Chapter 2

Function Approximation

We saw in the introductory chapter that one key step in the construction of a numerical method to approximate a definite integral is the approximation of the integrand by a simpler function, which we can integrate exactly.

The problem of function approximation is central to many numerical methods. Given a continuous function f in a closed, bounded interval $[a, b]$, we would like to find a good approximation to it by functions from a certain class, for example algebraic polynomials, trigonometric polynomials, rational functions, radial functions, splines, neural networks, etc. We are going to measure the accuracy of an approximation using norms and ask whether or not there is a best approximation out of functions from a given family of functions. These are the main topics of this introductory chapter in approximation theory.

2.1 Norms

A norm on a vector space V over a field $\mathbb{F}$ ($\mathbb{R}$ or $\mathbb{C}$ for our purposes) is a mapping

$$\|\cdot\| : V \to [0, \infty),$$

which satisfy the following properties:

(i) $\|x\| \geq 0 \; \forall x \in V$ and $\|x\| = 0$ iff $x = 0$.

(ii) $\|x + y\| \leq \|x\| + \|y\| \; \forall x, y \in V$.

(iii) $\|\lambda x\| = |\lambda| \, \|x\| \; \forall x \in V, \lambda \in \mathbb{F}$.

If we relax (i) to just $\|x\| \geq 0$, we get a *semi-norm.*

We recall first some of the most important examples of norms in the finite dimensional case $V = \mathbb{R}^n$ (or $V = \mathbb{C}^n$):

$$\|x\|_1 = |x_1| + \ldots + |x_n|, \tag{2.1}$$

$$\|x\|_2 = \sqrt{|x_1|^2 + \ldots + |x_n|^2}, \tag{2.2}$$

$$\|x\|_\infty = \max\{|x_1|, \ldots, |x_n|\}. \tag{2.3}$$

These are all special cases of the l^p norm:

$$\|x\|_p = \left(|x_1|^p + \ldots + |x_n|^p\right)^{1/p}, \qquad 1 \leq p \leq \infty. \tag{2.4}$$

If we have weights $w_i > 0$ for $i = 1, \ldots, n$ we can also define a weighted l^p norm by

$$\|x\|_{w,p} = \left(w_1|x_1|^p + \ldots + w_n|x_n|^p\right)^{1/p}, \qquad 1 \leq p \leq \infty. \tag{2.5}$$

All norms in a finite dimensional space V are equivalent, in the sense that for any two norms in V, $\|\cdot\|_\alpha$ and $\|\cdot\|_\beta$, there are two constants c and C such that

$$\|x\|_\alpha \leq C\|x\|_\beta, \tag{2.6}$$

$$\|x\|_\beta \leq c\|x\|_\alpha, \tag{2.7}$$

for all $x \in V$.

If V is a space of functions defined on a interval $[a, b]$, for example $C[a, b]$, the corresponding norms to (2.1)-(2.4) are given by

$$\|u\|_1 = \int_a^b |u(x)|dx, \tag{2.8}$$

$$\|u\|_2 = \left(\int_a^b |u(x)|^2 dx\right)^{1/2}, \tag{2.9}$$

$$\|u\|_\infty = \sup_{x\in[a,b]} |u(x)|, \tag{2.10}$$

$$\|u\|_p = \left(\int_a^b |u(x)|^p dx\right)^{1/p}, \qquad 1 \leq p \leq \infty \tag{2.11}$$

and are called the L^1, L^2, L^∞, and L^p norms, respectively. Similarly to (2.5) we can defined a weighted L^p norm by

$$\|u\|_p = \left(\int_a^b w(x)|u(x)|^p dx \right)^{1/p}, \qquad 1 \le p \le \infty, \tag{2.12}$$

where w is a given positive weight function defined in $[a, b]$. If $w(x) \ge 0$, we get a semi-norm.

Lemma 1. *Let $\|\cdot\|$ be a norm on a vector space V then*

$$|\, \|x\| - \|y\| \,| \le \|x - y\|. \tag{2.13}$$

This lemma implies that a norm is a continuous function (on V to $\mathbb{R}$).

Proof. $\|x\| = \|x - y + y\| \le \|x - y\| + \|y\|$ which gives that

$$\|x\| - \|y\| \le \|x - y\|. \tag{2.14}$$

By reversing the roles of x and y we also get

$$\|y\| - \|x\| \le \|x - y\|. \tag{2.15}$$

□

2.2 Uniform Polynomial Approximation

There is a fundamental result in approximation theory: *any continuous function on a closed, bounded interval can be approximated uniformly, i.e. in the $\|\cdot\|_\infty$ norm, with arbitrary accuracy by a polynomial.* This is the celebrated Weierstrass approximation theorem. We are going to present a constructive proof due to S. Bernstein, which uses a class of polynomials that have found widespread applications in computer graphics and animation. Historically, the use of these so-called Bernstein polynomials in computer assisted design (CAD) was introduced by two engineers working in the French car industry: Pierre Bézier at Renault and Paul de Casteljau at Citroën.

2.2.1 Bernstein Polynomials and Bézier Curves

Given a function f on $[0,1]$, the Bernstein polynomial of degree $n \geq 1$ is defined by

$$B_n f(x) = \sum_{k=0}^{n} f(k/n) \binom{n}{k} x^k (1-x)^{n-k}, \tag{2.16}$$

where

$$\binom{n}{k} = \frac{n!}{(n-k)!k!}, \quad k = 0, \ldots, n \tag{2.17}$$

are the binomial coefficients. Note that $B_n f(0) = f(0)$ and $B_n f(1) = f(1)$ for all n. The terms

$$b_{k,n}(x) = \binom{n}{k} x^k (1-x)^{n-k}, \quad k = 0, \ldots, n, \tag{2.18}$$

which are all nonnegative, are called the Bernstein basis polynomials and can be viewed as x-dependent weights that sum up to one:

$$\sum_{k=0}^{n} b_{k,n}(x) = \sum_{k=0}^{n} \binom{n}{k} x^k (1-x)^{n-k} = [x + (1-x)]^n = 1. \tag{2.19}$$

Thus, for each $x \in [0,1]$, $B_n f(x)$ represents a weighted average of the values of f at $0, 1/n, 2/n, \ldots, 1$. Moreover, as n increases the weights $b_{k,n}(x)$, for $0 < x < 1$, concentrate more and more around the points k/n close to x as Fig. 2.1 indicates for $b_{k,n}(0.5)$.

For $n = 1$, the Bernstein polynomial is just the straight line connecting $f(0)$ and $f(1)$, $B_1 f(x) = (1-x)f(0) + x f(1)$. Given two points $\mathbf{P_0}$ and $\mathbf{P_1}$ in the plane or in space, the segment of the straight line connecting them can be written in parametric form as

$$\mathbf{B}_1(t) = (1-t)\mathbf{P_0} + t\,\mathbf{P_1}, \qquad t \in [0,1]. \tag{2.20}$$

With three points, $\mathbf{P_0}, \mathbf{P_1}, \mathbf{P_2}$, we can employ the quadratic Bernstein basis polynomials to get a more useful parametric curve

$$\mathbf{B}_2(t) = (1-t)^2 \mathbf{P_0} + 2t(1-t)\mathbf{P_1} + t^2\,\mathbf{P_2}, \qquad t \in [0,1]. \tag{2.21}$$

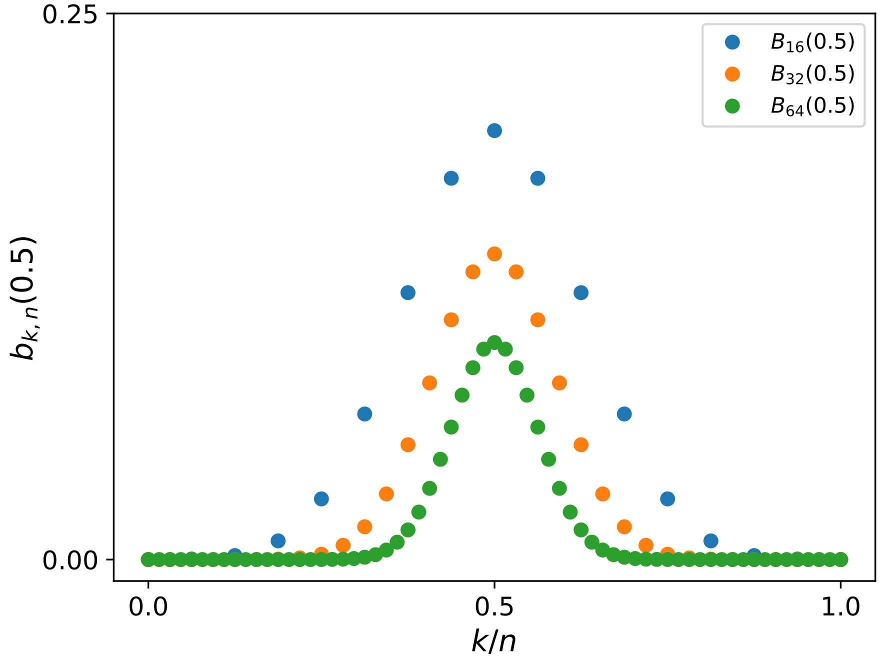

Figure 2.1: The Bernstein basis (weights) $b_{k,n}(x)$ for $x = 0.5$, $n = 16$, 32, and 64. Note how they concentrate more and more around $k/n \approx x$ as n increases.

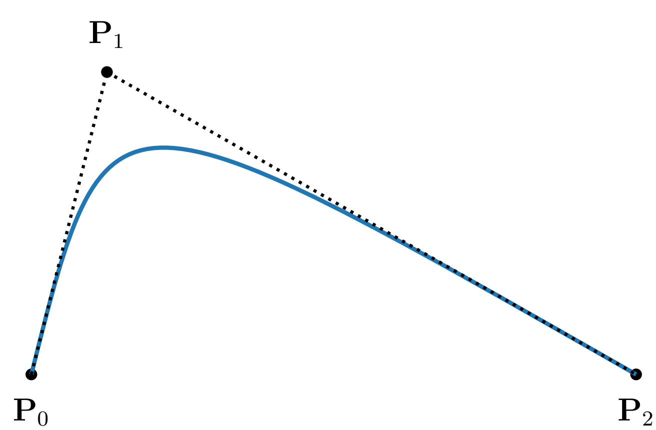

Figure 2.2: Quadratic Bézier curve.

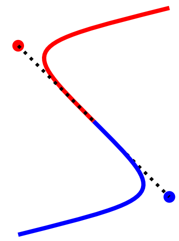

Figure 2.3: Example of a composite, quadratic C^1 Bézier curve with two pieces.

This curve connects again $\mathbf{P_0}$ and $\mathbf{P_2}$ but $\mathbf{P_1}$ can be used to control how the curve bends. More precisely, the tangents at the end points are $\mathbf{B}_2'(0) = 2(\mathbf{P_1} - \mathbf{P_0})$ and $\mathbf{B}_2'(1) = 2(\mathbf{P_2} - \mathbf{P_1})$, which intersect at $\mathbf{P_1}$, as Fig. 2.2 illustrates. These parametric curves formed with the Bernstein basis polynomials are called *Bézier curves* and have been widely employed in computer graphics, specially in the design of vector fonts, and in computer animation. A Bézier curve of degree $n \geq 1$ can be written in parametric form as

$$\mathbf{B}_n(t) = \sum_{k=0}^{n} b_{k,n}(t)\mathbf{P}_k, \qquad t \in [0,1]. \tag{2.22}$$

The points $\mathbf{P}_0, \mathbf{P}_1, \ldots, \mathbf{P}_n$ are called control points. Often, low degree (quadratic or cubic) Bézier curves are pieced together to represent of complex shapes. These *composite Bézier curves* are broadly used in font generation. For example, the TrueType font of most computers today is generated with composite, quadratic Bézier curves while the Metafont used in these pages, via LaTeX, employs composite, cubic Bézier curves. For each character, many pieces of Bézier curves are stitched together. To have some degree of smoothness (C^1), the common point for two pieces of a composite Bézier curve has to lie on the line connecting the two adjacent control points on either side as Fig. 2.3 shows.

Let us now do some algebra to prove some useful identities of the Bernstein polynomials. First, for $f(x) = x$ we have,

$$\begin{aligned}
\sum_{k=0}^{n} \frac{k}{n}\binom{n}{k} x^k (1-x)^{n-k} &= \sum_{k=1}^{n} \frac{kn!}{n(n-k)!k!} x^k (1-x)^{n-k} \\
&= x \sum_{k=1}^{n} \binom{n-1}{k-1} x^{k-1}(1-x)^{n-k} \\
&= x \sum_{k=0}^{n-1} \binom{n-1}{k} x^k (1-x)^{n-1-k} \\
&= x\left[x + (1-x)\right]^{n-1} = x.
\end{aligned} \tag{2.23}$$

Now for $f(x) = x^2$, we get

$$\sum_{k=0}^{n} \left(\frac{k}{n}\right)^2 \binom{n}{k} x^k (1-x)^{n-k} = \sum_{k=1}^{n} \frac{k}{n}\binom{n-1}{k-1} x^k (1-x)^{n-k} \tag{2.24}$$

and writing

$$\frac{k}{n} = \frac{k-1}{n} + \frac{1}{n} = \frac{n-1}{n}\frac{k-1}{n-1} + \frac{1}{n}, \tag{2.25}$$

we have

$$\begin{aligned}
\sum_{k=0}^{n} \left(\frac{k}{n}\right)^2 \binom{n}{k} x^k (1-x)^{n-k} &= \frac{n-1}{n} \sum_{k=2}^{n} \frac{k-1}{n-1}\binom{n-1}{k-1} x^k (1-x)^{n-k} \\
&\quad + \frac{1}{n} \sum_{k=1}^{n} \binom{n-1}{k-1} x^k (1-x)^{n-k} \\
&= \frac{n-1}{n} \sum_{k=2}^{n} \binom{n-2}{k-2} x^k (1-x)^{n-k} + \frac{x}{n} \\
&= \frac{n-1}{n} x^2 \sum_{k=0}^{n-2} \binom{n-2}{k} x^k (1-x)^{n-2-k} + \frac{x}{n}.
\end{aligned}$$

Thus,

$$\sum_{k=0}^{n} \left(\frac{k}{n}\right)^2 \binom{n}{k} x^k (1-x)^{n-k} = \frac{n-1}{n} x^2 + \frac{x}{n}. \tag{2.26}$$

Now, expanding $\left(\frac{k}{n}-x\right)^2$ and using (2.19), (2.23), and (2.26) it follows that

$$\sum_{k=0}^{n}\left(\frac{k}{n}-x\right)^2\binom{n}{k}x^k(1-x)^{n-k}=\frac{1}{n}x(1-x). \tag{2.27}$$

2.2.2 Weierstrass Approximation Theorem

Theorem 2.1. *(Weierstrass Approximation Theorem) Let f be a continuous function in a closed, bounded interval $[a,b]$. Given $\epsilon>0$, there is a polynomial p such that*

$$\max_{a\le x\le b}|f(x)-p(x)|<\epsilon.$$

Proof. We are going to work on the interval $[0,1]$. For a general interval $[a,b]$, we consider the change of variables $x=a+(b-a)t$ for $t\in[0,1]$ so that $F(t)=f(a+(b-a)t)$ is continuous in $[0,1]$.

Using (2.19), we have

$$f(x)-B_nf(x)=\sum_{k=0}^{n}\left[f(x)-f\left(\frac{k}{n}\right)\right]\binom{n}{k}x^k(1-x)^{n-k}. \tag{2.28}$$

Since f is continuous in $[0,1]$, it is also uniformly continuous. Thus, given $\epsilon>0$ there is $\delta(\epsilon)>0$, independent of x, such that

$$|f(x)-f(k/n)|<\frac{\epsilon}{2}\quad\text{if } |x-k/n|<\delta. \tag{2.29}$$

Moreover,

$$|f(x)-f(k/n)|\le 2\|f\|_\infty\quad\text{for all } x\in[0,1],\ k=0,1,\ldots,n. \tag{2.30}$$

We now split the sum in (2.28) in two sums, one over the points such that $|k/n-x|<\delta$ and the other over the points such that $|k/n-x|\ge\delta$:

$$\begin{aligned}f(x)-B_nf(x)=&\sum_{|k/n-x|<\delta}\left[f(x)-f\left(\frac{k}{n}\right)\right]\binom{n}{k}x^k(1-x)^{n-k}\\&+\sum_{|k/n-x|\ge\delta}\left[f(x)-f\left(\frac{k}{n}\right)\right]\binom{n}{k}x^k(1-x)^{n-k}.\end{aligned} \tag{2.31}$$

Using (2.29) and (2.19) it follows immediately that the first sum is bounded by $\epsilon/2$. For the second sum we have

$$\begin{aligned}
\sum_{|k/n-x|\geq\delta} &\left|f(x) - f\left(\frac{k}{n}\right)\right| \binom{n}{k} x^k(1-x)^{n-k} \\
&\leq 2\|f\|_\infty \sum_{|k/n-x|\geq\delta} \binom{n}{k} x^k(1-x)^{n-k} \\
&\leq \frac{2\|f\|_\infty}{\delta^2} \sum_{|k/n-x|\geq\delta} \left(\frac{k}{n}-x\right)^2 \binom{n}{k} x^k(1-x)^{n-k} \qquad (2.32)\\
&\leq \frac{2\|f\|_\infty}{\delta^2} \sum_{k=0}^{n} \left(\frac{k}{n}-x\right)^2 \binom{n}{k} x^k(1-x)^{n-k} \\
&= \frac{2\|f\|_\infty}{n\delta^2} x(1-x) \leq \frac{\|f\|_\infty}{2n\delta^2}.
\end{aligned}$$

Therefore, there is N such that for all $n \geq N$ the second sum in (2.31) is bounded by $\epsilon/2$ and this completes the proof. □

Figure 2.4 shows approximations of $f(x) = \sin(2\pi x)$ by Bernstein polynomials of degree $n = 10, 20, 40$. Observe that $\|f - B_n f\|_\infty$ decreases by roughly one half as n is doubled, suggesting a slow $O(1/n)$ convergence even for this smooth function.

2.3 Best Approximation

We just saw that any continuous function f on a closed, bounded interval can be approximated uniformly with arbitrary accuracy by a polynomial. Ideally, we would like to find the closest polynomial, say of degree at most n, to the function f when the distance is measured in the supremum (infinity) norm, or in any other norm we choose. There are three important elements in this general problem: the space of functions we want to approximate, the norm, and the family of approximating functions. The following definition makes this more precise.

Definition 2.1. *Given a normed, vector space V and a subspace W of V, $p^* \in W$ is called a best approximation of $f \in V$ by elements in W if*

$$\|f - p^*\| \leq \|f - p\|, \quad \text{for all } p \in W. \qquad (2.33)$$

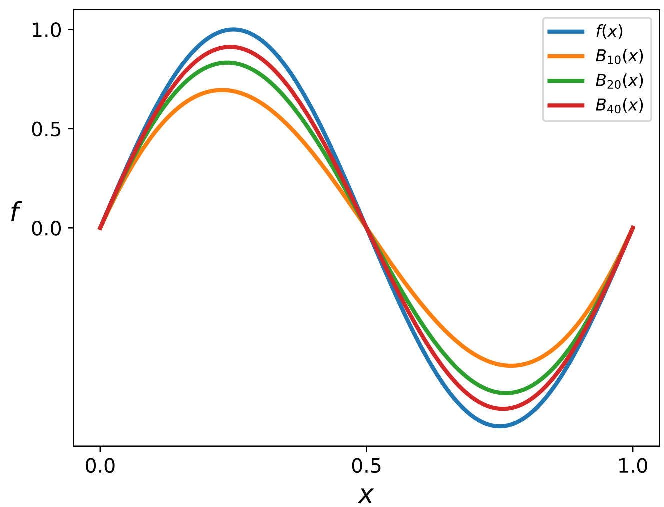

Figure 2.4: Approximation of $f(x) = \sin(2\pi x)$ on $[0, 1]$ by Bernstein polynomials.

For example, the normed, vector space V could be $C[a, b]$ with the supremum norm (2.10) and W could be the set of all polynomials of degree at most n, which henceforth we will denote by $\mathbb{P}_n$.

Theorem 2.2. *Let W be a finite-dimensional subspace of a normed, vector space V. Then, for every $f \in V$, there is at least one best approximation to f by elements in W.*

Proof. Since W is a subspace $0 \in W$ and for any candidate $p \in W$ for best approximation to f we must have

$$\|f - p\| \leq \|f - 0\| = \|f\|. \tag{2.34}$$

Therefore we can restrict our search to the set

$$F = \{p \in W : \|f - p\| \leq \|f\|\}. \tag{2.35}$$

F is closed and bounded and because W is finite-dimensional it follows that F is compact. Now, the function $p \mapsto \|f - p\|$ is continuous on this compact set and hence it attains its minimum in F. $\square$

If we remove the finite-dimensionality of W then we cannot guarantee that there is a best approximation as the following example shows.

