Six Myths of Polynomial Interpolation and Quadrature

Lloyd N. Trefethen FRS, FIMA, Oxford University

It is a pleasure to offer this essay for *Mathematics Today* as a record of my Summer Lecture on 29 June 2011 at the Royal Society.

Polynomials are as basic a topic as any in mathematics, and for numerical mathematicians like me, they are the starting point of numerical methods that in some cases go back centuries, like quadrature formulae for numerical integration and Newton iterations for finding roots. You would think that by now, the basic facts about computing with polynomials would be widely understood. In fact, the situation is almost the reverse. There are indeed widespread views about polynomials, but some of the important ones are wrong, founded on misconceptions entrenched by generations of textbooks.

Since 2006, my colleagues and I have been solving mathematical problems with polynomials using the Chebfun software system (`www.maths.ox.ac.uk/chebfun`). We have learned from daily experience how fast and reliable polynomials are. This entirely positive record has made me curious to try to pin down where these misconceptions come from, and this essay is an attempt to summarise some of my findings. Full details, including precise definitions and theorems, can be found in my draft book *Approximation Theory and Approximation Practice,* available at `www.maths.ox.ac.uk/~trefethen`.

The essay is organised around 'six myths'. Each myth has some truth in it – mathematicians rarely say things that are simply false! Yet each one misses something important.

Throughout the discussion, f is a continuous function defined on the interval $[-1,1]$, $n+1$ distinct points $x_0,\dots,x_n$ in $[-1,1]$ are given, and p_n is the unique polynomial of degree at most n with $p_n(x_j)=f(x_j)$ for each j. Two families of points will be of particular interest: *equispaced points,* $x_j=-1+2j/n$, and *Chebyshev points*, $x_j=\cos(j\pi/n)$. I will also mention *Legendre points*, defined as the zeros of the degree $n+1$ Legendre polynomial P_{n+1}.

The discussion of each myth begins with two or three representative quotations from leading textbooks, listed anonymously with the year of publication. Then I say a word about the mathematical truth underlying the myth, and after that, what that truth overlooks.

Myth 1. Polynomial interpolants diverge as $n\to\infty$

Textbooks regularly warn students not to expect $p_n\to f$ as $n\to\infty$.

> *'Polynomial interpolants rarely converge to a general continuous function.'* (1989)
>
> *'Unfortunately, there are functions for which interpolation at the Chebyshev points fails to converge.'* (1996)

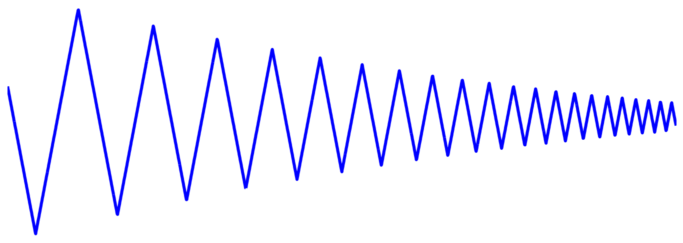
On the face of it, this caution is justified by two theorems. Weierstrass proved in 1885 [1] that any continuous function can be approximated arbitrarily closely by polynomials. On the other hand, Faber proved in 1914 [2] that no polynomial *interpolation* scheme, no matter how the points are distributed, will converge for all such functions.

So it sounds as if there is something wrong with polynomial interpolation. Yet the truth is, polynomial interpolants in Chebyshev points always converge *if f is a little bit smooth.* (We shall call them *Chebyshev interpolants*.) Lipschitz continuity is more than enough, that is, $|f(x)-f(y)|\le L|x-y|$ for some constant L and all $x,y\in[-1,1]$. So long as f is Lipschitz continuous, as it will be in almost any practical application, $p_n\to f$ is guaranteed.

There is indeed a big problem with convergence of polynomial interpolants, but it pertains to interpolation *in equispaced points.* As Runge showed in 1901 [3], equispaced interpolants may diverge exponentially, even if f is so smooth as to be analytic (holomorphic). This genuinely important fact, known as the Runge phenomenon, has confused people. With Faber's theorem seeming to provide justification, a real problem with equispaced polynomial interpolants has been overgeneralised so that people have suspected it of applying to polynomial interpolants in general.

