyshev-proxy scheme with moderate N on each subinterval.
3. Optionally, the roots of the proxy f_N can be refined by one or two Newton iterations of $f(x)$, which we dub "Newton-polishing."

The only restriction is that $f(x)$ must be a nice function on $x \in [a, b]$ in the sense that it possesses a rapidly convergent Chebyshev series on the interval as quantified in the next section. If $f(x)$ is analytic on the entire interval, then the error in truncating the Chebyshev series after the coefficient of $T_N(x)$ falls *geometrically* with N, and then a polynomial approximation $f_N(x)$ of modest degree N will be a very good proxy for the function.

A good idea is usually only half a good paper. The Chebyshev-proxy algorithm is a combination of at least three good ideas: rootfinding-by-proxy, Chebyshev interpolation, and a method for finding the roots of a polynomial in Chebyshev form. But even these three concepts are not enough. Ideas must be understood deeply and tested, and one must compile a dossier, as thorough as a legal brief or a spy's false identity before one can write an article that will sell others on its virtues. The goodness of *Romeo and*

Juliet is not the goodness of the theme of young love thwarted, but rather the genius of Shakespeare's instantiation of it. However, the articles [8, 10, 11, 12, 31, 19, 13] have filled in the necessary details. In the next section, we analyze difficulties, cures, and generalizations.

7. Resolving Complications.

7.1. When $f(x)$ Isn't Smooth. In engineering reality, often $f(x)$ isn't analytic everywhere on the desired target interval. What then?

First, if the singularities on $x \in [a, b]$ are "weak" in the sense that the first K derivatives exist and are bounded, then the $(N+1)$-point Chebyshev interpolation error falls proportionally to $1/N^K$ if the singularity is on the interior of the interval, or as $1/N^{2K-1}$ if at an endpoint, $x = a$ or $x = b$ (see Chapter 2 of [9]). If K is not too small, we can apply the Chebyshev algorithm without change except that the degree of the proxy will be larger than would be needed if $f(x)$ were analytic.

Second, if the singularities are too strong to ignore, one can split the interval into subintervals at each singularity. If $f(x)$ is *piecewise* analytic, analytic everywhere on each subinterval $[a, b]$ *including the endpoints*, and nonanalytic only on a larger interval because of a simple discontinuity or a discontinuity of slope, Chebyshev interpolation will converge exponentially fast on each subinterval. In other words, subdivision removes the convergence-killing effect of the discontinuity if the subdomain wall is placed at the discontinuity. If $f(x)$ has a branch point at either $x = a$ or $x = b$, but only derivatives are unbounded at the singularity, an exponential change-of-coordinate near each endpoint-that-is-a-branch-point can then recover an exponential rate of convergence [5, 9].

7.2. Slightly Complex Zeros. The weakness of Chebyshev interpolation on an interval $x \in [a, b]$ is that the accuracy of the approximation decays *exponentially* outside this interval for either real or complex x. More precisely, define a new coordinate z such that $x \in z(b-a)/2 + (b+a)/2$, thus mapping the interval $[a, b]$ in x to $z \in [-1, 1]$, and then define elliptical coordinates in the complex z-plane by

$$\begin{aligned} \Re(z) &= \cosh(\mu)\cos(\eta), \qquad \mu \in [0, \infty], \\ \Im(z) &= -\sinh(\mu)\sin(\eta), \qquad \eta \in [0, 2\pi]. \end{aligned} \tag{7.1}$$

The curve of constant elliptical coordinate μ is an ellipse with foci at $z = \pm 1$ with semimajor axis $\cosh(\mu)$ and semiminor axis $\sinh(\mu)$.

If $f(z)$ is free of singularities everywhere inside the ellipse $\mu = \tilde{\mu}$, but is singular somewhere on this ellipse, then the error in the Chebyshev series, truncated after the Nth term, will fall proportionally to $\exp(-N\tilde{\mu})$ on the real interval $z \in [-1, 1]$ [9, 16, 52].

Without approximation,

$$T_n(z) = \cos(n[\eta - i\mu]) = \cosh(n\mu)\cos(n\eta) - i\sinh(n\mu)\sin(n\eta), \tag{7.2}$$

which implies

$$|T_n(z)| = \sqrt{\cosh^2(n\mu)\cos^2(\eta) + \sinh^2(n\mu)\sin^2(n\eta)} \tag{7.3}$$

$$= \frac{1}{2}\sqrt{\exp(2n\mu) + \exp(-2n\mu) + \cos(2n\eta)} \tag{7.4}$$

$$\approx (1/2)\exp(n\mu), \qquad n\mu \gg 1. \tag{7.5}$$

This implies that along the ellipse with elliptical coordinate μ, the nth term of the Chebyshev series is no longer bounded by a_n but rather by $a_n \exp(n\mu)/2$. As we move further in any direction from the canonical approximation interval $z \in [-1, 1]$, the error of the truncated Chebyshev series will increase exponentially and will usually diverge for sufficiently large μ.

The implication is that in theory (and in *Chebfun*), Chebyshev interpolants can be used to find *slightly* complex roots, by which we mean roots whose imaginary parts are small.

In any event, to exclude the roots that are unreliable because they are too far from the Chebyshev target interval, $x \in [-1, 1]$, the companion matrix eigenvalues should be accepted only if they pass the dual tests

$$|\Re(x_j)| \le 1 + \sigma, \qquad |\Im(x_j)| \le \tau, \tag{7.6}$$

where σ and τ are tiny, user-choosable nonnegative constants. A small positive value for τ such as 10^{-6} is helpful to detect a real-valued root doublet which has been perturbed into a complex conjugate pair with tiny imaginary parts. Newton's iteration can refine genuine roots to high precision, or tag poor approximations as spurious, so there is some flexibility in excluding matrix eigenvalues that are outside the real interval $[-1, 1]$.

