Approximation Theory and Approximation Practice

Lloyd N. Trefethen

January 30, 2018

Contents

1 Introduction 1

2 Chebyshev points and interpolants 7

3 Chebyshev polynomials and series 15

4 Interpolants, projections, and aliasing 29

5 Barycentric interpolation formula 39

6 Weierstrass Approximation Theorem 51

7 Convergence for differentiable functions 59

8 Convergence for analytic functions 67

9 Gibbs phenomenon 79

10 Best approximation 91

11 Hermite integral formula 101

12 Potential theory and approximation 111

13 Equispaced points, Runge phenomenon 119

14 Discussion of high-order interpolation 129

15 Lebesgue constants 135

16 Best and near-best 149

17 Orthogonal polynomials 159

18 Polynomial roots and colleague matrices 171

19 Clenshaw–Curtis and Gauss quadrature 185

20 Caratheodory–Fejer approximation 199

21 Spectral methods 213

22 Linear approximation: beyond polynomials 229

23 Nonlinear approximation: why rational functions? 245

24 Rational best approximation 259

25 Two famous problems 271

26 Rational interpolation and linearized least-squares 287

27 Padé approximation 305

28 Analytic continuation and convergence acceleration 325

REFERENCES 341

Chapter 1

Introduction

Welcome to a beautiful subject!—the constructive approximation of functions. And welcome to a rather unusual book.

Approximation theory is an established field, and my aim is to teach you some of its most important ideas and results, centered on classical topics related to polynomials and rational functions. The style of this book, however, is quite different from what you will find elsewhere. Everything is illustrated computationally with the help of the Chebfun software package in Matlab, from Chebyshev interpolants to Lebesgue constants, from the Weierstrass approximation theorem to the Remez algorithm. Everything is practical and fast, so we will routinely compute polynomial interpolants or Gauss quadrature weights for tens of thousands of points. In fact, each chapter of this book is a single Matlab M-file, and the book has been produced by executing these files with the Matlab "publish" facility. The chapters come from M-files called `chap1.m`, . . . , `chap28.m` and you can download them and use them as templates to be modified for explorations of your own.

Beginners are welcome, and so are experts, who will find familiar topics approached from new angles and familiar conclusions turned on their heads. Indeed, the field of approximation theory came of age in an era of polynomials of degrees perhaps O(10). Now that O(1000) is easy and O(1,000,000) is not hard, different questions come to the fore. For example, we shall see that "best" approximants are hardly better than "near-best", though they are much harder to compute, and that, contrary to widespread misconceptions, numerical methods based on high-order polynomials can be extremely efficient and robust.

This is a book about approximation, not Chebfun, and for the most part we shall use Chebfun tools with little explanation. For information about Chebfun, see `http://www.maths.ox.ac.uk/chebfun`. In the course of the book we shall use Chebfun overloads of the following Matlab functions, among others:

```
CONV, CUMSUM, DIFF, INTERP1, NORM, POLY, POLYFIT, ROOTS, SPLINE
```

as well as additional Chebfun commands such as

```
CF, CHEBELLIPSEPLOT, CHEBPADE, CHEBPOLY, CHEBPTS,
LEBESGUE, LEGPOLY, LEGPTS, PADEAPPROX,
RATINTERP, REMEZ.
```

There are quite a number of excellent books on approximation theory. Three classics are [Cheney 1966], [Davis 1975], and [Meinardus 1967], and a slightly more recent computationally oriented classic is [Powell 1981]. Perhaps the first approximation theory text was [Borel 1905].

A good deal of my emphasis will be on ideas related to Chebyshev points and polynomials, whose origins go back more than a century to mathematicians including Chebyshev (1821–1894), de la Vallée Poussin (1866–1962), Bernstein (1880–1968), and Jackson (1888–1946). In the computer era, some of the early figures who developed "Chebyshev technology," in approximately chronological order, were Lanczos, Clenshaw, Good, Fox, Elliott, Mason, Orszag, Paszkowski, and V. I. Lebedev. Five books on Chebyshev polynomials are by Snyder [1966], Paszkowski [1975], Fox and Parker [1968], Rivlin [1990], and Mason and Handscomb [2003]. One reason we emphasize Chebyshev technology so much is that in practice, for working with functions on intervals, these methods are unbeatable. For example, we shall see in Chapter 16 that the difference in approximation power between Chebyshev and "optimal" interpolation points is utterly negligible. Another reason is that if you know the Chebyshev material well, this is the best possible foundation for work on other approximation topics, and for understanding the links with Fourier analysis.

My style is conversational, but that doesn't mean the material is all elementary. The book aims to be more readable than most, and the numerical experiments help achieve this. At the same time, theorems are stated and proofs are given, often rather tersely, without all the details spelled out. It is assumed that the reader is comfortable with rigorous mathematical arguments and familiar with ideas like continuous functions on compact sets, Lipschitz continuity, contour integrals in the complex plane, and norms of operators. If you are a student, I hope you are an advanced undergraduate or graduate who has taken courses in numerical analysis and complex analysis. If you are a seasoned mathematician, I hope you are also a Matlab user.

Each chapter has a collection of exercises, which span a wide range from mathematical theory to Chebfun-based numerical experimentation. Please do not skip the numerical exercises! If you are going to do that, you might as well put this book aside and read one of the classics from the 1960s.

To give readers easy access to all the examples in executable form, the book was produced using `publish` in LaTeX mode: thus this chapter, for example, can be generated with the Matlab command `publish('chap1','latex')`. To achieve the desired layout, we begin each chapter by setting a few default parameters concerning line widths for plots, etc., which are collected in an M-file called `ATAPformats` that is included with the standard distribution of Chebfun. Most readers can ignore these details and simply apply `publish` to each chapter. For the actual production of the printed book, `publish` was executed not chapter-by-chapter but on a concatenation of all the chapters, and a few tweaks were made to the resulting LaTeX file, including removal of Matlab commands whose effects are evident from looking at the figures, like `title`, `axis`, `hold off`, and `grid on`.

The Lagrange interpolation formula was discovered by Waring, the Gibbs phenomenon was discovered by Wilbraham, and the Hermite integral formula is due to Cauchy. These are just some of the instances of Stigler's Law in approximation theory, and in writing this book I have taken pleasure in trying to cite the originator of each of the main ideas. Thus the entries in the references section stretch back several centuries, and each has an editorial comment attached. Often the original papers are surprisingly readable and insightful, at least if you are comfortable with French or German, and in any case, it seems particularly important to pay heed to original sources in a book like this that aims to reexamine material that has grown too standardized in the textbooks. Another reason for looking at original sources is that in the last few years it has become far easier to track them down, thanks to the digitization of journals, though there are always difficult special cases like [Wilbraham 1848], which I finally found in an elegant leather-bound volume in the Balliol College library. No doubt I have missed originators of certain ideas, and I would be glad to be corrected on such points by readers. For a great deal of information about approximation theory, including links to dozens of classic papers, see the History of Approximation Theory web site at `http://www.math.technion.ac.il/hat/`.

Perhaps I may add a further personal comment. As an undergraduate and graduate student in the late 1970s and early 1980s, one of my main interests was approximation theory. I regarded this subject as the foundation of my wider field of numerical analysis, but as the years passed, research in approximation theory came to seem to me dry and academic, and I moved into other areas. Now times have changed, computers have changed, and my perceptions have changed. I now again regard approximation theory as exceedingly close to computing, and in this book we shall discuss many practical numerical problems, including interpolation, quadrature, rootfinding, analytic continuation, extrapolation of sequences and series, and solution of differential equations.

Why is approximation theory useful? The answer goes much further than the rather tired old fact that your computer relies on approximations to evaluate functions like $\sin(x)$ and $\exp(x)$. For my personal answer to the question, concerning polynomials and rational functions in particular, take a look at the last three pages of Chapter 23, beginning with the quotes of Runge and Kirchberger from the beginning of the 20th century. There are also many other fascinating and important topics of approximation theory not touched upon in this volume, including splines, wavelets, radial basis functions, compressed sensing, and multivariate approximations of all kinds.