The smoother f is, the faster its Chebyshev interpolants converge. If f has ν derivatives, with the νth derivative being of bounded variation V, then $\|f-p_n\|=O(Vn^{-\nu})$ as $n\to\infty$. (By $\|f-p_n\|$ I mean the maximum of $|f(x)-p_n(x)|$ for $x\in[-1,1]$.) If f is analytic, the convergence is geometric, with $\|f-p_n\|=O(\rho^{-n})$ for some $\rho>1$. I will give details about the parameter ρ under Myth 5.

For example, here is the degree 10,000 Chebyshev interpolant p_n to the sawtooth function $f(x)$ defined as the integral from -1 to x of $\mathrm{sign}(\sin(100t/(2-t)))$. This curve may not look like a polynomial, but it is! With $\|f-p_n\|\approx 0.0001$, the plot is indistinguishable from a plot of f itself.



There is not much use in polynomial interpolants to functions with so little smoothness as this, but mathematically they are trouble free. For smoother functions like e^x, $\cos(10x)$ or $1/(1+25x^2)$, we get convergence to 15 digits of accuracy for small values of n (14, 34 and 182, respectively).

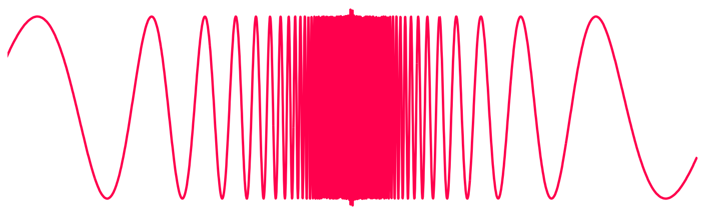
Myth 2. Evaluating polynomial interpolants numerically is problematic

Interpolants in Chebyshev points may converge in theory, but aren't there problems on a computer? Textbooks warn students about this.

> *'Polynomial interpolation has drawbacks in addition to those of global convergence. The determination and evaluation of interpolating polynomials of high degree can be too time-consuming and can also lead to difficulty problems associated with roundoff error.'* (1977)
>
> *'Although Lagrangian interpolation is sometimes useful in theoretical investigations, it is rarely used in practical computations.'* (1985)
>
> *'Interpolation is a notoriously tricky problem from the point of view of numerical stability.'* (1990)

The origin of this view is the fact that some of the methods one might naturally try for evaluating polynomial interpolants are slow or numerically unstable or both. For example, if you write down the Lagrange interpolation formula in its most obvious form and implement it on a computer as written, you get an algorithm that requires $O(n^2)$ work per evaluation point. (Partly because of this, books warn readers that they should use Newton interpolation formulae rather than Lagrange – another myth.) And if you compute interpolants 'linear algebra style', by setting up a Vandermonde matrix whose columns contain samples of $1, x, x^2, \dots, x^n$ at the grid points, your numerical method is exponentially unstable. This is the 'polyval/polyfit' algorithm of Matlab, and I am guilty of propagating Myth 2 myself by using this algorithm in my textbook *Spectral Methods in Matlab* [4]. Rounding errors on a computer destroy all accuracy of this method even for $n = 60$, let alone $n = 10{,}000$ as in the plot above.

Or how about $n = 1{,}000{,}000$? Here is a plot of the polynomial interpolant to $f(x) = \sin(10/x)$ in a million Chebyshev points. The plot was obtained in about 30 seconds on my laptop by evaluating the interpolant at 2,000 points clustered near zero.



The fast and stable algorithm that makes these calculations possible comes from a representation of the Lagrange interpolant known as the *barycentric formula*, published by Salzer in 1972 [5]:

$$p_n(x) = \sum_{j=0}^{n}{}' \frac{(-1)^j f_j}{x - x_j} \Bigg/ \sum_{j=0}^{n}{}' \frac{(-1)^j}{x - x_j},$$

with the special case $p_n(x) = f_j$ if $x = x_j$. The primes on the summation signs signify that the terms $j = 0$ and $j = n$ are multiplied by $1/2$. The work required is $O(n)$ per evaluation point, and though the divisions by $x - x_j$ may look dangerous for $x \approx x_j$, the formula is numerically stable, as was proved by Nick Higham in 2004 [6].

