

BARYCENTRIC LAGRANGE INTERPOLATION

JEAN-PAUL BERRUT* AND LLOYD N. TREFETHEN†

Dedicated to the memory of Peter Henrici (1923–1987)

Abstract. Barycentric interpolation is a variant of Lagrange polynomial interpolation that is fast and stable. It deserves to be known as the standard method of polynomial interpolation.

Key words. barycentric formula, interpolation

AMS subject classifications. 65D05, 65D25

1. Introduction. “Lagrangian interpolation is praised for analytic utility and beauty but deplored for numerical practice.” This statement, from the extended table of contents of one of the most enjoyable textbooks of numerical analysis [1], expresses a widespread view.

In the present work we shall show that, on the contrary, the Lagrange approach is in many cases the method of choice for dealing with polynomial interpolants. The key is that the Lagrange polynomial must be manipulated through the formulas of *barycentric interpolation*. Barycentric interpolation is not new, but most students, most mathematical scientists, and even many numerical analysts do not know about it. This simple and powerful idea deserves a place at the heart of introductory courses and textbooks in numerical analysis.¹

As always with polynomial interpolation, unless the degree of the polynomial is very low, it will not do to use uniformly spaced interpolation points. Instead one must use Chebyshev points or other systems of points clustered at the boundary of the interval of approximation.

2. Lagrange and Newton interpolation. Let $n + 1$ distinct interpolation points (nodes) x_j , $j = 0, \dots, n$, be given, together with corresponding numbers f_j , which may or may not be samples of a function f . Unless stated otherwise, we assume that the nodes are real, although most of the results and comments generalize to the complex plane. Let Π_n denote the vector space of all polynomials of degree at most n . The classical problem addressed here is that of finding the polynomial $p \in \Pi_n$ that interpolates f at the points x_j , i.e.,

$$p(x_j) = f_j, \quad j = 0, \dots, n.$$

The problem is well-posed, i.e., it has a unique solution that depends continuously on the data. Moreover, as explained in virtually every introductory numerical analysis

*Département de Mathématiques, Université de Fribourg, 1700 Fribourg/Pérolles, Switzerland, Jean-Paul.Berrut@unifr.ch. The work of this author has been supported by the Swiss National Science Foundation, grant No. 21-59272.99.

†Computing Laboratory, Oxford University, LNT@comlab.ox.ac.uk.

¹We are speaking here of a method of interpolating data in one space dimension by a polynomial. A distinct use of the term barycentric for representation of multidimensional data, e.g. on triangles and tetrahedra, is somewhat better known.

text, the solution can be written in *Lagrange form*,²

$$(2.1) \quad p(x) = \sum_{j=0}^n f_j \ell_j(x), \quad \ell_j(x) = \frac{\prod_{k=0, k \neq j}^n (x - x_k)}{\prod_{k=0, k \neq j}^n (x_j - x_k)}.$$

The *Lagrange polynomial* ℓ_j corresponding to the node x_j has the property

$$(2.2) \quad \ell_j(x_k) = \begin{cases} 1, & j = k, \\ 0, & \text{otherwise,} \end{cases} \quad j, k = 0, \dots, n.$$

At this point, many authors specialize Lagrange's formula (2.1) for small numbers n of nodes, before asserting that certain shortcomings make it a bad choice for practical computations:

1. Each evaluation of $p(x)$ requires $O(n^2)$ additions and multiplications.
2. Adding a new data pair x_{n+1}, f_{n+1} requires a new computation from scratch.
3. The computation is numerically unstable.

From here it is commonly concluded that the Lagrange form of p is mainly a theoretical tool for proving theorems. For computations, it is generally recommended that one should instead use Newton's formulas, which require only $O(n)$ flops for each evaluation of p once some numbers, which are independent of the evaluation point x , have been computed.³

Newton's approach consists of two steps. First, compute the *Newton tableau* of divided differences

$$\begin{array}{ccccccc} f[x_0] & & & & & & \\ & f[x_0, x_1] & & & & & \\ f[x_1] & & f[x_0, x_1, x_2] & & & & \\ & f[x_1, x_2] & & f[x_0, x_1, x_2, x_3] & & & \\ f[x_2] & & f[x_1, x_2, x_3] & & \ddots & & \\ & f[x_2, x_3] & & \vdots & & f[x_0, x_1, x_2, \dots, x_n] & \\ f[x_3] & & \vdots & & f[x_{n-3}, \dots, x_n] & & \\ \vdots & \vdots & f[x_{n-2}, x_{n-1}, x_n] & & & & \\ f[x_n] & & & & & & \end{array}$$

by the recursive formula

$$(2.3) \quad f[x_j, x_{j+1}, \dots, x_{k-1}, x_k] = \frac{f[x_{j+1}, \dots, x_k] - f[x_j, \dots, x_{k-1}]}{x_k - x_j}$$

²Lagrange's original treatment can be found in [42]. On p. 171 of [27] it is stated that Lagrange attributes the formula to Newton, but this is not the way we read Lagrange's text. Lagrange recalls a formula of Newton with the words "Newton est le premier qui se soit proposé ce Problème; voici la solution qu'il en donne;" but then he introduces the derivation of his own formula with "Mais on peut réduire cette solution à une plus grande simplicité par la considération suivante."

³We define a *flop*—floating point operation—as a multiplication or division plus an addition or subtraction. (In practice, divisions are slower than multiplications on many computers.) As usual, $O(n^p)$ means $\leq cn^p$ for some constant c .

with initial condition $f[x_j] = f_j$. This requires about n^2 subtractions and $n^2/2$ divisions. Second, for each x , evaluate $p(x)$ by the *Newton interpolation formula*

$$(2.4) \quad \begin{aligned} p(x) = & f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) + \cdots \\ & + f[x_0, x_1, \dots, x_n](x - x_0)(x - x_1) \dots (x - x_{n-1}). \end{aligned}$$

This requires a mere n flops, much less than the $O(n^2)$ required for direct application of (2.1).