Example 2.1. *Let $V = C[0,1/2]$ and W be the space of all polynomials (clearly a subspace of V). Take $f(x) = 1/(1-x)$ for $x \in [0,1/2]$ and note that*

$$\frac{1}{1-x} - (1 + x + x^2 + \ldots + x^N) = \frac{x^{N+1}}{1-x}. \tag{2.36}$$

Therefore, given $\epsilon > 0$ there is N such that

$$\max_{x\in[0,1/2]} \left| \frac{1}{1-x} - (1 + x + x^2 + \ldots + x^N) \right| = \left(\frac{1}{2}\right)^N < \epsilon. \tag{2.37}$$

Thus, if there is a best approximation p^* in the supremum norm, necessarily $\|f - p^*\|_\infty = 0$, which implies

$$p^*(x) = \frac{1}{1-x} \tag{2.38}$$

This is of course impossible since p is a polynomial.

Theorem 2.2 does not guarantee uniqueness of best approximation. Strict convexity of the norm gives us a sufficient condition.

Definition 2.2. *A norm $\|\cdot\|$ on a vector space V is strictly convex if for all $f \neq g$ in V with $\|f\| = \|g\| = 1$ then*

$$\|\theta f + (1-\theta)g\| < 1, \quad \textit{for all } 0 < \theta < 1.$$

In other words, a norm is strictly convex if its unit ball is strictly convex.

Note the use of the strict inequality $\|\theta f + (1-\theta)g\| < 1$ in the definition. The p-norm is strictly convex for $1 < p < \infty$ but not for $p = 1$ or $p = \infty$.

Theorem 2.3. *Let V be a vector space with a strictly convex norm, W a subspace of V, and $f \in V$. If p^* and q^* are best approximations of f in W then $p^* = q^*$.*

Proof. Let $M = \|f - p^*\| = \|f - q^*\|$. If $p^* \neq q^*$, by the strict convexity of the norm

$$\left\|\theta\left(\frac{f - p^*}{M}\right) + (1-\theta)\left(\frac{f - q^*}{M}\right)\right\| < 1, \quad \text{for all } 0 < \theta < 1. \tag{2.39}$$

That is,

$$\|\theta(f - p^*) + (1-\theta)(f - q^*)\| < M, \quad \text{for all } 0 < \theta < 1. \tag{2.40}$$

Taking $\theta = 1/2$ we get

$$\|f - \frac{1}{2}(p^* + q^*)\| < M, \tag{2.41}$$

which is impossible because $\frac{1}{2}(p^* + q^*)$ is in W and cannot be a better approximation. □

2.3.1 Best Uniform Polynomial Approximation

Given a continuous function f on a closed, bounded interval $[a, b]$ we know there is at least one best approximation p_n^* to f, in any given norm, by polynomials of degree at most n because the dimension of $\mathbb{P}_n$ is finite. The norm $\|\cdot\|_\infty$ is not strictly convex so Theorem 2.3 does not apply. However, due to a special property (called the *Haar property*) of the vector space $\mathbb{P}_n$, which is that the only element of $\mathbb{P}_n$ that has more than n roots is the zero element, we will see that the best uniform approximation out of $\mathbb{P}_n$ is unique and is characterized by a very peculiar property. Specifically, the error function

$$e_n(x) = f(x) - p_n^*(x), \quad x \in [a, b], \tag{2.42}$$

has to *equioscillate* at least $n+2$ points, between $+\|e_n\|_\infty$ and $-\|e_n\|_\infty$. That is, there are k points, $x_1, x_2, \ldots, x_k$, with $k \geq n+2$, such that

$$\begin{aligned}
e_n(x_1) &= \pm\|e_n\|_\infty \\
e_n(x_2) &= -e_n(x_1), \\
e_n(x_3) &= -e_n(x_2), \\
&\vdots \\
e_n(x_k) &= -e_n(x_{k-1}).
\end{aligned} \tag{2.43}$$

For if not, it would be possible to find a polynomial of degree at most n, with the same sign at the extremal points of e_n (at most n sign changes), and use this polynomial to decrease the value of $\|e_n\|_\infty$. This would contradict the fact that p_n^* is a best approximation. This is easy to see for $n = 0$ as it is impossible to find a polynomial of degree 0 (a constant) with one change of sign. This is the content of the next result.

Theorem 2.4. *The error $e_n = f - p_n^*$ has at least two extremal points, x_1 and x_2, in $[a, b]$ such that $|e_n(x_1)| = |e_n(x_2)| = \|e_n\|_\infty$ and $e_n(x_1) = -e_n(x_2)$ for all $n \geq 0$.*

Proof. The continuous function $|e_n(x)|$ attains its maximum $\|e_n\|_\infty$ in at least one point x_1 in $[a, b]$. Suppose $\|e_n\|_\infty = e_n(x_1)$ and that $e_n(x) > -\|e_n\|_\infty$ for all $x \in [a, b]$. Then, $m = \min_{x\in[a,b]} e_n(x) > -\|e_n\|_\infty$ and we have some room to decrease $\|e_n\|_\infty$ by shifting down e_n a suitable amount c. In particular, if take c as one half the gap between the minimum m of e_n and $-\|e_n\|_\infty$,

$$c = \frac{1}{2}\left(m + \|e_n\|_\infty\right) > 0, \tag{2.44}$$

and subtract it to e_n, as shown in Fig. 2.5, we have

$$-\|e_n\|_\infty + c \leq e_n(x) - c \leq \|e_n\|_\infty - c. \tag{2.45}$$

Therefore, $\|e_n - c\|_\infty = \|f - (p_n^* + c)\|_\infty = \|e_n\|_\infty - c < \|e_n\|_\infty$ but $p_n^* + c \in \mathbb{P}_n$ so this is impossible since p_n^* is a best approximation. A similar argument can used when $e_n(x_1) = -\|e_n\|_\infty$. □

Before proceeding to the general case, let us look at the $n = 1$ situation. Suppose there are only two alternating extremal points x_1 and x_2 for e_1 as described in (2.43). We are going to construct a linear polynomial that has the same sign as e_1 at x_1 and x_2 and which can be used to decrease $\|e_1\|_\infty$. Suppose $e_1(x_1) = \|e_1\|_\infty$ and $e_1(x_2) = -\|e_1\|_\infty$. Since e_1 is continuous, we can find small closed intervals I_1 and I_2, containing x_1 and x_2, respectively, and such that

$$e_1(x) > \frac{\|e_1\|_\infty}{2} \quad \text{for all } x \in I_1, \tag{2.46}$$

$$e_1(x) < -\frac{\|e_1\|_\infty}{2} \quad \text{for all } x \in I_2. \tag{2.47}$$

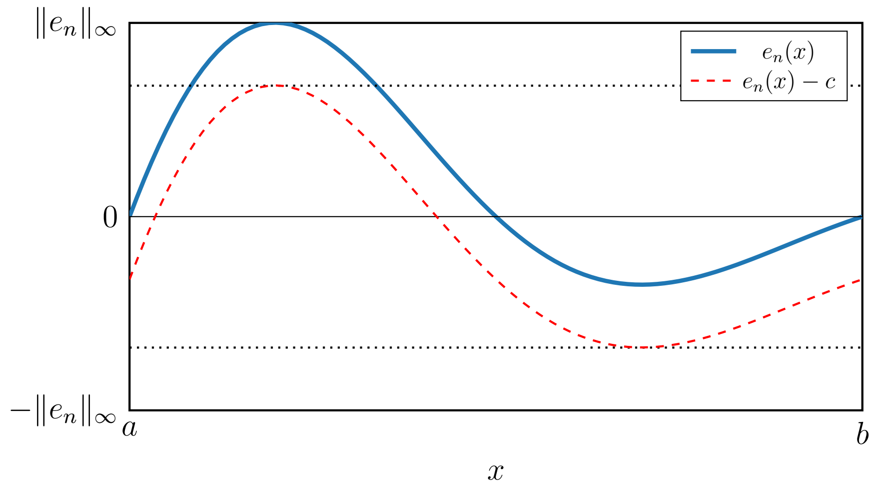

Figure 2.5: If the error function e_n does not equioscillate at least twice we could lower $\|e_n\|_\infty$ by an amount $c > 0$.

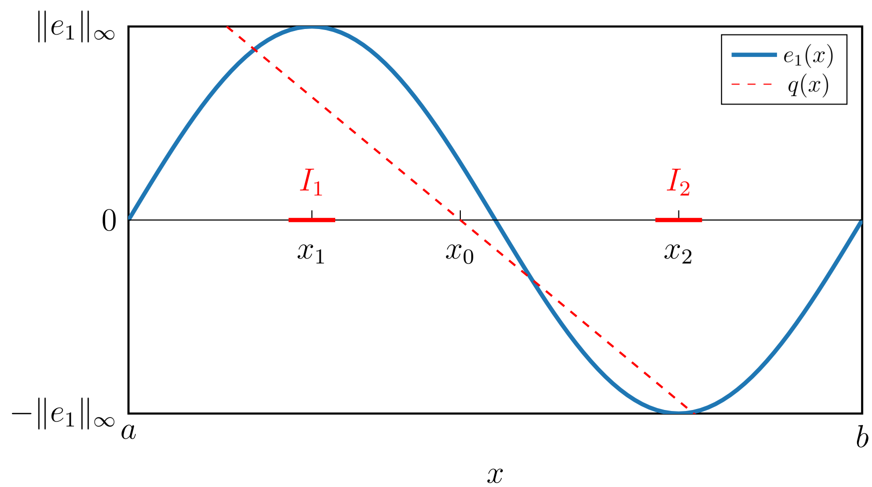

Figure 2.6: If e_1 equioscillates only twice, it would be possible to find a polynomial $q \in \mathbb{P}_1$ with the same sign around x_1 and x_2 as that of e_1 and, after a suitable scaling, use it to decrease the error.

Since I_1 and I_2 are disjoint sets, we can choose a point x_0 between the two intervals. Then, it is possible to find $q \in \mathbb{P}_1$ that passes through x_0 and that is positive in I_1 and negative in I_2 as Fig. 2.6 depicts. We are now going to pick a suitable constant $\alpha > 0$ such that $\|f - p_1^* - \alpha q\|_\infty < \|e_1\|_\infty$. Since $p_1^* + \alpha q \in \mathbb{P}_1$ this would be a contradiction to the fact that p_1^* is a best approximation.

Let $R = [a, b] \setminus (I_1 \cup I_2)$ and $d = \max_{x \in R} |e_1(x)|$. Clearly $d < \|e_1\|_\infty$. Choose α such that

$$0 < \alpha < \frac{1}{2\|q\|_\infty} (\|e_1\|_\infty - d) . \tag{2.48}$$

On I_1, we have

$$0 < \alpha q(x) < \frac{1}{2\|q\|_\infty} (\|e_1\|_\infty - d) q(x) \leq \frac{1}{2} (\|e_1\|_\infty - d) < e_1(x). \tag{2.49}$$

Therefore

$$|e_1(x) - \alpha q(x)| = e_1(x) - \alpha q(x) < \|e_1\|_\infty, \quad \text{for all } x \in I_1. \tag{2.50}$$

Similarly, on I_2, we can show that $|e_1(x) - \alpha q(x)| < \|e_1\|_\infty$. Finally, on R we have

$$|e_1(x) - \alpha q(x)| \leq |e_1(x)| + |\alpha q(x)| \leq d + \frac{1}{2} (\|e_1\|_\infty - d) < \|e_1\|_\infty. \tag{2.51}$$

Therefore, $\|e_1 - \alpha q\|_\infty = \|f - (p_1^* + \alpha q)\|_\infty < \|e_1\|_\infty$, which contradicts the best approximation assumption on p_1^*.

Theorem 2.5. *(Chebyshev Equioscillation Theorem) Let $f \in C[a, b]$. Then, p_n^* in $\mathbb{P}_n$ is a best uniform approximation of f if and only if there are at least $n + 2$ points in $[a, b]$, where the error $e_n = f - p_n^*$ equioscillates between the values $\pm\|e_n\|_\infty$ as defined in (2.43).*

Proof. We first prove that if the error $e_n = f - p_n^*$, for some $p_n^* \in \mathbb{P}_n$, equioscillates at least $n + 2$ times then p_n^* is a best approximation. Suppose the contrary. Then, there is $q_n \in \mathbb{P}_n$ such that

$$\|f - q_n\|_\infty < \|f - p_n^*\|_\infty. \tag{2.52}$$

Let $x_1, \ldots, x_k$, with $k \geq n+2$, be the points where e_n equioscillates. Then

$$|f(x_j) - q_n(x_j)| < |f(x_j) - p_n^*(x_j)|, \quad j = 1, \ldots, k \tag{2.53}$$

and since

$$f(x_j) - p_n^*(x_j) = -[f(x_{j+1}) - p_n^*(x_{j+1})], \quad j = 1, \ldots, k-1 \tag{2.54}$$

we have that

$$q_n(x_j) - p_n^*(x_j) = f(x_j) - p_n^*(x_j) - [f(x_j) - q_n(x_j)] \tag{2.55}$$

changes signs $k-1$ times, i.e. at least $n+1$ times. But $q_n - p_n^* \in \mathbb{P}_n$. Therefore $q_n = p_n^*$, which contradicts (2.52), and consequently p_n^* has to be a best uniform approximation of f.

For the other half of the proof the idea is the same as for $n = 1$ but we need to do more bookkeeping. We are going to partition $[a, b]$ into the union of sufficiently small subintervals so that we can guarantee that $|e_n(t) - e_n(s)| \leq \|e_n\|_\infty/2$ for any two points t and s in each of the subintervals. Let us label by $I_1, \ldots, I_k$, the subintervals on which $|e_n(x)|$ achieves its maximum $\|e_n\|_\infty$. Then, on each of these subintervals either $e_n(x) > \|e_n\|_\infty/2$ or $e_n(x) < -\|e_n\|_\infty/2$. We need to prove that e_n changes sign at least $n+1$ times.

Going from left to right, we can label the subintervals $I_1, \ldots, I_k$ as a $(+)$ or $(-)$ subinterval depending on the sign of e_n. For definiteness, suppose I_1 is a $(+)$ subinterval then we have the groups

$$\begin{array}{ll}
\{I_1, \ldots, I_{k_1}\}, & (+) \\
\{I_{k_1+1}, \ldots, I_{k_2}\}, & (-) \\
\vdots & \\
\{I_{k_m+1}, \ldots, I_k\}, & (-)^m.
\end{array}$$

We have m changes of sign so let us assume that $m \leq n$. We already know $m \geq 1$. Since the sets, I_{k_j} and I_{k_j+1} are disjoint for $j = 1, \ldots, m$, we can select points $t_1, \ldots, t_m$, such that $t_j > x$ for all $x \in I_{k_j}$ and $t_j < x$ for all $x \in I_{k_j+1}$. Then, the polynomial

$$q(x) = (t_1 - x)(t_2 - x) \cdots (t_m - x) \tag{2.56}$$

has the same sign as e_n in each of the extremal intervals $I_1, \ldots, I_k$ and $q \in \mathbb{P}_n$. The rest of the proof is as in the $n = 1$ case to show that $p_n^* + \alpha q$ would be a better approximation to f than p_n^*. $\square$

Theorem 2.6. *Let $f \in C[a,b]$. The best uniform approximation p_n^* to f by elements of $\mathbb{P}_n$ is unique.*

Proof. Suppose q_n^* is also a best approximation, i.e.

$$\|e_n\|_\infty = \|f - p_n^*\|_\infty = \|f - q_n^*\|_\infty.$$

Then, the midpoint $r = \frac{1}{2}(p_n^* + q_n^*)$ is also a best approximation, for $r \in \mathbb{P}_n$ and

$$\begin{aligned} \|f - r\|_\infty &= \|\frac{1}{2}(f - p_n^*) + \frac{1}{2}(f - q_n^*)\|_\infty \\ &\le \frac{1}{2}\|f - p_n^*\|_\infty + \frac{1}{2}\|f - q_n^*\|_\infty = \|e_n\|_\infty. \end{aligned} \tag{2.57}$$

Let $x_1, \ldots, x_{n+2}$ be extremal points of $f - r$ with the alternating property (2.43), i.e. $f(x_j) - r(x_j) = (-1)^{m+j}\ \|e_n\|_\infty$ for some integer m and $j = 1, \ldots n+2$. This implies that

$$\frac{f(x_j) - p_n^*(x_j)}{2} + \frac{f(x_j) - q_n^*(x_j)}{2} = (-1)^{m+j}\ \|e_n\|_\infty, \quad j = 1, \ldots, n+2. \tag{2.58}$$

But $|f(x_j) - p_n^*(x_j)| \le \|e_n\|_\infty$ and $|f(x_j) - q_n^*(x_j)| \le \|e_n\|_\infty$. As a consequence,

$$f(x_j) - p_n^*(x_j) = f(x_j) - q_n^*(x_j) = (-1)^{m+j}\ \|e_n\|_\infty, \quad j = 1, \ldots, n+2, \tag{2.59}$$

and it follows that

$$p_n^*(x_j) = q_n^*(x_j), \quad j = 1, \ldots, n+2. \tag{2.60}$$

Therefore, $q_n^* = p_n^*$. $\square$

2.4 Chebyshev Polynomials

The best uniform approximation of $f(x) = x^{n+1}$ in $[-1,1]$ by polynomials of degree at most n can be found explicitly and the solution introduces one of the most useful and remarkable polynomials, the Chebyshev polynomials.

Let $p_n^* \in \mathbb{P}_n$ be the best uniform approximation to x^{n+1} in the interval $[-1,1]$ and as before define the error function as $e_n(x) = x^{n+1} - p_n^*(x)$. Note that since e_n is a monic polynomial (its leading coefficient is 1) of degree $n+1$, the problem of finding p_n^* is equivalent to finding, among all monic polynomials of degree $n+1$, the one with the smallest deviation (in absolute value) from zero in $[-1,1]$.

According to Theorem 2.5, there exist $n+2$ distinct points,

$$-1 \leq x_1 < x_2 < \cdots < x_{n+2} \leq 1, \tag{2.61}$$

such that

$$e_n^2(x_j) = \|e_n\|_\infty^2, \quad \text{for } j = 1, \ldots, n+2. \tag{2.62}$$

Now consider the polynomial

$$q(x) = \|e_n\|_\infty^2 - e_n^2(x). \tag{2.63}$$

Then, $q(x_j) = 0$ for $j = 1, \ldots, n+2$. Each of the points x_j in the interior of $[-1,1]$ is also a local minimum of q, then necessarily $q'(x_j) = 0$ for $j = 2, \ldots n+1$. Thus, the n points $x_2, \ldots, x_{n+1}$ are zeros of q of multiplicity at least two. But q is a nonzero polynomial of degree $2n+2$ exactly. Therefore, x_1 and x_{n+2} have to be simple zeros and so $x_1 = -1$ and $x_{n+2} = 1$. Note that the polynomial $p(x) = (1-x^2)[e_n'(x)]^2 \in \mathbb{P}_{2n+2}$ has the same zeros as q and so $p = cq$, for some constant c. Comparing the coefficient of the leading order term of p and q it follows that $c = (n+1)^2$. Therefore, e_n satisfies the ordinary differential equation

$$(1-x^2)[e_n'(x)]^2 = (n+1)^2 \left[\|e_n\|_\infty^2 - e_n^2(x)\right]. \tag{2.64}$$

We know $e_n' \in \mathbb{P}_n$ and its n zeros are the interior points $x_2, \ldots, x_{n+1}$. Therefore, e_n' cannot change sign in $[-1, x_2]$. Suppose it is nonnegative for $x \in [-1, x_2]$ (we reach the same conclusion if we assume $e_n'(x) \leq 0$) then, taking square roots in (2.64) we get

$$\frac{e_n'(x)}{\sqrt{\|e_n\|_\infty^2 - e_n^2(x)}} = \frac{n+1}{\sqrt{1-x^2}}, \quad \text{for } x \in [-1, x_2]. \tag{2.65}$$

We can integrate this ordinary differential equation using the trigonometric substitutions $e_n(x) = \|e_n\|_\infty \cos\phi$ and $x = \cos\theta$, for the left and the right hand side respectively, to obtain

$$-\cos^{-1}\left(\frac{e_n(x)}{\|e_n\|_\infty}\right) = -(n+1)\theta + C, \tag{2.66}$$

where C is a constant of integration. Choosing $C = 0$ (so that $e_n(1) = \|e_n\|_\infty$) we get

$$e_n(x) = \|e_n\|_\infty \cos\left[(n+1)\theta\right] \tag{2.67}$$

for $x = \cos\theta \in [-1, x_2]$ with $0 < \theta \leq \pi$. Recall that e_n is a polynomial of degree $n+1$ then so is $\cos[(n+1)\cos^{-1} x]$. Since these two polynomials agree in $[-1, x_2]$, (2.67) must also hold for all x in $[-1, 1]$.