Myth 3. Best approximations are optimal

This one sounds true by definition!

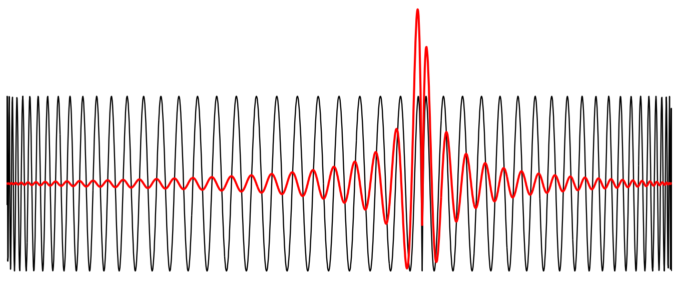
> *'Since the Remes algorithm, or indeed any other algorithm for producing genuine best approximations, requires rather extensive computations, some interest attaches to other more convenient procedures to give good, if not optimal, polynomial approximations.'* (1968)
>
> *'Minimal polynomial approximations are clearly suitable for use in functions evaluation routines, where it is advantageous to use as few terms as possible in an approximation.'* (1968)
>
> *'Ideally, we would want a best uniform approximation.'* (1980)

Though the statement of this myth looks like a tautology, there is content in it. The 'best approximation' is a common name for the unique polynomial p_n^* that minimises $\|f - p_n^*\|$. So a best approximant is optimal in the maximum norm, but is it really the best in practice?

As the first quotation suggests, computing p_n^* is not a trivial matter, since the dependence on f is nonlinear. By contrast, computing a Chebyshev interpolant with the barycentric formula is easy. Here our prejudices about value for money begin to intrude. If best approximations are hard to compute, they must be valuable!

Two considerations make the truth not so simple. First of all, the maximum-norm accuracy difference between Chebyshev interpolants and best approximations can never be large, for Ehlich and Zeller proved in 1966 [7] that $\|f - p_n\|$ cannot exceed $\|f - p_n^*\|$ by more than the factor $2 + (2/\pi)\log(n+1)$. Usually the difference is less than that, and in fact, the best known error bounds for functions that have ν derivatives or are analytic, the two smoothness classes mentioned under Myth 1, are just a factor of 2 larger for Chebyshev interpolants as for best approximations.

Secondly, according to the equioscillation theorem going back to Chebyshev in the 1850s, the maximum error of a best approximation is always attained at at least $n + 2$ points in $[-1, 1]$. For example, the black curve below is the error curve $f(x) - p_n^*(x)$, $x \in [-1, 1]$, for the best approximant to $|x - 1/4|$ of degree 100, equioscillating between 102 extreme values $\approx \pm 0.0027$. The red curve corresponds to the polynomial interpolant p_n in Chebyshev points. It is clear that for most values of x, $|f(x) - p_n(x)|$ is much smaller than $|f(x) - p_n^*(x)|$. Which approximation would be more useful in an application? I think the only reasonable answer is, it depends. Sometimes one really does need a guarantee about worst-case behaviour. In other situations, it would be wasteful to sacrifice so much accuracy over 95% of the range just to gain one bit of accuracy in a small subinterval.



Myth 4. Gauss quadrature has twice the order of accuracy of Clenshaw–Curtis

Quadrature formulae are usually derived from polynomials. We approximate an integral by a finite sum,

$$I = \int_{-1}^{1} f(x)dx \quad \approx \quad I_n = \sum_{k=0}^{n} w_k f(x_k), \tag{1}$$

and the weights $\{w_k\}$ are determined by the principle of interpolating f by a polynomial at the points $\{x_k\}$ and integrating the interpolant. Newton–Cotes quadrature corresponds to equispaced points, Clenshaw–Curtis quadrature to Chebyshev points, and Gauss quadrature to Legendre points. Almost every textbook first describes Newton–Cotes, which achieves $I_n = I$ exactly if f is a polynomial of degree n, and then shows that Gauss has twice this order of exactness: $I_n = I$ if f is a polynomial of degree $2n+1$. Clenshaw–Curtis is occasionally mentioned, but it only has order of exactness n, no better than Newton–Cotes.