3. An improved Lagrange formula. The first purpose of the present work is to emphasize that the Lagrange formula (2.1) can be rewritten in such a way that it too can be evaluated and updated in $O(n)$ operations, just like its Newton counterpart (2.4). To this end, it suffices to notice that the numerator of ℓ_j in (2.1) can be written as the quantity

$$(3.1) \quad \ell(x) = (x - x_0)(x - x_1) \cdots (x - x_n)$$

divided by $x - x_j$. If we define the *barycentric weights* by

$$(3.2) \quad w_j = \frac{1}{\prod_{k \neq j} (x_j - x_k)}, \quad j = 0, \dots, n,$$

we can thus write ℓ_j as

$$\ell_j(x) = \ell(x) \frac{w_j}{x - x_j}.$$

Now we note that all the terms of the sum in (2.1) contain the factor $\ell(x)$, which does not depend on j . This factor may therefore be brought in front of the sum to yield

$$(3.3) \quad p(x) = \ell(x) \sum_{j=0}^n \frac{w_j}{x - x_j} f_j.$$

That's it! Now Lagrange interpolation too is a formula requiring $O(n^2)$ flops for calculating some quantities independent of x , the numbers w_j , followed by $O(n)$ flops for evaluating p once these numbers are known. Rutishauser [49] calls (3.3) the "first form of the barycentric interpolation formula".

What about updating? A look at (3.2) shows that incorporating a new node x_{n+1} entails two calculations:

- Divide each w_j , $j = 0, \dots, n$, by $x_j - x_{n+1}$ (one flop for each point), for a cost of $n + 1$ flops;
- Compute w_{n+1} with formula (3.2), another $n + 1$ flops.

The Lagrange interpolant can thus also be updated with $O(n)$ flops! By applying this update recursively, we get the following algorithm for computing $w_j = w_j^{(j)}$:

```

 $w_0^{(0)} = 1;$ 
for  $j = 1$  to  $n$  do
  for  $k = 1$  to  $j - 1$  do
     $w_j^{(k)} = w_j^{(k-1)} / (x_k - x_j);$ 

```

end
 $w_j^{(j)} = 1 / \prod_{k=0}^{j-1} (x_j - x_k)$
end

This algorithm performs precisely the same operations as (3.2), only in a different order. It therefore yields the same numerical values, apart from any differences in rounding errors on a computer.

A remarkable advantage of Lagrange over Newton interpolation, rarely mentioned in the literature, is that *the quantities that have to be computed in $O(n^2)$ operations do not depend on the data f_j* . This feature permits the interpolation of as many functions as desired in $O(n)$ operations each once the weights w_j are known, whereas Newton interpolation requires the re-computation of the divided difference tableau for each new function.

Another advantage of the Lagrange formula is that it does not depend on the order in which the nodes are arranged. In the Newton formula, the divided differences do have such a dependence. This seems odd aesthetically, and it has a computational consequence: for larger values of n , most orderings lead to numerical instability. For stability, it is necessary to select the points in a *van der Corput* or *Leja sequence* or in another related sequence based on a certain equidistribution property [20, 59, 65].

This is not to say that Newton interpolation has no advantages. One is the elegant way in which it can incorporate derivative data in the case of confluent interpolation points (e.g., $x_k = x_{k+1}$). Another may be its ability to accommodate vector or matrix interpolation from scalar data [58]; using a representation like (3.3) would require costly matrix inversion. There are also some theoretical instances in which Newton interpolation is preferable, e.g., the construction of multistep formulas for the solution of ordinary differential equations [32, 43].

4. The barycentric formula. Eq. (3.3) is not the end of the story: it can be modified to a even better formula, the one that should be used in practice, for reasons of numerical stability.

Suppose we interpolate, besides the data f_j , the constant function 1, whose interpolant is of course itself. Inserted into (3.3), we get

$$(4.1) \quad 1 = \sum_{j=0}^n \ell_j(x) = \ell(x) \sum_{j=0}^n \frac{w_j}{x - x_j}.$$

Dividing (3.3) by this expression and cancelling the common factor $\ell(x)$, we obtain the *barycentric formula* for p :

$$(4.2) \quad \boxed{p(x) = \frac{\sum_{j=0}^n \frac{w_j}{x - x_j} f_j}{\sum_{j=0}^n \frac{w_j}{x - x_j}}} \quad \begin{array}{l} \text{BARYCENTRIC} \\ \text{FORMULA} \end{array}$$

where w_j is still defined by (3.2). Rutishauser [49] calls (4.2) the “second (true) form of the barycentric formula”. See p. 229 of [38] for comments on this terminology.

We see that the barycentric formula is a Lagrange formula, but one with a special and beautiful symmetry. The weights w_j appear in the denominator exactly as in the numerator, except without the data factors f_j . This means that any common factor

in all the weights w_j may be cancelled without affecting the value of p_n , and in the discussion to follow we will make use of this freedom.

Like (3.3), (4.2) can also take advantage of the updating of the weights w_j in $O(n)$ flops to incorporate a new data pair x_{n+1}, f_{n+1} .

5. Chebyshev and other point distributions. For certain special sets of nodes x_j , one can give explicit formulas for the barycentric weights w_j . The obvious place to start is *equidistant nodes* with spacing $h = 2/n$ on the interval $[-1, 1]$. Here the weights can be directly calculated to be $w_j = (-1)^{n-j} \binom{n}{j} / (h^n n!)$ [53], which after cancelling the factors independent of j yields

$$(5.1) \quad w_j = (-1)^j \binom{n}{j}.$$

For the interval $[a, b]$ we would multiply the original formula for w_j by $2^n(b-a)^{-n}$, but this constant factor too can be dropped, so we end up with (5.1) again, regardless of a and b .

A glance at (5.1) reveals that if n is large, the weights w_j for equispaced barycentric interpolation vary by exponentially large factors, of order approximately 2^n . This sounds dangerous, and it is. The effect will be that even small data near the center of the interval are associated with wild oscillations in the interpolant, on the order of 2^n times bigger, near the edge of the interval [39, 63]. This so-called *Runge phenomenon* is not a problem with the barycentric formula, but is intrinsic in the underlying interpolation problem. Among other things it implies that polynomial interpolation in equally spaced points is highly *ill-conditioned*: small changes in the data may cause huge changes in the interpolant.

For polynomial interpolation to be a well-conditioned process, unless n is very small, one must dispense with equally spaced points [57, Thm. 6.21.3]. As is well known in approximation theory, the right approach is to use point sets that are clustered at the endpoints of the interval with an asymptotic density proportional to $(1-x^2)^{-1/2}$ as $n \rightarrow \infty$. Remarkably, this is the same asymptotic density one gets if $[-1, 1]$ is interpreted as a conducting wire and the points x_j are interpreted as point charges repelling one another with an inverse-linear force, which are allowed to move along the wire to find their equilibrium configuration [62, chap. 5]. And it is precisely the same asymptotic density that is required to make the weights w_j of comparable scale in the sense that although they may not all be exactly equal, they do not vary by factors exponentially large in n .