The rootfinding-by-polynomial-proxy idea can be extended to search for roots within a target region in the complex plane. Geddes and Mason showed that for approximation within a *disk*, interpolation on points evenly spaced around the boundary circle of the disk has the same optimum properties as does Chebyshev interpolation on a real interval [37]. The disk or disks must be sufficiently small to be free of singularities (poles, branch points, etc.) of $f(x)$. One flaw is that overlapping disks do not tessellate the complex plane efficiently. Disks can be replaced by ellipses by interpolating at points on the ellipse [36] (not a line segment inside the ellipse) and by more general regions by using Faber polynomials [54, 28].

7.3. Expense and Subdivision. The QZ algorithm for computing the eigenvalues of a matrix, the default in MATLAB, has a cost which is proportional to the *cube* of the dimension of the matrix. The *Chebfun* system routinely employs $N > 1000$, but its rootfinding is relatively inexpensive, because it automatically replaces a single expansion on $[a, b]$ by subdividing the interval into several subintervals, computing lower degree Chebyshev interpolants on each subinterval, and finding the roots on the original interval as the *union* of the zeros on the collection of subintervals. This greatly reduces cost because with M subdomains, the number of floating point operations is $O(M(N/M)^3)$, roughly M^2 cheaper than using a single Chebyshev interpolant of large degree N [20]. As noted earlier, it is easy to append automatic subdivision to automatic adaptation of the degree N by splitting the interval in two whenever the Clenshaw–Curtis strategy calls for N larger then some user-specified limit N_{max}.

7.4. Dynamic Range. Subdivision can solve another difficulty. Denote machine precision by $\epsilon_{machine}$. Define the maximum of $f(x)$ on the interval as $f_{max} \equiv \max_{x\in[a,b]} |f(x)|$. If a function $f(x)$ has on a subinterval within $[a, b]$ (excluding tiny subintervals around a zero) a magnitude as small as $|f(x)| \le O(\epsilon_{machine} f_{max})$, then $f(x)$ has, to borrow a term from photography, a large "dynamic range." An example on $x \in [-1, 1]$ is

$$f(x) = \exp(-25(x-1)^2)\,\sin(10\pi x), \tag{7.7}$$

whose magnitude varies wildly from the neighborhood of $x = 1$, where $f(x)$ is $O(1)$, to near $x = -1$, where $f(x)$ is $O(\exp(-100))$. A useful pragmatic criterion for a large dynamic range is that two or more adjacent samples of $f(x)$, $f(x_j)$, and $f(x_{j+1})$ are $O(\epsilon_{machine} f_{max})$.

Photographers must battle a similar "dynamic range" problem in the sense that it is difficult for a single photograph to capture a scene which contains portions in bright sunlight and also areas of deep shadow. If the shadows are captured by a long exposure, the bright areas will be solid white without detail, but a short exposure renders the shadows as solid black. The only completely satisfactory strategy is to portray the scene using *multiple* photographs with *different exposures*.

Similarly, subdividing the interpolation interval into subintervals such that $f(x)$ varies by only a few orders of magnitude on each subinterval eliminates the dynamic range difficulty. Because machine precision is 2.2×10^{-16} in MATLAB, for example, the dynamic range problem arises for Chebyshev interpolants only occasionally, but there are nontrivial examples. The first and second Wilkinson polynomials, defined as $W_1 = \prod_{k=1}^{20}(x-k)$ and $W_2 = \prod_{k=0}^{19}(x-1/2^k))$, are Chebyshev-analyzed in [8] and [33], respectively.

7.5. Roots on an Infinite Real Interval. If $f(y)$ is nonoscillatory for large y and thus has only a *finite* number of real zeros, one can find a proxy valid on all of $y \in [-\infty, \infty]$ by making the change of coordinate $y = Lx/\sqrt{1-x^2}$ for some positive constant L so as to map the infinite interval in y into $x \in [-1, 1]$ [7, 6, 9]. We can then apply the Chebyshev rootfinding procedure in x and invert the map to translate the roots x_k back to the original coordinate y.

As an example, we attacked

(7.8)
$$f(y) = \exp(-0.5y^2)(12 - 48y^2 + 16y^4) \;\; \rightarrow \frac{df}{dy} = -4y\exp(-0.5y^2)(27 - 28y^2 + 4y^4).$$

After mapping $y \in [-\infty, \infty]$ to the finite interval, we expanded $f(y[x])$ as a Chebyshev series of degree 60.

The map parameter L is usually chosen experimentally (but see [7, 6]); the user should pick the value of L that gives the fastest rate of Chebyshev convergence. Here, $L = 4$ was a good choice. It is not necessary to carefully optimize L; usually $L = 2L_{optimum}$ and $L = (1/2)L_{optimum}$ are nearly as good as the optimal L.

As illustrated in Figure 7.1, the complication ensues because the Hermite function $f(y)$ decays exponentially fast as $|y| \rightarrow \infty$, and this implies that $f(y[x])$ is exponentially tiny near the endpoints $x = \pm 1$ as evident in the zoom plot in the lower right of the figure. Although the Chebyshev expansion has tiny *absolute* errors over the whole interval, the *relative* errors are large near the endpoints. The result is that although $f(y)$ has only four real-valued zeros and no complex roots at all, its Chebyshev proxy has 22 real-valued zeros on $x \in [-1, 1]$.