In summary, here are some distinctive features of this book:

- The emphasis is on topics close to numerical algorithms.
- Everything is illustrated with Chebfun.
- Each chapter is a publishable M-file, available online.
- There is a bias toward theorems and methods for analytic functions, which appear so often in applications, rather than on functions at the edge of discontinuity with their seductive theoretical challenges.
- Original sources are cited rather than textbooks, and each item in the bibliography is listed with an editorial comment.

At a more detailed level, virtually every chapter contains mathematical and scholarly novelties. Examples are the use of barycentric formulas beginning in Chapter 5, the tracing of barycentric formulas and the Hermite integral formula back to Jacobi in 1825 and Cauchy in 1826, Theorem 7.1 on the size of Chebyshev coefficients, the introduction to potential theory in Chapter 12, the discussion in Chapter 14 of prevailing misconceptions about interpolation, the presentation of colleague matrices for rootfinding in Chapter 18 with Jacobi matrices for quadrature as a special case in Chapter 19, Theorem 19.5 showing that Clenshaw–Curtis quadrature converges about as fast as Gauss quadrature, the first textbook presentation of Carathódory–Fejér approximation in Chapter 20, the explanation in Chapter 22 of why polynomials are not optimal functions for linear approximation, the extensive discussion in Chapter 23 of the uses of rational approximations, and the SVD-based algorithms for robust rational interpolation and linearized least-squares fitting and Padé approximation in Chapters 26 and 27.

All in all, we shall see that there is scarcely an idea in classical approximation theory that cannot be illustrated in a few lines of Chebfun code, and as I first imagined around 1975, anyone who wants to be expert at numerical computation really does need to know this material.

Dozens of people have helped me in preparing this book. I cannot name them all, but I would like to thank in particular Serkan Gugercin, Nick Higham, Jörg Liesen, Ricardo Pachón, and Ivo Panayotov for reading the whole text and making many useful suggestions, Jean-Paul Berrut for teaching me about rational functions and barycentric formulas, Folkmar Bornemann for bringing to light historical surprises involving Jacobi, Cauchy, and Marcel Riesz, and Volker Mehrmann for hosting a sabbatical visit to the Technical University of Berlin in 2010 during which much of the work was done. I am grateful to Max Jensen of the University of Durham, whose invitation to give a 50-minute talk in March 2009 sparked the whole project, and to Marlis Hochbruck and Caroline Lasser for testing a draft of the book with their students in Karlsruhe and Munich. Here in the Numerical Analysis Group at Oxford, Endre Süli and Andy Wathen have been the finest colleagues one could ask for these past fifteen years, and the remarkable Lotti Ekert makes everything run smoothly. Finally, none of this would have been possible without the team who have made Chebfun so powerful and beautiful, my good friends Zachary Battles, Ásgeir Birkisson, Toby Driscoll, Pedro Gonnet, Stefan Güttel, Nick Hale, Ricardo Pachón, Rodrigo Platte, Mark Richardson, and Alex Townsend.

Exercise 1.1. Chebfun download. Download Chebfun from the web site at `http://www.maths.ox.ac.uk/chebfun` and install it in your Matlab path as instructed there. Execute `chebtest` to make sure things are working, and note the time taken. Execute `chebtest` again and note how much speedup there is now that various files have been brought into memory. Now read Chapter 1 of the online *Chebfun Guide,* and look at the list of Examples.

Exercise 1.2. The publish command. Execute `help publish` and `doc publish` in Matlab to learn the basics of how the `publish` command works. Then download the files `chap1.m` and `chap2.m` from `http://www.maths.ox.ac.uk/chebfun/ATAP` and publish them with `publish('chap1','latex')` followed by appropriate LaTeX commands. (You will probably find that `chap1.tex` and `chap2.tex` appear in a subdirectory on your computer labeled `html`.) If you are a student taking a course for which you are expected to turn in writeups of the exercises, I recommend that you make it your habit to produce them with `publish`.

Exercise 1.3. Textbook X. Buy or borrow a copy of an approximation theory textbook, which we shall call X; good examples are the books of Achieser, Braess, Cheney, Davis, Lorentz, Meinardus, Natanson, Powell, Rice, Rivlin, Schönhage, Timan, and Watson listed in the References. As you work through *Approximation Theory and Approximation Practice*, keep X at your side and get in the habit of comparing treatments of each topic between *ATAP* and X. (a) What are the author, title, and publication date of X? (b) Where did/does the author work and what were/are his/her dates? (c) Look at the first three theorems in X and write down one of them that interests you. You do not have to write down the proof.

Chapter 2

Chebyshev points and interpolants

```
ATAPformats
```

Any interval $[a, b]$ can be scaled to $[-1, 1]$, so most of the time, we shall just talk about $[-1, 1]$.

Let n be a positive integer:

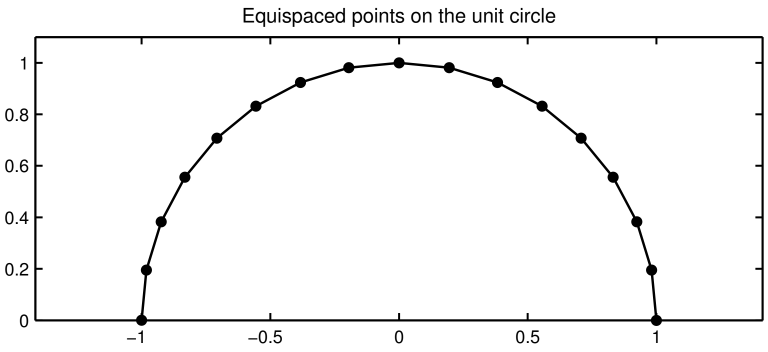
```
n = 16;
```

Consider $n + 1$ equally spaced angles $\{\theta_j\}$ from 0 to π:

```
tt = linspace(0,pi,n+1);
```

We can think of these as the arguments of $n + 1$ points $\{z_j\}$ on the upper half of the unit circle in the complex plane. These are the $(2n)$th roots of unity lying in the closed upper half-plane:

```
zz = exp(1i*tt);
hold off, plot(zz,'.-k'), axis equal, ylim([0 1.1])
FS = 'fontsize';
title('Equispaced points on the unit circle',FS,9)
```



The *Chebyshev points* associated with the parameter n are the real parts of these points,

$$x_j = \operatorname{Re} z_j = \frac{1}{2}(z_j + z_j^{-1}), \quad 0 \le j \le n : \tag{2.1}$$

```
xx = real(zz);
```

Some authors use the terms *Chebyshev–Lobatto points*, *Chebyshev extreme points*, or *Chebyshev points of the second kind*, but as these are the points most often used in practical computation, we shall just say Chebyshev points.

Another way to define the Chebyshev points is in terms of the original angles,

$$x_j = \cos(j\pi/n), \quad 0 \le j \le n, \tag{2.2}$$

```
xx = cos(tt);
```

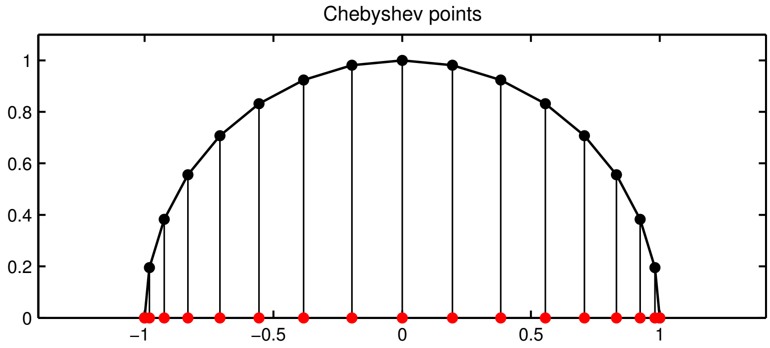
and the problem of polynomial interpolation in these points was considered at least as early as [Jackson 1913]. There is also an equivalent Chebfun command `chebpts`:

```
xx = chebpts(n+1);
```

Actually this result isn't exactly equivalent, as the ordering is left-to-right rather than right-to-left. Concerning rounding errors when these numbers are calculated numerically, see Exercise 2.3.