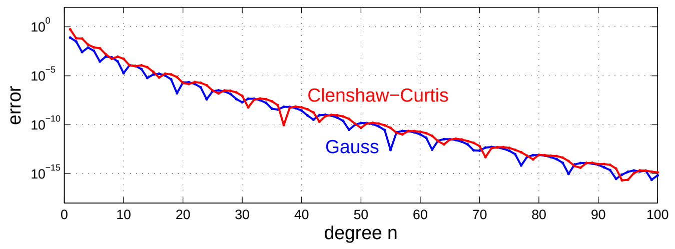
'However, the degree of accuracy for Clenshaw–Curtis quadrature is only $n-1$.' (1997)

'Clenshaw–Curtis rules are not optimal in that the degree of an n-point rule is only $n-1$, which is well below the maximum possible.' (2002)

This ubiquitous emphasis on order of exactness is misleading. Textbooks suggest that the reason Gauss quadrature gives better results than Newton–Cotes is its higher order of exactness, but this is not correct. The problem with Newton–Cotes is that the sample points are equally spaced: the Runge phenomenon. In fact, as Pólya proved in 1933 [8], Newton–Cotes quadrature does not converge as $n \to \infty$, in general, even if f is analytic.

Clenshaw–Curtis and Gauss quadratures behave entirely differently. Both schemes converge for all continuous integrands, and if the integrand is analytic, the convergence is geometric. Clenshaw–Curtis is easy to implement, using either the Fast Fourier Transform or by an algorithm of Waldvogel in 2006 [9], and one reason it gets little attention may be that Gauss quadrature had a big head start, invented in 1814 [10] instead of 1960 [11]. Both Clenshaw–Curtis and Gauss quadrature are practical even if n is in the millions, in the latter case because nodes and weights can be calculated by an algorithm implemented in Chebfun due to Glaser, Liu and Rokhlin in 2007 [12]. (Indeed, since Glaser–Liu–Rokhlin, it is another myth to imagine that Gauss quadrature is only practicable for small values of n.)

With twice the order of exactness, we would expect Gauss quadrature to converge twice as fast as Clenshaw–Curtis. Yet it does not. Unless f is analytic in a large region surrounding $[-1,1]$ in the complex plane, one typically finds that Clenshaw–Curtis quadrature converges at about the same rate as Gauss, as illustrated by these curves for $f(x) = \exp(-1/x^2)$:



A theorem of mine from 2008 makes this observation precise. If f has a νth derivative of bounded variation V, Gauss quadrature can be shown to converge at the rate $O(V(2n)^{-\nu})$, the factor of 2 reflecting its doubled order of exactness. The theorem asserts that the same rate $O(V(2n)^{-\nu})$, with the same factor of 2, is also achieved by Clenshaw–Curtis. (Folkmar Bornemann (private communication) has pointed out that both of these rates can probably be improved by one further power of n.)

The explanation for this surprising result goes back to O'Hara and Smith in 1968 [13]. It is true that $(n+1)$-point Gauss quadrature integrates the Chebyshev polynomials $T_{n+1}, T_{n+2}, \dots$ exactly whereas Clenshaw–Curtis does not. However, the error that Clenshaw–Curtis makes consists of aliasing them to the Chebyshev polynomials $T_{n-1}, T_{n-2}, \dots$ and integrating these correctly. As it happens, the integrals of T_{n+k} and T_{n-k} differ by only $O(n^{-3})$, and that is why Clenshaw–Curtis is more accurate than its order of exactness seems to suggest.

Myth 5. Gauss quadrature is optimal

Gauss quadrature may not be much better than Clenshaw–Curtis, but at least it would appear to be as accurate as possible, the gold standard of quadrature formulae.