The 18 spurious solutions can be readily identified by "Newton-polishing." The upper panel in the figure shows the absolute values of the of the corrections δ in the first Newton iteration to each of the roots. Four corrections do not appear because they are MATLAB "NaN"—Not-a-Number. The other spurious roots have Newton corrections which are $O(1)$. In contrast, the two smaller genuine roots have Newton corrections of only 1.7×10^{-10}, which are also the errors in the Chebyshev approximation of these roots; the two larger genuine zeros have errors and corrections of only 5.5×10^{-10}.

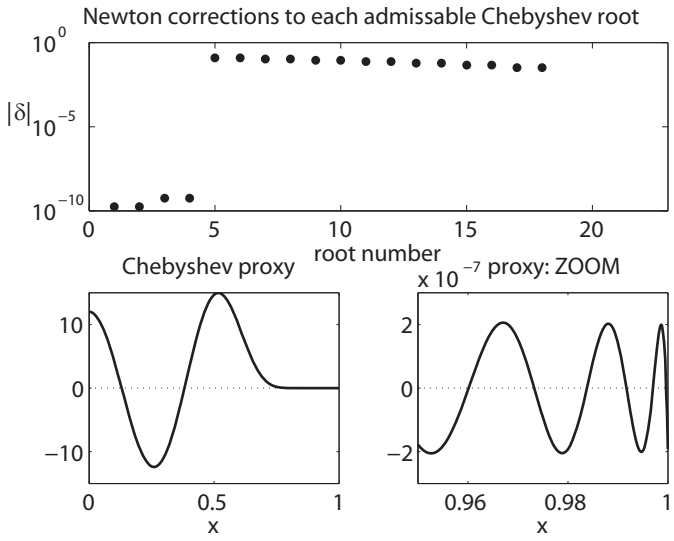

Fig. 7.1 *Lower left: $f(y[x])$. Only the right half of $x \in [-1, 1]$ is shown because the function is symmetric about the origin. Lower right: same but a zoom plot showing just $x \in [0.95, 1]$ and graphing the Chebyshev approximation. Note that the maximum of $f(y[x])$ on the zoom plot is only 2×10^{-7}, although the global maximum (lower left panel) is about 15. Upper plot: the Newton corrections $|\delta|$ for each root easily distinguish the many spurious zeros (large $|\delta|$) from the four genuine roots (tiny $|\delta|$).*

These spurious roots are a manifestation of the dynamic range difficulty; a highly accurate Chebyshev expansion is likely to have spurious zeros in regions where the amplitude of the function is many orders of magnitude smaller than its peak. It is fortunate that such spurious roots can be so easily identified and rejected by using Newton's iteration. If, in contrast to this example, there is an *infinite* number of roots, one can often find all the roots by combining CPR with an *asymptotic approximation*, as illustrated in the next section.

8. Additional Numerical Examples.

8.1. Roots of $x\,\tan(x) - 1$. Rootfinding can never *safely* rely on automated "black box" solvers; in the words of Field Marshal the Viscount Sir William Slim: "Thinking is the cheapest and one of the most effective long-range weapons." We can illustrate both the strengths and the limits of the Chebyshev rootfinder by finding the zeros of

$$f(x) \equiv x\,\tan(x) - 1. \tag{8.1}$$

First, $f(x)$ is infinite at the poles of $\tan(x)$, and this wrecks Chebyshev interpolation. However, the function

$$g(x) \equiv \cos(x) f(x) = x\,\sin(x) - \cos(x) \tag{8.2}$$

has the same roots as $f(x)$ but is analytic on the entire real axis and thus can be expanded in a rapidly convergent Chebyshev series on any finite real interval. The two functions are illustrated in Figure 8.1.

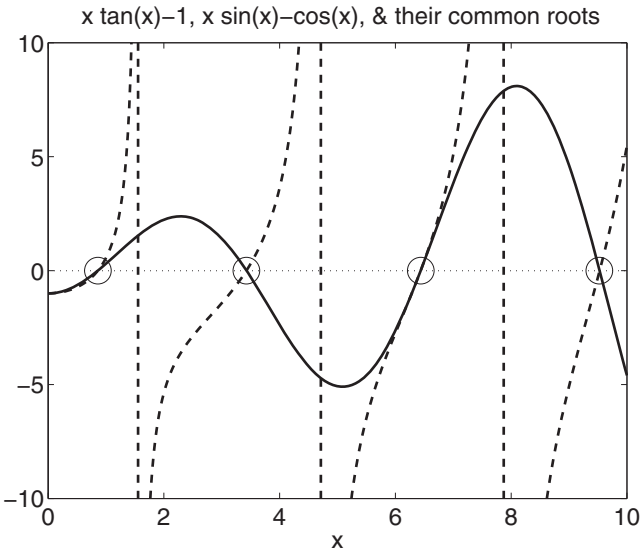

Fig. 8.1 *Graphs of* $f(x) = x\tan(x) - 1$ *(solid) and* $g(x) = x\sin(x) - \cos(x)$ *(dashed). Their zeros are everywhere the same; the four common roots on* $x \in [0, 10]$ *are circled.*

The second point is that if x_* is a root, then so is $-x_*$, as follows from the property that

$$f(-x) = -x\,\tan(-x) - 1 = f(x) \tag{8.3}$$

because $\tan(-x) = -\tan(x)$. Thus, it is sufficient to restrict our search to *positive* roots only.