Let us add the Chebyshev points to the plot:

```
hold on
for j = 2:n
  plot([xx(n+2-j) zz(j)],'k','linewidth',0.7)
end
plot(xx,0*xx,'.r'), title('Chebyshev points',FS,9)
```



They cluster near 1 and -1, with the average spacing as $n \to \infty$ being given by a density function with square root singularities at both ends (Exercise 2.2).

Let $\{f_j\}$, $0 \le j \le n$, be a set of numbers, which may or may not come from sampling a function $f(x)$ at the Chebyshev points. Then there exists a unique polynomial p of degree n that interpolates these data, i.e., $p(x_j) = f_j$ for each j. When we say "of degree n," we mean of degree less than or equal to n, and we let $\mathcal{P}_n$ denote the set of all such polynomials:

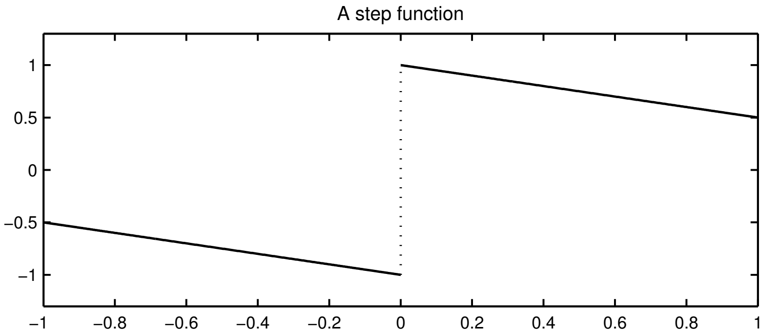
$$\mathcal{P}_n = \{\text{polynomials of degree at most } n\}. \tag{2.3}$$

As we trust the reader already knows, the existence and uniqueness of polynomial interpolants applies for any distinct set of interpolation points. In the case of Chebyshev points, we call the polynomial the *Chebyshev interpolant.*

Polynomial interpolants through equally spaced points have terrible properties, as we shall see in Chapters 11–15. Polynomial interpolants through Chebyshev points, however, are excellent. It is the clustering near the ends of the interval that makes the difference, and other sets of points with similar clustering, like Legendre points (Chapter 17), have similarly good behavior. The explanation of this fact has a lot to do with potential theory, a subject we shall introduce in Chapter 12. Specifically, what makes Chebyshev or Legendre points effective is that each one has approximately the same average distance from the others, as measured in the sense of the geometric mean. On the interval $[-1, 1]$, this distance is about 1/2 (Exercise 2.6).

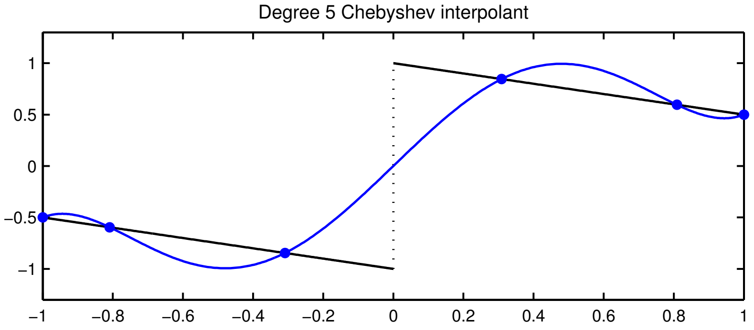
Chebfun is built on Chebyshev interpolants [Battles & Trefethen 2004]. For example, here is a certain step function:

```
x = chebfun('x');
f = sign(x) - x/2;
hold off, plot(f,'k'), ylim([-1.3 1.3])
title('A step function',FS,9)
```



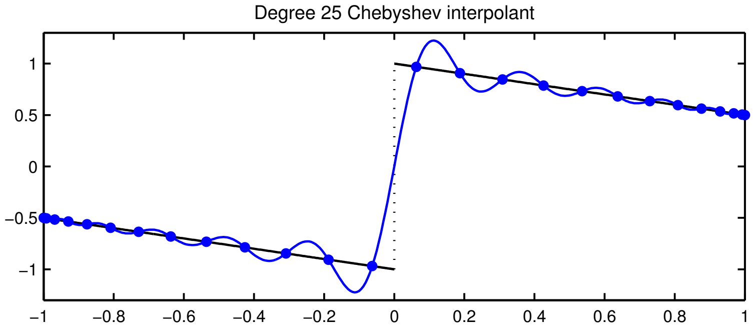
By calling **chebfun** with a second explicit argument of 6, we can construct the Chebyshev interpolant to f through 6 points, that is, of degree 5:

```
p = chebfun(f,6); hold on, plot(p,'.-'), ylim([-1.3 1.3])
title('Degree 5 Chebyshev interpolant',FS,9)
```



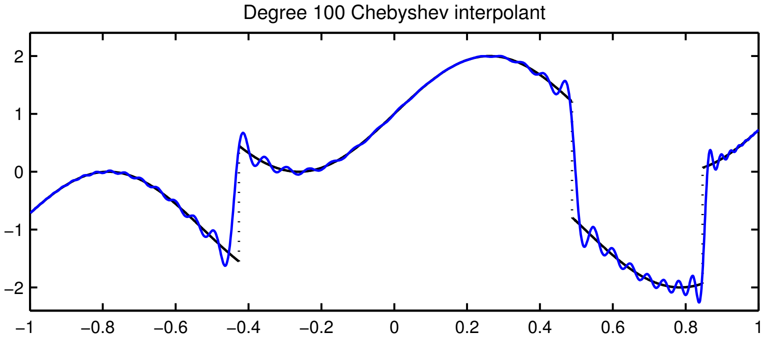
Similarly, here is the Chebyshev interpolant of degree 25:

```
hold off, plot(f,'k')
p = chebfun(f,26); hold on, plot(p,'.-')
ylim([-1.3 1.3]), title('Degree 25 Chebyshev interpolant',FS,9)
```



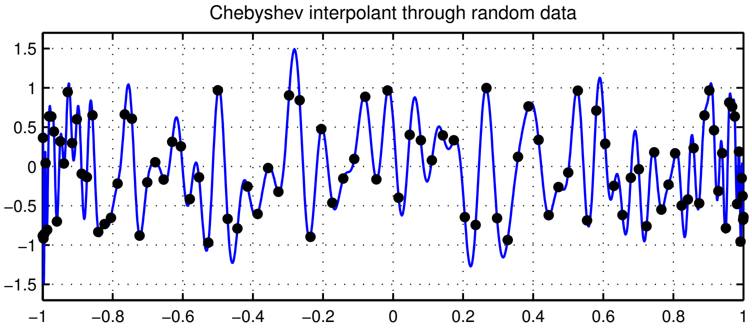
Here are a more complicated function and its interpolant of degree 100:

```
f = sin(6*x) + sign(sin(x+exp(2*x)));
hold off, plot(f,'k')
p = chebfun(f,101); hold on, plot(p), ylim([-2.4 2.4])
title('Degree 100 Chebyshev interpolant',FS,9)
```



Another way to use the `chebfun` command is by giving it an explicit vector of data rather than a function to sample, in which case it interprets the vector as data for a Chebyshev interpolant of the appropriate order. Here for example is the interpolant of degree 99 through random data values at 100 Chebyshev points in $[-1, 1]$:

```
p = chebfun(2*rand(100,1)-1);
hold off, plot(p,'-'), hold on, plot(p,'.k')
ylim([-1.7 1.7]), grid on
title('Chebyshev interpolant through random data',FS,9)
```



This experiment illustrates how robust Chebyshev interpolation is. If we had taken a million points instead of 100, the result would not have been much different mathematically, though it would have been a mess to plot. We shall return to this figure in Chapter 15.

For illustrations like these it is interesting to pick data with jumps or wiggles, and Chapter 9 discusses such interpolants systematically. In applications where polynomial interpolants are most useful, however, the data will typically be smooth.