The simplest examples of clustered point sets are the families of *Chebyshev points*, obtained by projecting equally spaced points on the unit circle down to the unit interval $[-1, 1]$. Four standard varieties of such points have been defined, and for each, there is an explicit formula for ℓ in (3.1) which may be easily differentiated [8, 34, 45]. From the identity

$$(5.2) \quad w_j = \frac{1}{\ell'(x_j)}$$

[39, p. 243] we accordingly obtain explicit formulas for the weights w_j .

The *Chebyshev points of the first kind* are defined by

$$x_j = \cos \frac{(2j+1)\pi}{2n+2}, \quad j = 0, \dots, n.$$

In this case we find [39, p. 249]

$$(5.3) \quad w_j = (-1)^j \sin \frac{(2j+1)\pi}{2n+2}.$$

Note that these numbers vary algebraically, not exponentially, reflecting the good distribution of the points. The *Chebyshev points of the second kind* are defined by

$$x_j = \cos \frac{j\pi}{n}, \quad j = 0, \dots, n.$$

Here we find [50]

$$(5.4) \quad w_j = (-1)^j \delta_j, \quad \delta_j = \begin{cases} 1/2, & j = 0 \text{ or } j = n, \\ 1, & \text{otherwise;} \end{cases}$$

all but two of the weights are exactly equal. Formulas for the less important Chebyshev points of the third and fourth kinds can be found in [8].

For all of these sets of Chebyshev points, if the interval $[-1, 1]$ is linearly transformed to $[a, b]$, the weights as defined by (3.2) all get multiplied by $2^n(b-a)^{-n}$. However, as this factor cancels out in the barycentric formula, there is as usual no need to include it (and it is safer not to—see §7).

One sees that, with equidistant or Chebyshev points, no expensive computations are needed to get the weights w_j , and thus only $O(n)$ operations are required for evaluating p_n . No other interpolation method seems to achieve this, for as mentioned in §3, Newton interpolation always requires $O(n^2)$ operations for the divided differences.

Here is a MATLAB code segment which samples the function $f(x) = |x| + x/2 - x^2$ in 1001 Chebyshev points of the second kind and evaluates the barycentric interpolant in 5000 points. One would rarely use so many points in practice; we pick such large values just to illustrate the effectiveness of the formula. (At the end of §7 we shall modify this program for numerical stability.)

```
n = 1000;
fun = inline('abs(x)+.5*x-x.^2');
x = cos(pi*(0:n)'/n);
f = fun(x);
c = [1/2; ones(n-1,1); 1/2].*(-1).^((0:n)');
xx = linspace(-1,1,5000)';
numer = zeros(size(xx));
denom = zeros(size(xx));
for j = 1:n+1
    xdiff = xx-x(j);
    numer = numer + c(j)*f(j)./xdiff;
    denom = denom + c(j)./xdiff;
end
ff = numer./denom;
plot(x,f,'.',xx,ff,'-')
```

On one of our workstations this code runs in about 1 second, producing a plot like those shown in Figure 5.1 but for the case $n = 1000$. For further numerical examples of barycentric interpolation, see [39].

Other point sets with the asymptotic distribution $(1-x^2)^{-1/2}$ also lead to well-conditioned polynomial approximation, notably the *Legendre points*—zeros or extrema of the Legendre polynomials. However, explicit formulas for the weights w_j are not known for Legendre points.

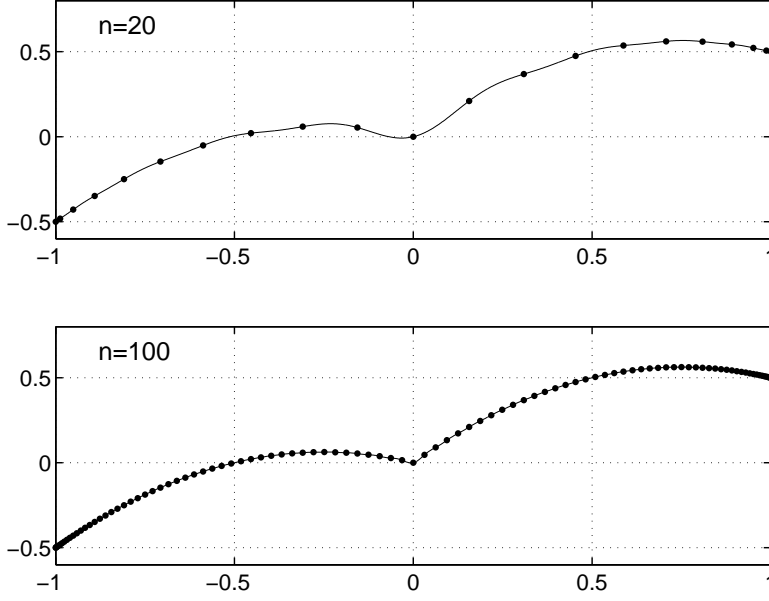


FIG. 5.1. Barycentric interpolation of the function $f(x) = |x| + x/2 - x^2$ in 21 and 101 Chebyshev points of the second kind on $[-1, 1]$. The dots mark the interpolated values f_j .

6. Convergence rates for smooth functions. If a smooth function f is interpolated by polynomials in Chebyshev points or in any other system of points with the asymptotic density $(1 - x^2)^{-1/2}$, the rate of convergence of the interpolants to f as $n \rightarrow \infty$ is remarkably fast. In particular, suppose that f is a function that can be analytically continued to a function $f(z)$ that is analytic (holomorphic) in a neighborhood of $[-1, 1]$ in the complex plane. Then the interpolants p_n satisfy an error estimate

$$(6.1) \quad \max_{x \in [-1, 1]} |f(x) - p_n(x)| \leq CK^{-n}$$

for some constants C and $K > 1$ [21, p. 173]. The bigger the region of analyticity, the bigger we may take K . To be precise, if f is analytic on and inside an ellipse in the complex plane with foci ± 1 and axis lengths $2L$ and 2ℓ , then we may take $K = L + \ell$ [62, chap. 5].