An automated procedure still cannot obtain all the real roots of $f(x)$ because these are *infinite* in number. However, for large x, one of the terms in $g(x)$, $x\sin(x)$, is very large compared to the other term, $\cos(x)$, except very close to the roots of the sine function. It follows that for large $|x|$, the roots must be *near* those of the sine function, and this makes it possible to derive an asymptotic series for the large roots.

Define $\delta(n)$ by writing, with x_n denoting the root near the nth positive root of the sine,

$$x_n \equiv n\pi + \delta(n). \tag{8.4}$$

Using the identities $\sin(x) = \sin(n\pi + \delta) = \cos(n\pi)\sin(\delta)$ and $\cos(x) = \cos(n\pi + \delta) = \cos(n\pi)\cos(\delta)$, $g(x)$ becomes

$$g(x) = \cos(n\pi)\left\{(n\pi + \delta)\sin(\delta) - \cos(\delta)\right\}. \tag{8.5}$$

For sufficiently large n, $|\delta(n)| \ll 1$. Expanding δ as a series of negative powers of n and matching powers of n allows the coefficients to be calculated to whatever order is desired (see Figure 8.2):

$$x_n = n\pi + \frac{1}{\pi}\frac{1}{n} - \frac{4}{3\pi^3}\frac{1}{n^3} + \frac{53}{15\pi^5}\frac{1}{n^5} + O\left(\frac{1}{n^7}\right). \tag{8.6}$$

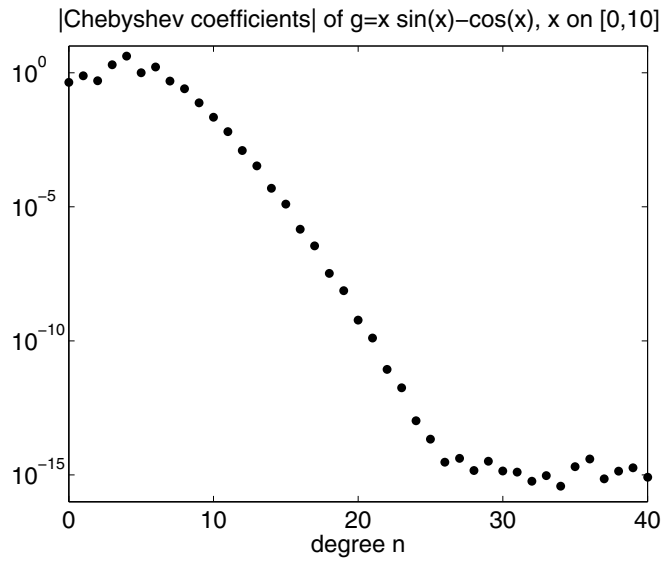

Fig. 8.2 *Absolute values of the Chebyshev coefficients* $|a_n|$ *of the function* $g(x) = x\,\sin(x) - \cos(x)$ *for the expansion on* $x \in [0, 10]$. *The coefficients fall to a magnitude of roughly* $\max|g(x)|$ *multiplied by machine epsilon, which is* 2.2×10^{-16} *in MATLAB, within the first* 25 *coefficients and then plateau. This "roundoff plateau"(see Chapter* 2 *of* [9]*) would be replaced by continued descent if the floating point precision were increased.*

The expansion through n^{-5} gives x_4 to an absolute error of only 2.30×10^{-7}, so the problem is reduced to approximating the first four roots. Since $x_3 \approx 9.4$, we therefore compute the Chebyshev interpolant on $x \in [0, 10]$. The 18-point interpolant is

$$\begin{aligned} g(x) \approx\ & 0.043727769 + 0.7738004103T_1(x/5-1) + 0.5091114626T_2(x/5-1) \\ & - 1.9877461313T_3(x/5-1) - 4.1206433584T_4(x/5-1) - 1.0075530243T_5(x/5-1) \\ & + 1.6256793844T_6(x/5-1) + 0.4910421844T_7(x/5-1) - 0.2548086862T_8(x/5-1) \\ & - 0.0761365984T_9(x/5-1) + 0.0222395898T_{10}(x/5-1) + 0.0063494104T_{11}(x/5-1) \\ & - 0.0012508791T_{12}(x/5-1) - 0.0003383213T_{13}(x/5-1) + 0.0000493403T_{14}(x/5-1) \\ & + 0.0000126223T_{15}(x/5-1) - 0.0000014441T_{16}(x/5-1) - 3.421E-7T_{17}(x/5-1). \end{aligned} \tag{8.7}$$

We observe that, as is generic when $f(x)$ is analytic on the expansion interval, the coefficients fall *geometrically*, that is, the coefficient a_n of T_n is bounded by $p\exp(-\mu n)$ for some positive constants p and μ. The nth term is bounded on the expansion interval by $|a_n|$, so it is easy to estimate the error merely by inspecting the coefficients. Choosing the polynomial degree by systematic adaptation was described in section 4.

Table 8.1 shows that the first four zeros of $x\tan(x) - 1$ are computed to an accuracy of 7×10^{-9} or better by the Chebyshev interpolant of degree 17 on $x \in [0, 10]$. To increase the accuracy, we can of course increase the degree of the Chebyshev approximation. Since the roots are already very accurate, it is preferable to "polish" the zeros by Newton's iteration or the secant iteration.