> SUMMARY OF CHAPTER 2. *Polynomial interpolants in equispaced points in* $[-1,1]$ *have very poor approximation properties, but interpolants in Chebyshev points, which cluster near* ± 1*, are excellent.*

Exercise 2.1. Chebyshev interpolants through random data. (a) Repeat the experiment of interpolation through random data for 10, 100, 1000, and 10000 points. In each case use `minandmax(p)` to determine the minimum and maximum values of the interpolant and measure the computer time required for this computation (e.g. using `tic` and `toc`). You may find it helpful to increase Chebfun's standard plotting resolution with a command like `plot(p,'numpts',10000)`. (b) In addition to the four plots over $[-1,1]$, use `plot(p,'.-','interval',[0.9999 1])` to produce another plot of the 10000-point interpolant in the interval $[0.9999, 1]$. How many of the 10000 grid points fall in this interval?

Exercise 2.2. Limiting density as $n \to \infty$. (a) Suppose $x_0, \dots, x_n$ are $n+1$ points equally spaced from -1 to 1. If $-1 \le a < b \le 1$, what fraction of the points fall in the interval $[a,b]$ in the limit $n \to \infty$? Give an exact formula. (b) Give the analogous formula for the case where $x_0, \dots, x_n$ are the Chebyshev points. (c) How does the result of (b) match the number found in $[0.9999, 1]$ in the last exercise for the case $n = 9999$? (d) Show that in the limit $n \to \infty$, the density of the Chebyshev points near $x \in (-1,1)$ approaches $N/(\pi\sqrt{1-x^2})$ (see equation (12.10)).

Exercise 2.3. Rounding errors in computing Chebyshev points. On a computer in floating point arithmetic, the formula (2.2) for the Chebyshev points is not so good, because it lacks the expected symmetries. (a) Write a Matlab program that finds the smallest even value $n \geq 2$ for which, on your computer as computed by this formula, $x_{n/2} \neq 0$. (You will probably find that $n = 2$ is the first such value.) (b) Find the line in the code `chebpts.m` in which Chebfun computes Chebyshev points. What alternative formula does it use? Explain why this formula achieves perfect symmetry for all n in floating point arithmetic. (c) Show that this formula is mathematically equivalent to (2.2).

Exercise 2.4. Chebyshev points of the first kind. The Chebyshev points of the first kind, also known as **Gauss–Chebyshev points**, are obtained by taking the real parts of points on the unit circle mid-way between those we have considered, i.e. $x_j = \cos((j+\frac{1}{2})\pi/(n+1))$ for integers $0 \leq j \leq n$. Call `help chebpts` and `help legpts` to find out how to generate these points in Chebfun and how to generate Legendre points for comparison (these are roots of Legendre polynomials—see Chapter 17). For $n+1 = 100$, what is the maximum difference between a Chebyshev point of the first kind and the corresponding Legendre point? Draw a plot to illustrate as informatively as you can how close these two sets of points are.

Exercise 2.5. Convergence of Chebyshev interpolants. (a) Use Chebfun to produce a plot on a log scale of $\|f - p_n\|$ as a function of n for $f(x) = e^x$ on $[-1,1]$, where p_n is the Chebyshev interpolant in $\mathcal{P}_n$. Take $\|\cdot\|$ to be the supremum norm, which can be computed by `norm(f-p,inf)`. How large must n be for accuracy at the level of machine precision? What happens if n is increased beyond this point? (b) The same questions for $f(x) = 1/(1+25x^2)$. Convergence rates like these will be analyzed in Chapters 7 and 8.

Exercise 2.6. Geometric mean distance between points. Write a code `meandistance` that takes as input a vector of points $x_0, \ldots, x_n$ in $[-1,1]$ and produces a plot with x_j on the horizontal axis and the geometric mean of the distances of x_j to the other points on the vertical axis. (The Matlab command `prod` may be useful.) (a) What are the results for Chebyshev points with $n = 5, 10, 20$? (b) The same for Legendre points (see Exercise 2.4). (c) The same for equally spaced points from $x_0 = -1$ to $x_n = 1$.

Exercise 2.7. Chebyshev points scaled to the interval $[a,b]$. (a) Use `chebpts(10)` to print the values of the Chebyshev points in $[-1,1]$ for $n = 9$. (b) Use `chebfun(@sin,10)` to compute the degree 9 interpolant $p(x)$ to $\sin(x)$ in these points. Make a plot showing $p(x)$ and $\sin(x)$ over the larger interval $[-6,6]$, and also a semilog plot of $|f(x) - p(x)|$ over that interval. Comment on the results. (c) Now use `chebpts(10,[0 6])` to print the values of the Chebyshev points for $n = 9$ scaled to the interval $[0,6]$. (d) Use `chebfun(@sin,[0 6],10)` to compute the degree 9 interpolant to $\sin(x)$ in these points, and make the same two plots as before over $[-6, 6]$. Comment.

Chapter 3

Chebyshev polynomials and series

`ATAPformats`

Throughout applied mathematics, one encounters three closely analogous canonical settings associated with the names of Fourier, Laurent, and Chebyshev. In fact, if we impose certain symmetries in the Fourier and Laurent cases, the analogies become equivalences. The Chebyshev setting is the one of central interest in this book, concerning a variable x and a function f defined on $[-1,1]$:

$$\textit{Chebyshev:}\quad x\in[-1,1],\quad f(x)\approx\sum_{k=0}^{n}a_kT_k(x). \tag{3.1}$$

Here T_k is the kth Chebyshev polynomial, which we shall discuss in a moment. For the equivalent Laurent problem, let z be a variable that ranges over the unit circle in the complex plane. Given $f(x)$, define a transplanted function $F(z)$ on the unit circle by the condition $F(z)=f(x)$, where $x=(z+z^{-1})/2$ as in (2.1). Note that this means that there are two values of z for each value of x, and F satisfies the symmetry property $F(z)=F(z^{-1})$. The series now involves a polynomial in both z and z^{-1}, known as a *Laurent polynomial:*

$$\textit{Laurent:}\quad |z|=1,\quad F(z)=F(z^{-1})\approx\frac{1}{2}\sum_{k=0}^{n}a_k(z^k+z^{-k}). \tag{3.2}$$

For the equivalent Fourier problem, let θ be a variable that ranges over $[-\pi,\pi]$, which we regard as a 2π-periodic domain. Transplant f and F to a function $\mathcal{F}$ defined on $[-\pi,\pi]$ by setting $\mathcal{F}(\theta)=F(e^{i\theta})=f(\cos(\theta))$ as in (2.2). Now we have a 1-to-1 correspondence $z=e^{i\theta}$ between θ and z and a

2-to-1 correspondence between θ and x, with the symmetry $\mathcal{F}(\theta) = \mathcal{F}(-\theta)$, and the series is a trigonometric polynomial:

$$\textit{Fourier:} \quad \theta \in [-\pi, \pi], \quad \mathcal{F}(\theta) = \mathcal{F}(-\theta) \approx \frac{1}{2}\sum_{k=0}^{n} a_k(e^{ik\theta} + e^{-ik\theta}). \tag{3.3}$$

One can carry (3.1)–(3.3) further by introducing canonical systems of grid points in the three settings. We have already seen the $(n{+}1)$-point Chebyshev grid,

$$\textit{Chebyshev points:} \quad x_j = \cos(j\pi/n), \quad 0 \le j \le n, \tag{3.4}$$

and we have interpreted these in terms of the $(2n)$th roots of unity:

$$\textit{Roots of unity:} \quad z_j = e^{ij\pi/n}, \quad -n+1 \le j \le n. \tag{3.5}$$

These grids are transplants of the set of $2n$ equispaced points in $[-\pi, \pi]$:

$$\textit{Equispaced points:} \quad \theta_j = j\pi/n, \quad -n+1 \le j \le n. \tag{3.6}$$

All three of these settings are unassailably important. Real analysts cannot do without Fourier, complex analysts cannot do without Laurent, and numerical analysts cannot do without Chebyshev. Moreover, the mathematics of the connections between the three frameworks is beautiful. But all this symmetry presents an expository problem. Without a doubt, a fully logical treatment should consider x, z and θ in parallel. Each theorem should appear in three forms. Each application should be one of a trio.