Figure 6.1 illustrates this fast convergence for two smooth functions. The convergence rate for $f(x) = \exp(x)/\cos(x)$ is determined by the poles of $f(z)$ at $z = \pm\pi/2$; we have

$$K = \frac{\pi}{2} + \sqrt{\pi^2 - 1} \approx 2.7822.$$

The convergence rate for $f(x) = (1 + 16x^2)^{-1}$ is determined by the poles of this function at $z = \pm i/4$; we have

$$K = \frac{1}{4} + \sqrt{\frac{17}{16}} \approx 1.2808.$$

These facts about convergence pertain to polynomial interpolation in general, not barycentric interpolation per se, but the latter has a special feature well suited to

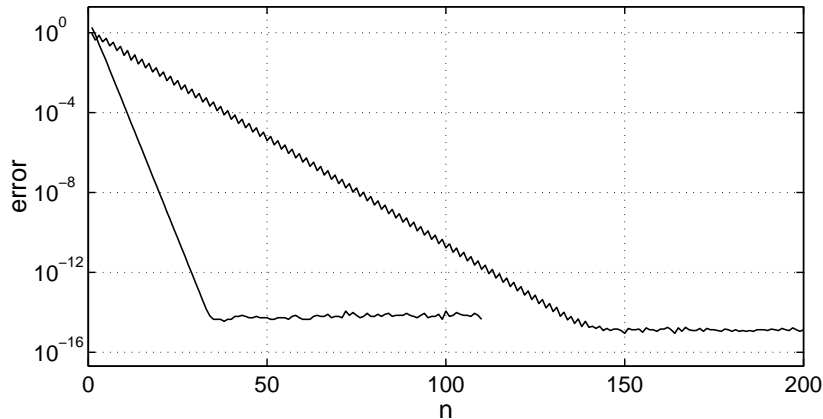


FIG. 6.1. Convergence of the error (6.1) in polynomial interpolation of two smooth functions f in Chebyshev points of the second kind in $[-1, 1]$. The lower curve corresponds to $f(x) = \exp(x)/\cos(x)$, and the upper curve to $f(x) = (1 + 16x^2)^{-1}$. In both cases we observe steady convergence down to the level of rounding errors. The wiggles in the second function arise from the missing central node for odd n .

taking advantage of such convergence rates. Suppose we apply the barycentric formula to Chebyshev points associated with values $n = 2, 4, 8, 16, \dots$. Then it is possible to compute the interpolants recursively, reusing the previous function values f_j each time n is doubled [38]. If f is analytic in a neighborhood of $[-1, 1]$, each doubling of n results approximately in a squaring of the error; one has *quadratic convergence* of the overall process.

A function f may be less smooth than the class we have considered (e.g., twice continuously differentiable, or infinitely differentiable); or it may be smoother (e.g., analytic in the entire complex plane). A great deal is known about exactly how the convergence rates of polynomial interpolants depend on the precise degree of smoothness of f . See chap. 5 of [62] for an introduction and [56] for a more advanced treatment.

7. Numerical stability. Polynomial interpolation is a subject in which instability is often a problem. One of the major advantages of the barycentric formula is its stability when properly applied. But this is a subject where confusion is widespread, and one must be sure to distinguish several phenomena that are distinct.

One phenomenon to be clear about is that for improperly distributed sets of nodes, as discussed in §5, the underlying interpolation problem is ill-conditioned: small changes in the data may cause large changes in the interpolant, typically manifested as wild oscillations near the endpoints. Unless the nodes converge to the $(1 - x^2)^{-1/2}$ distribution, the condition numbers grow exponentially as $n \rightarrow \infty$. In such circumstances polynomial interpolation is simply not a useful method for applications, regardless of whether one uses a Lagrange, Newton, or any other formulation. Even though the interpolants may converge to f in theory as $n \rightarrow \infty$ when f is sufficiently smooth, rounding errors will destroy the convergence in practice.

Let us suppose then that we are working with a properly distributed set of points, such as Chebyshev points. The barycentric formula is now stable provided that two matters are attended to.

The first is the question of underflow and overflow in the computation of the

weights in cases where we are working from the general formula (3.2) rather than from well-behaved explicit expressions like (5.3) and (5.4). Suppose we are working on an interval $[a, b]$ of length $2C$. (In the field of complex analysis, C is called the *capacity* of the interval. The following remarks generalize to approximation on more complicated sets in the complex plane besides intervals.) Then as $n \rightarrow \infty$, the scale of the weights w_j as defined by (3.2) will grow or decay exponentially at the rate C^{-n} . If n is large or C is far from 1, the result may be weights that underflow or overflow on a computer—in IEEE double precision arithmetic, the thresholds are about 10^{-308} and 10^{308} . This problem can be solved easily: one need only rescale the weights uniformly by a factor roughly of the order C^n . There is no need to determine this factor exactly—if you get it right to within a factor of, say, 10^{290} , no digits of accuracy will be lost!

The other matter of stability is less trivial but also easily coped with: what if the value of x in (4.2) is very close to one of the interpolation points x_k , or in an extreme case, exactly equal? Consider first the case in which $x \approx x_k$ but $x \neq x_k$. The quotient $w_k/(x - x_k)$ will be very large, and it would seem that there might be a risk of inaccuracy in this number caused by cancellation error in the denominator. However, as pointed out by Henrici [39], this is not in fact a problem. There is indeed cancellation and inaccuracy, but the same inaccurate number appears in both numerator and denominator of (4.2), and these inaccuracies cancel out; the formula remains stable overall.

If $x = x_k$ exactly, on the other hand, something must certainly be done: the barycentric formula requires a division by zero, and if the MATLAB code segment given earlier is run in standard IEEE arithmetic, the corresponding values of the interpolant **ff** will come out as NaN, i.e., not-a-number. A somewhat old-fashioned solution to this problem is to perturb x by a number on the order of machine precision. In the MATLAB code, this can be achieved by changing the definition of **xdiff** to

```
xdiff = xx-x(i); xdiff(xdiff==0) = eps;
```

A better solution, however, is simply to replace the affected values of **ff** by their correct exact values. One way to do this is to add three lines containing a new variable **exact** to the kernel of the MATLAB code, as follows:

```
numer = zeros(size(xx));
denom = zeros(size(xx));
exact = zeros(size(xx));
for j = 1:n+1
    xdiff = xx-x(j);
    numer = numer + c(j)*f(j)./xdiff;
    denom = denom + c(j)./xdiff;
    exact(xdiff==0) = i;
end
ff = numer./denom;
jj = find(exact); ff(jj) = f(exact(jj));
```

The above observations seem to make barycentric interpolation entirely stable in practice, but it would appear that, despite convincing explanations and experiments by several authors, a rigorous analysis of the stability of these formulas has not been published.

The barycentric interpolation formula has a further kind of stability or robustness property that proves advantageous in some applications. Suppose (4.2) is applied

with an arbitrary set of positive weights w_j , not necessarily those given by (3.2). For example, the weights might have been computed somehow with large errors. It is easily seen that the resulting function $p(x)$ still interpolates the data f , even though it is no longer in general a polynomial (see §9.2).