Table 8.1 *Errors in the roots of $x\tan(x) - 1 = 0$ as approximated by the roots of the degree-17 Chebyshev interpolant* (8.7).

n	Exact	Approximate	Error
0	0.8603335890	0.8603335864	2.61×10^{-9}
1	3.4256184595	3.4256184659	6.46×10^{-9}
2	6.4372981792	6.4372981762	2.92×10^{-9}
3	9.52933440536	9.5293344094	8.78×10^{-11}

Table 8.2 *Approximate roots of $(x - 1/10000)(x + 1/100000)J_0(x)$ as approximated by Chebyshev interpolants of various degrees.*

n	Exact	10th degree error	20th degree error	30th degree error
1	−5.520078110	1.4136e-03	−2.8873e-10	1.1546e-14
2	−2.404825558	8.9391e-03	−5.5956e-10	3.5527e-15
3	−0.00001	2.0223e-07	7.8818e-11	−7.2297e-11
4	0.0001	−2.0223e-07	−7.8830e-11	7.2288e-11
5	2.404825558	−8.9391e-03	5.5957e-10	4.4409e-16
6	5.520078110	−1.4136e-03	2.8874e-10	−6.2172e-15

8.2. Multiple Roots and Doublets. Denoting the usual zeroth-order Bessel function by J_0,

$$f(x) \equiv (x - 10^{-4})(x + 10^{-5})J_0(x) \tag{8.8}$$

is challenging because it has a "doublet," which is a pair of closely spaced roots. Doublets and multiple roots are inherently very sensitive to perturbation, whether by a small physical change in $f(x)$ or a numerical perturbation due to the inherent errors of numerical approximation and floating point arithmetic. For example, perturbing

$$x^4 \Rightarrow x^4 - \epsilon \tag{8.9}$$

changes a single quadruple root to four simple roots at $x = \epsilon^{1/4}\exp(ji\pi/2)$, $j = 0, 1, 2, 3$. Note that if the perturbation $\epsilon = 10^{-16}$, for example, the roots shift by 10^{-4}, which is twelve orders of magnitude larger than the perturbation!

No rootfinding algorithm can annul these harsh mathematical realities. However, the doublet-Bessel example shows that the CPR does as well as one might reasonably hope. The errors for the roots near the origin, the doublet, are larger than the other roots when the polynomial degree is 30, and increasing degree from 20 to 30 yields no improvement for the doublet, but an accuracy better than 10^{-10} is quite satisfactory for most engineering purposes!

9. Summary. The simple algorithm described above can be extended in a number of directions. The roots of a general *trigonometric* polynomial can be found by a companion matrix method [14, 17] using the "degree-doubling" algorithm that has been independently invented several times [10]; an extension to a trigonometric polynomial augmented by a nonperiodic or "secular" term is in [21]. The review article [13] gives the companion matrices for non-Chebyshev spectral series including trigonometric functions, Hermite functions, Legendre polynomials, and so on. These other expansions can be used as proxies instead of a Chebyshev interpolant. Polynomial-proxy methods can be extended to search for roots in regions of the complex plane, too.

The literature on solving a single equation in a single unknown is already vaster than the 2.5 million word Indian epic, the *Mahabharata*, and is still growing. John McNamee's online bibliography, restricted to *polynomial* equations only, is more than 900 pages long [46]. Many engineers have happily made do with Newton's iteration, perhaps augmented with pseudoarclength continuation; it would be silly to suppose that the algorithms discussed here have a monopoly on success. (In particular, Newton's iteration is outstanding if one can devise an always-convergent first guess for all the desired roots, which is often possible when $f(x)$ is a special function [18].) A narrative like this necessarily has the selectivity of a novel.

Still, after so many centuries, it is gratifying that a reliable method is finally available for finding the real zeros of a smooth transcendental function on an interval.

Appendix A. Chebyshev Interpolation of a Function $f(x)$: Chebyshev–Lobatto Grid. Goal: to 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:

$$x_k \equiv \frac{b-a}{2}\cos\left(\pi\frac{k}{N}\right) + \frac{b+a}{2}, \qquad k = 0, 1, 2, \ldots, N. \tag{A.1}$$

Step 2: Compute the grid point values of $f(x)$, the function to be approximated:

$$f_k \equiv f(x_k), \qquad k = 0, 1, \ldots, N. \tag{A.2}$$

Step 3: 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

$$\mathcal{I}_{jk} = \frac{2}{p_j\, p_k\, N}\,\cos\left(j\pi\frac{k}{N}\right). \tag{A.3}$$

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

$$a_j = \sum_{k=0}^{N} \mathcal{I}_{jk}\, f_k, \qquad j = 0, 1, 2, \ldots, N. \tag{A.4}$$

(The vector-matrix multiply can be accelerated by the FFT, but since this vector-matrix multiply costs $O(2N^2)$ while the eigensolving cost is $O(10N^3)$, the FFT is scarcely worth the bother.)

The approximation is

$$f \approx \sum_{j=0}^{N} a_j T_j\left(\frac{2x-(b+a)}{b-a}\right) = \sum_{j=0}^{N} a_j \cos\left\{j\arccos\left(\frac{2x-(b+a)}{b-a}\right)\right\}. \tag{A.5}$$

Appendix B. The Chebyshev Companion Matrix.