Definition 2.3. *The Chebyshev polynomial (of the first kind) of degree n, T_n is defined by*

$$T_n(x) = \cos n\theta, \quad x = \cos\theta, \quad 0 \leq \theta \leq \pi. \tag{2.68}$$

Note that (2.68) only defines T_n for $x \in [-1, 1]$. However, once the coefficients of this polynomial are determined we can define it for any real (or complex) x.

Using the trigonometry identity

$$\cos(n+1)\theta + \cos(n-1)\theta = 2\cos n\theta \cos\theta, \tag{2.69}$$

we immediately get

$$T_{n+1}(\cos\theta) + T_{n-1}(\cos\theta) = 2T_n(\cos\theta)\cdot\cos\theta \tag{2.70}$$

and going back to the x variable we obtain the recursion formula

$$\begin{aligned} T_0(x) &= 1, \\ T_1(x) &= x, \\ T_{n+1}(x) &= 2xT_n(x) - T_{n-1}(x), \quad n \geq 1, \end{aligned} \tag{2.71}$$

which makes it more evident the T_n for $n = 0, 1, \ldots$ are indeed polynomials of exactly degree n. Let us generate a few of them.

$$\begin{aligned} &T_0(x) = 1, \\ &T_1(x) = x, \\ &T_2(x) = 2x\cdot x - 1 = 2x^2 - 1, \\ &T_3(x) = 2x\cdot(2x^2 - 1) - x = 4x^3 - 3x, \\ &T_4(x) = 2x(4x^3 - 3x) - (2x^2 - 1) = 8x^4 - 8x^2 + 1 \\ &T_5(x) = 2x(8x^4 - 8x^2 + 1) - (4x^3 - 3x) = 16x^5 - 20x^3 + 5x. \end{aligned} \tag{2.72}$$

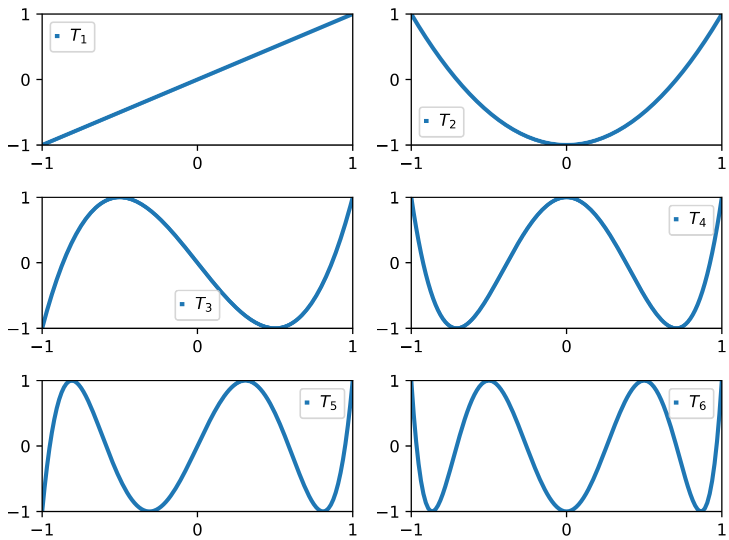

Figure 2.7: The Chebyshev polynomials T_n for $n = 1, 2, 3, 4, 5, 6$.

From these few Chebyshev polynomials, and from (2.71), we see that

$$T_n(x) = 2^{n-1}x^n + \text{lower order terms} \tag{2.73}$$

and that T_n is an even (odd) function of x if n is even (odd), i.e.

$$T_n(-x) = (-1)^n T_n(x). \tag{2.74}$$

The Chebyshev polynomials T_n, for $n = 1, 2, \ldots, 6$ are plotted in Fig. 2.7. Going back to (2.67), since the leading order coefficient of e_n is 1 and that of T_{n+1} is 2^n, it follows that $\|e_n\|_\infty = 2^{-n}$. Therefore

$$p_n^*(x) = x^{n+1} - \frac{1}{2^n}T_{n+1}(x) \tag{2.75}$$

is the best uniform approximation of x^{n+1} in $[-1, 1]$ by polynomials of degree at most n. Equivalently, as noted in the beginning of this section, the monic polynomial of degree n with smallest supremum norm in $[-1, 1]$ is

$$\tilde{T}_n(x) = \frac{1}{2^{n-1}}T_n(x). \tag{2.76}$$

Hence, for any other monic polynomial p of degree n

$$\max_{x\in[-1,1]} |p(x)| > \frac{1}{2^{n-1}}. \tag{2.77}$$

The zeros and extremal points of T_n are easy to find. Because $T_n(x) = \cos n\theta$ and $0 \leq \theta \leq \pi$, the zeros occur when θ is an odd multiple of $\pi/2$. Therefore,

$$\bar{x}_j = \cos\left(\frac{(2j+1)}{n}\frac{\pi}{2}\right) \quad j = 0, \ldots, n-1 \tag{2.78}$$

are the zeros of T_n.

The extremal points of T_n (the points x where $T_n(x) = \pm 1$) correspond to $n\theta = j\pi$ for $j = 0, 1, \ldots, n$, that is

$$x_j = \cos\left(\frac{j\pi}{n}\right), \quad j = 0, 1, \ldots, n. \tag{2.79}$$

These points are called Chebyshev, Chebyshev-Lobatto, or Gauss-Lobatto points and are very useful in applications. We will simply call them Chebyshev points or Chebyshev nodes. Figure 2.8 shows the Chebyshev nodes for $n = 16$. Note that they are more clustered at the end points of the interval and that x_j for $j = 1, \ldots, n-1$ are local extremal points. Therefore

$$T_n'(x_j) = 0, \quad \text{for } j = 1, \ldots, n-1. \tag{2.80}$$

In other words, the Chebyshev points (2.79) are the $n-1$ zeros of T_n' plus the end points $x_0 = 1$ and $x_n = -1$.

Using the Chain Rule we can differentiate T_n with respect to x:

$$T_n'(x) = -n \sin n\theta \frac{d\theta}{dx} = n\frac{\sin n\theta}{\sin\theta}, \quad (x = \cos\theta). \tag{2.81}$$

Therefore

$$\frac{T_{n+1}'(x)}{n+1} - \frac{T_{n-1}'(x)}{n-1} = \frac{1}{\sin\theta}\left[\sin(n+1)\theta - \sin(n-1)\theta\right] \tag{2.82}$$

and since $\sin(n+1)\theta - \sin(n-1)\theta = 2\sin\theta\cos n\theta$, we get that

$$\frac{T_{n+1}'(x)}{n+1} - \frac{T_{n-1}'(x)}{n-1} = 2T_n(x). \tag{2.83}$$

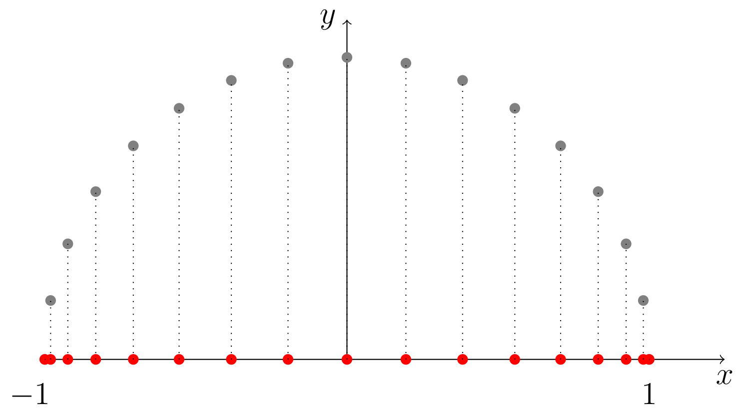

Figure 2.8: The Chebyshev nodes (red dots) $x_j = \cos(j\pi/n)$, $j = 0, 1, \ldots, n$ for $n = 16$. The gray dots on the semi-circle correspond to the equispaced angles $\theta_j = j\pi/n$, $j = 0, 1, \ldots, n$.

The polynomial

$$U_n(x) = \frac{T'_{n+1}(x)}{n+1} = \frac{\sin(n+1)\theta}{\sin\theta}, \quad (x = \cos\theta) \tag{2.84}$$

is called the second kind Chebyshev polynomial of degree n. Thus, the Chebyshev nodes (2.79) are the zeros of the polynomial

$$q_{n+1}(x) = (1 - x^2)U_{n-1}(x). \tag{2.85}$$

2.5 Bibliographical Notes

Section 2.1. A simple proof that all norms on a finite dimensional vector space are equivalent can be found in [Hac94], Section 2.6.

Section 2.2. A historical account of the invention of Bezier curves and surfaces used in CAD is given by G. Farin [Far02]. The excellent book on approximation theory by Rivlin [Riv81] contains Berstein's proof of Weierstrass theorem. Other fine textbooks on approximation theory that are the main sources used in this chapter and the next one are the classical books by Cheney [Che82] and Davis [Dav75]. There are many proofs of Weierstrass approximation theorem. One of great simplicity, due to H. Lebesgue, is masterfully presented by de la Vallée Poussin in his lectures on function approximation [dLVP19].

Section 2.3. This section follows the material on best approximation in [Riv81] (Introduction and Chapter 1) and in [Dav75] (Chapter 7). Example 2.1 is from Rivlin's book [Riv81].

Section 2.4. The construction of the solution to the best uniform approximation of x^{n+1} by polynomials of degree at most n, or equivalently the polynomial of degree $\leq n$ that deviates the least from zero, is given in [Riv81, Tim94]. In particular, Timan [Tim94] points out that Chebyshev arrived at his equi-oscillation theorem by considering this particular problem. An excellent reference for Chebyshev polynomials is the monograph by Rivlin [Riv20].

Chapter 3

Interpolation

One of the most useful tools for approximating a function or a given data set is interpolation, where the approximating function is required to coincide with a give set of values. In this chapter, we focus on polynomial and piecewise polynomial interpolation (splines), and trigonometric interpolation.

3.1 Polynomial Interpolation

The polynomial *interpolation problem* can be stated as follows: Given $n+1$ data points, $(x_0, f_0), (x_1, f_1)..., (x_n, f_n)$, where $x_0, x_1, \ldots, x_n$ are distinct, find a polynomial $p_n \in \mathbb{P}_n$, which satisfies the interpolation conditions:

$$
\begin{aligned}
p_n(x_0) &= f_0,\\
p_n(x_1) &= f_1,\\
&\vdots\\
p_n(x_n) &= f_n.
\end{aligned}
$$

The points $x_0, x_1, \ldots, x_n$ are called interpolation *nodes* and the values $f_0, f_1, \ldots, f_n$ are data supplied to us or can come from a function f we would like to approximate, in which case $f_j = f(x_j)$ for $j = 0, 1, \ldots, n$. Figure 3.1 illustrates the interpolation problem for $n = 6$.

Let us represent such polynomial as $p_n(x) = a_0 + a_1 x + \cdots + a_n x^n$. Then, the interpolation conditions implie

$$a_0 + a_1 x_0 + \cdots + a_n x_0^n = f_0,$$

$$a_0 + a_1 x_1 + \cdots + a_n x_1^n = f_1,$$

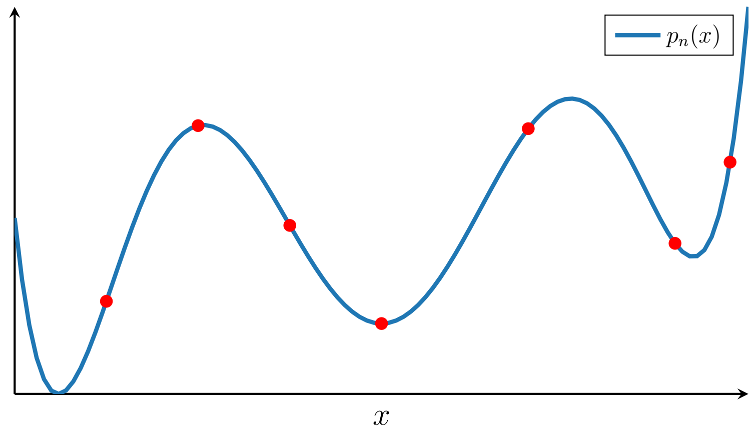

Figure 3.1: Given the data points $(x_0, f_0), \ldots, (x_n, f_n)$ (• , $n = 6$), the polynomial interpolation problem consists in finding a polynomial $p_n \in \mathbb{P}_n$ such that $p_n(x_j) = f_j$, for $j = 0, 1, \ldots, n$.

$$\vdots$$

$$a_0 + a_1 x_n + \cdots + a_n x_n^n = f_n.$$

This is a linear system of $n+1$ equations in $n+1$ unknowns (the polynomial coefficients $a_0, a_1, \ldots, a_n$). In matrix form:

$$\begin{bmatrix} 1 & x_0 & x_0^2 \cdots x_0^n \\ 1 & x_1 & x_1^2 \cdots x_1^n \\ \vdots & & \\ 1 & x_n & x_n^2 \cdots x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_n. \end{bmatrix} \tag{3.1}$$

Does this linear system have a solution? Is this solution unique? The answer is yes to both. Here is a simple proof. Take $f_j = 0$ for $j = 0, 1, \ldots, n$. Then $p_n(x_j) = 0$, for $j = 0, 1, ..., n$ but p_n is a polynomial of degree at most n, it cannot have $n+1$ zeros unless $p_n \equiv 0$, which implies $a_0 = a_1 = \cdots = a_n = 0$. That is, the homogenous problem associated with (3.1) has only the trivial solution. Therefore, (3.1) has a unique solution.

Example 3.1. *As an illustration let us consider interpolation by a polynomial $p_1 \in \mathbb{P}_1$. Suppose we are given (x_0, f_0) and (x_1, f_1) with $x_0 \neq x_1$. We wrote p_1 explicitly in* (1.2) *[with $x_0 = a$ and $x_1 = b$]. We write it now in a* *different form:*

$$p_1(x) = \left(\frac{x - x_1}{x_0 - x_1}\right) f_0 + \left(\frac{x - x_0}{x_1 - x_0}\right) f_1. \tag{3.2}$$

Clearly, this polynomial has degree at most 1 and satisfies the interpolation conditions:

$$p_1(x_0) = f_0, \tag{3.3}$$
$$p_1(x_1) = f_1. \tag{3.4}$$

Example 3.2. *Given (x_0, f_0), (x_1, f_1), and (x_2, f_2), with x_0, x_1 and x_2 distinct, let's construct $p_2 \in \mathbb{P}_2$ that interpolates these points. The form we have used for p_1 in (3.2) is suggestive of how we can write p_2:*

$$p_2(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} f_0 + \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} f_1 + \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} f_2.$$

If we define

$$l_0(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)}, \tag{3.5}$$
$$l_1(x) = \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)}, \tag{3.6}$$
$$l_2(x) = \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)}, \tag{3.7}$$

then we simply have

$$p_2(x) = l_0(x) f_0 + l_1(x) f_1 + l_2(x) f_2. \tag{3.8}$$

Note that each of the polynomials (3.5), (3.6), and (3.7) are exactly of degree 2 and they satisfy $l_j(x_k) = \delta_{jk}$ [1]. Therefore, it follows that p_2 given by (3.8) satisfies the desired interpolation conditions:

$$\begin{aligned} p_2(x_0) &= f_0, \\ p_2(x_1) &= f_1, \\ p_2(x_2) &= f_2. \end{aligned} \tag{3.9}$$

[1] δ_{jk} is the Kronecker delta, i.e. $\delta_{jk} = 0$ if $k \neq j$ and 1 if $k = j$.

We can now write down the polynomial p_n of degree at most n that interpolates $n+1$ given values, $(x_0, f_0), \ldots, (x_n, f_n)$, where the interpolation nodes $x_0, \ldots, x_n$ are assumed distinct. Define

$$\begin{aligned} l_j(x) &= \frac{(x-x_0)\cdots(x-x_{j-1})(x-x_{j+1})\cdots(x-x_n)}{(x_j-x_0)\cdots(x_j-x_{j-1})(x_j-x_{j+1})\cdots(x_j-x_n)} \\ &= \prod_{\substack{k=0\\k\neq j}}^{n} \frac{(x-x_k)}{(x_j-x_k)}, \qquad \text{for } j=0,1,...,n. \end{aligned} \tag{3.10}$$

These polynomials are called (polynomial) cardinal functions or fundamental polynomials of degree n. For simplicity, we are omitting in the notation their dependence on the $n+1$ nodes $x_0, x_1, \ldots, x_n$. Since $l_j(x_k) = \delta_{jk}$,

$$p_n(x) = l_0(x)f_0 + l_1(x)f_1 + \cdots + l_n(x)f_n = \sum_{j=0}^{n} l_j(x)f_j \tag{3.11}$$

interpolates the given data, i.e., it satisfies $p_n(x_j) = f_j$ for $j = 0, 1, 2, \ldots, n$. Relation (3.11) is called the *Lagrange form* of the interpolating polynomial.

The following result summarizes our discussion.

Theorem 3.1. *Given the $n+1$ values $(x_0, f_0), \ldots, (x_n, f_n)$, for $x_0, x_1, ..., x_n$ distinct. There is a unique polynomial p_n of degree at most n such that $p_n(x_j) = f_j$ for $j = 0, 1, \ldots, n$.*

Proof. p_n in (3.11) is of degree at most n and interpolates the data. Uniqueness follows from the fundamental theorem of algebra, as noted earlier. Suppose there is another polynomial q_n of degree at most n such that $q_n(x_j) = f_j$ for $j = 0, 1, \ldots, n$. Consider $r = p_n - q_n$. This is a polynomial of degree at most n and $r(x_j) = p_n(x_j) - q_n(x_j) = f_j - f_j = 0$ for $j = 0, 1, 2, \ldots, n$, which is impossible unless $r \equiv 0$. This implies $q_n = p_n$. □

3.1.1 Equispaced and Chebyshev Nodes

There are two special sets of nodes that are particularly important in applications. The uniform or equispaced nodes in an interval $[a, b]$ are given by

$$x_j = a + jh, \quad j = 0, 1, \ldots, n \;\text{ with } h = (b-a)/n. \tag{3.12}$$

These nodes yield very accurate and efficient *trigonometric* polynomial interpolation but are generally not good for (algebraic) polynomial interpolation as we will see later.

One of the preferred set of nodes for high order, accurate, and computationally efficient polynomial interpolation is the *Chebyshev* nodes, introduced in Section 2.4. In $[-1, 1]$, they are given by

$$x_j = \cos\left(\frac{j\pi}{n}\right), \qquad j = 0, \ldots, n, \tag{3.13}$$

and are the extremal points of the Chebyshev polynomial (2.68) of degree n . Note that these nodes are obtained from the equispaced points $\theta_j = j(\pi/n)$, $j = 0, 1, \ldots, n$ in $[0, \pi]$ by the one-to-one relation $x = \cos\theta$, for $\theta \in [0, \pi]$. As defined in (3.13), the nodes go from 1 to -1 so sometimes the alternative definition $x_j = -\cos(j\pi/n)$ is used. The Chebyshev nodes are not equally spaced and tend to cluster toward the end points of the interval (see Fig. 2.8). For a general interval $[a, b]$, we can do the simple change of variables

$$x = \frac{1}{2}(a + b) + \frac{1}{2}(b - a)t, \quad t \in [-1, 1], \tag{3.14}$$

to obtain the corresponding Chebyshev nodes in $[a, b]$.

3.2 Connection to Best Uniform Approximation

Given a continuous function f in $[a, b]$, its best uniform approximation p_n^* in $\mathbb{P}_n$ is characterized by an error, $e_n = f - p_n^*$, which equioscillates, as defined in (2.43), at least $n + 2$ times. Therefore e_n has a minimum of $n + 1$ zeros and consequently, there exists $x_0, \ldots, x_n$ such that

$$\begin{aligned} p_n^*(x_0) &= f(x_0), \\ p_n^*(x_1) &= f(x_1), \\ &\vdots \\ p_n^*(x_n) &= f(x_n). \end{aligned} \tag{3.15}$$

In other words, p_n^* is the polynomial of degree at most n that interpolates the function f at $n+1$ zeros of e_n. Rather than finding these zeros, a natural and more practical question is: given $(x_0, f(x_0)), (x_1, f(x_1)), \ldots, (x_n, f(x_n))$, where $x_0, \ldots, x_n$ in $[a, b]$ are distinct, how close is the interpolating polynomial $p_n \in \mathbb{P}_n$ of f at these nodes to the best uniform approximation $p_n^* \in \mathbb{P}_n$ of f?