'The precision is maximised when the quadrature is Gaussian.' (1982)

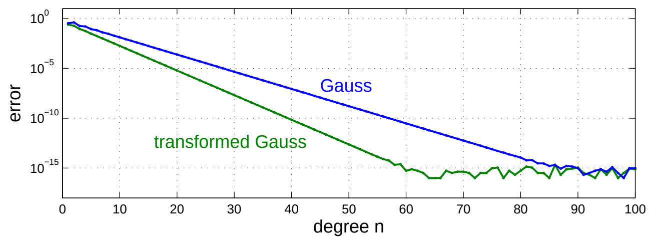
'In fact, it can be shown that among all rules using n function evaluations, the n-point Gaussian rule is likely to produce the most accurate estimate.' (1989)

There is another misconception here, quite different from the one just discussed as Myth 4. Gauss quadrature is optimal as measured by polynomial order of exactness, but that is a skewed measure.

The power of polynomials is nonuniform: they have greater resolution near the ends of an interval than in the middle. Suppose, for example, that $f(x)$ is an analytic function on $[-1,1]$, which means it can be analytically continued into a neighbourhood of $[-1,1]$ in the complex x-plane. Then polynomial approximants to f, whether Chebyshev or Legendre interpolants or best approximants, will converge at a geometric rate $O(\rho^{-n})$ determined by how close any singularities of f in the plane are to $[-1,1]$. To be precise, ρ is the sum of the semiminor and semimajor axis lengths of the largest ellipse with foci ± 1 inside which f is analytic and bounded. But ellipses are narrower near the ends than in the middle. If f has a singularity at $x_0 = i\varepsilon$ for some small ε, then we get $O(\rho^{-n})$ convergence with $\rho \approx 1+\varepsilon$. If f has a singularity at $1+\varepsilon$, on the other hand, the parameter becomes $\rho \approx 1+\sqrt{2\varepsilon}$, corresponding to much faster convergence. A function with a singularity at 1.01 converges 14 times faster than one with a singularity at $0.01i$.

Quadrature rules generated by polynomials, including both Gauss and Clenshaw–Curtis, show the same nonuniformity. This might seem unavoidable, but in fact, there is no reason why a quadrature formula (1) needs to be derived from polynomials. By introducing a change of variables, one can generate alternative formulae based on interpolation by transplanted polynomials, which may converge up to $\pi/2$ times faster than Gauss or Clenshaw–Curtis quadrature for many functions. This idea was developed in a paper of mine with Nick Hale in 2008 [14] and is related to earlier work by Kosloff and Tal-Ezer in 1993 [15].

The following theorem applies to one of the transformed methods Hale and I proposed. Let f be a function that is analytic in an ε-neighborhood of $[-1,1]$ for $\varepsilon \leq 0.05$. Then whereas Gauss quadrature converges at the rate $I_n - I = O((1+\varepsilon)^{-2n})$, transformed Gauss quadrature converges 50% faster, $I_n - I = O((1+\varepsilon)^{-3n})$. Here is an illustration for $f(x) = 1/(1+25x^2)$.



The fact that some quadrature formulae converge up to $\pi/2$ times faster than Gauss as $n \to \infty$ is probably not of much practical importance. The importance is conceptual.

Myth 6. Polynomial root-finding is dangerous

Our final myth originates with Jim Wilkinson (1919–1986), a hero of mine who taught me two courses in graduate school at Stanford. Working with Alan Turing on the Pilot Ace computer in 1950, Wilkinson found that attempts to compute roots of even some low-degree polynomials failed dramatically. He publicised this discovery widely.

'Our main object in this chapter has been to focus attention on the severe inherent limitations of all numerical methods for finding the zeros of polynomials.' (1963)

'Beware: Some polynomials are ill-conditioned!' (1992)

The first of these quotations comes from Wilkinson's book on rounding errors [16], and he also coined the memorable phrase 'the perfidious polynomial' as the title of a 1984 article that won the Chauvenet Prize for outstanding mathematical exposition [17].