8. Trigonometric, sinc, and Laurent interpolation. Polynomial interpolation in Chebyshev points on $[-1, 1]$ is equivalent to other familiar interpolation problems, and accordingly, these too have barycentric formulas. For example, one can transplant any function f defined on $[-1, 1]$ to a 2π -periodic, even function F defined on $[-\pi, \pi]$ by changing variables to $\theta = \cos^{-1} x$ and defining $F(\theta) = f(x) = f(\cos \theta)$ [56]. The barycentric formula becomes [8]

$$(8.1) \quad t_n(\theta) = \frac{\sum_{j=0}^n \frac{w_j}{\cos \theta - \cos \theta_j} F_j}{\sum_{j=0}^n \frac{w_j}{\cos \theta - \cos \theta_j}}, \quad w_j = \frac{1}{\prod_{k \neq j} (\cos \theta_j - \cos \theta_k)},$$

with $\theta_j = \cos^{-1} x_j$ and $F_j = F(\theta_j)$. Since $t_n(\theta) = p_n(x)$ is a polynomial of degree $\leq n$ in $\cos \theta$, it is a trigonometric polynomial of degree $\leq n$ in θ [47, p. 3], and (8.1) is the *trigonometric interpolant* of F . For Chebyshev points of the second kind $\{x_j\}$, the transplanted nodes $\{\theta_j\}$ are equally spaced, and by (5.4), (8.1) becomes

$$t_n(\theta) = \sum_{j=0}^n \frac{(-1)^j \delta_j}{\cos \theta - \cos \theta_j} F_j \bigg/ \sum_{j=0}^n \frac{(-1)^j \delta_j}{\cos \theta - \cos \theta_j}.$$

This formula is a special case of the trigonometric interpolant of degree $\leq n$ of an arbitrary function F in an even number $2n$ of *equidistant* nodes $\theta_j = j\pi/n$, $j = 0, \dots, 2n-1$, on the interval $[0, 2\pi]$ [38]:

$$(8.2) \quad t(\theta) = \sum_{j=0}^{2n-1} (-1)^j \cot \frac{\theta - \theta_j}{2} F(\theta_j) \bigg/ \sum_{j=0}^{2n-1} (-1)^j \cot \frac{\theta - \theta_j}{2}.$$

These formulas are essentially equivalent to those that arise in the Discrete Fourier Transform.

The last expression can be further generalized. The function t is nothing but the so-called *cardinal* or *sinc interpolant* S_h of the 2π -periodically extended function F between the equidistant points $\theta_j = jh$, $h = \pi/n$, $j = -\infty, \dots, \infty$ on the whole real line R [54]. After replacement of the cotangent with its Mittag-Leffler series, (8.2) becomes the barycentric formula for S_h [9],

$$S_h(\theta) = \sum_{j=-\infty}^{\infty} \frac{(-1)^j}{\theta - \theta_j} F(\theta_j) \bigg/ \sum_{j=-\infty}^{\infty} \frac{(-1)^j}{\theta - \theta_j}.$$

In this way we recover the sinc interpolant for more general functions on R (i.e., non-periodic). Gautschi has recently found another way of evaluating S_h that is more efficient in many cases [26].

Another closely related problem is that of polynomial interpolation in equally spaced points on the unit circle in the complex plane. For the n roots of unity $z_j = \exp(2\pi i j/n)$, $j = 0, \dots, n-1$, one has $\ell(z) = \prod_j (z - z_j) = z^n - 1$, and the

simplified weights are easily computed from (5.2) as $w_j = z_j$, yielding the barycentric formula [31, 39]

$$p_n(z) = \sum_{j=0}^{n-1} \frac{z_j}{z - z_j} f(z_j) \bigg/ \sum_{j=0}^{n-1} \frac{z_j}{z - z_j}.$$

This is a starting point for all kinds of computations related to Cauchy integrals and Taylor and Laurent series in complex analysis [37].

9. Further uses of the Lagrange representation. Lagrange and barycentric formulas for polynomial interpolation have many other uses. Here are a few.

9.1. Estimation of the Lebesgue constant. We mentioned in §5 that some sets of interpolation points are better than others. Famous mathematicians, among them P. Erdős, have given much effort to studies of which point sets are good or best, and how such matters can be quantified. A crucial observation is that, for a fixed set of $n+1$ nodes x_j , the operator P_n that maps a function f to its interpolating polynomial p_n is a linear projection, with $P_n p_n = p_n$. A good measure of the conditioning of the interpolation problem is the norm of P_n , known as the *Lebesgue constant*,

$$(9.1) \quad \Lambda_n = \|P_n\| = \sup_{f \in C([-1,1])} \frac{\|P_n f\|}{\|f\|},$$

where $\|f\| = \max_{x \in [-1,1]} |f(x)|$ and $C([-1,1])$ denotes the space of all continuous functions on $[-1,1]$. This number can be shown to be a function of the Lagrange polynomials ℓ_j of (2.1) [46, 56]:

$$\Lambda_n = \max_{x \in [-1,1]} \sum_{j=0}^n |\ell_j(x)|.$$

For some sets of interpolation points, the values of Λ_n have been determined or estimated theoretically; see [15], [25, p. 121], and [35]. For any nodes, we can use the computed weights (3.2) to yield a lower bound:

$$(9.2) \quad \Lambda_n \geq \frac{1}{2n^2} \frac{\max_{0 \leq j \leq n} |w_j|}{\min_{0 \leq j \leq n} |w_j|}.$$

This inequality can be derived by applying Markov's inequality on the size of derivatives of polynomials [11]. Notice that it quantifies the observation of §5 that if the barycentric weights vary widely, the interpolation problem must be ill-conditioned. Other illustrations of the relevance of the quotient of the barycentric weights can be found in [24] and [65].

9.2. Rational interpolation. It was mentioned in §7 that if the barycentric formula (4.2) is applied with an arbitrary set of weights w_j , not necessarily those given by (3.2), then the resulting function $p(x)$ still interpolates the data f even though it is no longer in general a polynomial. Back-multiplication of the numerator and denominator by $\ell(x)$ shows that $p(x)$ is in fact the quotient of two polynomials of degree at most n , i.e., a rational interpolant of f . It is easy to see that, conversely, every rational function r that interpolates the data f_j in the points x_j can be written in the form (4.2) for some weights w_j [11, p. 79]. These observations lead to an efficient method for computing rational interpolants based on computing the kernel

of a variant type of Vandermonde matrix [12]. The barycentric representation of rational interpolants has several advantages over other representations, besides those it shares with polynomial interpolation. In particular, it allows for an easier detection of so-called unattainable points and of poles in the interval of interpolation [10, 52].