To obtain a bound for $\|p_n - p_n^*\|_\infty$ we note that $p_n - p_n^*$ is a polynomial of degree at most n which interpolates $f - p_n^*$. Therefore, we can use Lagrange formula to represent it:

$$p_n(x) - p_n^*(x) = \sum_{j=0}^{n} l_j(x)[f(x_j) - p_n^*(x_j)]. \tag{3.16}$$

It then follows that

$$\|p_n - p_n^*\|_\infty \le \Lambda_n \|f - p_n^*\|_\infty, \tag{3.17}$$

where

$$\Lambda_n = \max_{a \le x \le b} \sum_{j=0}^{n} |l_j(x)| \tag{3.18}$$

is called the *Lebesgue constant* and depends only on the interpolation nodes, not on f. On the other hand, we have that

$$\|f - p_n\|_\infty = \|f - p_n^* - p_n + p_n^*\|_\infty \le \|f - p_n^*\|_\infty + \|p_n - p_n^*\|_\infty. \tag{3.19}$$

Using (3.17) we obtain

$$\|f - p_n\|_\infty \le (1 + \Lambda_n)\|f - p_n^*\|_\infty. \tag{3.20}$$

This inequality connects the interpolation error $\|f - p_n\|_\infty$ with the best approximation error $\|f - p_n^*\|_\infty$. What happens to these errors as we increase n? To make it more concrete, suppose we have a triangular array of nodes as follows:

$$\begin{array}{llll}
x_0^{(0)} & & & \\
x_0^{(1)} & x_1^{(1)} & & \\
x_0^{(2)} & x_1^{(2)} & x_2^{(2)} & \\
\vdots & & & \\
x_0^{(n)} & x_1^{(n)} & \ldots & x_n^{(n)} \\
\vdots & & &
\end{array} \tag{3.21}$$

where $a \leq x_0^{(n)} < x_1^{(n)} < \cdots < x_n^{(n)} \leq b$ for $n = 0, 1, \ldots$. Let p_n be the interpolating polynomial of degree at most n of f at the nodes corresponding to the $n+1$ row of (3.21). By the Weierstrass Approximation Theorem (p_n^* is a better approximation or at least as good as that provided by the Bernstein polynomial),

$$\|f - p_n^*\|_\infty \to 0 \quad \text{as } n \to \infty. \tag{3.22}$$

However, it can be proved that

$$\Lambda_n > \frac{2}{\pi^2} \log n - 1 \tag{3.23}$$

and hence the Lebesgue constant is not bounded in n. Therefore, we cannot conclude from (3.20) and (3.22) that $\|f - p_n\|_\infty$ as $n \to \infty$, i.e. that the interpolating polynomial, as we add more and more nodes, converges uniformly to f. In fact, given any distribution of points, organized in a triangular array (3.21), it is possible to construct a continuous function f for which its interpolating polynomial p_n (corresponding to the nodes on the n-th row of (3.21)) will not converge uniformly to f as $n \to \infty$.

Convergence of polynomial interpolation depends on both *the regularity of f and the distribution of the interpolation nodes*. We will discuss this further in Section 3.8

3.3 Barycentric Formula

The Lagrange form of the interpolating polynomial

$$p_n(x) = \sum_{j=0}^{n} l_j(x) f_j$$

is not convenient for computations. The evaluation of each l_j costs $O(n)$ operations and there are n of these evaluations for a total cost of $O(n^2)$ operations. Also, if we want to increase the degree of the polynomial we cannot reuse the work done in getting and evaluating a lower degree one. However, we can obtain a more efficient formula by rewriting the interpolating polynomial in the following way. Let

$$\omega(x) = (x - x_0)(x - x_1) \cdots (x - x_n). \tag{3.24}$$

Then, differentiating this polynomial of degree $n+1$ and evaluating at $x = x_j$ we get

$$\omega'(x_j) = \prod_{\substack{k=0 \\ k\neq j}}^{n} (x_j - x_k), \quad \text{for } j = 0, 1, \ldots, n, \tag{3.25}$$

Therefore, each of the fundamental polynomials may be written as

$$l_j(x) = \frac{\frac{\omega(x)}{x - x_j}}{\omega'(x_j)} = \frac{\omega(x)}{(x - x_j)\omega'(x_j)}, \quad \text{for } j = 0, 1, \ldots, n, \tag{3.26}$$

for $x \neq x_j$ and $l_j(x_j) = 1$ follows from L'Hôpital rule.

Defining

$$\lambda_j = \frac{1}{\omega'(x_j)}, \quad \text{for } j = 0, 1, \ldots, n, \tag{3.27}$$

we can recast Lagrange formula as

$$p_n(x) = \omega(x) \sum_{j=0}^{n} \frac{\lambda_j}{x - x_j} f_j. \tag{3.28}$$

This *modified Lagrange formula* is computationally more efficient than the original formula if we need to evaluate p_n at more than one point. This is because the λ_j's only depend on the interpolation nodes and can be precomputed for a one-time cost of $O(n^2)$ operations. After that, each evaluation of p_n only costs $O(n)$ operations. Unfortunately, the λ_j's as defined in (3.27) grow exponentially with the length of the interpolation interval so that (3.28) can only be used for moderate size n, without having to rescale the interval. We can eliminate this problem by noting that from (3.11) with $f(x) \equiv 1$ it follows that

$$1 = \sum_{j=0}^{n} l_j(x) = \omega(x) \sum_{j=0}^{n} \frac{\lambda_j}{x - x_j}. \tag{3.29}$$

Dividing (3.28) by (3.29), we get the so-called *barycentric formula* for interpolation:

$$p_n(x) = \frac{\displaystyle\sum_{j=0}^{n} \frac{\lambda_j}{x - x_j} f_j}{\displaystyle\sum_{j=0}^{n} \frac{\lambda_j}{x - x_j}}, \qquad \text{for } x \neq x_j, \quad j = 0, 1, \ldots, n. \tag{3.30}$$

If x coincides with one of the nodes x_j, the interpolation property $p_n(x_j) = f_j$ should be used.

The numbers λ_j depend only on the nodes $x_0, x_1, ..., x_n$ and not on given values $f_0, f_1, ..., f_n$. We can obtain them explicitly for both the Chebyshev nodes (3.13) and for the equally spaced nodes (3.12) and can be precomputed efficiently for a general set of nodes.

3.3.1 Barycentric Weights for Chebyshev Nodes

The Chebyshev nodes are the zeros of $q_{n+1}(x) = (1 - x^2)U_{n-1}(x)$, where $U_{n-1}(x) = \sin n\theta / \sin\theta$, $x = \cos\theta$ is the Chebyshev polynomial of the second kind of degree $n - 1$, with leading order coefficient 2^{n-1} [see Section 2.4]. Since the λ_j's can be defined up to a multiplicative constant (which would cancel out in the barycentric formula) we can take λ_j to be proportional to $1/q'_{n+1}(x_j)$. Since

$$q_{n+1}(x) = \sin\theta \sin n\theta, \tag{3.31}$$

differentiating we get

$$q'_{n+1}(x) = -n\cos n\theta - \sin n\theta \cot\theta. \tag{3.32}$$

Thus,

$$q'_{n+1}(x_j) = \begin{cases} -2n, \text{ for } j = 0, \\ -(-1)^j n, \text{ for } j = 1, \ldots, n-1, \\ -2n\ (-1)^n \text{ for } j = n. \end{cases} \tag{3.33}$$

We can factor out $-n$ in (3.33) to obtain the barycentric weights for the Chebyshev points

$$\lambda_j = \begin{cases} \frac{1}{2}, \text{ for } j = 0, \\ (-1)^j, \text{ for } j = 1, \ldots, n-1, \\ \frac{1}{2}\ (-1)^n \text{ for } j = n. \end{cases} \tag{3.34}$$

Note that for a general interval $[a, b]$, the term $(a + b)/2$ in the change of variables (3.14) cancels out in (3.25) but we gain an extra factor of $[(b-a)/2]^n$. However, this factor can be omitted as it does not alter the barycentric formula. Therefore, the same barycentric weights (3.34) can also be used for the Chebyshev nodes in an interval $[a, b]$.

3.3.2 Barycentric Weights for Equispaced Nodes

For equispaced points, $x_j = x_0 + jh$, $j = 0, 1, \ldots, n$ we have

$$
\begin{aligned}
\lambda_j &= \frac{1}{(x_j - x_0)\cdots(x_j - x_{j-1})(x_j - x_{j+1})\cdots(x_j - x_n)} \\
&= \frac{1}{(jh)[(j-1)h]\cdots(h)(-h)(-2h)\cdots(j-n)h} \\
&= \frac{1}{(-1)^{n-j}h^n[j(j-1)\cdots 1][1\cdot 2\cdots(n-j)]} \\
&= \frac{1}{(-1)^{n-j}h^n n!}\;\frac{n!}{j!(n-j)!} \\
&= \frac{1}{(-1)^n h^n n!}\;(-1)^j\binom{n}{j}.
\end{aligned}
$$

We can omit the factor $1/((-1)^n h^n n!)$ because it cancels out in the barycentric formula. Thus, for equispaced nodes we can use

$$
\lambda_j = (-1)^j\binom{n}{j}, \quad j = 0, 1, \ldots n. \tag{3.35}
$$

Note that in this case the λ_j's grow very rapidly with n, limiting the use of the barycentric formula to only moderate size n for equispaced nodes. However, as we will see, equispaced nodes are not a good choice for accurate, high order polynomial interpolation in the first place.

3.3.3 Barycentric Weights for General Sets of Nodes

The barycentric weights for a general set of nodes can be computed efficiently by using the definition (3.27), i.e.

$$\lambda_j = \frac{1}{\prod_{\substack{k=0\\k\neq j}}^{n}(x_j - x_k)}, \quad j = 0, 1, \dots n \tag{3.36}$$

and by noting the following. Suppose we have the barycentric weights for the nodes $x_0, x_1, \dots, x_{m-1}$ and let's call these $\lambda_j^{(m-1)}$, for $j = 0, 1, \dots, m-1$. Then, the barycentric weights $\lambda_j^{(m)}$ for the set of nodes $x_0, x_1, \dots, x_m$ can be computed reusing the previous values:

$$\lambda_j^{(m)} = \frac{\lambda_j^{(m-1)}}{x_j - x_m}, \quad \text{for } j = 0, 1, \dots m-1 \tag{3.37}$$

and for $j = m$ we employ directly the definition:

$$\lambda_m^{(m)} = \frac{1}{\prod_{k=0}^{m-1}(x_m - x_k)}. \tag{3.38}$$

Algorithm 3.1 shows the procedure in pseudo-code.

Algorithm 3.1 Barycentric weights for general nodes

$\lambda_0^{(0)} \leftarrow 1$
for $m = 1, \dots, n$ **do**
  for $j = 0, \dots, m-1$ **do**
    $\lambda_j^{(m)} \leftarrow \frac{\lambda_j^{(m-1)}}{x_j - x_m}$
  end for
  $\lambda_m^{(m)} \leftarrow \frac{1}{\prod_{k=0}^{m-1}(x_m - x_k)}$
end for

3.4 Newton's Form and Divided Differences

There is another representation of the interpolating polynomial p_n that is convenient for the derivation of some numerical methods and for the evaluation of relatively low order p_n. The idea of this representation, due to Newton, is to use successively lower order polynomials for constructing p_n.

Suppose we have gotten $p_{n-1} \in \mathbb{P}_{n-1}$, the interpolating polynomial of $(x_0, f_0), (x_1, f_1), \ldots, (x_{n-1}, f_{n-1})$ and we would like to obtain $p_n \in \mathbb{P}_n$, the interpolating polynomial of $(x_0, f_0), (x_1, f_1), \ldots, (x_n, f_n)$ by reusing p_{n-1}. The difference between these polynomials, $r = p_n - p_{n-1}$, is a polynomial of degree at most n. Moreover, for $j = 0, \ldots, n-1$

$$r(x_j) = p_n(x_j) - p_{n-1}(x_j) = f_j - f_j = 0. \tag{3.39}$$

Therefore, r can be factored as

$$r(x) = c_n(x - x_0)(x - x_1)\cdots(x - x_{n-1}). \tag{3.40}$$

The constant c_n is called the *n-th divided difference* of $f = [f_0, f_1, \ldots, f_n]$ with respect to $x_0, x_1, ..., x_n$, and is usually denoted by $f[x_0, \ldots, x_n]$. Thus, we have

$$p_n(x) = p_{n-1}(x) + f[x_0, \ldots, x_n](x - x_0)(x - x_1)\cdots(x - x_{n-1}). \tag{3.41}$$

By the same argument, we have

$$p_{n-1}(x) = p_{n-2}(x) + f[x_0, \ldots, x_{n-1}](x - x_0)(x - x_1)\cdots(x - x_{n-2}), \tag{3.42}$$

etc. So we arrive at Newton's Form of p_n:

$$p_n(x) = f[x_0] + f[x_0, x_1](x - x_0) + \ldots + f[x_0, \ldots, x_n](x - x_0)\cdots(x - x_{n-1}). \tag{3.43}$$

Note that for $n = 1$,

$$p_1(x) = f[x_0] + f[x_0, x_1](x - x_0) \tag{3.44}$$

and the interpolation property gives

$$f_0 = p_1(x_0) = f[x_0], \tag{3.45}$$

$$f_1 = p_1(x_1) = f[x_0] + f[x_0, x_1](x_1 - x_0). \tag{3.46}$$

$$\tag{3.47}$$

Therefore

$$f[x_0] = f_0, \tag{3.48}$$

$$f[x_0, x_1] = \frac{f_1 - f_0}{x_1 - x_0}, \tag{3.49}$$

and

$$p_1(x) = f_0 + \left(\frac{f_1 - f_0}{x_1 - x_0}\right)(x - x_0)\,. \tag{3.50}$$

Define $f[x_j] = f_j$ for $j = 0, 1, ...n$. The following identity will allow us to compute all the required divided differences, order by order.

Theorem 3.2.

$$f[x_0, x_1, ..., x_k] = \frac{f[x_1, x_2, ..., x_k] - f[x_0, x_1, ..., x_{k-1}]}{x_k - x_0}. \tag{3.51}$$

Proof. Let p_{k-1} be the interpolating polynomial of degree at most $k-1$ of $(x_0, f_0), \ldots, (x_{k-1}, f_{k-1})$ and q_{k-1} the interpolating polynomial of degree at most $k-1$ of $(x_1, f_1), \ldots, (x_k, f_k)$. Then

$$p(x) = q_{k-1}(x) + \left(\frac{x - x_k}{x_k - x_0}\right)[q_{k-1}(x) - p_{k-1}(x)]. \tag{3.52}$$

is a polynomial of degree at most k and for $j = 1, 2, ...k-1$

$$p(x_j) = f_j + \left(\frac{x_j - x_k}{x_k - x_0}\right)[f_j - f_j] = f_j.$$

Moreover, $p(x_0) = p_{k-1}(x_0) = f_0$ and $p(x_k) = q_{k-1}(x_k) = f_k$. Therefore, $p = p_k$, the interpolation polynomial of degree at most k of the points $(x_0, f_0), (x_1, f_1), \ldots, (x_k, f_k)$. From (3.43), the leading order coefficient of p_k is $f[x_0, ..., x_k]$. Equating this with the leading order coefficient of p

$$\frac{f[x_1, ..., x_k] - f[x_0, x_1, ...x_{k-1}]}{x_k - x_0},$$

gives (3.51). □

To obtain the divided differences of p_n we construct a table using (3.51), computing all first order divided differences, then the second order ones, etc. This process is illustrated in Table 3.1 for $n = 3$.

Table 3.1: Table of divided differences for $n = 3$.

x_j	f_j	1st order	2nd order	3rd order
x_0	f_0			
x_1	f_1	$f[x_0, x_1] = \frac{f_1 - f_0}{x_1 - x_0}$		
x_2	f_2	$f[x_1, x_2] = \frac{f_2 - f_1}{x_2 - x_1}$	$f[x_0, x_1, x_2] = \frac{f[x_1,x_2] - f[x_0,x_1]}{x_2 - x_0}$	
x_3	f_3	$f[x_2, x_3] = \frac{f_3 - f_2}{x_3 - x_2}$	$f[x_1, x_2, x_3] = \frac{f[x_2,x_3] - f[x_1,x_2]}{x_3 - x_1}$	$f[x_0, x_1, x_2, x_3] = \frac{f[x_1,x_2,x_3] - f[x_0,x_1,x_2]}{x_3 - x_0}$

Example 3.3. *Take the data set* $(0, 1), (1, 2), (2, 5), (3, 10)$. *Then*

x_j	f_j			
0	1			
1	2	$\frac{2-1}{1-0} = 1$		
2	5	$\frac{5-2}{2-1} = 3$	$\frac{3-1}{2-0} = 1$	
3	10	$\frac{10-5}{3-2} = 5$	$\frac{5-3}{3-1} = 1$	$\frac{1-1}{3-0} = 0$

so

$$p_3(x) = 1 + 1(x - 0) + 1(x - 0)(x - 1) + 0(x - 0)(x - 1)(x - 2) = 1 + x^2.$$

After computing the divided differences, we need to evaluate p_n at a given point x. This can be done efficiently by suitably factoring it. For example, for $n = 3$ we have

$$\begin{aligned} p_3(x) &= c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1) + c_3(x - x_0)(x - x_1)(x - x_2) \\ &= c_0 + (x - x_0)\{c_1 + (x - x_1)[c_2 + (x - x_2)c_3]\} \end{aligned}$$

For general n we can use the *Horner*-like scheme in Algorithm 3.2 to get $y = p_n(x)$, given the divided difference coefficients $c_0, c_1, \ldots, c_n$ and the evaluation point x.

Algorithm 3.2 Horner Scheme to evaluate p_n at x in Newton's form

$y \leftarrow c_n$
for $k = n - 1, \ldots, 0$ **do**
  $y \leftarrow c_k + (x - x_k) * y$
end for

3.5 Cauchy Remainder

We now assume the data $f_j = f(x_j)$, $j = 0, 1, \ldots, n$ come from a sufficiently smooth function f, which we are trying to approximate with an interpolating polynomial p_n, and we focus on the error $f - p_n$ of such approximation.

In Chapter 1 we proved that if x_0, x_1, and x are in $[a, b]$ and $f \in C^2[a, b]$ then

$$f(x) - p_1(x) = \frac{1}{2} f''(\xi(x))(x - x_0)(x - x_1),$$

where p_1 is the polynomial of degree at most 1 that interpolates $(x_0, f(x_0))$, $(x_1, f(x_1))$ and $\xi(x) \in (a, b)$. The general result about the interpolation error is the following theorem:

Theorem 3.3. *Let $f \in C^{n+1}[a, b]$, $x_0, x_1, ..., x_n \in [a, b]$ distinct, $x \in [a, b]$, and p_n be the interpolation polynomial of degree at most n of f at $x_0, ..., x_n$ then*

$$f(x) - p_n(x) = \frac{1}{(n+1)!} f^{(n+1)}(\xi(x))(x - x_0)(x - x_1) \cdots (x - x_n), \quad (3.53)$$

where $\min\{x_0, \ldots, x_n, x\} < \xi(x) < \max\{x_0, \ldots, x_n, x\}$.

Proof. The right hand side of (3.53) is known as the Cauchy remainder.

For x equal to one of the nodes x_j the result is trivially true. Take x fixed not equal to any of the nodes and define

$$\phi(t) = f(t) - p_n(t) - [f(x) - p_n(x)] \frac{(t - x_0)(t - x_1) \cdots (t - x_n)}{(x - x_0)(x - x_1) \cdots (x - x_n)}. \quad (3.54)$$

Clearly, $\phi \in C^{n+1}[a, b]$ and vanishes at $t = x_0, x_1, ..., x_n, x$. That is, ϕ has at least $n + 2$ distinct zeros. Applying Rolle's Theorem $n + 1$ times we conclude that there exists a point $\xi(x) \in (a, b)$ such that $\phi^{(n+1)}(\xi(x)) = 0$ (see Fig. reffig:CauchyThm for an illustration of the $n = 4$ case). Therefore,

$$0 = \phi^{(n+1)}(\xi(x)) = f^{(n+1)}(\xi(x)) - [f(x) - p_n(x)] \frac{(n+1)!}{(x - x_0)(x - x_1) \cdots (x - x_n)}$$

from which (3.53) follows. Note that the repeated application of Rolle's theorem implies that $\xi(x)$ is between $\min\{x_0, x_1, ..., x_n, x\}$ and $\max\{x_0, x_1, ..., x_n, x\}$. $\square$

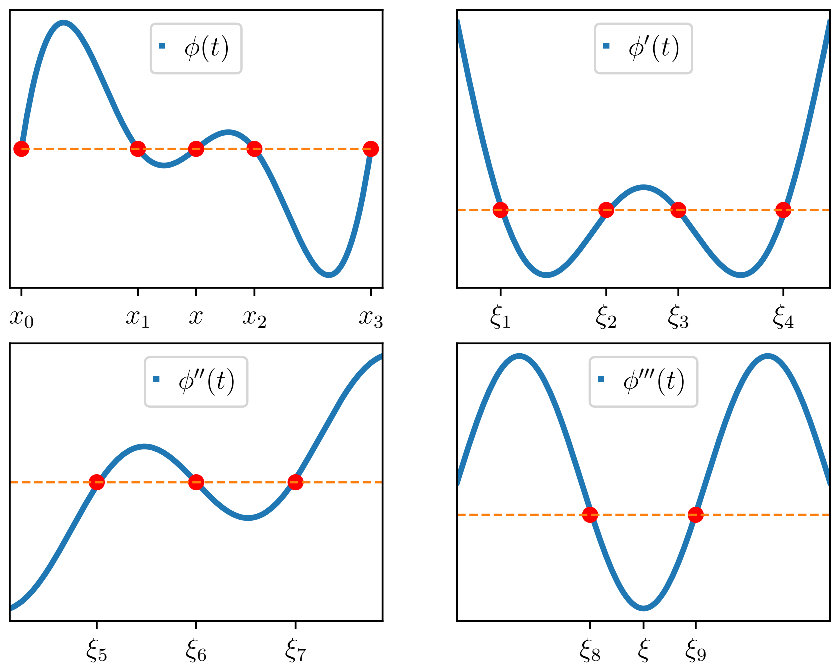

Figure 3.2: Successive application of Rolle's Theorem on $\phi(t)$ for Theorem 3.3, $n = 3$.