THEOREM B.1 (Chebyshev companion matrix). *Let $f_N(x)$ denote a polynomial of degree N written in "Chebyshev form" as*

$$f_N(x) = \sum_{j=0}^{N} a_j\, T_j(x). \tag{B.1}$$

Then all roots of $f_N(x)$, both on and off the canonical expansion interval, $x \in [-1,1]$, are eigenvalues of the $N \times N$ matrix $\vec{\vec{A}}$ whose elements are

$$A_{jk} = \begin{cases} \delta_{2,k}, & j = 1, \quad k = 1,2,\ldots,N, \\ \frac{1}{2}\{\delta_{j,k+1} + \delta_{j,k-1}\}, & j = 2,\ldots,(N-1),\ k = 1,2,\ldots,N, \\ (-1)\dfrac{a_{j-1}}{2\,a_N} + (1/2)\delta_{k,N-1}, & j = N,\ k = 1,2,\ldots,N, \end{cases} \tag{B.2}$$

where δ_{jk} is the usual Kronecker delta function such that $\delta_{jk} = 0$ if $j \neq k$ while $\delta_{jj} = 1$ for all j. (See [49, 50, 39, 1, 38, 51, 31]*.)*

For a quintic polynomial, for example,

$$\begin{vmatrix} 0 & 1 & 0 & 0 & 0 \\ (1/2) & 0 & (1/2) & 0 & 0 \\ 0 & (1/2) & 0 & (1/2) & 0 \\ 0 & 0 & (1/2) & 0 & (1/2) \\ (-1)\dfrac{a_0}{2\,a_5} & (-1)\dfrac{a_1}{2\,a_5} & (-1)\dfrac{a_2}{2\,a_5} & (-1)\dfrac{a_3}{2\,a_5} + (1/2) & (-1)\dfrac{a_4}{2\,a_5} \end{vmatrix}. \tag{B.3}$$

Corless has devised a companion matrix whose elements depend only on the values of $f(x)$ at a discrete set of arbitrary interpolation points [26, 25, 27] and has demonstrated its usefulness. However, adaptation from samples of $f(x)$ is impossible without computation of the spectral coefficients, so the Lagrange basis companion matrix will not be discussed further here.

Appendix C. Why Eigenvalues Are Roots: A Derivation of the One-Dimensional Companion Matrix for General Orthogonal Polynomials. Let $\phi_j(x), j = 0,1,\ldots,$ denote a family of orthogonal polynomials. Any polynomial of an arbitrary degree N can be represented exactly as the finite sum of the first $N+1$ basis functions. In particular, if we seek the roots of a polynomial $f_N(x)$, it can be written

$$f_N(x) = \sum_{j=0}^{N} a_j \phi_j(x) \tag{C.1}$$

for some coefficients a_j that could be found by imposing $N+1$ interpolation conditions, among other ways.

If we multiply each basis function $\phi_j(x)$ by x, the result is obviously a polynomial of degree $j+1$. This in turn can be reexpanded in terms of the basis functions to yield

$$x\phi_j(x) = \sum_{k=0}^{j+1} H_{j+1,k+1}\phi_k(x) \tag{C.2}$$

for some coefficients $H_{j+1,k+1}$.

Define a vector $\vec{\Phi}$ whose N elements are the basis functions. The expansions of the first N polynomials $x\phi_j(x)$ can be organized into an inhomogeneous matrix equation

$$\vec{\vec{H}}\vec{\Phi} = x\vec{\Phi} - H_{N,N+1}\phi_N(x)\vec{e}_N, \tag{C.3}$$

where the double arrow labels the matrix whose elements are the $H_{j,k}$, single arrows denote vectors, and $\vec{e}_N$ denotes the unit vector whose elements are all zeros except

for a one in row N. The rightmost term arises because the product $x\phi_{N-1}$ is a polynomial of degree N and thus requires $\phi_N(x)$ for its representation, but this is not one of the N basis functions in the vector $\vec{\Phi}$, which includes only the basis functions up to $\phi_{N-1}(x)$.

We can remove the extra term in ϕ_N by adding and subtracting $qf_N(x)$ to and from the last equation of the matrix system, which becomes

(C.4)
$$\sum_{k=0}^{N-1} H_{N,k+1}\phi_k(x) = x\phi_{N-1}(x) - H_{N,N+1}\phi_N(x) + q\left\{a_N\phi_N(x) + \sum_{j=0}^{N-1} a_j\phi_j(x)\right\} - qf_N(x).$$

If we choose $q = H_{N,N+1}/a_N$, the terms in ϕ_N will *cancel*, and the terms involving a_j for j up to $(N-1)$ can be absorbed into the matrix by modifying the elements of the lowest row. The only rub is the leftover term $-qf_N(x)$. If, however, x is not a general value but happens to be one of the *roots* x_r of $f_N(x)$, then $-qf_N(x_r)$ is zero and the matrix problem becomes a classic matrix eigenvalue problem with the parameter x_r as the eigenvalue. We have thus proved the following proposition.

THEOREM C.1 (generalized companion matrix). *Let the $\phi_j(x)$ be a set of polynomial basis functions such that the subscript j is the degree. Then the roots x_j of the degree-N polynomial*

$$f_N(x) = \sum_{n=0}^{N} a_n\phi_n(x) \tag{C.5}$$

are the eigenvalues of

$$\vec{\vec{M}}\,\vec{\phi} = x\vec{\phi}, \tag{C.6}$$

where the elements of $\vec{\vec{M}}$ are

$$M_{jk} = H_{jk}, \qquad j = 1, 2, \ldots N-1, \tag{C.7}$$

$$M_{Nk} = H_{Nk} - H_{N,N+1}\frac{a_{k-1}}{a_N}, \qquad k = 1, 2, \ldots N, \tag{C.8}$$

where

$$x\phi_j(x) = \sum_{k=0}^{j+1} H_{j+1,k+1}\phi_k(x) \tag{C.9}$$

for some coefficients $H_{j+1,k+1}$.

Although not proved here, it can be shown that the N eigenvalues always give the N roots if both eigenvalues and roots are counted according to the multiplicity; a double root thus appears twice in the list of eigenvalues.