9.3. Differentiation of polynomial interpolants. Suppose we have a function u represented by a polynomial interpolant in Lagrange form, $u(x) = \sum_{j=0}^n u_j \ell_j(x)$. Then the first and second derivatives of u are

$$(9.3) \quad u'(x) = \sum_{j=0}^n u_j \ell_j'(x), \quad u''(x) = \sum_{j=0}^n u_j \ell_j''(x).$$

Now by (4.2), the barycentric representation of ℓ_j is

$$\ell_j(x) = \frac{w_j}{x - x_j} \bigg/ \sum_{k=0}^n \frac{w_k}{x - x_k}.$$

Multiplying through, and then multiplying both sides by $x - x_i$ to render them differentiable at $x = x_i$, we have

$$\ell_j(x) \sum_{k=0}^n w_k \frac{x - x_i}{x - x_k} = w_j \frac{x - x_i}{x - x_j},$$

from which it follows with $s(x) = \sum_{k=0}^n w_k (x - x_i)/(x - x_k)$ that

$$\ell_j'(x)s(x) + \ell_j(x)s'(x) = w_j \left(\frac{x - x_i}{x - x_j} \right)'$$

and

$$\ell_j''(x)s(x) + 2\ell_j'(x)s'(x) + \ell_j(x)s''(x) = w_j \left(\frac{x - x_i}{x - x_j} \right)''.$$

General formulas for $\ell_j'(x)$ and $\ell_j''(x)$ can be derived from these expressions. At $x = x_i$, straightforward computations yield $s(x_i) = w_i$, $s'(x_i) = \sum_{k \neq i} w_k/(x_i - x_k)$ and $s''(x_i) = -2 \sum_{k \neq i} w_k/(x_i - x_k)^2$, from which, together with $\ell_j(x_i) = 0$, we get for $i \neq j$

$$(9.4) \quad \ell_j'(x_i) = \frac{w_j/w_i}{x_i - x_j}, \quad \ell_j''(x_i) = -2 \frac{w_j/w_i}{x_i - x_j} \left[\sum_{k \neq i} \frac{w_k/w_i}{x_i - x_k} - \frac{1}{x_i - x_j} \right].$$

As for the case $i = j$, (4.1) implies $\sum_{j=0}^n \ell^{(m)}(x) = 0$ for each differentiation order m and thus

$$(9.5) \quad \ell_j'(x_j) = - \sum_{i \neq j} \ell_j'(x_i), \quad \ell_j''(x_j) = - \sum_{i \neq j} \ell_j''(x_i).$$

The advantages of (9.5) from the point of view of numerical stability are discussed in [2, 3, 5, 6].

What we have just achieved with the aid of Lagrange interpolation formulas is the computation of the entries of what are commonly known as first- and second-order *differentiation matrices* $D^{(1)}$ and $D^{(2)}$:

$$(9.6) \quad D_{ij}^{(1)} = \ell_j'(x_i), \quad D_{ij}^{(2)} = \ell_j''(x_i).$$

These matrices have a simple interpretation. If f is a vector of function values associated with the grid $\{x_j\}$, then $D^{(1)}f$ is the vector obtained by interpolating these data, then differentiating the interpolant at the grid points; and similarly for $D^{(2)}$. These formulas hold virtually unchanged for rational interpolation [4].

9.4. Spectral methods for differential equations. Differentiation of interpolants is the basis of one of the most important applications of polynomial interpolation: *spectral collocation methods* for the numerical solution of ordinary and partial differential equations. Suppose f is a function on $[-1, 1]$, for example, and we want to solve numerically the boundary value problem

$$(9.7) \quad \frac{d^2 u}{dx^2} = f, \quad -1 < x < 1$$

together with boundary conditions $u(-1) = u(1) = 0$. One way to do this is to set up a Chebyshev grid and consider the $(n-1) \times (n-1)$ matrix problem

$$Dv = f,$$

where D is the matrix consisting of the interior rows and columns of the matrix $D^{(2)}$ of (9.6), v is an $(n-1)$ -vector of unknown approximations to u at the grid points, and f is now the $(n-1)$ -vector of values of $f(x)$ sampled at these same grid points. If a continuous function $v(x)$ is constructed by barycentric interpolation from the solution vector v together with boundary values zero, then v may be an extremely good approximation to the exact solution u . If f is analytic in a neighborhood of $[-1, 1]$, for example, the accuracy will improve geometrically at the rate discussed in §6 as $n \rightarrow \infty$.

Spectral collocation methods have aroused great interest in recent decades and have given rise to a large body of literature, of which we mention here just the books [14, 16, 21, 22, 29, 62, 64]. These are the high-accuracy, “ p versions” of the more general class of finite element (and spectral element) methods for the numerical solution of differential equations, an even larger subject in which polynomials written in Lagrange form also play an important role [19].

9.5. Fast multipole methods. Finally, the barycentric formula has natural advantages for applications to fast multipole methods, which are fast algorithms invented by Rokhlin [48] and Greengard [30] for evaluating certain sums. In [18], multipole methods are applied to the evaluation of interpolating polynomials of large degrees by means of the first form of the barycentric formula (3.3). The true barycentric formula (4.2) should lead to even better stability, and first tests in this direction are described in [13].

10. Historical notes. Why is barycentric interpolation not better known?

The first appearance of the barycentric representation is in the article [60], in which W. Taylor restricted himself to equidistant points. The term “barycentric” seems to appear for the first time in [17] (although Hamming implicitly attributes it to Taylor in [33]). Did Dupuy know of Taylor’s work? He does not cite it. Kuntzmann wrote the first book with extensive discussion of barycentric formulas [41], but his work did not have much impact in the USA.

In the first edition of his very influential text [33], Hamming presents the barycentric formula very nicely just after introducing “the Lagrange method of interpolation” and comments that the formula is “easier to use than the Lagrange formula”. Why

did he discard this paragraph in the second edition? His comment may show that he mainly had hand calculations in mind and believed the advent of the computer would render obsolete any better way of evaluating the formula. This very unfortunate omission would probably have been avoided had Winrich's 1969 paper [66] appeared a little earlier, which clearly documented the superiority of the barycentric formula for every $n > 3$. In [36], another influential book of the 1960s and 1970s, Henrici did not mention barycentric formulas either.