It was on this basis that I started to write a book in 2008. The symmetries were elegant, but as the chapters accumulated, I came to realize that this would be a very long book and not a lovable one. The excellent logic was just a dead weight. The next year, I started again with the decision that the book would focus on $x \in [-1, 1]$. This is the setting closest to much of approximation theory and numerical analysis, and it has a further special feature: it is the one least familiar to people. Nobody is surprised if you compute a Fourier transform of a million points, but the fact that you can compute a polynomial interpolant through a million Chebyshev points surprises people indeed.

Here then is the mathematical plan for this book. Our central interest will be the approximation of functions $f(x)$ on $[-1, 1]$. When it comes to deriving formulas and proving theorems, however, we shall generally transplant to $F(z)$ on the unit circle so as to make the tools of complex analysis most conveniently available.

Now let us turn to the definitions, already implicit in (3.1)–(3.3). The kth *Chebyshev polynomial* can be defined as the real part of the function z^k on the unit circle:

$$x = \mathrm{Re}(z) = \tfrac{1}{2}(z + z^{-1}) = \cos\theta, \quad \theta = \cos^{-1} x, \tag{3.7}$$

$$T_k(x) = \mathrm{Re}(z^k) = \tfrac{1}{2}(z^k + z^{-k}) = \cos(k\theta). \tag{3.8}$$

(Chebyshev polynomials were introduced by Chebyshev in the 1850s, though without the connection to the variables z and θ [Chebyshev 1854 & 1859]. The label T was apparently chosen by Bernstein, following French transliterations such as "Tchebischeff.") The Chebyshev polynomials are a family of orthogonal polynomials with respect to a certain weight function (Exercise 3.7), but we shall not make much use of orthogonality until Chapters 17–19.

It follows from (3.8) that T_k satisfies $-1 \le T_k(x) \le 1$ for $x \in [-1, 1]$ and takes alternating values ±1 at the $k+1$ Chebyshev points. What is not obvious is that T_k is a polynomial. We can verify this property by the computation

$$\tfrac{1}{2}(z + z^{-1})(z^k + z^{-k}) = \tfrac{1}{2}(z^{k+1} + z^{-k-1}) + \tfrac{1}{2}(z^{k-1} + z^{-k+1})$$

for any $k \ge 1$, that is,

$$2xT_k(x) = T_{k+1}(x) + T_{k-1}(x), \tag{3.9}$$

or in other words

$$T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x). \tag{3.10}$$

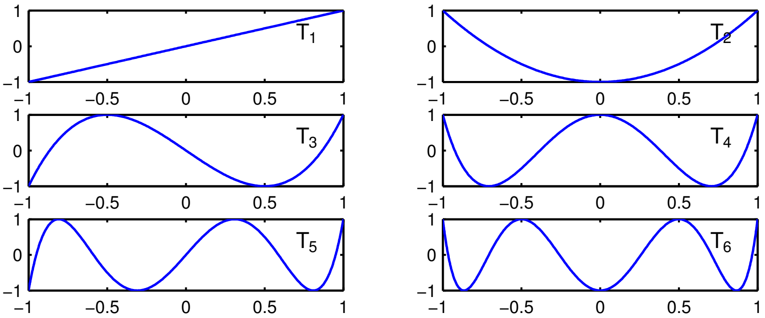
By induction, this three-term recurrence relation implies that for each $k \ge 1$, T_k is a polynomial of degree exactly k with leading coefficient 2^{k-1}. In Chapters 18 and 19 the coefficients of this recurrence will be taken as the entries of a "colleague matrix," whose eigenvalues can be computed to find roots of polynomials or quadrature nodes.

The Chebfun command `chebpoly(n)` returns the chebfun corresponding to T_n.[1] Here for example are $T_1, \dots, T_6$:

```
FS = 'fontsize';
for n = 1:6
```

[1] The name of the software system is Chebfun, with a capital C. A representation of a particular function in Chebfun is called a chebfun, with a lower-case c.

```
    T{n} = chebpoly(n);
    subplot(3,2,n), plot(T{n}), axis([-1 1 -1 1])
    text(.7,.41,'T',FS,10), text(.78,.24,int2str(n),FS,7)
end
```



These plots do not show the Chebyshev points, which are the extremes of each curve: thus the numbers of Chebyshev points in the six plots are 2, 3, 4, 5, 6, and 7.

Here are the coefficients of these polynomials with respect to the monomial basis $1, x, x^2, \ldots$. As usual, Matlab orders coefficients from highest degree down to degree zero.

```
for n = 1:6, disp(poly(T{n})), end
```

```
     1     0
     2     0    -1
     4     0    -3     0
     8     0    -8     0     1
    16     0   -20     0     5     0
    32     0   -48     0    18     0    -1
```

So, for example,

$$T_5(x) = 16x^5 - 20x^3 + 5x.$$

The monomial basis is familiar and comfortable, but you should never use it for numerical work with functions on an interval. Use the Chebyshev basis instead (Exercise 3.8). (If the domain is $[a, b]$ rather than $[-1, 1]$, the Chebyshev polynomials must be scaled accordingly, and Chebfun does this automatically when it works on other intervals.) For example, x^5 has the Chebyshev expansion

$$x^5 = \frac{5}{80}T_5(x) + \frac{5}{16}T_3(x) + \frac{5}{8}T_1(x).$$

We can calculate such expansion coefficients by using the command `chebpoly(p)`, where p is the chebfun whose coefficients we want to know:

```
format short, x = chebfun('x'); chebpoly(x.^5)
```

```
Warning: CHEBPOLY is deprecated. Please use CHEBCOEFFS instead.
ans =
    0.0625         0    0.3125         0    0.6250         0
```

Any polynomial p can be written uniquely like this as a finite Chebyshev series: the functions $T_0(x), T_1(x), \ldots, T_n(x)$ form a basis for $\mathcal{P}_n$. Since p is determined by its values at Chebyshev points, it follows that there is a one-to-one linear mapping between values at Chebyshev points and Chebyshev expansion coefficients. This mapping can be applied in $O(n \log n)$ operations with the aid of the Fast Fourier Transform (FFT) or the Fast Cosine Transform, a crucial observation for practical work that was perhaps first made by Ahmed and Fisher and Orzsag around 1970 [Ahmed & Fisher 1970, Orszag 1971a and 1971b, Gentleman 1972b, Geddes 1978]. This is what Chebfun does every time it constructs a chebfun. We shall not give details.

Just as a polynomial p has a finite Chebyshev series, a more general function f has an infinite Chebyshev series. Exactly what kind of "more general function" can we allow? For an example like $f(x) = e^x$ with a rapidly converging Taylor series, everything will surely be straightforward, but what if f is merely differentiable rather than analytic? Or what if it is continuous but not differentiable? Analysts have studied such cases carefully, identifying exactly what degrees of smoothness correspond to what kinds of convergence of Chebyshev series. We shall not concern ourselves with trying to state the sharpest possible result but will just make a particular assumption that covers most applications. We shall assume that f is *Lipschitz continuous* on $[-1, 1]$. Recall that this means that there is a constant C such that $|f(x) - f(y)| \le C|x - y|$ for all $x, y \in [-1, 1]$. Recall also that a series is *absolutely convergent* if it remains convergent if each term is replaced by its absolute value, and that this implies that one can reorder the terms arbitrarily without changing the result. Such matters are discussed in analysis textbooks such as [Rudin 1976].

Here is our basic theorem about Chebyshev series and their coefficients.

Theorem 3.1. Chebyshev series. *If f is Lipschitz continuous on $[-1, 1]$, it has a unique representation as a Chebyshev series,*

$$f(x) = \sum_{k=0}^{\infty} a_k T_k(x), \tag{3.11}$$

which is absolutely and uniformly convergent, and the coefficients are given for $k \geq 1$ by the formula

$$a_k = \frac{2}{\pi} \int_{-1}^{1} \frac{f(x)T_k(x)}{\sqrt{1-x^2}} dx, \tag{3.12}$$

and for $k = 0$ by the same formula with the factor $2/\pi$ changed to $1/\pi$.