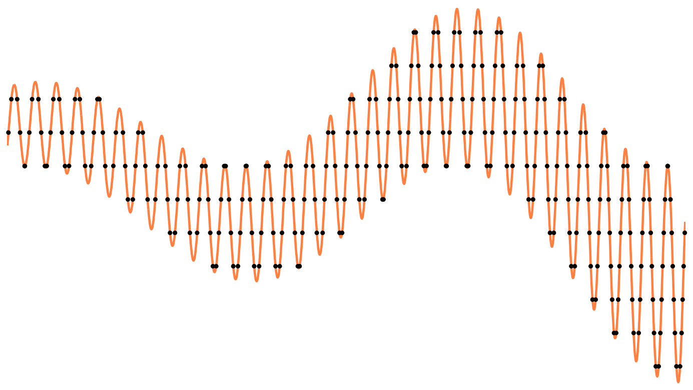
What Wilkinson discovered was the extreme ill-conditioning of roots of certain polynomials *as functions of their coefficients*. Specifically, suppose a polynomial p_n is specified by its coefficients in the form $a_0 + a_1x + \cdots + a_nx^n$. If p_n has roots near the unit circle in the complex plane, these pose no difficulties: they are well-conditioned functions of the coefficients a_k and can be computed accurately by Matlab's 'roots' command, based on the calculation of eigenvalues of a companion matrix containing the coefficients. Roots far from the circle, however, such as roots on the interval $[-1,1]$, can be so ill-conditioned as to be effectively uncomputable. The monomials x^k form exponentially bad bases for polynomials on $[-1,1]$.

The flaw in the argument is that it says nothing about the condition of roots of polynomials *as functions of their values*. For effective root-finding on $[-1,1]$ based on pointwise samples, all one must do is fix the basis: replace the monomials x^k, which are orthogonal polynomials on the unit circle, by the Chebyshev polynomials $T_k(x)$, which are orthogonal on the interval. Suppose a polynomial p_n is specified by its coefficients in the form $a_0T_0(x) + a_1T_1(x) + \cdots + a_nT_n(x)$. If p_n has roots near $[-1,1]$, these are well-conditioned functions of the coefficients a_k, and they can be computed accurately by solving an eigenvalue problem involving a 'colleague matrix'. The details were worked out by Specht in 1957 [18] and Good in 1961 [19].

Chebfun finds roots of a function f on $[-1,1]$ by approximating it by a polynomial expressed in Chebyshev form and then solving a colleague-matrix eigenvalue problem, and if the degree is greater than 100, first subdividing the interval recursively to reduce it. These ideas originate with John Boyd in 2002 [20] and are extraordinarily effective. Far from being exceptionally troublesome, polynomial root-finding when posed in this fashion begins to emerge as the most tractable of all root-finding problems, for we can solve the problem globally with just $O(n^2)$ work to get all the roots in an interval to high accuracy.

For example, the function $f(x) = \sin(1000\pi x)$ on $[-1,1]$ is represented in Chebfun by a polynomial of degree 4091. It takes 2 seconds on my laptop to find all 2001 of its roots in $[-1,1]$, and the maximum deviation from the exact values is 4.4×10^{-16}.

Here is another illustration of the robustness of polynomial root-finding on an interval. In Chebfun, we have plotted the function $f(x) = \exp(x/2)(\sin(5x) + \sin(101x))$ and then executed the commands `r = roots(f-round(f)), plot(r,f(r),'.')`. This sequence solves a collection of hundreds of polynomial root-finding problems to locate all the points where f takes a value equal to an integer or a half-integer, and plots them as dots. The computation took 2/3 of a second.



Conclusion

Perhaps I might close by mentioning another perspective on the misconceptions that have affected the study of computation with polynomials. By the change of variables $x = \cos\theta$, one can show that interpolation by polynomials in Chebyshev points is equivalent to interpolation of periodic functions by series of sines and cosines in equispaced points. The latter is the subject of discrete Fourier analysis, and one cannot help noting that whereas there is widespread suspicion that it is not safe to compute with polynomials, nobody worries about the Fast Fourier Transform! In the end this may be the biggest difference between Fourier and polynomial interpolants, the difference in their reputations.