Barycentric formulas were certainly noticed by Rutishauser and Stiefel at the ETH in Zurich. Rutishauser coauthored the chapter on interpolation in [51] with Bulirsch, and he presented barycentric formulas in his classes at the ETH in the 1960s. However, his lecture notes were only published much later by M. Gutknecht [49]. Stiefel emphasized the barycentric formula in [55], as did Henrici in his later text [39]. However, the former is in German, and the second did not have the impact of [36] in the USA. Schwarz's book, first in German and then in an English translation, has followed Stiefel's work and retained the barycentric formula [53]. Some French and German authors describe the barycentric representation [44, 61], and in the USA it at least sometimes appears as an exercise [40, 62]. A recent (transatlantic) text that treats barycentric formulas nicely is the textbook by Gautschi [25], who, as it happens, was the translator of Rutishauser's lecture notes into English [49].

The first publication mentioning that the Lagrange representation can be updated seems to be the important paper by Werner [65]. (If Werner's paper had attracted more attention, there would have been less need for this one!) Werner's algorithm is in fact twice as fast as the one we give in §3, but replaces the quotient defining $w_j^{(j)}$ with a long, unstable sum, a drawback also experienced by Gautschi in the related application of the construction of quadrature rules [24]. It should be noted that in 1973 Gander had already updated Lagrange and barycentric formulas for the special case of the extrapolation to the limit [23].

Schneider and Werner were the first to notice that the advantages of the barycentric representation carry over from polynomial to rational interpolation, as mentioned in §9.2. The first publication of a differentiation matrix as in §9.3 seems to have been by Bellman et al. [7] as a special case of the general method of *differential quadrature* introduced by these authors. In the context of spectral methods, the first appearance may have been in [28]. In general, however, the literature on spectral methods makes almost no mention of barycentric formulas.

If you look in the index of a book of numerical analysis, you probably won't find "barycentric". Let us hope it will be different a generation from now.

Acknowledgments. We are indebted to Walter Gautschi for reading and amending a first draft of this work.

REFERENCES

- [1] F. S. ACTON, *Numerical Methods that Work*, Amer. Math. Soc., Providence, 1990.
- [2] R. BALTENSPERGER, *Improving the accuracy of the matrix differentiation method for arbitrary collocation points*, Appl. Numer. Math., 33 (2000), pp. 143–149.
- [3] R. BALTENSPERGER R. AND J.-P. BERRUT, *The errors in calculating the pseudospectral differentiation matrices for Čebyšev–Gauss–Lobatto points*, Computers Math. Applic., 37 (1999), pp. 41–48. Errata **38** (1999) 119.
- [4] R. BALTENSPERGER R. AND J.-P. BERRUT, *The linear rational collocation method*, J. Comput. Appl. Math., 134 (2001), pp. 243–258.
- [5] R. BALTENSPERGER R. AND M. R. TRUMMER, *Spectral differencing with a twist*, SIAM J. Sci. Comp., to appear.

- [6] A. BAYLISS, A. CLASS, AND B. MATKOWSKY, *Roundoff error in computing derivatives using the Chebyshev differentiation matrix*, J. Comput. Phys., 116 (1994), pp. 380–383.
- [7] R. BELLMAN, B. G. KASHEF, AND J. CASTI, *Differential quadrature: a technique for the rapid solution of nonlinear partial differential equations*, J. Comput. Phys., 10 (1972), pp. 40–52.
- [8] J.-P. BERRUT, *Baryzentrische Formeln zur trigonometrischen Interpolation I*, Z. angew. Math. Phys. (ZAMP), 35 (1984), pp. 91–105.
- [9] J.-P. BERRUT, *Barycentric formulae for cardinal (SINC-) interpolants*, Numer. Math., 54 (1989), pp. 703–718. (Erratum 55 (1989) 747.)
- [10] J.-P. BERRUT, *Linear rational interpolation of continuous functions over an interval*, W. Gautschi, ed., Mathematics of Computation 1943–1993: a Half-Century of Computational Mathematics, Proceedings of Symposia in Applied Mathematics, American Mathematical Society, Providence, 1994, 261–264.
- [11] J.-P. BERRUT AND H. MITTELMANN, *Lebesgue constant minimizing linear rational interpolation of continuous functions over the interval*, Computers Math. Applic., 33 (1997), pp. 77–86.
- [12] J.-P. BERRUT AND H. MITTELMANN, *Matrices for the direct determination of the barycentric weights of rational interpolation*, J. Comput. Appl. Math., 78 (1997), pp. 355–370.
- [13] M. BLÖCHLIGER, *Ein Spezialfall der Multipolmethode für baryzentrische Formeln*, Diploma thesis, University of Fribourg (Switzerland), 1998.
- [14] J. P. BOYD, *Chebyshev and Fourier Spectral Methods*, 2nd ed., Dover, 2001.
- [15] L. BRUTMAN, *Lebesgue functions for polynomial interpolation—a survey*, in *The heritage of P. L. Chebyshev: A Festschrift in honor of the 70th birthday of T. J. Rivlin*, Ann. Numer. Math., 4 (1997), pp. 111–127.
- [16] C. CANUTO, M. Y. HUSSAINI, A. QUARTERONI, AND T. A. ZANG, *Spectral Methods in Fluid Dynamics*, Springer, New York, 1988.
- [17] M. DUPUY, *Le calcul numérique des fonctions par l'interpolation barycentrique*, C. R. Acad. Sci., 226 (1948), pp. 158–159.
- [18] A. DUTT, M. GU, AND V. ROKHLIN, *Fast algorithms for polynomial interpolation, integration and differentiation*, SIAM J. Numer. Anal., 33 (1996), pp. 1689–1711.
- [19] K. ERIKSSON, D. ESTEP, P. HANSBO, AND C. JOHNSON, *Introduction to Computational Methods for Differential Equations*, Clarendon Press, Oxford, 1995.
- [20] B. FISCHER AND L. REICHEL, *Newton interpolation in Fejér and Chebyshev points*, Math. Comp., 53 (1989), pp. 265–278.
- [21] B. FORNBERG, *A Practical Guide to Pseudospectral Methods*, Cambridge Univ. Press, Cambridge 1996.
- [22] D. FUNARO, *Polynomial Approximation of Differential Equations*, Springer, Berlin, 1992.
- [23] W. GANDER, *Numerische Implementationen des Rombergschen Extrapolationsverfahrens, mit Anwendungen auf die Summation unendlicher Reihen*, ETH Doctoral Thesis Nr. 5172, Zürich, 1973.
- [24] W. GAUTSCHI, *Moments in quadrature problems*, Computers Math. Applic., 33 (1997), pp. 105–118.
- [25] W. GAUTSCHI, *Numerical Analysis: an Introduction*, Birkhäuser, Boston, 1997.
- [26] W. GAUTSCHI, *Barycentric formulae for cardinal (SINC-)interpolants by Jean-Paul Berrut (Remark)*, Numer. Math., 87 (2001), pp. 791–792.
- [27] H. H. GOLDSTINE, *A History of Numerical Analysis from the 16th through the 19th Century*, Springer, New York, 1977.
- [28] D. GOTTLIEB D., M. Y. HUSSAINI AND S. A. ORSZAG, *Theory and applications of spectral methods*, in R. J. Voigt, D. Gottlieb, and M. Y. Hussaini, eds., *Spectral Methods for Partial Differential Equations*, SIAM, Philadelphia, 1984, pp. 1–54.
- [29] D. GOTTLIEB D. AND S. A. ORSZAG, *Numerical Analysis of Spectral Methods, Theory and Applications*, SIAM, Philadelphia, 1977.
- [30] L. GREENGARD, *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press, Cambridge, 1988.
- [31] M. H. GUTKNECHT, *Numerical conformal mapping methods based on function conjugation*, J. Comput. Appl. Math., 14 (1986), pp. 31–78.
- [32] E. HAIRER, S. P. NØRSETT, AND G. WANNER, *Solving Ordinary Differential Equations I. Non-stiff Problems*, Springer, Berlin, 1987.
- [33] R. W. HAMMING, *Numerical Methods for Scientists and Engineers*, McGraw-Hill, New York, 1962; 2d ed. 1973.
- [34] D. C. HANDSCOMB AND J. C. MASON, *Chebyshev Polynomials*, CRC Press, 2003.
- [35] W. HEINRICHS, *Strong convergence estimates for pseudospectral methods*, Appl. Math., 37 (1992), pp. 401–417.
- [36] P. HENRICI, *Elements of Numerical Analysis*, Wiley, New York, 1962.