Proof. Formula (3.12) will come from the Cauchy integral formula, and to make this happen, we begin by transplanting f to F on the unit circle as described above: $F(z) = F(z^{-1}) = f(x)$ with $x = \operatorname{Re} z = (z + z^{-1})/2$. To convert between integrals in x and z, we have to convert between dx and dz:

$$dx = \tfrac{1}{2}(1 - z^{-2})\, dz = \tfrac{1}{2} z^{-1}(z - z^{-1})\, dz.$$

Since

$$\tfrac{1}{2}(z - z^{-1}) = i \operatorname{Im} z = \pm i\sqrt{1-x^2},$$

this implies

$$dx = \pm i z^{-1}\sqrt{1-x^2}\, dz.$$

In these equations the plus sign applies for $\operatorname{Im} z \geq 0$ and the minus sign for $\operatorname{Im} z \leq 0$.

These formulas have implications for smoothness. Since $\sqrt{1-x^2} \leq 1$ for all $x \in [-1,1]$, they imply that if $f(x)$ is Lipschitz continuous, then so is $F(z)$. By a standard result in Fourier analysis, this implies that F has a unique representation as an absolutely and uniformly convergent Laurent series on the unit circle,

$$F(z) = \frac{1}{2}\sum_{k=0}^{\infty} a_k (z^k + z^{-k}) = \sum_{k=0}^{\infty} a_k T_k(x).$$

Recall that a *Laurent series* is an infinite series in both positive and negative powers of z, and that if F is analytic, such a series converges in the interior of an annulus. A good treatment of Laurent series for analytic functions can be found in [Markushevich 1985]; see also other complex variables texts such as [Hille 1973, Priestley 2003, Saff & Snider 2003].

The kth Laurent coefficient of a Lipschitz continuous function $G(z) = \sum_{k=-\infty}^{\infty} b_k z^k$ on the unit circle can be computed by the Cauchy integral formula,

$$b_k = \frac{1}{2\pi i} \int_{|z|=1} z^{-1-k} G(z)\, dz.$$

(We shall make more substantial use of the Cauchy integral formula in Chapters 11–12.) The notation $|z| = 1$ indicates that the contour consists of the unit circle traversed once in the positive (counterclockwise) direction. Here we have a function F with the special symmetry property $F(z) = F(z^{-1})$, and we have also introduced a factor 1/2 in front of the series. Accordingly, we can compute the coefficients a_k from either of two contour integrals,

$$a_k = \frac{1}{\pi i}\int_{|z|=1} z^{-1+k}F(z)\,dz = \frac{1}{\pi i}\int_{|z|=1} z^{-1-k}F(z)\,dz, \qquad (3.13)$$

with πi replaced by $2\pi i$ for $k = 0$.

In particular, we can get a formula for a_k that is symmetric in k and $-k$ by combining the two integrals like this:

$$a_k = \frac{1}{2\pi i}\int_{|z|=1} (z^{-1+k} + z^{-1-k})F(z)\,dz = \frac{1}{\pi i}\int_{|z|=1} z^{-1}\,T_k(x)F(z)\,dz, \quad (3.14)$$

with πi replaced by $2\pi i$ for $k = 0$. Replacing $F(z)$ by $f(x)$ and $z^{-1}dz$ by $-i\,dx/(\pm\sqrt{1-x^2})$ gives

$$a_k = -\frac{1}{\pi}\int_{|z|=1} \frac{f(x)T_k(x)}{\pm\sqrt{1-x^2}}\,dx,$$

with π replaced by 2π for $k = 0$. We have now almost entirely converted to the x variable, except that the contour of integration is still the circle $|z| = 1$. When z traverses the circle all the way around in the positive direction, x decreases from 1 to -1 and then increases back to 1 again. At the turning point $z = x = -1$, the ± sign attached to the square root switches from + to −. Thus instead of cancelling, the two traverses of $x \in [-1, 1]$ contribute equal halves to a_k. Converting to a single integration from -1 to 1 in the x variable multiplies the integral by $-1/2$, hence multiplies the formula for a_k by -2, giving (3.12). ∎

We now know that any function f, so long as it is Lipschitz continous, has a Chebyshev series. Chebfun represents a function as a finite series of some degree n, storing both its values at Chebyshev points and also, equivalently, their Chebyshev coefficients. How does it figure out the right value of n? Given a set of $n + 1$ samples, it converts the data to a Chebyshev expansion of degree n and examines the resulting Chebyshev coefficients. If several of these in a row fall below a relative level of approximately 10^{-15}, then the grid is judged to be fine enough. For example, here are the Chebyshev coefficients of the chebfun corresponding to e^x:

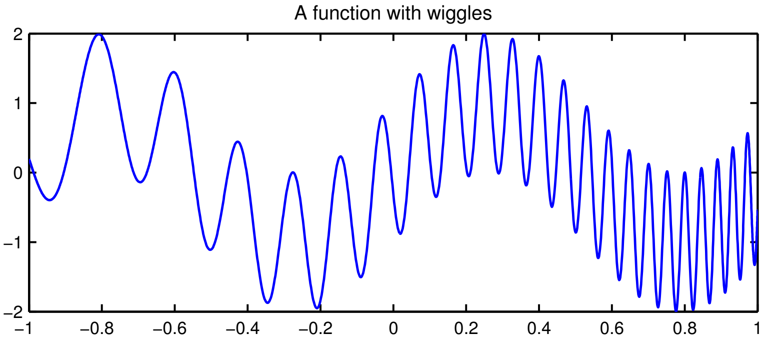
```
f = exp(x); a = chebpoly(f); format long, a(end:-1:1)'
```

```
ans =
   1.266065877752008
   1.130318207984970
   0.271495339534077
   0.044336849848664
   0.005474240442094
   0.000542926311914
   0.000044977322954
   0.000003198436462
   0.000000199212481
   0.000000011036772
   0.000000000550590
   0.000000000024980
   0.000000000001039
   0.000000000000040
   0.000000000000001
```

Notice that the last coefficient is about at the level of machine precision.

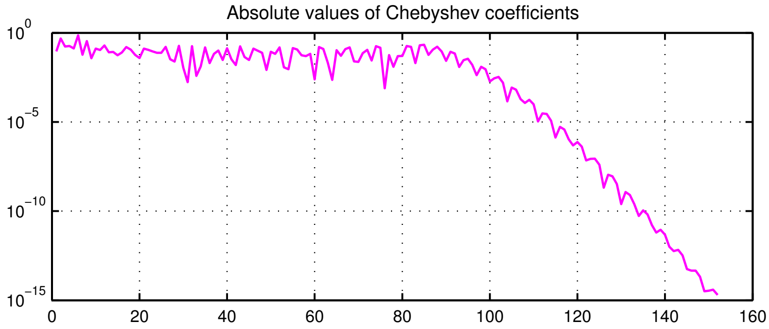
For complicated functions it is often more interesting to plot the coefficients than to list them. For example, here is a function with a number of wiggles:

```
f = sin(6*x) + sin(60*exp(x));
clf, plot(f), title('A function with wiggles',FS,9)
```



If we plot the absolute values of the Chebyshev coefficients, here is what we find:

```
a = chebpoly(f); semilogy(abs(a(end:-1:1)),'m')
grid on, title('Absolute values of Chebyshev coefficients',FS,9)
```



One can explain this plot as follows. Up to degree about $k = 80$, a Chebyshev series cannot resolve f much at all, for the oscillations occur on too short wavelengths. After that, the series begins to converge rapidly. By the time we reach $k = 150$, the accuracy is about 15 digits, and the computed Chebyshev series is truncated there. We can find out exactly where the truncation took place with the command `length(f)`:

```
length(f)
```

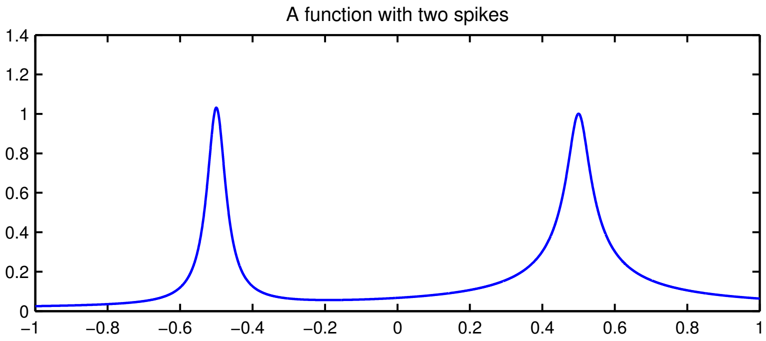
```
ans =
   152
```

This tells us that the chebfun is a polynomial interpolant through 151 points, that is, of degree 150.