Example 3.4. *Let us find an approximation to* $\cos(0.8{*}\pi)$ *using interpolation of the values* $(0,1),(0.5,0),(1,-1),(1.5,0),(2,1)$. *We first employ Newton's divided differences to find* p_4.

x_j	f_j				
0.0	1				
0.5	0	-2			
1.0	-1	-2	0		
1.5	-1	2	4	$8/3$	
2.0	-1	2	0	$-8/3$	$-8/3$

Thus,

$$p_4(x) = 1 - 2x + \frac{8}{3}x(x-0.5)(x-1) - \frac{8}{3}x(x-0.5)(x-1)(x-1.5).$$

Then, $\cos(0.8\pi) \approx p_4(0.8) = -0.8176$. *Let us find an upper bound for the error using the Cauchy remainder. Since* $f(x) = cos(\pi x)$, $|f^5(x)| \leq \pi^5$ *for all* x. *Therefore,*

$$\begin{aligned} |\cos(0.8\pi) - p_4(0.8)| &\leq \frac{\pi^5}{5!}\,|(0.8-0)(0.8-0.5)(0.8-1)(0.8-1.5)(0.8-2)| \\ &\approx 0.10. \end{aligned} \tag{3.55}$$

This is a significant overestimate of the actual error $|\cos(0.8\pi) - p_4(0.8)| \approx 0.0086$ *because we replaced* $f^{(5)}(\xi(x))$ *with a global bound of the fifth derivative. Figure 3.3 shows a plot of* f *and* p_4. *Note that the interpolation nodes are equispaced and the largest error is produced toward the end of the interpolation interval.*

We have no control on the term $f^{(n+1)}(\xi(x))$ but we can choose the interpolation nodes $x_0, \ldots, x_n$ so that the factor

$$w(x) = (x - x_0)(x - x_1)\cdots(x - x_n) \tag{3.56}$$

is smallest as possible in the infinity norm. The function w is a monic polynomial of degree $n+1$ and we have proved in Section 2.4 that the Chebyshev

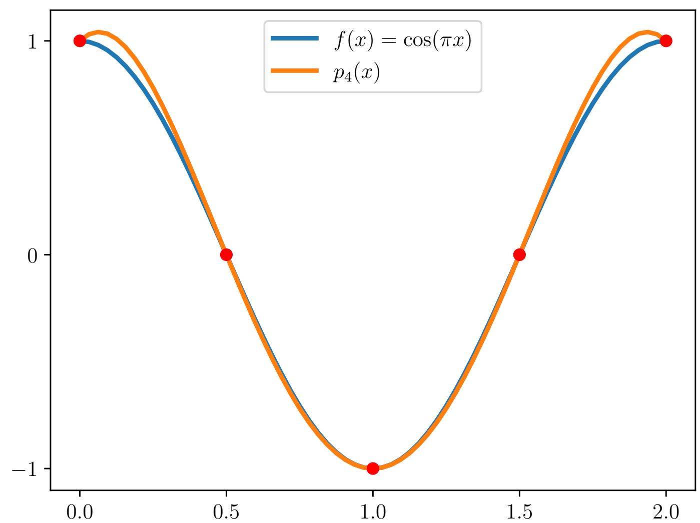

Figure 3.3: $f(x) = \cos(\pi x)$ in $[0, 2]$ and its interpolating polynomial p_4 at $x_j = j/2$, $j = 0, 1, 2, 3, 4$.

polynomial $\widetilde{T}_{n+1}$, defined in (2.76), is the monic polynomial of degree $n+1$ with smallest infinity norm in $[-1, 1]$. Hence, if the interpolation nodes are taken to be the zeros of $\widetilde{T}_{n+1}$, namely

$$x_j = \cos\left(\frac{(2j+1)}{n+1}\frac{\pi}{2}\right), \quad j = 0, 1, \ldots n. \tag{3.57}$$

$\|w\|_\infty$ is minimized and $\|w\|_\infty = 2^{-n}$. Figure 3.4 shows a plot of w for equispaced nodes and for the nodes (3.57) for $n = 10$ in $[-1, 1]$. For equispaced nodes, w oscillates unevenly with much larger (absolute) values toward the end of the interval than around the center. In contrast, for the nodes (3.57), w equioscillates between $\pm 1/2^n$, which is a small fraction of maximum amplitude of the equispaced-node w. The following theorem summarizes this observation.

Theorem 3.4. *Let Π_n be the interpolating polynomial of degree at most n of $f \in C^{n+1}[-1, 1]$ with respect to the nodes (3.57) then*

$$\|f - \Pi_n\|_\infty \leq \frac{1}{2^n(n+1)!}\|f^{n+1}\|_\infty. \tag{3.58}$$

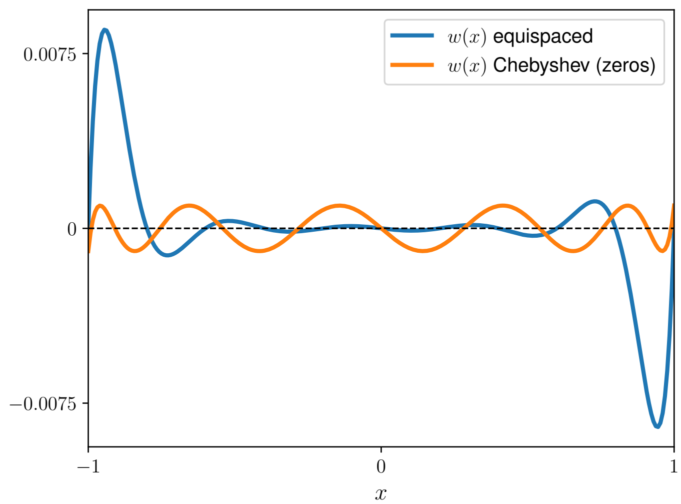

Figure 3.4: The node polynomial $w(x) = (x - x_0)\cdots(x - x_n)$, for equispaced nodes and for the zeros of T_{n+1} taken as nodes, $n = 10$.

The Chebyshev points,

$$x_j = \cos\left(\frac{j\pi}{n}\right), \quad j = 0, 1, \ldots, n, \tag{3.59}$$

which are the extremal points and not the zeros of the corresponding Chebyshev polynomial, do not minimize $\max_{x\in[-1,1]} |w(x)|$. However, they are nearly optimal. More precisely, since the Chebyshev nodes (3.59) are the zeros of the (monic) polynomial [see (2.85) and (3.31)]

$$\frac{1}{2^{n-1}}(1 - x^2)U_{n-1}(x) = \frac{1}{2^{n-1}} \sin\theta \sin n\theta, \quad x = \cos\theta. \tag{3.60}$$

We have that

$$\|w\|_\infty = \max_{x\in[-1,1]} \left| \frac{1}{2^{n-1}}(1 - x^2)U_{n-1}(x) \right| \leq \frac{1}{2^{n-1}}. \tag{3.61}$$

Thus, the Chebyshev nodes yield a $\|w\|_\infty$ of no more than a factor of two from the optimal value. Figure 3.5 compares w for equispaced nodes and for the Chebyshev nodes. For the latter, w is qualitatively very similar to that with the (3.57) nodes but, as we just proved, with an amplitude twice as large.

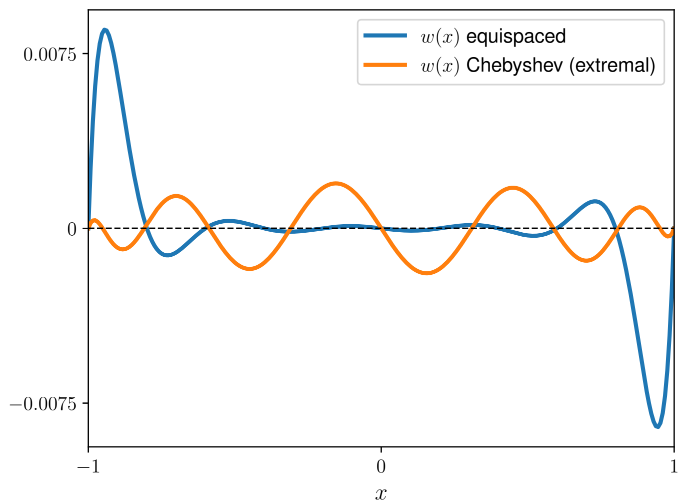

Figure 3.5: The node polynomial $w(x) = (x - x_0) \cdots (x - x_n)$, for equispaced nodes and for the Chebyshev nodes, the extremal points of T_n, $n = 10$.

3.6 Divided Differences and Derivatives

We now relate divided differences to the derivatives of f using the Cauchy remainder. Take an arbitrary point t distinct from $x_0, \ldots, x_n$. Let p_{n+1} be the interpolating polynomial of f at $x_0, \ldots, x_n, t$ and p_n that at $x_0, \ldots, x_n$. Then, Newton's formula (3.41) implies

$$p_{n+1}(x) = p_n(x) + f[x_0, \ldots, x_n, t](x - x_0)(x - x_1) \cdots (x - x_n). \tag{3.62}$$

Noting that $p_{n+1}(t) = f(t)$ we get

$$f(t) = p_n(t) + f[x_0, \ldots, x_n, t](t - x_0)(t - x_1) \cdots (t - x_n). \tag{3.63}$$

Since t was arbitrary we can set $t = x$ and obtain

$$f(x) = p_n(x) + f[x_0, \ldots, x_n, x](x - x_0)(x - x_1) \cdots (x - x_n), \tag{3.64}$$

and upon comparing with the Cauchy remainder we get

$$f[x_0, ..., x_n, x] = \frac{f^{(n+1)}(\xi(x))}{(n+1)!}. \tag{3.65}$$

If we set $x = x_{n+1}$ and relabel $n+1$ by k we have

$$f[x_0, ..., x_k] = \frac{1}{k!} f^{(k)}(\xi), \tag{3.66}$$

where $\min\{x_0, \dots, x_k\} < \xi < \max\{x_0, \dots, x_k\}$.

Suppose that we now let $x_1, ..., x_k \to x_0$. Then $\xi \to x_0$ and

$$\lim_{x_1,...,x_k \to x_0} f[x_0, ..., x_k] = \frac{1}{k!} f^{(k)}(x_0). \tag{3.67}$$

We can use this relation to define a divided difference where there are *coincident* nodes. For example $f[x_0, x_1]$ when $x_0 = x_1$ by $f[x_0, x_0] = f'(x_0)$, etc. This is going to be very useful for interpolating both function and derivative values.

3.7 Hermite Interpolation

The Hermite interpolation problem is: given values of f and some of its derivatives at the nodes $x_0, x_1, ..., x_n$, find the polynomial of smallest degree interpolating those values. This polynomial is called the *Hermite Interpolation Polynomial* and can be obtained with a minor modification to the Newton's form representation.

For example: Suppose we look for a polynomial p of lowest degree which satisfies the interpolation conditions:

$$\begin{aligned} p(x_0) &= f(x_0), \\ p'(x_0) &= f'(x_0), \\ p(x_1) &= f(x_1), \\ p'(x_1) &= f'(x_1). \end{aligned}$$

We can view this problem as a limiting case of polynomial interpolation of f at two pairs of coincident nodes, x_0, x_0, x_1, x_1 and we can use Newton's Interpolation form to obtain p. The table of divided differences, in view of (3.67), is

x_j	f_j			
x_0	$f(x_0)$			
x_0	$f(x_0)$	$f'(x_0)$		
x_1	$f(x_1)$	$f[x_0, x_1]$	$f[x_0, x_0, x_1]$	
x_1	$f(x_1)$	$f'(x_1)$	$f[x_0, x_1, x_1]$	$f[x_0, x_0, x_1, x_1]$

(3.68)

and

$$
\begin{aligned}
p(x) = f(x_0) + f'(x_0)(x - x_0) + f[x_0, x_0, x_1](x - x_0)^2 \\
+ f[x_0, x_0, x_1, x_1](x - x_0)^2(x - x_1).
\end{aligned}
\tag{3.69}
$$

Example 3.5. *Let $f(0) = 1$, $f'(0) = 0$ and $f(1) = \sqrt{2}$. Find the Hermite Interpolation Polynomial.*

We construct the table of divided differences as follows:

x_j	f_j		
0	1		
0	1	$f'(0) = 0$	
1	$\sqrt{2}$	$(\sqrt{2} - 1)/(1 - 0) = \sqrt{2} - 1$	$(\sqrt{2} - 1 - 0)/(1 - 0) = \sqrt{2} - 1$

and therefore

$$
p(x) = 1 + 0(x - 0) + (\sqrt{2} - 1)(x - 0)^2 = 1 + (\sqrt{2} - 1)x^2. \tag{3.70}
$$

3.8 Convergence of Polynomial Interpolation

From the Cauchy Remainder formula

$$
f(x) - p_n(x) = \frac{1}{(n+1)!} f^{(n+1)}(\xi(x))(x - x_0)(x - x_1) \cdots (x - x_n)
$$

it is clear that the accuracy of the interpolating polynomial p_n of f depends on both the *regularity of f* and the *distribution of the interpolation nodes* $x_0, x_1, \ldots, x_n$.

The function

$$
f(x) = \frac{1}{1 + 25x^2} \qquad x \in [-1, 1], \tag{3.71}
$$

provides a classical example, due to Runge, that illustrates the importance of node distribution. It has an infinite number of continuous derivatives, i.e. $f \in C^\infty[-1, 1]$ (in fact f is real analytic in the whole real line, i.e. it has a convergent Taylor series to $f(x)$ for every $x \in \mathbb{R}$). Nevertheless, for the equispaced nodes (3.12) p_n does not converge uniformly to $f(x)$ as $n \to \infty$. In fact it diverges quite dramatically toward the end points of the interval as Fig. 3.6 demonstrates. In contrast, as Fig. 3.7 shows, there is fast

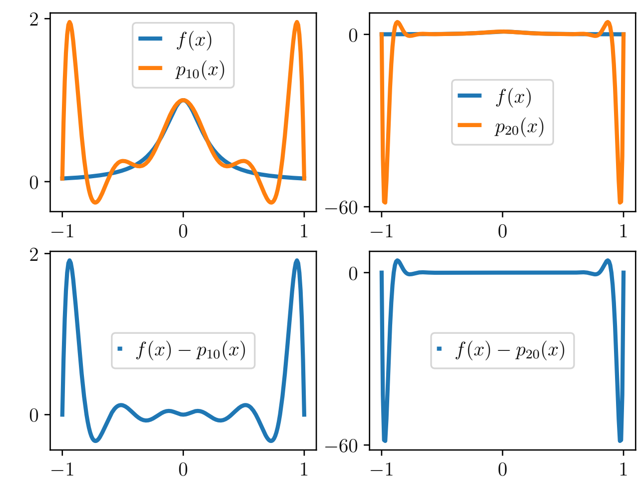

Figure 3.6: Lack of convergence of the interpolant p_n for $f(x) = 1/(1+25x^2)$ in $[-1, 1]$ using equispaced nodes. The first row shows plots of f and p_n ($n = 10, 20$) and the second row shows the corresponding error $f - p_n$.

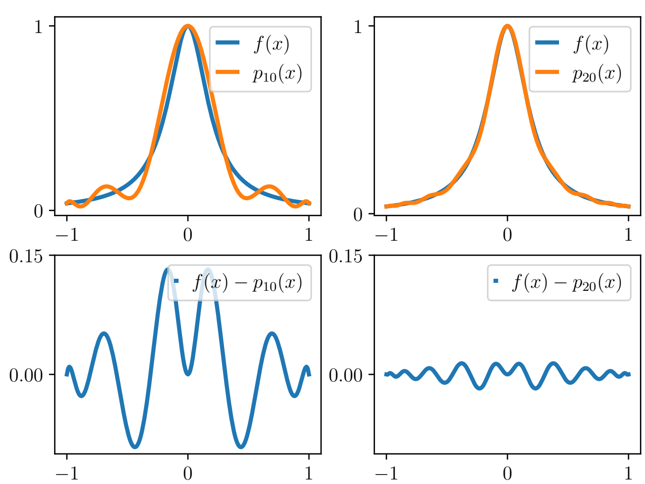

Figure 3.7: Convergence of the interpolant p_n for $f(x) = 1/(1 + 25x^2)$ in $[-1, 1]$ using Chebyshev nodes. The first row shows plots of f and p_n ($n = 10, 20$) and the second row shows the corresponding error $f - p_n$.

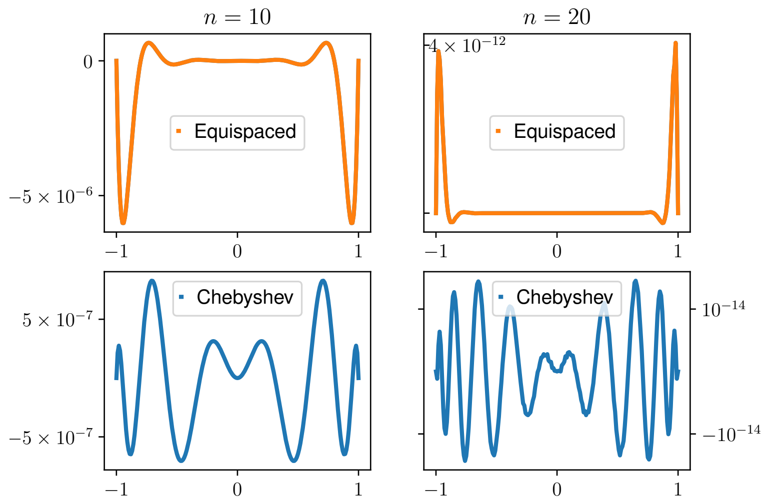

Figure 3.8: Fast convergence of the interpolant p_n for $f(x) = e^{-x^2}$ in $[-1, 1]$. Plots of the error $f - p_n$, $n = 10, 20$ for both the equispaced (first row) and the Chebyshev nodes (second row).

and uniform convergence of p_n to f when the Chebyshev nodes (3.13) are employed.

Now consider

$$f(x) = e^{-x^2}, \quad x \in [-1, 1]. \tag{3.72}$$

The interpolating polynomial p_n converges f, even when equispaced nodes are used. In fact, the convergence is noticeably fast. Figure 3.8 shows plots of the error $f - p_n$, $n = 10, 20$, for both equispaced and Chebyshev nodes. The interpolant p_{10} has already more than 5 and 6 digits of accuracy for the equispaced and Chebyshev nodes, respectively. Note that the error when using Chebyshev nodes is significantly smaller and more equidistributed throughout the interval $[-1, 1]$ than when using equispaced nodes. For the latter, as we have seen earlier, the error is substantially larger toward the endpoints of the interval than around the center.

What is so special about $f(x) = e^{-x^2}$? The function $f(z) = e^{-z^2}$, $z \in \mathbb{C}$ is analytic in the entire complex plane[2]. Using complex variables analysis, it

[2]A function of a complex variable $f(z)$ is said to be analytic in an open set D if it has

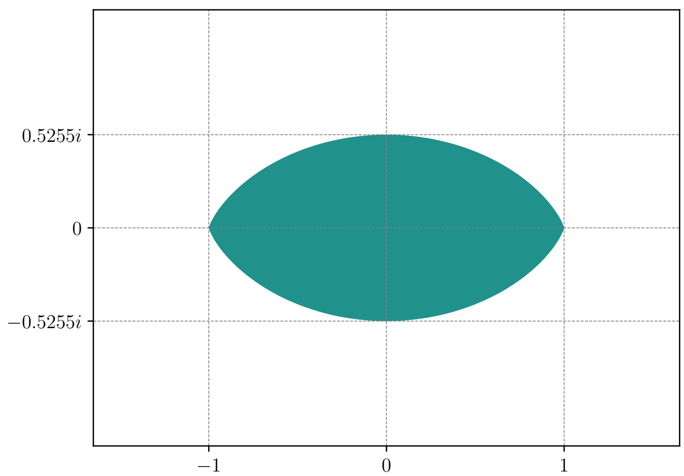

Figure 3.9: For uniform convergence of the interpolants p_n, $n = 1, 2, \ldots$ to f on $[-1, 1]$, with equi-spaced nodes, f must be analytic in the shaded, football-like region.

can be shown that if f is analytic in a sufficiently large region of the complex plane containing $[-1, 1]$ [3] then $\|f - p_n\|_\infty \to 0$. Just how large the region of analyticity needs to be? it depends on the asymptotic distribution of the nodes as $n \to \infty$. We will show next that for equispaced nodes, f must be analytic in the football-like region shown in Fig. 3.9 for p_n to converge uniformly to f in $[-1, 1]$, as $n \to \infty$. The Runge function (3.71) is not analytic in this region (it has singularities at $\pm i/5$) and hence the divergence of p_n. In contrast, for the Chebyshev nodes, it suffices that f be analytic in any region containing $[-1, 1]$, however thin this region might be, to guarantee the uniform convergence of p_n to f in $[-1, 1]$, as $n \to \infty$.