And here's a bonus, free of charge.

Myth 7. Lagrange discovered Lagrange interpolation

It was Waring in 1779 [21]. Euler used the formula in 1783, and Lagrange in 1795. ❏

References

1 Weierstrass, K. (1885) *Über die analytische Darstellbarkeit sogenannter willkürlicher Funktionen einer reellen Veränderlichen*, Sitzungsberichte der Akademie zu Berlin, pp. 633–639 and 789–805.

2 Faber, G. (1914) Über die interpolatorische Darstellung stetiger Funktionen, *Jahresber. Deutsch. Math. Verein.*, vol. 23, pp. 190–210.

3 Runge, C. (1901) Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten, *Z. Math. Phys.*, vol. 46, pp. 224–243.

4 Trefethen, L.N. (2000) *Spectral Methods in MATLAB*, SIAM, Philadelphia.

5 Salzer, H.E. (1972) Lagrangian interpolation at the Chebyshev points $x_{n,v} \equiv \cos(v\pi/n), v = 0(1)n$; some unnoted advantages, *Comp. J.* vol. 15, pp. 156–159.

6 Higham, N.J. (2004) The numerical stability of barycentric Lagrange interpolation *IMA J. Numer. Anal.*, vol. 24, no. 4, pp. 547–556.

7 Ehlich, H. and Zeller, K. (1966) Auswertung der Normen von Interpolationsoperatoren, *Math. Ann.*, vol. 164, pp. 105–112.

8 Pólya, G. (1933) Über die Konvergenz von Quadraturverfahren, *Math. Z.*, vol. 37, pp. 264–286.

9 Waldvogel, J. (2006) Fast construction of the Fejér and Clenshaw–Curtis quadrature rules, *BIT Num. Math.*, vol. 46, no. 1. pp. 195–202.

10 Gauss, C. F. (1814) Methodus nova integralium valores per approximationem inveniendi, *Comment. Soc. R. Scient. Göttingensis Rec.*, vol. 3, pp. 39–76.

11 Clenshaw, C. W. and Curtis, A. R. (1960) A method for numerical integration on an automatic computer, *Numer. Math.*, vol. 2, pp. 197–205.

12 Glaser, A., Liu, X. and Rokhlin, V. (2007) A fast algorithm for the calculation of the roots of special functions, *SIAM J. Sci. Comp.*, vol. 29, no. 4, pp. 1420–1438.

13 O'Hara, H. and Smith, F. J. (1968) Error estimation in the Clenshaw–Curtis quadrature formula, *Comp. J.*, vol. 11, pp. 213–219.

14 Hale, N. and Trefethen, L. N. (2008) New quadrature formulas from conformal maps, *SIAM J. Numer. Anal.*, vol. 46, pp. 930–948.

15 Kosloff, D. and Tal-Ezer, H. (1993) A modified Chebyshev pseudospectral method with an $O(N^1)$ time step restriction, *J. Comp. Phys.*, vol. 104, pp. 457–469.

16 Wilkinson, J.H. (1994) *Rounding Errors in Algebraic Processes (Prentice-Hall Series in Automatic Computation)* Dover Publications.

17 Wilkinson, J.H. (1984) The perfidious polynomial, *MAA Stud. Num. Anal.*

18 Specht, W. (1957) Die Lage der Nullstellen eines Polynoms. III, *Math. Nachr.*, vol. 16, pp. 363–389.

19 Good, I. J. (1961) The colleague matrix, a Chebyshev analogue of the companion matrix, *Quart. J. Math.*, vol. 12, pp. 61–68.

20 Boyd, J.P. (2002) Computing zeros on a real interval through Chebyshev expansion and polynomial rootfinding, *SIAM J. Numer. Anal.*, vol. 40, no. 5, pp. 1666–1682.

21 Waring, E. (1779) Problems concerning interpolations, *Phil. Trans.*, vol. 69, pp. 59–67.