Without giving all the engineering details, here is a fuller description of how Chebfun constructs its approximation. First it calculates the polynomial interpolant through the function sampled at 9 Chebyshev points, i.e., a polynomial of degree 8, and checks whether the Chebyshev coefficients appear to be small enough. For the example just given, the answer is no. Then it tries 17 points, then 33, then 65, and so on. In this case Chebfun judges at 257 points that the Chebyshev coefficients have fallen to the level of rounding error. At this point it truncates the tail of terms deemed to be negligible, leaving a series of 151 terms (Exercise 3.13). The corresponding degree 150 polynomial is then evaluated at 151 Chebyshev points via FFT, and these 151 numbers become the data defining this particular chebfun. Engineers would say that the signal has been *downsampled* from 257 points to 151.

For another example we consider a function with two spikes:

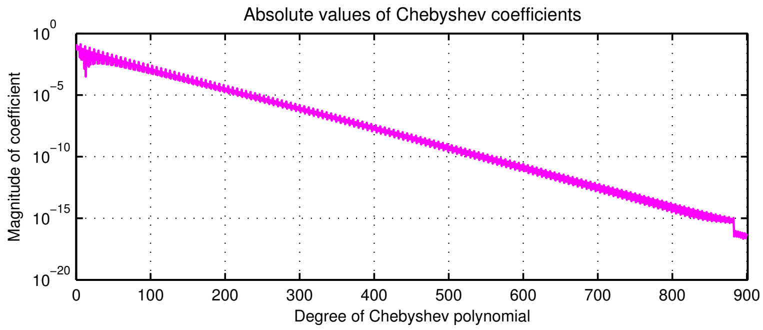
```
f = 1./(1+1000*(x+.5).^2) + 1./sqrt(1+1000*(x-.5).^2);
clf, plot(f), title('A function with two spikes',FS,9)
```



Here are the Chebyshev coefficients of the chebfun. This time, instead of `chebpoly` and `semilogy`, we execute the special command `chebpolyplot`, which has the same effect.

```
chebpolyplot(f,'m'), grid on
title('Absolute values of Chebyshev coefficients',FS,9)
```

```
Warning: CHEBPOLYPLOT is deprecated. Please use PLOTCOEFFS instead.
```



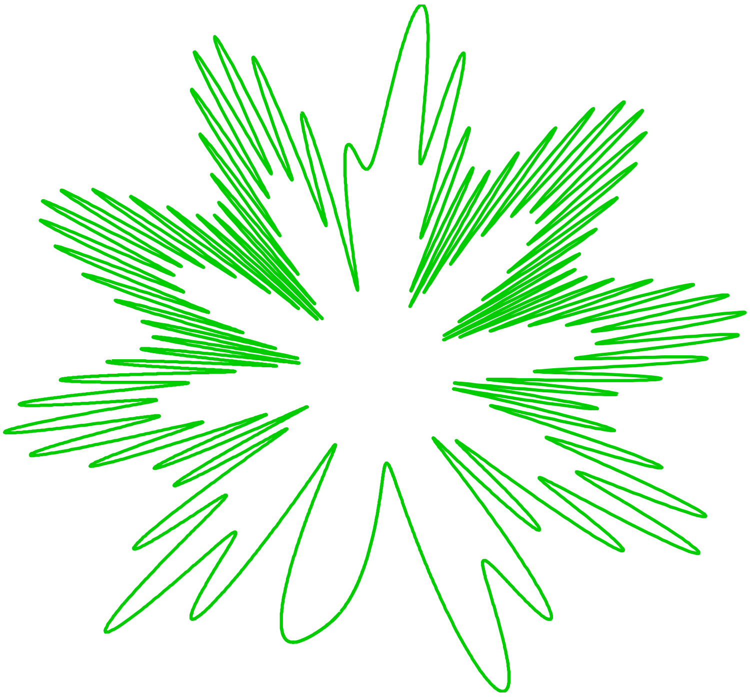
Note that although it is far less wiggly, this function needs six times as many points to resolve as the previous one (Exercise 3.13). We shall explain these polynomial degrees in Chapter 8.

Chebyshev interpolants are effective for complex functions (still defined on a real interval) as well as real ones. Here, for example, is a complex function that happens to be periodic, though the Chebyshev representation does not take advantage of this fact.

```
f = (3+sin(10*pi*x)+sin(61*exp(.8*sin(pi*x)+.7))).*exp(1i*pi*x);
```

A plot shows the image of $[-1,1]$ under f, which appears complicated:

```
plot(f,'linewidth',0.6,'color',[0 .8 0]), axis equal off
```



Yet the degree of the polynomial is not so high:

```
length(f)
```

```
ans =
   599
```

People often ask, is there anything special about Chebyshev points and Chebyshev polynomials? Could we equally well interpolate in other points and expand in other sets of polynomials? From an approximation point of view, the answer is yes, and in particular, Legendre points and Legendre polynomials have much the same power for representing a general function f, as we shall see in Chapters 17–19. Legendre points and polynomials are neither much better than Chebyshev for approximating functions, nor much worse; they are essentially the same. One can improve upon both Legendre and Chebyshev, shrinking the number of sample points needed to represent a given function by a factor of up to $\pi/2$, but to do so one must leave the class of polynomials. See Chapter 22.

Nevertheless, there is a big advantage of Chebyshev over Legendre points, and this is that one can use the FFT to go from point values to coefficients and back again. There are algorithms that make such computations practicable for Legendre interpolants too [Piessens 1974, Alpert & Rokhlin 1991, Dutt, Gu & Rokhlin 1996, Potts, Steidl & Tasche 1998, Iserles 2011]—see also Theorem 19.6 of this book—but Chebyshev remains the easy case.

SUMMARY OF CHAPTER 3. *The Chebyshev polynomial $T_k(x)$ is an analogue for $[-1,1]$ of the monomial z^k on the unit circle. Each Lipschitz continuous function f on $[-1,1]$ has an absolutely and uniformly convergent Chebyshev series, that is, an expansion $f(x) = a_0T_0(x) + a_1T_1(x) + \dots$.*

Exercise 3.1. Monomial and Chebyshev coefficients. Let $p \in \mathcal{P}_n$ have coefficient vectors $a = (a_0, a_1, \dots, a_n)^T$ for a Chebyshev series and $b = (b_0, b_1, \dots, b_n)^T$ for a series in the monomials $1, x, \dots, x^n$. Show that a and b are related by $Aa = b$, where A is an upper-triangular matrix, whose entries you should describe precisely, though you don't have to give explicit formulas for them. Prove that any $p \in \mathcal{P}_n$ has uniquely defined coefficient vectors a and b for both representations.

Exercise 3.2. A Chebyshev coefficient. Use Chebfun to determine numerically the coefficient of T_5 in the Chebyshev expansion of $\tan^{-1}(x)$ on $[-1,1]$.

Exercise 3.3. Chebyshev coefficients and "rat". (a) Use Chebfun to determine numerically the coefficients of the Chebyshev series for $1 + x^3 + x^4$. By inspection, identify these rational numbers. Use the Matlab command `[n,d] = rat(c)` to confirm this. (b) Use Chebfun and `rat` to make good guesses as to the Chebyshev coefficients of $x^7/7 + x^9/9$. (Of course it is not hard to figure them out analytically.)