Let us consider the interpolation error, evaluated at a complex point

a derivative at every point of D. If f is analytic in D then all its derivatives exist and are analytic in D.

[3] Of course, the same arguments can be applied for a general interval $[a, b]$.

$z \in \mathbb{C}$ [4]:

$$f(z) - p_n(z) = f(z) - \sum_{j=0}^{n} l_j(z) f(x_j). \tag{3.73}$$

Employing (3.26), we can rewrite this as

$$f(z) - p_n(z) = f(z) - \sum_{j=0}^{n} \frac{\omega(z)}{(z - x_j)\omega'(x_j)} f(x_j), \tag{3.74}$$

where $\omega(z) = (z - x_0)(z - x_1) \cdots (z - x_n)$. Using the calculus of residues, the right hand side of (3.74) can be expressed as a contour integral:

$$f(z) - p_n(z) = \frac{1}{2\pi i} \oint_C \frac{\omega(z)}{\omega(\xi)} \frac{f(\xi)}{\xi - z} d\xi, \tag{3.75}$$

where C is a positively oriented closed curve that encloses $[-1, 1]$ and z but not any singularity of f. The integrand has a simple pole at $\xi = z$ with residue $f(z)$. It also has simple poles at $\xi = x_j$ for $j = 0, 1, \ldots, n$ with corresponding residues $-f(x_j)\omega(z)/[(z - x_j)\omega'(x_j)]$, which produces $-p_n(z)$. Expression (3.75) is called Hermite formula for the interpolation remainder.

To estimate $|f(z) - p_n(z)|$ using (3.75) we need to estimate $|\omega(z)|/|\omega(\xi)|$ for $\xi \in C$ and z inside C. To this end, it is convenient to choose a contour C on which $|\omega(\xi)|$ is approximately constant for sufficiently large n. Note that

$$|\omega(\xi)| = \prod_{j=0}^{n} |\xi - x_j| = \exp\left(\sum_{j=0}^{n} \log |\xi - x_j| \right). \tag{3.76}$$

In the limit as $n \to \infty$, we can view the interpolation nodes as a continuum of a density ρ (or limiting distribution), with

$$\int_{-1}^{1} \rho(x) dx = 1, \tag{3.77}$$

so that, for sufficiently large n,

$$\text{the total number of nodes in } [\alpha, \beta] = (n + 1) \int_{\alpha}^{\beta} \rho(x) dx, \tag{3.78}$$

[4] The rest of this section uses complex variables theory.

for $-1 \leq \alpha < \beta \leq 1$. Therefore, assuming the interpolation nodes have a limiting distribution ρ, we have

$$\frac{1}{n+1}\sum_{j=0}^{n} \log|\xi - x_j| \xrightarrow[n\to\infty]{} \int_{-1}^{1} \log|\xi - x|\rho(x)dx. \tag{3.79}$$

Let us define the function

$$\phi(\xi) = -\int_{-1}^{1} \log|\xi - x|\rho(x)dx. \tag{3.80}$$

Then, for sufficiently large n, $|\omega(z)|/|\omega(\xi)| \approx e^{-(n+1)[\phi(z)-\phi(\xi)]}$. The level curves of ϕ, i.e. the set of points $\xi \in \mathbb{C}$ such that $\phi(\xi) = c$, with c constant, approximate large circles for very large and negative values of c. As c is increased, the level curves shrink. Let z_0 be the singularity of f closest to the origin. Then, we can take any $\epsilon > 0$ and select C to be the level curve $\phi(\xi) = \phi(z_0)+\epsilon$ so that f is analytic on and inside C. Take z inside C. From (3.75), (3.79), and (3.80)

$$\begin{aligned} |f(z) - p_n(z)| &\leq \frac{1}{2\pi}\oint_C \frac{|\omega(z)|}{|\omega(\xi)|}\frac{|f(\xi)|}{|\xi - z|}ds \\ &\leq \text{constant } e^{-(n+1)[\phi(z)-(\phi(z_0)+\epsilon)]}. \end{aligned} \tag{3.81}$$

Therefore, it follows that $|f(z) - p_n(z)| \to 0$ as $n \to \infty$ and the convergence is exponential. Note that this holds as long as z is inside the chosen contour C. If z is outside the level curve $\phi(\xi) = \phi(z_0)$, i.e. $\phi(z) < \phi(z_0)$, then $|f(z)-p_n(z)|$ diverges exponentially. Therefore, p_n converges (uniformly) to f in $[-1, 1]$ if and only if f is analytic on and inside the smallest level curve of ϕ that contains $[-1, 1]$. More precisely, let γ be the supremum over all the values of c for which $[-1, 1]$ lies inside the level set curve $\phi(\xi) = c$. Define the region

$$D_\gamma = \{z \in \mathbb{C} : \phi(z) \geq \gamma\}. \tag{3.82}$$

Then, we have the following result.

Theorem 3.5. *The f be analytic in any region containing D_γ in its interior. Then,*

$$|f(z) - p_n(z)| \xrightarrow[n\to\infty]{} 0, \textit{ uniformily, for } z \in D_\gamma. \tag{3.83}$$

For equispaced nodes, the number of nodes is the same (asymptotically) for all intervals of the same length. Therefore, ρ is a constant. The normalization condition (3.77) implies that $\rho(x) = 1/2$ for equispaced points in $[-1, 1]$. It can be shown that with $\rho(x) = 1/2$ we get

$$\phi(\xi) = 1 - \frac{1}{2}\mathrm{Re}\left\{(\xi + 1)\log(\xi + 1) - (\xi - 1)\log(\xi - 1)\right\}. \tag{3.84}$$

The curve of ϕ that bounds D_γ for equispaced nodes is the one that passes through ± 1, has value $1 - \log 2$, and is shown in Fig. 3.9. It crosses the imaginary axis at $\pm 0.5255...i$. On the hand, the level curve that passes through $\pm i/5$ crosses the real axis at about $\pm 0.7267...$. Thus, there is uniform convergence of p_n to f in the reduced interval $[-0.72, 0.72]$.

The Chebyshev points $x_j = \cos\theta_j$, $j = 0, 1, \ldots, n$, are equispaced in θ ($\theta_j = j\pi/n$) and since

$$\int_\alpha^\beta \rho(x)dx = \int_{\cos^{-1}\beta}^{\cos^{-1}\alpha} \rho(\cos\theta)\sin\theta d\theta, \tag{3.85}$$

then $\rho(\cos\theta)\sin\theta = \rho(x)\sqrt{1 - x^2}$ must be constant. Using (3.77), it follows the density for Chebyshev nodes is

$$\rho(x) = \frac{1}{\pi\sqrt{1 - x^2}}, \quad x \in [-1, 1]. \tag{3.86}$$

With this node distribution it can be shown that

$$\phi(\xi) = \log\frac{2}{|\xi + \sqrt{\xi^2 - 1}|}. \tag{3.87}$$

The level curves of ϕ in this case are the points $\xi \in \mathbb{C}$ such that $|\xi + \sqrt{\xi^2 - 1}| = c$, with c constant. These are ellipses with foci at ± 1 as shown in Fig. 3.10. The level curve that passes through ± 1 degenerates into the interval $[-1, 1]$.

3.9 Piecewise Polynomial Interpolation

One way to avoid the oscillatory behavior of high-order interpolation when the interpolation nodes do not cluster appropriately is to employ low order polynomials in small subintervals.

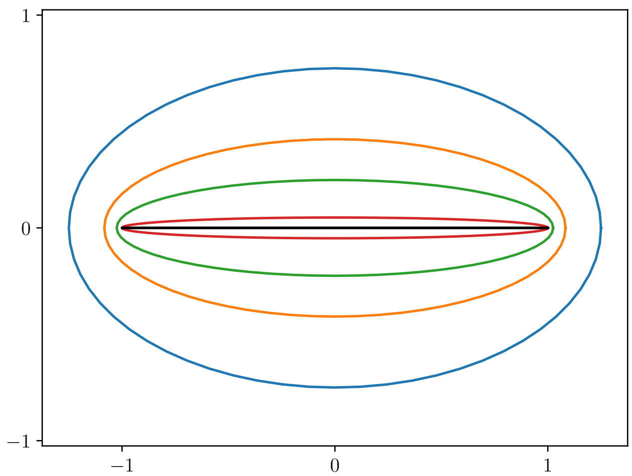

Figure 3.10: Some level curves of ϕ for the Chebyshev node distribution.

Given the nodes $a = x_0 < x_1 \ldots < x_n = b$ we can consider the subintervals $[x_0, x_1], ..., [x_{n-1}, x_n]$ and construct in each a polynomial degree at most k (for $k \geq 1$ small) that interpolates f. For $k = 1$, on each $[x_j, x_{j+1}]$, $j = 0, 1, \ldots, n-1$, we know there is a unique polynomial $s_j \in \mathbb{P}_1$ that interpolates f at x_j and x_{j+1}. Thus, there is a unique, continuous piecewise linear interpolant s of f at the given $n + 1$ nodes. We simply use $\mathbb{P}_1$ interpolation for each of its pieces:

$$s_j(x) = f_j + \frac{f_{j+1} - f_j}{x_{j+1} - x_j}(x - x_j), \quad x \in [x_j, x_{j+1}], \tag{3.88}$$

for $j = 0, 1, \ldots, n-1$ and we have set $f_j = f(x_j)$. Figure 3.11 shows an illustration of this piecewise linear interpolant s.

Assuming that $f \in C^2[a, b]$, we know that

$$f(x) - s(x) = \frac{1}{2} f''(\xi(x))(x - x_j)(x - x_{j+1}), \qquad x \in [x_j, x_{j+1}], \tag{3.89}$$

where $\xi(x)$ is some point between x_j and x_{j+1}. Then,

$$\max_{x_j \leq x \leq x_{j+1}} |f(x) - p(x)| \leq \frac{1}{2} \|f''\|_\infty \max_{x_j \leq x \leq x_{j+1}} |(x - x_j)(x - x_{j+1})|, \tag{3.90}$$

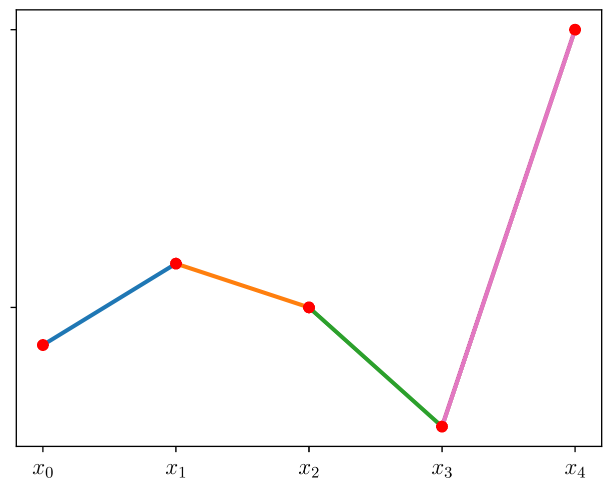

Figure 3.11: Piecewise linear interpolation.

where $\|f''\|_\infty$ is the sup norm of f'' over $[a, b]$. Now, the max at the right hand side is attained at the midpoint $(x_j + x_{j+1})/2$ and

$$\max_{x_j \leq x \leq x_{j+1}} |(x - x_j)(x - x_{j+1})| = \left(\frac{x_{j+1} - x_j}{2}\right)^2 = \frac{1}{4}h_j^2, \tag{3.91}$$

where $h_j = x_{j+1} - x_j$. Therefore

$$\max_{x_j \leq x \leq x_{j+1}} |f(x) - p(x)| \leq \frac{1}{8}\|f''\|_\infty \, h_j^2. \tag{3.92}$$

If we add more nodes, we can make h_j sufficiently small so that the error is smaller than a prescribed tolerance δ. That is, we can pick h_j such that $\frac{1}{8}\|f''\|_\infty h_j^2 \leq \delta$, which implies

$$h_j \leq \sqrt{\frac{8\delta}{\|f''\|_\infty}}. \tag{3.93}$$

This gives us an adaptive procedure to obtain a desired accuracy.

Continuous, piecewise quadratic interpolants ($k = 2$) can be obtained by adding an extra point in each subinverval, say its midpoint, so that each piece

$s_j \in \mathbb{P}_2$ is the one that interpolates f at $x_j, \frac{1}{2}(x_j + x_{j+1}), x_{j+1}$. For $k = 3$, we need to add 2 more points on each subinterval, etc. This procedure allows us to construct continuous, piecewise polynomial interpolants of f and if $f \in C^{k+1}[a, b]$ one can simply use the Cauchy remainder on each subinterval to get a bound for the error, as we did for the piecewise linear case.

Sometimes a smoother piecewise polynomial interpolant s is needed. If we want $s \in C^m[a, b]$ then on the first subinterval, $[x_0, x_1]$, we can take an arbitrary polynomial of degree at most k ($k + 1$ degrees of freedom) but in the second subinterval the corresponding polynomial has to match $m + 1$ (continuity plus m derivatives) conditions at x_1 so we only have $k-m$ degrees of freedom for it, and so on. Thus, in total we have $k+1+(n-1)(k-m) = n(k-m)+m+1$ degrees of freedom. For $m = k$ we only have $k+1$ degrees of freedom and since $s \in \mathbb{P}_k$ on each subinterval, it must be a polynomial of degree at most k in the entire interval $[a, b]$. Moreover, since polynomials are C^∞ it follows that $s \in \mathbb{P}_k$ on $[a, b]$ for $m \geq k$. So we restrict ourselves to $m < k$ and specifically focus on the case $m = k - 1$. These functions are called *splines*.

Definition 3.1. *Given a partition*

$$\Delta = \{a = x_0 < x_1 \ldots < x_n = b\} \tag{3.94}$$

of $[a, b]$*, the functions in the set*

$$\mathbb{S}^k_\Delta = \left\{ s : s \in C^{k-1}[a, b],\ s\,\Big|_{[x_j, x_{j+1}]} \in \mathbb{P}_k, j = 0, 1, \ldots, n-1 \right\} \tag{3.95}$$

are called splines of degree k *(or order* $k+1$*). The nodes* x_j*,* $j = 0, 1, \ldots, n$*, are called knots or breakpoints.*

Note that if s and r are in $\mathbb{S}^k_\Delta$ so is $as + br$, i.e. $\mathbb{S}^k_\Delta$ is a linear space, a subspace of $C^{k-1}[a, b]$. The piecewise linear interpolant is a spline of degree 1. We are going to study next splines of degree 3.

3.10 Cubic Splines

Several applications require smoother approximations than that provided by a piece-wise linear interpolation. For example, continuity up to the second derivative is generally desired in computer graphics applications. With the C^2 requirement, we need to consider splines of degree $k \geq 3$. The case $k = 3$

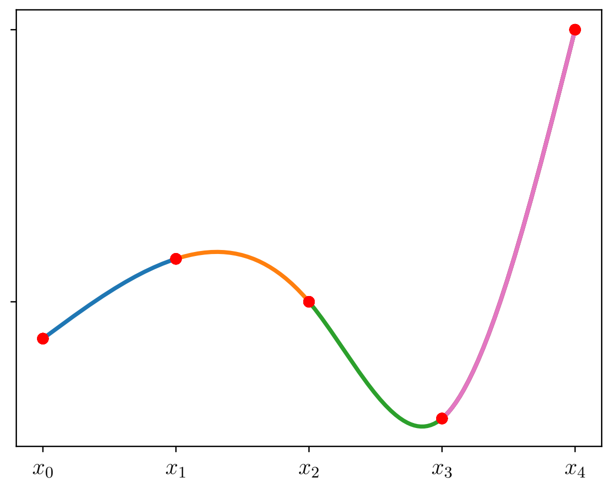

Figure 3.12: Cubic spline s interpolating 5 data points. Each color represents a cubic polynomial constructed so that s interpolates the given data, has two continuous derivatives, and $s''(x_0) = s"(x_4) = 0$.

is the most widely used and the corresponding splines are simply called *cubic splines*.

We consider here cubic splines that interpolate a set of values $f_0, f_1, \ldots, f_n$ at the nodes $a = x_0 < x_1 \ldots < x_n = b$, i.e. $s \in \mathbb{S}^3_\Delta$ with $s(x_j) = f_j$, $j = 0, 1, \ldots, n$. We call such a function a *cubic spline interpolant*. Figure 3.12 shows an example of a cubic spline interpolating 5 data points. The cubic polynomial pieces (s_j for $j = 0, 1, 2, 3$), appearing in different colors, are stitched together so that s interpolates the given data and has two continuous derivatives. The same data points have been used in both Fig. 3.11 and Fig. 3.12. Note the striking difference of the two interpolants.

As we saw in Section 3.9, there are $n+3$ degrees of freedom to determine $s \in \mathbb{S}^3_\Delta$, two more than the $n+1$ interpolation conditions. The two extra conditions could be the first or the second derivative of s at the end points ($x = a, x = b$). Note that if $s \in \mathbb{S}^3_\Delta$ then $s'' \in \mathbb{S}^1_\Delta$, i.e. the second derivative of a cubic spline is a continuous, piece-wise linear spline. Consequently, s'' is determined uniquely by its $(n+1)$ values

$$m_j = s''(x_j), \quad j = 0, 1, \ldots, n. \tag{3.96}$$

In the following construction of cubic spline interpolants we impose the $n+1$ interpolation conditions plus two extra conditions to find the unique values m_j, $j = 0, 1, \ldots, n$ that s'' must have at the nodes in order for s to be $C^2[a, b]$.

3.10.1 Natural Splines

Cubic splines with a vanishing second derivative at the first and last node, $m_0 = 0$ and $m_n = 0$, are called natural cubic splines. They are useful in graphics but not good for approximating a function f, unless f happens to also have vanishing second derivatives at x_0 and x_n.

We are now going to derive a linear system of equations for the values $m_1, m_2, \ldots, m_{n-1}$ that define the natural cubic spline interpolant. Once this system is solved we obtain the spline piece by piece.