The argument presented here is very similar to those of Barnett [1] and Day and Romero [31]. The proof uses almost no information about the basis functions except that the $x\phi_j$ are polynomials of degree at most $N-1$ for $j \le N-2$ and $x\phi_{N-1}$ is a polynomial of at most degree N. It is not surprising that Corless generalized the companion matrix theorem to nonorthogonal bases such as the Newton functions of interpolation theory [27] and to polynomial Lagrange ("cardinal") bases [26].

Acknowledgments. I thank one referee for returning a scanned copy of my paper, marked up with many useful corrections, and also for pointing out the companion matrix work of Corless. I thank the second reviewer for many detailed and helpful comments. I also thank Burhan Sadiq for his careful proofreading.

REFERENCES

[1] S. Barnett, *Companion matrix analog for orthogonal polynomials*, Linear Algebra Appl., 12 (1975), pp. 197–208.
[2] Z. Battles and L. N. Trefethen, *An extension of MATLAB to continuous functions and operators*, SIAM J. Sci. Comput., 25 (2004), pp. 1743–1770.
[3] S. N. Bernstein, *Quelques remarques sur linterpolation*, J. Math. Ann., 79 (1918), pp. 1–12.
[4] J. P. Boyd, *Complex coordinate methods for hydrodynamic instabilities and Sturm-Liouville problems with an interior singularity*, J. Comput. Phys., 57 (1985), pp. 454–471.
[5] J. P. Boyd, *Polynomial series versus sinc expansions for functions with corner or endpoint singularities*, J. Comput. Phys., 64 (1986), pp. 266–269.
[6] J. P. Boyd, *Orthogonal rational functions on a semi-infinite interval*, J. Comput. Phys., 70 (1987), pp. 63–88.
[7] J. P. Boyd, *Spectral methods using rational basis functions on an infinite interval*, J. Comput. Phys., 69 (1987), pp. 112–142.
[8] J. P. Boyd, *A Chebyshev polynomial interval-searching method ("Lanczos economization") for solving a nonlinear equation with application to the nonlinear eigenvalue problem*, J. Comput. Phys., 118 (1995), pp. 1–8.
[9] J. P. Boyd, *Chebyshev and Fourier Spectral Methods*, 2nd ed., Dover, Mineola, New York, 2001.
[10] J. P. Boyd, *Computing zeros on a real interval through Chebyshev expansion and polynomial rootfinding*, SIAM J. Numer. Anal., 40 (2002), pp. 1666–1682.
[11] J. P. Boyd, *Computing real roots of a polynomial in Chebyshev series form through subdivision*, Appl. Numer. Math., 56 (2006), pp. 1077–1091.
[12] J. P. Boyd, *Computing real roots of a polynomial in Chebyshev series form through subdivision with linear testing and cubic solves*, Appl. Math. Comput., 174 (2006), pp. 1642–1648.
[13] J. P. Boyd, *Computing the zeros, maxima and inflection points of Chebyshev, Legendre and Fourier series: Solving transcendental equations by spectral interpolation and polynomial rootfinding*, J. Engrg. Math., 56 (2006), pp. 203–219.
[14] J. P. Boyd, *Computing the zeros of a Fourier series or a Chebyshev series or general orthogonal polynomial series with parity symmetries*, Comput. Math. Appl., 54 (2007), pp. 336–349.
[15] J. P. Boyd, *Polynomialization of Kepler's equation through Chebyshev polynomial expansion of the sine: A non-iterative rootfinder*, Appl. Numer. Math., 57 (2007), pp. 12–18.
[16] J. P. Boyd, *Large-degree asymptotics and exponential asymptotics for Fourier coefficients and transforms, Chebyshev and other spectral coefficients*, J. Engrg. Math., 63 (2009), pp. 355–399.
[17] J. P. Boyd, *A comparison of companion matrix methods to find roots of a trigonometric polynomial*, J. Comput. Phys., to appear.
[18] J. P. Boyd, *Numerical, perturbative and Chebyshev inversion of the incomplete elliptic integral of the second kin*, Appl. Math. Comput., 218 (2012), pp. 7005–7013.
[19] J. P. Boyd and D. H. Gally, *Numerical experiments on the accuracy of the Chebyshev-Frobenius companion matrix method for finding the zeros of a truncated series of Chebyshev polynomials*, J. Comput. Appl. Math., 205 (2007), pp. 281–295.
[20] J. P. Boyd and J. R. Ong, *Exponentially-convergent strategies for defeating the Runge phenomenon for the approximation of non-periodic functions, Part* II: *Multi-interval schemes*, Appl. Numer. Math., 61 (2011), pp. 460–472.
[21] J. P. Boyd and B. A. Sadiq, *Computing the real roots of a Fourier series-plus-linear-polynomial: A Chebyshev companion matrix approach*, Appl. Math. 219 (2012), pp. 819–826.
[22] J. D. Brown, M. T. Chu, D. C. Ellison, and R. J. Plemmons, eds., *Proceedings of the Cornelius Lanczos International Centenary Conference*, SIAM, Philadelphia, 1994.
[23] C. W. Clenshaw and A. R. Curtis, *A method for numerical integration on an automatic computer*, Numer. Math., 2 (1960), pp. 197–205.
[24] J. W. Cooley and J. W. Tukey, *An algorithm for the machine calculation of complex Fourier series*, Math. Comp., 19 (1965), pp. 297–301.