- [37] P. HENRICI, *Fast Fourier methods in computational complex analysis*, SIAM Review, 21 (1979), pp. 481–527.
- [38] P. HENRICI, *Barycentric formulas for interpolating trigonometric polynomials and their conjugates*, Numer. Math., 33 (1979), pp. 225–234.
- [39] P. HENRICI, *Essentials of Numerical Analysis*, Wiley, New York, 1982.
- [40] D. KINCAID AND W. CHENEY, *Numerical Analysis: Mathematics of Scientific Computing*, Wadsworth, Belmont, 1991.
- [41] J. KUNTZMANN, *Méthodes Numériques: Interpolation—Dérivées*, Dunod, Paris, 1959.
- [42] J. L. LAGRANGE, *Leçons élémentaires sur les mathématiques, données à l'Ecole Normale en 1795*, in Oeuvres VII, Gauthier-Villars, Paris, 1877, 183–287.
- [43] J. D. LAMBERT, *Numerical Methods for Ordinary Differential Systems*, Wiley, Chichester, 1991.
- [44] G. MAESS, *Vorlesungen über numerische Mathematik, II. Analysis*, Birkhäuser, Basel, 1988.
- [45] J. C. MASON, *Chebyshev polynomials of the second, third and fourth kinds in approximation, indefinite integration and integral transforms*, J. Comput. Appl. Math., 49 (1993), pp. 169–178.
- [46] M. J. D. POWELL, *Approximation Theory and Methods*, Cambridge University Press, 1981.
- [47] T. J. RIVLIN, *The Chebyshev Polynomials*, John Wiley, New York, 1974.
- [48] V. ROKHLIN, *Rapid solution of integral equations of classical potential theory*, J. Comput. Phys., 60 (1985), pp. 187–207.
- [49] H. RUTISHAUSER, *Vorlesungen über numerische Mathematik, Vol. 1*, Birkhäuser, Basel–Stuttgart, 1976; English translation *Lectures on Numerical Mathematics* by Walter Gautschi, Birkhäuser, Boston, 1990.
- [50] H. E. SALZER, *Lagrangian interpolation at the Chebyshev points $x_{n,\nu} = \cos(\nu\pi/n)$, $\nu = 0(1)n$; some unnoted advantages*, Computer J., 15 (1972), pp. 156–159.
- [51] R. SAUER AND I. SZABÓ, *Mathematische Hilfsmittel des Ingenieurs*, Springer, Berlin–Heidelberg, 1968.
- [52] C. SCHNEIDER AND W. WERNER, *Some new aspects of rational interpolation*, Math. Comp., 47 (1986), pp. 285–299.
- [53] H. R. SCHWARZ, *Numerische Mathematik*, 4th ed., Teubner, 1997; English translation of the 2nd edition: *Numerical Analysis: A Comprehensive Introduction*, Wiley, New York, 1989.
- [54] F. STENGER, *Numerical Methods Based on Sinc and Analytic Functions*, Springer, 1993.
- [55] E. STIEFEL, *Einführung in die numerische Mathematik*, Teubner, Stuttgart, 1961.
- [56] J. SZABADOS AND P. VÉRTESI, *Interpolation of Functions*, World Scientific, Singapore, 1990.
- [57] G. SZEGŐ, *Orthogonal Polynomials*, Amer. Math. Soc., Providence, 1978.
- [58] H. TAL-EZER, *Polynomial approximation of functions of matrices and applications*, J. Sci. Comput., 4 (1989), pp. 25–60.
- [59] H. TAL-EZER, *High degree polynomial interpolation in Newton form*, SIAM J. Sci. Stat. Comput., 12 (1991), pp. 648–667.
- [60] W. J. TAYLOR, *Method of Lagrangian curvilinear interpolation*, J. Res. Nat. Bur. Standards, 35 (1945), pp. 151–155.
- [61] R. THÉODOR, *Initiation à l'Analyse Numérique*, Masson, Paris, 1989.
- [62] L. N. TREFETHEN, *Spectral Methods in MATLAB*, SIAM, Philadelphia, 2000.
- [63] L. N. TREFETHEN AND J. A. C. WEIDEMAN, *Two results on polynomial interpolation in equally spaced points*, J. Approx. Theory, 65 (1991), pp. 247–260.
- [64] R. J. VOIGT, D. GOTTLIEB, AND M. Y. HUSSAINI, *Spectral Methods for Partial Differential Equations*, SIAM, Philadelphia, 1984.
- [65] W. WERNER, *Polynomial interpolation: Lagrange versus Newton*, Math. Comp., 43 (1984), pp. 205–217.
- [66] L. B. WINNICH, *Note on a comparison of evaluation schemes for the interpolating polynomial*, Computer J., 12 (1969), pp. 154–155.