In each subinterval $[x_j, x_{j+1}]$, s is a polynomial $s_j \in \mathbb{P}_3$, which we may represent as

$$s_j(x) = A_j(x - x_j)^3 + B_j(x - x_j)^2 + C_j(x - x_j) + D_j, \tag{3.97}$$

for $j = 0, 1, \ldots, n-1$. To simplify the formulas below we let

$$h_j = x_{j+1} - x_j. \tag{3.98}$$

The spline s interpolates the given data. Thus, for $j = 0, 1, \ldots n-1$

$$s_j(x_j) = D_j = f_j, \tag{3.99}$$

$$s_j(x_{j+1}) = A_j h_j^3 + B_j h_j^2 + C_j h_j + D_j = f_{j+1}. \tag{3.100}$$

Now $s_j'(x) = 3A_j(x - x_j)^2 + 2B_j(x - x_j) + C_j$ and $s_j''(x) = 6A_j(x - x_j) + 2B_j$. Therefore, for $j = 0, 1, \ldots n-1$

$$s_j'(x_j) = C_j, \tag{3.101}$$

$$s_j'(x_{j+1}) = 3A_j h_j^2 + 2B_j h_j + C_j, \tag{3.102}$$

and

$$s_j''(x_j) = 2B_j, \tag{3.103}$$

$$s_j''(x_{j+1}) = 6A_j h_j + 2B_j. \tag{3.104}$$

Since s'' is continuous

$$m_{j+1} = s''(x_{j+1}) = s_{j+1}''(x_{j+1}) = s_j''(x_{j+1}) \tag{3.105}$$

and we can write (3.103)-(3.104) as

$$m_j = 2B_j, \tag{3.106}$$
$$m_{j+1} = 6A_j h_j + 2B_j. \tag{3.107}$$

We now write A_j, B_j, C_j, and D_j in terms of the unknown values m_j and m_{j+1}, and the known values f_j and f_{j+1}. We have

$$\begin{aligned} D_j &= f_j, \\ B_j &= \frac{1}{2} m_j, \\ A_j &= \frac{1}{6h_j}(m_{j+1} - m_j) \end{aligned}$$

and substituting these values in (3.100) we get

$$C_j = \frac{1}{h_j}(f_{j+1} - f_j) - \frac{1}{6} h_j (m_{j+1} + 2m_j).$$

Let us collect all our formulas for the spline coefficients:

$$A_j = \frac{1}{6h_j}(m_{j+1} - m_j), \tag{3.108}$$
$$B_j = \frac{1}{2} m_j, \tag{3.109}$$
$$C_j = \frac{1}{h_j}(f_{j+1} - f_j) - \frac{1}{6} h_j (m_{j+1} + 2m_j), \tag{3.110}$$
$$D_j = f_j, \tag{3.111}$$

for $j = 0, 1, \ldots, n-1$. So far we have only used that s and s'' are continuous and that s interpolates the given data. We are now going to impose the continuity of the first derivative of s to determine equations for the unknown values m_j, $j = 1, 2, \ldots, n-1$. Substituting (3.108)-(3.111) in (3.102) we get

$$\begin{aligned} s_j'(x_{j+1}) &= 3A_j h_j^2 + 2B_j h_j + C_j \\ &= 3\frac{1}{6h_j}(m_{j+1} - m_j) h_j^2 + 2\frac{1}{2} m_j h_j + \frac{1}{h_j}(f_{j+1} - f_j) \\ &\quad - \frac{1}{6} h_j (m_{j+1} + 2m_j) \\ &= \frac{1}{h_j}(f_{j+1} - f_j) + \frac{1}{6} h_j (2m_{j+1} + m_j) \end{aligned} \tag{3.112}$$

and decreasing the index by 1

$$s'_{j-1}(x_j) = \frac{1}{h_{j-1}}(f_j - f_{j-1}) + \frac{1}{6}h_{j-1}(2m_j + m_{j-1}). \tag{3.113}$$

Continuity of the first derivative means $s'_{j-1}(x_j) = s'_j(x_j)$ for $j = 1, 2, ..., n-1$. Therefore, for $j = 1, \ldots, n-1$

$$\begin{aligned} \frac{1}{h_{j-1}}(f_j - f_{j-1}) + \frac{1}{6}h_{j-1}(2m_j + m_{j-1}) = C_j \\ = \frac{1}{h_j}(f_{j+1} - f_j) - \frac{1}{6}h_j(m_{j+1} + 2m_j) \end{aligned} \tag{3.114}$$

which can be written as

$$\begin{aligned} h_{j-1}m_{j-1} + 2(h_{j-1} + h_j)m_j + h_j m_{j+1} = \\ -\frac{6}{h_{j-1}}(f_j - f_{j-1}) + \frac{6}{h_j}(f_{j+1} - f_j), \quad j = 1, \ldots, n-1. \end{aligned} \tag{3.115}$$

This is a linear system of $n-1$ equations for the $n-1$ unknowns $m_1, m_2, \ldots, m_{n-1}$. In matrix form

$$\begin{bmatrix} a_1 & b_1 & & & & \\ c_1 & a_2 & b_2 & & & \\ & c_2 & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & \ddots & \\ & & & & \ddots & \ddots & b_{n-2} \\ & & & & & c_{n-2} & a_{n-1} \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ m_{n-1} \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ d_{n-1} \end{bmatrix}, \tag{3.116}$$

where

$$a_j = 2(h_{j-1} + h_j), \qquad j = 1, 2, \ldots, n-1, \tag{3.117}$$

$$b_j = h_j, \qquad j = 1, 2, \ldots, n-2, \tag{3.118}$$

$$c_j = h_j, \qquad j = 1, 2, \ldots, n-2, \tag{3.119}$$

$$d_j = -\frac{6}{h_{j-1}}(f_j - f_{j-1}) + \frac{6}{h_j}(f_{j+1} - f_j), \quad j = 1, \ldots, n-1. \tag{3.120}$$

Note that we have used $m_0 = m_n = 0$ in the first and last equation of this linear system. The matrix of the linear system (3.116) is strictly diagonally dominant, a concept we make precise in the definition below. A consequence of this property is that the matrix is nonsingular and therefore the linear system (3.116) has a unique solution. Moreover, this tridiagonal linear system can be solved efficiently with Algorithm 9.5. Once $m_1, m_2, \ldots, m_{n-1}$ are found, the spline coefficients can be computed from (3.108)-(3.111).

Definition 3.2. *An $n \times n$ matrix A with entries a_{ij}, $i, j = 1, \ldots, n$ is strictly diagonally dominant if*

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|, \quad \text{for } i = 1, \ldots, n. \tag{3.121}$$

Theorem 3.6. *Let A be a strictly diagonally dominant matrix. Then A is nonsingular.*

Proof. Suppose the contrary, that is there is $x \neq 0$ such that $Ax = 0$. Let k be an index such that $|x_k| = \|x\|_\infty$. Then, the k-th equation in $Ax = 0$ gives

$$a_{kk} x_k + \sum_{\substack{j=1 \\ j \neq k}}^{n} a_{kj} x_j = 0 \tag{3.122}$$

and consequently

$$|a_{kk}||x_k| \leq \sum_{\substack{j=1 \\ j \neq k}}^{n} |a_{kj}||x_j|. \tag{3.123}$$

Dividing by $|x_k|$, which by assumption in nonzero, and using that $|x_j|/|x_k| \leq 1$ for all $j = 1, \ldots, n$, we get

$$|a_{kk}| \leq \sum_{\substack{j=1 \\ j \neq k}}^{n} |a_{kj}|, \tag{3.124}$$

which contradicts the fact that A is strictly diagonally dominant. $\square$

Example 3.6. *Find the natural cubic spline that interpolates* $(0, 0), (1, 1), (2, 0)$. *We know $m_0 = 0$ and $m_2 = 0$. We only need to find m_1 (only 1 interior node). The system (3.115) degenerates to just one equation. With $h_0 = h_1 = 1$ we have*

$$m_0 + 4m_1 + m_2 = 6[f_0 - 2f_1 + f_2] \Rightarrow m_1 = -1/3$$

In $[0,1]$*:*

$$
\begin{aligned}
A_0 &= \frac{1}{6}(m_1 - m_0) = \left(\frac{1}{6}\right)\left(-\frac{1}{3}\right) = -\frac{1}{18},\\
B_0 &= \frac{1}{2}m_0 = 0\\
C_0 &= f_1 - f_0 - \frac{1}{6}(m_1 + 2m_0) = 1 + \frac{1}{18} = \frac{19}{18},\\
D_0 &= f_0 = 0.
\end{aligned}
$$

Thus, $s_0(x) = A_0(x-0)^3 + B_0(x-0)^2 + C_0(x-0) + D_0 = -\frac{1}{18}x^3 + \frac{19}{18}x.$

In $[1,2]$*:*

$$
\begin{aligned}
A_1 &= \frac{1}{6}(m_2 - m_1) = \left(\frac{1}{6}\right)\left(\frac{1}{3}\right) = \frac{1}{18},\\
B_1 &= \frac{1}{2}m_1 = -\frac{1}{6},\\
C_1 &= f_2 - f_1 - \frac{1}{6}(m_2 + 2m_1) = 0 - 1 - \frac{1}{6}\left(-\frac{2}{3}\right) = -\frac{8}{9},\\
D_1 &= f_1 = 1.
\end{aligned}
$$

and $s_1(x) = \frac{1}{18}(x-1)^3 - \frac{1}{6}(x-1)^2 - \frac{8}{9}(x-1) + 1$*. Therefore the natural cubic spline that interpolates the given data is*

$$
s(x) = \begin{cases} -\frac{1}{18}x^3 + \frac{19}{18}x & x \in [0,1],\\[2ex] \frac{1}{18}(x-1)^3 - \frac{1}{6}(x-1)^2 - \frac{8}{9}(x-1) + 1 & x \in [1,2]. \end{cases}
$$

3.10.2 Complete Splines

If we are interested in approximating a function with a cubic spline interpolant it is generally more accurate to specify the first derivative at the endpoints instead of imposing a vanishing second derivative. A cubic spline where we specify $s'(a)$ and $s'(b)$ is called a *complete spline*.

In a complete spline the values m_0 and m_n of s'' at the endpoints become unknowns together with $m_1, m_2, \ldots, m_{n-1}$. Thus, we need to add two more equations to have a complete system for all the $n+1$ unknown values $m_0, m_1, \ldots, m_n$. Recall that

$$
s_j(x) = A_j(x - x_j)^3 + B_j(x - x_j)^2 + C_j(x - x_j) + D_j
$$

and so $s_j'(x) = 3A_j(x-x_j)^2 + 2B_j(x-x_j) + C_j$. Therefore

$$s_0'(x_0) = C_0 = f_0', \tag{3.125}$$
$$s_{n-1}'(x_n) = 3A_{n-1}h_{n-1}^2 + 2B_{n-1}h_{n-1} + C_{n-1} = f_n', \tag{3.126}$$

where $f_0' = f'(x_0)$ and $f_n' = f'(x_n)$. Substituting C_0, A_{n-1}, B_{n-1}, and C_{n-1} from (3.108)-(3.110) we get

$$2h_0m_0 + h_0m_1 = \frac{6}{h_0}(f_1 - f_0) - 6f_0', \tag{3.127}$$
$$h_{n-1}m_{n-1} + 2h_{n-1}m_n = -\frac{6}{h_{n-1}}(f_n - f_{n-1}) + 6f_n'. \tag{3.128}$$

If we append (3.127) and (3.127) at the top and the bottom of the system (3.115), respectively and set $h_{-1} = h_n = 0$ we obtain the following tridiagonal linear system for the values of the second derivative of the complete spline at the knots:

$$\begin{bmatrix} a_0 & b_0 & & & & & \\ c_0 & a_1 & b_1 & & & & \\ & c_1 & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & \ddots & \ddots & b_{n-1} \\ & & & & & c_{n-1} & a_n \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ m_n \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ d_n \end{bmatrix}, \tag{3.129}$$

where

$$a_j = 2(h_{j-1} + h_j), \qquad j = 0, 1, \ldots, n, \tag{3.130}$$
$$b_j = h_j, \qquad j = 0, 1, \ldots, n-1, \tag{3.131}$$
$$c_j = h_j, \qquad j = 0, 1, \ldots, n-1, \tag{3.132}$$
$$d_0 = \frac{6}{h_0}(f_1 - f_0) - 6f_0', \tag{3.133}$$
$$d_j = -\frac{6}{h_{j-1}}(f_j - f_{j-1}) + \frac{6}{h_j}(f_{j+1} - f_j), \quad j = 1, \ldots, n-1, \tag{3.134}$$
$$d_n = -\frac{6}{h_{n-1}}(f_n - f_{n-1}) + 6f_n'. \tag{3.135}$$

As in the case of natural cubic splines, this linear system is also diagonally dominant (hence nonsingular) and can be solved efficiently with Algorithm 9.5.

It can be proved that if f is sufficiently smooth its complete spline interpolant s produces an error $\|f - s\|_\infty \leq Ch^4$, where $h = \max_i h_i$, whereas for the natural cubic spline interpolant the error deteriorates to $O(h^2)$ near the endpoints.

3.10.3 Minimal Bending Energy

Consider a curve given by $y = f(x)$ for $x \in [a, b]$, where $f \in C^2[a, b]$. Its curvature is

$$\kappa(x) = \frac{f''(x)}{[1 + (f'(x))^2]^{3/2}} \tag{3.136}$$

and a measure of how much the curve "curves" or bends is its bending energy

$$E_b = \int_a^b \kappa^2(x)dx. \tag{3.137}$$

For curves with small $|f'|$ compared to 1, $\kappa(x) \approx f''(x)$ and $E_b \approx \|f''\|_2^2$. We are going to show that cubic splines interpolants are C^2 functions that have minimal $\|f''\|_2$, in a sense we make more precise below. To show this we are going to use the following two results.

Lemma 2. *Let $s \in \mathbb{S}^3_\Delta$ be a cubic spline interpolant of $f \in C^2[a, b]$ at the nodes $\Delta = \{a = x_0 < x_1 \ldots < x_n = b\}$. Then, for all $g \in \mathbb{S}^1_\Delta$*

$$\int_a^b [f''(x) - s''(x)]g(x)dx = [f'(b) - s'(b)]g(b) - [f'(a) - s'(a)]g(a). \tag{3.138}$$

Proof.

$$\int_a^b [f''(x) - s''(x)]g(x)dx = \sum_{j=0}^{n-1} \int_{x_j}^{x_{j+1}} [f''(x) - s''(x)]g(x)dx. \tag{3.139}$$

We can integrate by parts on each interval:

$$\begin{aligned} \int_{x_j}^{x_{j+1}} [f''(x) - s''(x)]g(x)dx &= [f'(x) - s'(x)]g(x)\Big|_{x_j}^{x_{j+1}} \\ &\quad - \int_{x_j}^{x_{j+1}} [f'(x) - s'(x)]g'(x)dx. \end{aligned} \tag{3.140}$$

Substituting this in (3.139) the boundary terms telescope and we obtain

$$\begin{aligned}\int_a^b [f''(x) - s''(x)]g(x)dx &= [f'(b) - s'(b)]g(b) - [f'(a) - s'(a)]g(a) \\ &\quad - \sum_{j=0}^{n-1} \int_{x_j}^{x_{j+1}} [f'(x) - s'(x)]g'(x)dx.\end{aligned} \tag{3.141}$$

On each subinterval $[x_j, x_{j+1}]$, g' is constant and $f - s$ vanishes at the end-points. Therefore, the last term is zero. □

Theorem 3.7. *Let $s \in \mathbb{S}^3_\Delta$ be the (natural or complete) cubic spline interpolant of $f \in C^2[a,b]$ at the nodes $\Delta = \{a = x_0 < x_1 \ldots < x_n = b\}$. Then,*

$$\|s''\|_2 \leq \|f''\|_2. \tag{3.142}$$

Proof.

$$\begin{aligned}\|f'' - s''\|_2^2 = \int_a^b [f''(x) - s''(x)]^2 dx &= \|f''\|_2^2 + \|s''\|_2^2 - 2\int_a^b f''(x)s''(x)dx \\ &= \|f''\|_2^2 - \|s''\|_2^2 - 2\int_a^b [f''(x) - s''(x)]s''(x)dx.\end{aligned} \tag{3.143}$$

By Lemma 2 with $g = s''$ the last term vanishes for the natural spline ($s''(a) = s''(b) = 0$) and for the complete spline ($s'(a) = f'(a)$ and $s'(b) = f'(b)$) and we get the identify

$$\|f'' - s''\|_2^2 = \|f''\|_2^2 - \|s''\|_2^2 \tag{3.144}$$

from which the results follows. □

In Theorem 3.7 f could be substituted for any sufficiently smooth interpolant g of the given data.

Theorem 3.8. *Let $s \in \mathbb{S}^3_\Delta$ and $g \in C^2[a,b]$ both interpolate the values $f_0, f_1, \ldots, f_n$ at the nodes $\Delta = \{a = x_0 < x_1 \ldots < x_n = b\}$. Then,*

$$\|s''\|_2 \leq \|g''\|_2, \tag{3.145}$$

if either $s''(a) = s''(b) = 0$ (natural spline) or $s'(a) = g'(a)$ and $s'(b) = g'(b)$ (complete spline).

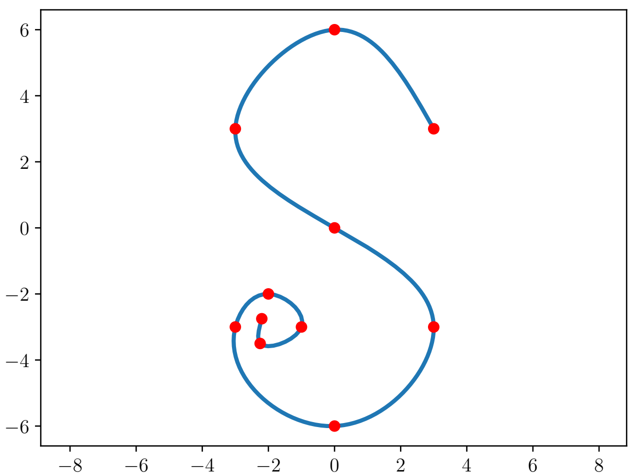

Figure 3.13: Example of a parametric spline representation to interpolate the given data points (in red).

3.10.4 Splines for Parametric Curves

In computer graphics and animation it is often required to construct smooth curves that are not necessarily the graph of a function but that have a parametric representation $x = x(t)$ and $y = y(t)$ for $t \in [a, b]$. Hence we need to determine two splines interpolating (t_j, x_j) and (t_j, y_j) ($j = 0, 1, \ldots n$), respectively. Usually, only the position of the "control points" $(x_0, y_0), \ldots (x_n, y_n)$ is given and not the parameter values $t_0, t_1, \ldots, t_n$. In such cases, we can use the distances of consecutive control points to generate appropriate t_j's as follows:

$$t_0 = 0, \quad t_j = t_{j-1} + \sqrt{(x_j - x_{j-1})^2 + (y_j - y_{j-1})^2}, \quad j = 1, 2, \ldots n. \tag{3.146}$$

Figure 3.13 shows an example of this approach.

3.11 Trigonometric Interpolation

We consider now the important case of interpolation of a periodic array of data $(x_0, f_0), (x_1, f_1), \ldots, (x_N, f_N)$ with $f_N = f_0$, and $x_j = j(2\pi/N)$, $j = 0, 1, \ldots, N$, by a trigonometric polynomial.

Definition 3.3. *A function of the form*

$$s_n(x) = \sum_{k=-n}^{n} c_k e^{ikx}, \tag{3.147}$$

where $c_0, c_1, c_{-1}, \ldots, c_n, c_{-n}$ are complex, or equivalently of the form[5]

$$s_n(x) = \frac{1}{2}a_0 + \sum_{k=0}^{n} (a_k \cos kx + b_k \sin kx) \tag{3.148}$$

where the coefficients $a_0, a_1, b_1, \ldots, a_n, b_n$ are real is called a trigonometric polynomial of degree (at most) *n.*

The values f_j, $j = 0, 1, \ldots, N$, could come from a 2π-periodic function, $f(j\,2\pi/N) = f_j$, or can simply be given data. Note that the interpolation nodes are equi-spaced points in $[0, 2\pi]$. One can accommodate any other period by doing a simple scaling. Because of periodicity ($f_N = f_0$), we only have N independent data points $(x_0, f_0), \ldots, (x_{N-1}, f_{N-1})$ or $(x_1, f_1), \ldots, (x_N, f_N)$. The interpolation problem is then to find a trigonometric polynomial s_n of lowest degree such that $s_n(x_j) = f_j$, for $j = 0, 1, \ldots, N-1$. Such polynomial has $2n+1$ coefficients. If we take $n = N/2$ (assuming N even), we have $N+1$ coefficients to be determined but only N interpolation conditions. An additional condition arises by noting that the sine term of highest wavenumber, $k = N/2$, vanishes at the equi-spaced nodes, $\sin(\frac{N}{2}x_j) = \sin(j\pi) = 0$. Thus, the coefficient $b_{N/2}$ is irrelevant for interpolation and we can set it to zero. Consequently, we look for a trigonometric polynomial of the form

$$s_{N/2}(x) = \frac{1}{2}a_0 + \sum_{k=1}^{N/2-1} (a_k \cos kx + b_k \sin kx) + \frac{1}{2}a_{N/2} \cos\left(\frac{N}{2}x\right). \tag{3.149}$$

The convenience of the $1/2$ factor in the last term will be seen in the formulas we obtain below for the coefficients.

It is conceptually and computationally simpler to work with the corresponding trigonometric polynomial in complex form

$$s_{N/2}(x) = {\sum_{k=-N/2}^{N/2}}'' c_k e^{ikx}, \tag{3.150}$$

[5] Recall $2\cos kx = e^{ik} + e^{-ik}$ and $2i \sin kx = e^{ik} - e^{-ik}$

where the double prime in the summation sign means that the first and last terms ($k = -N/2$ and $k = N/2$) have a factor of 1/2. It is also understood that $c_{-N/2} = c_{N/2}$, which is equivalent to the $b_{N/2} = 0$ condition in (3.149).

Theorem 3.9.

$$s_{N/2}(x) = \sum_{k=-N/2}^{N/2}{}'' c_k e^{ikx} \tag{3.151}$$

interpolates $(j2\pi/N, f_j), j = 0, \ldots, N-1$ *if and only if*

$$c_k = \frac{1}{N} \sum_{j=0}^{N-1} f_j e^{-ik2\pi j/N}, \quad k = -\frac{N}{2}, \ldots, \frac{N}{2}. \tag{3.152}$$

Proof. Substituting (3.152) in (3.151) we get

$$s_{N/2}(x) = \sum_{k=-N/2}^{N/2}{}'' c_k e^{ikx} = \sum_{j=0}^{N-1} f_j \frac{1}{N} \sum_{k=-N/2}^{N/2}{}'' e^{ik(x-x_j)},$$

with $x_j = j2\pi/N$ and defining the cardinal functions

$$l_j(x) = \frac{1}{N} \sum_{k=-N/2}^{N/2}{}'' e^{ik(x-x_j)} \tag{3.153}$$

we obtain

$$s_{N/2}(x) = \sum_{j=0}^{N-1} l_j(x) f_j. \tag{3.154}$$

Note that we have written $s_{N/2}$ in a form similar to the Lagrange form of polynomial interpolation. We will prove that for j and m in the range $0, \ldots, N-1$

$$l_j(x_m) = \begin{cases} 1 & \text{for } m = j, \\ 0 & \text{for } m \neq j, \end{cases} \tag{3.155}$$

and in view of (3.154), $s_{N/2}$ satisfies the interpolation conditions.