[25] R. M. Corless, *Generalized companion matrices in the Lagrange basis*, in Proceedings of Encuentros de Álgebra Computacional y Aplicaciones (EACA 2004), L. Gonzalez-Vega and T. Recio, eds., 2004, pp. 317–322.
[26] R. M. Corless, *On a generalized companion matrix pencil for matrix polynomials expressed in the Lagrange basis*, in Symbolic-Numeric Computation, D. Wang and L.-H. Zhi, eds., Trends in Mathematics, Birkhäuser, Basel, 2007, pp. 1–15.
[27] R. M. Corless and G. Litt, *Generalized Companion Matrices for Polynomials Not Expressed in Monomial Bases*, unpublished, University of Western Ontario, 2000; available online from http://www.apmaths.uwo.ca/~rcorless/frames/PAPERS/PABV/CMP.pdf.
[28] J. H. Curtiss, *Faber polynomials and the Faber series*, Amer. Math. Monthly, 78 (1971), pp. 577–596.
[29] P. J. Davis, *Interpolation and Approximation*, Dover, New York, 1975.
[30] W. R. Davis, M. T. Chu, P. Dolan, J. R. McConnell, L. K. Norris, E. Ortiz, R. J. Plemmons, D. Ridgeway, B. K. P. Scaife, W. J. Stewart, J. W. Yorke, Jr., W. O. Doggett, B. M. Gellai, A. A. Gsponer, and C. A. Prioli, eds., *Cornelius Lanczos: Collected Published Papers with Commentaries*, North Carolina State University, Raleigh, NC, 1998.
[31] D. Day and L. Romero, *Roots of polynomials expressed in terms of orthogonal polynomials*, SIAM J. Numer. Anal., 43 (2005), pp. 1969–1987.
[32] T. A. Driscoll, *Automatic spectral collocation for integral, integro-differential, and integrally reformulated differential equations*, J. Comput. Phys., 229 (2010), pp. 5980–5998.
[33] R. T. Farouki and T. N. T. Goodman, *On the optimal stability of the Bernstein basis*, Math. Comput., 65 (1996), pp. 1553–1566.
[34] L. Fox and I. B. Parker, *Chebyshev Polynomials in Numerical Analysis*, Oxford University Press, London, 1968.
[35] W. Gautschi, *The condition of polynomials in power form*, Math. Comput., 33 (1979), pp. 343–352.
[36] K. O. Geddes, *Chebyshev nodes for interpolation on a class of ellipses*, in Theory of Approximations with Applications, A. G. Law and B. N. Sahney, eds., Academic Press, New York, 1976, pp. 155–170.
[37] K. O. Geddes and J. C. Mason, *Polynomial approximation by projections on the unit circle*, SIAM J. Numer. Anal., 12 (1975), pp. 111–120.
[38] E. M. Gol'berg and V. N. Malozemov, *Estimates for the zeros of certain polynomials*, Vestmol Leningrad Univ. Math., 6 (1979), pp. 127–135. Translated from Vestnik Leningrad Univ. Mat. Mekh. Astr., 7 (1973), pp. 18–24.
[39] I. J. Good, *The colleague matrix, a Chebyshev analogue of the companion matrix*, Quart. J. Math., 12 (1961), pp. 61–68.
[40] A. S. Householder, *The Numerical Treatment of a Single Nonlinear Equation*, McGraw-Hill, New York, 1970.
[41] S. J. Jacobs, *A variable order pseudospectral method for two-point boundary value problems*, J. Comput. Phys., 88 (1990), pp. 169–182.
[42] R. B. King, *Beyond the Quartic Equation*, Birkhäuser, Boston, 1996.
[43] R. B. King and E. R. Canfield, *An algebraic algorithm for calculating the roots of a general quintic equation from its coefficients*, J. Math. Phys., 32 (1991), pp. 823–825.
[44] R. B. King and E. R. Canfield, *Icosahedral symmetry and the quintic equation*, Comput. Math. Appl., 24 (1992), pp. 13–28.
[45] C. Lanczos, *Trigonometric interpolation of empirical and analytical functions*, J. Math. Phys., 17 (1938), pp. 123–199. Reprinted in Cornelius Lanczos: Collected Papers with Commentaries, Vol. 3, W. R. Davis et al., eds., North Carolina State University, Raleigh, NC, 1997, pp. 3-221 to 3-297.
[46] J. M. McNamee, *Numerical Methods for Roots of Polynomials—Part* I, Elsevier, 2007.
[47] R. Pachón, R. B. Platte, and L. N. Trefethen, *Piecewise-smooth Chebfuns*, IMA J. Numer. Anal., 30 (2010), pp. 898–916.
[48] R. E. Shafer, *On quadratic approximation*, SIAM J. Numer. Anal., 11 (1974), pp. 447–460.
[49] W. Specht, *Die Lage der Nullstellen eines Polynoms* III, Math. Nach., 16 (1957), pp. 369–389.
[50] W. Specht, *Die Lage der Nullstellen eines Polynoms* IV, Math. Nach., 21 (1960), pp. 201–222.
[51] H. J. Stetter, *Numerical Polynomial Algebra*, SIAM, Philadelphia, 2004.
[52] L. N. Trefethen, *Spectral Methods in MATLAB*, SIAM, Philadelphia, 2000.
[53] L. N. Trefethen et al., *Chebfun Version* 4.0, The Chebfun Development Team, 2011, http://www.maths.ox.ac.uk/chebfun/.
[54] J. Zhang, *Symbolic computation on complex polynomial solution of differential equations*, J. Symbolic Comput., 22 (1996), pp. 345–354.