Now,

$$l_j(x_m) = \frac{1}{N} \sum_{k=-N/2}^{N/2}{}'' e^{ik(m-j)2\pi/N} \tag{3.156}$$

and $e^{i(\pm N/2)(m-j)2\pi/N} = e^{\pm i(m-j)\pi} = (-1)^{(m-j)}$ so we can combine the first and the last term and remove the double prime from the sum:

$$\begin{aligned} l_j(x_m) &= \frac{1}{N} \sum_{k=-N/2}^{N/2-1} e^{ik(m-j)2\pi/N} \\ &= \frac{1}{N} \sum_{k=-N/2}^{N/2-1} e^{i(k+N/2)(m-j)2\pi/N} e^{-i(N/2)(m-j)2\pi/N} \\ &= e^{-i(m-j)\pi} \frac{1}{N} \sum_{k=0}^{N-1} e^{ik(m-j)2\pi/N}. \end{aligned}$$

Recall that (see Section 1.3)

$$\frac{1}{N} \sum_{k=0}^{N-1} e^{-ik(j-m)2\pi/N} = \begin{cases} 1 & \text{if } \left(\frac{j-m}{N}\right) \in \mathbb{Z}, \\ 0 & \text{otherwise.} \end{cases} \tag{3.157}$$

Then, (3.155) follows and

$$s_{N/2}(x_m) = f_m, \quad m = 0, 1, \ldots N-1. \tag{3.158}$$

Now suppose $s_{N/2}$ interpolates $(j2\pi/N, f_j), j = 0, \ldots, N-1$. Then, the c_k coefficients of $s_{N/2}$ satisfy

$$\sum_{k=-N/2}^{N/2}{}'' c_k e^{ik2\pi j/N} = f_j, \quad j = 0, 1, \ldots, N-1. \tag{3.159}$$

Since $c_{-N/2} = c_{N/2}$, we can write (3.159) equivalently as the linear system

$$\sum_{k=-N/2}^{N/2-1} c_k e^{ik2\pi j/N} = f_j, \quad j = 0, 1, \ldots, N-1. \tag{3.160}$$

From the discrete orthogonality of the complex exponential (3.157), it follows that the matrix of coefficients of (3.160) has orthogonal columns and hence it is nonsingular. Therefore, (3.160) has a unique solution and thus the c_k coefficients must be those given by (3.152). □

Using the relations $c_0 = \frac{1}{2}a_0$, $c_k = \frac{1}{2}(a_k - ib_k)$, $c_{-k} = \bar{c}_k$, we find that

$$s_{N/2}(x) = \frac{1}{2}a_0 + \sum_{k=1}^{N/2-1}(a_k \cos kx + b_k \sin kx) + \frac{1}{2}a_{N/2}\cos\left(\frac{N}{2}x\right)$$

interpolates $(j2\pi/N, f_j), j = 0, \ldots, N-1$ if and only if

$$a_k = \frac{2}{N}\sum_{j=0}^{N-1} f_j \cos kx_j, \quad k = 0, 1, \ldots, N/2, \tag{3.161}$$

$$b_k = \frac{2}{N}\sum_{j=0}^{N-1} f_j \sin kx_j, \quad k = 1, \ldots, N/2-1. \tag{3.162}$$

Let us go back to the complex, interpolating trigonometric polynomial (3.150). Its coefficients c_k are periodic of period N,

$$c_{k+N} = \frac{1}{N}\sum_{j=0}^{N-1} f_j e^{-i(k+N)x_j} = \frac{1}{N}\sum_{j=0}^{N-1} f_j e^{-ikx_j}e^{-ij2\pi} = c_k. \tag{3.163}$$

Now, from (3.160) we have

$$\begin{aligned} f_j = \sum_{k=-N/2}^{N/2-1} c_k e^{ikx_j} &= \sum_{k=-N/2}^{-1} c_k e^{ikx_j} + \sum_{k=0}^{N/2-1} c_k e^{ikx_j} \\ &= \sum_{k=N/2}^{N-1} c_k e^{ikx_j} + \sum_{k=0}^{N/2-1} c_k e^{ikx_j} \\ &= \sum_{k=0}^{N-1} c_k e^{ikx_j}, \end{aligned} \tag{3.164}$$

where we have used that $c_{k+N} = c_k$ to shift the sum from $-N/2$ to -1 to the sum from $N/2$ to $N-1$. Combining this with the formula for the c_k's we get

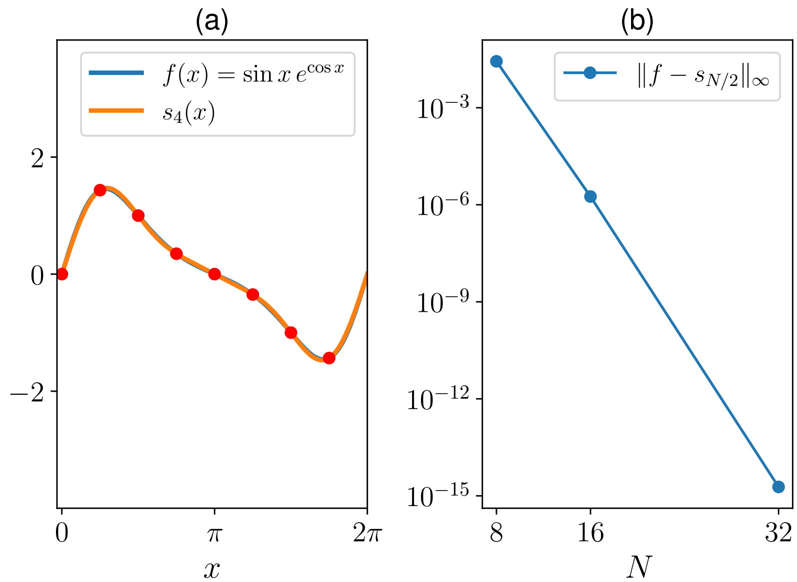

Figure 3.14: (a) $f(x) = \sin x\, e^{\cos x}$ and its interpolating trigonometric polynomial $s_4(x)$ and (b) the maximum error $\|f - s_{N/2}\|_\infty$ for $N = 8, 16, 32$.

the discrete Fourier transform (DFT) pair

$$c_k = \frac{1}{N} \sum_{j=0}^{N-1} f_j e^{-ikx_j}, \quad k = 0, \ldots, N-1, \tag{3.165}$$

$$f_j = \sum_{k=0}^{N-1} c_k e^{ikx_j}, \quad j = 0, \ldots, N-1. \tag{3.166}$$

The set of coefficients (3.165) is known as the DFT of the periodic array $f_0, f_1, \ldots, f_{N-1}$ and (3.166) is called the inverse DFT. It is important to note that the DFT coefficients for $k = N/2, \ldots, N-1$ correspond to those for $k = -N/2, \ldots, -1$ of the interpolating trigonometric polynomial $s_{N/2}$.

A smooth periodic function f can be approximated accurately by its interpolating trigonometric polynomial of low to moderate degree. Figure 3.14(a) shows the approximation of $f(x) = \sin x\, e^{\cos x}$ on $[0, 2\pi]$ by s_4 $(N = 8)$. The graphs of f and $s_{N/2}$ are almost indistinguishable. In fact, the interpolating trigonometric polynomial $s_{N/2}$ converges uniformly to f exponentially fast as Fig. 3.14(b) demonstrates (note that the vertical axis uses a logarithmic scale).

Note also that derivatives of $s_{N/2}$ can be easily computed

$$s_{N/2}^{(p)}(x) = \sum_{k=-N/2}^{N/2}{}'' (ik)^p c_k e^{ikx}. \tag{3.167}$$

The Fourier coefficients of the p-th derivative of $s_{N/2}$ can thus be readily obtained from the DFT of f (the c_k's) and $s_{N/2}^{(p)}$ yields an accurate approximation of $f^{(p)}$ if this is smooth. We discuss the implementations details of this approach in Section 6.4.

3.12 The Fast Fourier Transform

The direct evaluation of the DFT or the inverse DFT is computationally expensive, it requires $O(N^2)$ operations. However, there is a remarkable algorithm which achieves this in merely $O(N \log_2 N)$ operations. This algorithm is known as the Fast Fourier Transform.

We now look at the main ideas of this widely used algorithm.

Let us define $d_k = Nc_k$ for $k = 0, 1, \ldots, N-1$, and $\omega_N = e^{-i2\pi/N}$. Then we can rewrite the DFT (3.165) as

$$d_k = \sum_{j=0}^{N-1} f_j \omega_N^{kj}, \quad k = 0, 1, \ldots, N-1. \tag{3.168}$$

Let $N = 2n$. If we split the even-numbered and the odd-numbered points we have

$$d_k = \sum_{j=0}^{n-1} f_{2j} \omega_N^{2jk} + \sum_{j=0}^{n-1} f_{2j+1} \omega_N^{(2j+1)k} \tag{3.169}$$

But

$$\omega_N^{2jk} = e^{-i2jk\frac{2\pi}{N}} = e^{-ijk\frac{2\pi}{\frac{N}{2}}} = e^{-ijk\frac{2\pi}{n}} = \omega_n^{kj}, \tag{3.170}$$

$$\omega_N^{(2j+1)k} = e^{-i(2j+1)k\frac{2\pi}{N}} = e^{-ik\frac{2\pi}{N}} e^{-i2jk\frac{2\pi}{N}} = \omega_N^k \omega_n^{kj}. \tag{3.171}$$

Thus, denoting $f_j^e = f_{2j}$ and $f_j^o = f_{2j+1}$, we get

$$d_k = \sum_{j=0}^{n-1} f_j^e \omega_n^{jk} + \omega_N^k \sum_{j=0}^{n-1} f_j^o \omega_n^{jk} \tag{3.172}$$

We have reduced the problem to two DFT's of size $n = \frac{N}{2}$ plus N multiplications (and N sums). The numbers ω_N^k, $k = 0, 1, \ldots, N-1$ depend only on N so they can be precomputed once and stored for other DFT's of the same size N.

If $N = 2^p$, for p positive integer, we can repeat the process to reduce each of the DFT's of size n to a pair of DFT's of size $n/2$ plus n multiplications (and n additions), etc. We can do this p times so that we end up with 1-point DFT's, which require no multiplications!

Let us count the number of operations in the FFT algorithm. For simplicity, let is count only the number of multiplications (the numbers of additions is of the same order). Let m_N be the number of multiplications to compute the DFT for a periodic array of size N and assume that $N = 2^p$. Then

$$\begin{aligned}
m_N &= 2m_{\frac{N}{2}} + N \\
&= 2m_{2^{p-1}} + 2^p \\
&= 2(2m_{2^{p-2}} + 2^{p-1}) + 2^p \\
&= 2^2 m_{2^{p-2}} + 2 \cdot 2^p \\
&= \cdots \\
&= 2^p m_{2^0} + p \cdot 2^p = p \cdot 2^p \\
&= N \log_2 N,
\end{aligned}$$

where we have used that $m_{2^0} = m_1 = 0$ (no multiplication is needed for DFT of 1 point). To illustrate the savings, if $N = 2^{20}$, with the FFT we can obtain the DFT (or the inverse DFT) in order 20×2^{20} operations, whereas the direct methods requires order 2^{40}, i.e. a factor of $\frac{1}{20}2^{20} \approx 52429$ more operations. The FFT can also be implemented efficiently when N is the product of small primes.

3.13 The Chebyshev Interpolant and the DCT

We take now a closer look at polynomial interpolation of a function f in $[-1,1]$[6] at the Chebyshev nodes

$$x_j = \cos\left(\frac{j\pi}{n}\right), \quad j = 0, 1, \ldots, n. \tag{3.173}$$

The unique interpolating polynomial $p_n \in \mathbb{P}_n$ of f at the $n+1$ Chebyshev nodes, which we will call the *Chebyshev interpolant*, can be evaluated efficiently using its barycentric representation (Section 3.3). However, there is another representation of p_n that is also computationally efficient and useful for obtaining fast converging methods for integration and differentiation. This alternative representation is based on an expansion of Chebyshev polynomials and the DCT, the discrete cosine transform.

Since $p_n \in \mathbb{P}_n$, there are unique coefficients $c_0, c_1, \ldots, c_n$ such that

$$p_n(x) = \frac{1}{2}c_0 + \sum_{k=1}^{n-1} c_k T_k(x) + \frac{1}{2}c_n T_n(x) := \sum_{k=0}^{n}{}'' c_k T_k(x). \tag{3.174}$$

The $1/2$ factor for $k = 0, n$ is introduced for convenience to have one formula for all the c_k's, as we will see below. Under the change of variable $x = \cos\theta$, for $\theta \in [0, \pi]$ we get

$$p_n(\cos\theta) = \frac{1}{2}c_0 + \sum_{k=1}^{n-1} c_k \cos k\theta + \frac{1}{2}c_n \cos n\theta. \tag{3.175}$$

Let $\Pi_n(\theta) = p_n(\cos\theta)$ and $F(\theta) = f(\cos\theta)$. By extending F evenly over $[\pi, 2\pi]$ and using Theorem 3.9, we conclude that $\Pi_n(\theta)$ interpolates $F(\theta) = f(\cos\theta)$ at the equally spaced points $\theta_j = j\pi/n,\ j = 0, 1, ...n$ if and only if

$$c_k = \frac{2}{n}\sum_{j=0}^{n}{}'' F(\theta_j)\cos k\theta_j, \qquad k = 0, 1, .., n. \tag{3.176}$$

These are the (type I) Discrete Cosine Transform (DCT) coefficients of F and we can compute them efficiently in $O(n\log_2 n)$ operations with the fast

[6]For a function defined in an interval $[a,b]$ the change of variables $t = \frac{1}{2}(1-x)a + \frac{1}{2}(1+x)b$ could be used.

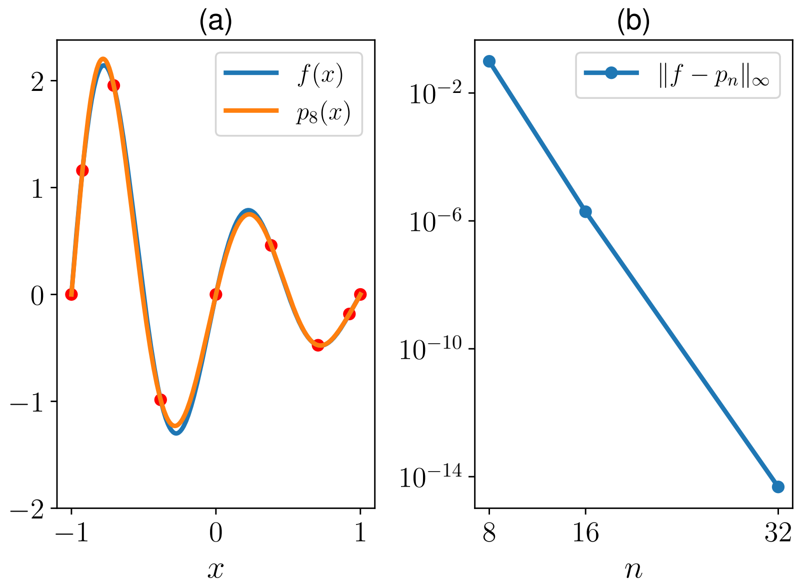

Figure 3.15: (a) $f(x) = \sin(2\pi x)\, e^{-x}$ and its Chebychev interpolant $p_8(x)$ and (b) the maximum error $\|f - p_n\|_\infty$ for $n = 8, 16, 32$.

DCT, an FFT-based algorithm which exploits that F is even and real. [7] Figure 3.15(a) presents a plot of $f(x) = \sin(2\pi x)e^{-x}$ on $[-1, 1]$ and its Chebyshev interpolant p_8, whose coefficients c_k were obtained with the fast DCT. The two graphs almost overlap. Figure. 3.15(a) shows the fast, uniform convergence of the Chebychev interpolant. With just $n = 32$, near machine precision is obtained.

One application of Chebyshev interpolation and its connection with the DCT is the Clenshaw-Curtis quadrature, which we consider in Section 7.4.

3.14 Bibliographical Notes

Section 3.1 The simple proof of existence and uniqueness of the interpolating polynomial using (3.1) appears in the book by Davis [Dav75].

[7] Using the full FFT requires extending F evenly to $[\pi, 2\pi]$, doubling the size of the arrays, and is thus computationally less efficient than the fast DCT.

Section 3.2. Rivlin [Riv81] provides a derivation of the bound for the Lebesgue constant $\Lambda > \frac{2}{\pi^2}\log n - 1$. There is a sharper estimate $\Lambda > \frac{2}{\pi}\log n - c$ for some positive constant c due to Erdös [Erd64]. Davis [Dav75] has a deeper discussion of the issue of convergence given a triangular system of nodes. He points to the independent discovery by Faber and Bernstein in 1914 that given any triangular system in advance, it is possible to construct a continuous function for which the interpolating polynomial does not converge uniformly to this function.

Section 3.3. Berrut and Trefethen [BT04] provide an excellent review of barycentric interpolation, including a discussion of numerical stability and historical notes. They also show that in most cases this is the method of choice for repeated evaluation of the interpolating polynomial. For historical reasons explained in [BT04], barycentric interpolation has rarely appeared in numerical analysis textbooks. Among the rare exceptions are the textbooks by Schwarz [SW89], Greenbaum and Chartier [GC12], and Gautschi [Gau11]. Our presentation here follows the latter. Our derivation of the barycentric weights for the Chebyshev nodes follows that of Salzer [Sal72].

Section 3.4. Divided differences receive considerable attention as an interpolation topic in most classical, numerical analysis textbooks (see for example [CB72, Hil13, RR01, IK94]). Here, we keep our presentation to a minimum to devote more space to barycentric interpolation (which is more efficient for the evaluation of the interpolating polynomial) and to other interpolation topics not extensively treated in most traditional textbooks. The emphasis of this section is to establish the connection of divided differences with the derivatives of f and later to Hermite interpolation.

Section 3.5. The elegant proof of Theorem 3.3 has been attributed to Cauchy (see for example [Gau11]). The interpolation error in the form (3.64) was derived by Cauchy in 1840 [Cau40]. The minimization of the polynomial $w(x) = (x - x_0)\cdots(x - x_n)$ in the error by the zeros of T_{n+1} is covered in many textbooks (e.g. [Dav75, Hil13, SW89, Gau11]). However, the more practical bound (3.61) for the Chebyshev nodes (the extremal points of T_n) is more rarely found. The derivation here follows that of Salzer [Sal72].

Section 3.6. Gautschi [Gau11] makes the observation that (3.64) is a tautology because $f[x_0, \ldots, x_n, x]$ involves itself the value $f(x)$ so it really reduces to a trivial identity. However, the connection of divided differences with the derivatives of f obtained from (3.64) and the Cauchy remainder has important consequences and applications; one of them is Hermite interpolation.

Section 3.7. Hermite interpolation is treated more extensively in [SB02, KC02]. Here, we make use of the notion of coincident nodes (see e.g. [Dav75]) and the connection of divided differences with derivatives to link Hermite interpolation with Newton's interpolation form.

Section 3.8. Runge [Run01] presented his famous example $f(x) = 1/(1+x^2)$ in the interval $[-5, 5]$. Here, we have rescaled it for the interval $[-1, 1]$. He employs Hermite formula [Her78] for the interpolation error for the analysis of interpolation with equispaced nodes. The convergence theorem for polynomial interpolation and its proof have been adapted from [Kry12, For96].

Section 3.9 and *Section 3.10.* The canonical reference for splines is de Boor's monograph [dB78]. This interpolation subject is also excellently treated in the textbooks by Kincaid and Cheney [KC02], Schwarz [SW89], and Gautschi [Gau11], whose presentations inspired these two sections. The use of (3.146) for obtaining the parameter values t_j in splines for parametric, smooth curves is proposed in [SW89].

Section 3.11. Trigonometric interpolation appears in most modern numerical analysis textbooks, e.g. [SW89, KC02, SB02, Sau12]. It is a central topic of spectral methods.

Section 3.12. The FFT algorithm was proposed by Cooley and Tukey [CT65] in 1965. It is now understood [HJB85] that this famous algorithm was discovered much earlier by Gauss, around 1805. The sorting out of the coefficients (not described in this text) using binary representation of the indices is provided in [CT65]. Sauer's book [Sau12] has an excellent section on the FFT and signal processing and a chapter on the DCT and compression.

Section 3.13. Despite its usefulness, Chebyshev interpolation is rarely found in introductory numerical analysis textbooks. One exception is the book by Schwarz [SW89].