Exercise 3.4. Dependence on wave number. (a) Calculate the length $L(k)$ of the chebfun corresponding to $f(x) = \sin(kx)$ on $[-1,1]$ for $k = 1, 2, 4, 8, \dots, 2^{10}$. (You can do this elegantly by defining a Matlab anonymous function `f = @(k)....`) Make a loglog plot of $L(k)$ as a function of k and comment on the result. (b) Do the same for $g(x) = 1/(1 + (kx)^2)$.

Exercise 3.5. Chebyshev series of a complicated function. (a) Make chebfuns of the three functions $f(x) = \tanh(x)$, $g(x) = 10^{-5}\tanh(10x)$, $h(x) = 10^{-10}\tanh(100x)$ on $[-1,1]$, and call `chebpolyplot` to show their Chebyshev coefficients. Comment on the results. (b) Now define $s = f + g + h$ and comment on the result of `chebpolyplot` applied to s. Chebfun does not automatically chop the tail of a Chebyshev series obtained by summation, but applying the `simplify` command will do this. What happens with `chebpolyplot(simplify(s))`?

Exercise 3.6. Chebyshev series of sign(x) and $|x|$ [Bernstein 1914]. Derive the following Chebyshev series coefficients by using the first equality in (3.14). (a)

For $f(x) = \text{sign}(x)$, $a_k = 0$ for k even and $a_k = (4/\pi)(-1)^{k-1}/k$ for k odd. (b) For $f(x) = |x|$, $a_k = 0$ for k odd, $a_0 = 2/\pi$, and $a_k = (4/\pi)(-1)^{(k/2)}/(1-k^2)$ for $k \geq 2$ even.

Exercise 3.7. Orthogonality of Chebyshev polynomials. Equation (3.12) gives the Chebyshev coefficient a_k of f by integration of f against just the single Chebyshev polynomial T_k. This formula implies an orthogonality property for $\{T_j\}$ involving a weighted integral. State exactly what this orthogonality property is and show carefully how it follows from the equations of this chapter.

Exercise 3.8. Conditioning of the Chebyshev basis. Although the Chebyshev polynomials are not orthogonal with respect to the standard unweighted inner product, they are close enough to orthogonal to provide a well-behaved basis. Set `T = chebpoly(0:10)` and explore the Chebfun "quasimatrix" that results with commands like `size(T)`, `spy(T)`, `plot(T)`, `svd(T)`. Explain the meaning of `T` (you may find Chapter 6 of the Chebfun Guide helpful) and determine the condition number of this basis with `cond(T)`. (b) Now construct the corresponding quasimatrix of monomials by executing `x = chebfun('x'); M = T; for j = 0:10, M(:,j+1) = x.^j; end`. What is the condition number of `M`? (c) Produce a plot of these two condition numbers for quasimatrices whose columns span $\mathcal{P}_n$ over $[-1,1]$ for $n = 0, 1, \ldots, 10$. (d) What happens to the condition numbers if M is constructed from monomials on $[0,1]$ rather than $[-1,1]$ via `x = chebfun('x',[0,1])`?

Exercise 3.9. Derivatives at endpoints. Prove from (3.10) that the derivatives of the Chebyshev polynomials satisfy $T_n'(1) = n^2$ for each $n \geq 0$. (*Markov's inequality* asserts that for any $p \in \mathcal{P}_n$, $\|p'\| \leq n^2\|p\|$, where $\|\cdot\|$ is the supremum norm.)

Exercise 3.10. Odd and even functions. Show that if f is an odd function on $[-1,1]$, its Chebyshev coefficients of even order are zero; show similarly that if f is even, its odd order coefficients are zero.

Exercise 3.11. A function neither even nor odd. Apply `chebpolyplot` to the chebfun for $f(x) = \exp(x)/(1+10000x^2)$. Why does the plot have the appearance of a stripe?

Exercise 3.12. Extrema and roots of Chebyshev polynomials. Give formulas for the extrema and roots of T_n in $[-1,1]$.

Exercise 3.13. Chebyshev coefficients and machine precision. By a command like `f = chebfun('exp(x)',np)`, one can force Chebfun to produce a chebfun of length `np` (i.e., degree `np`-1) rather than determine the length automatically. (a) Do this for the "function with wiggles" of this section with `np` $= 257$, and comment on how the `chebpolyplot` result differs from that shown in the text. (b) Likewise for the "function with two spikes" with `np` $= 2049$.

Exercise 3.14. Chebyshev series for a simple pole. (a) Let t be a complex number with $|t| < 1$ and define $F(z) = (z-t)^{-1} + (z^{-1}-t)^{-1}$. What is the Laurent series for F? (b) For the same t, show further that

$$1 + 2\sum_{k=1}^{\infty} t^k T_k(x) = \frac{1-t^2}{1-2tx+t^2}. \tag{3.15}$$

(This formula can be interpreted as a generating function for the Chebyshev polynomials.) (c) Let $a \notin [-1,1]$ be a real or complex number and let t be a real or complex number with $|t| < 1$ such that $(t+t^{-1})/2 = a$. Show that

$$\frac{1}{x-a} = \frac{2}{t-t^{-1}} \left[1 + 2\sum_{k=1}^{\infty} t^k T_k(x) \right]. \tag{3.16}$$

Exercise 3.15. Chebyshev series of e^{ax}. It can be shown that the Chebyshev series of e^{ax} is

$$e^{ax} = 2\sum_{k=0}^{\infty}{}' I_k(a) T_k(x), \tag{3.17}$$

where I_k is the modified Bessel function of the first kind and the prime indicates that the term $k = 0$ is to be multiplied by $1/2$. Derive the Chebyshev series for $\sinh(ax)$ and $\cosh(ax)$.

Chapter 4

Interpolants, projections, and aliasing

`ATAPformats`

Suppose $f(x)$ is a Lipschitz continuous function on $[-1,1]$ with Chebyshev series coefficients $\{a_k\}$ as in Theorem 3.1,

$$f(x)=\sum_{k=0}^{\infty} a_k T_k(x). \tag{4.1}$$

One approximation to f in $\mathcal{P}_n$ is the polynomial obtained by **interpolation** in Chebyshev points:

$$p_n(x)=\sum_{k=0}^{n} c_k T_k(x). \tag{4.2}$$

Another is the polynomial obtained by *truncation* or *projection* of the series to degree n, whose coefficients through degree n are the same as those of f itself:

$$f_n(x)=\sum_{k=0}^{n} a_k T_k(x). \tag{4.3}$$

The relationship of the Chebyshev coefficients of f_n to those of f is obvious, and in a moment we shall see that the Chebyshev coefficients of p_n have simple expressions too. In computational work generally, and in particular in Chebfun, the polynomials $\{p_n\}$ are usually almost as good approximations to f as the polynomials $\{f_n\}$, and easier to work with, since one does not need to evaluate the integral (3.12). The polynomials $\{f_n\}$, on the other hand, are also interesting. In this book, most of our computations will make use of $\{p_n\}$, but many of our theorems will treat both cases. A typical example is

Theorem 8.2, which asserts that if f is analytic on $[-1,1]$, then both $\|f-f_n\|$ and $\|f-p_n\|$ decrease geometrically to 0 as $n\to\infty$.

The key to understanding $\{c_k\}$ is the phenomenon of *aliasing,* a term that originated with radio engineers early in the 20th century. On the $(n+1)$ -point Chebyshev grid, it is obvious that any function f is indistinguishable from a polynomial of degree n. But something more is true: any Chebyshev polynomial T_N, no matter how big N is, is indistinguishable on the grid from a *single* Chebyshev polynomial T_m for some m with $0\le m\le n$. We state this as a theorem.

Theorem 4.1. Aliasing of Chebyshev polynomials. *For any $n\ge 1$ and $0\le m\le n$, the following Chebyshev polynomials take the same values on the $(n+1)$-point Chebyshev grid:*

$$T_m,\ T_{2n-m},\ T_{2n+m},\ T_{4n-m},\ T_{4n+m},\ T_{6n-m},\ldots.$$

Equivalently, for any $k\ge 0$, T_k takes the same value on the grid as T_m with

$$m=|(k+n-1)(\text{mod} 2n)-(n-1)|, \tag{4.4}$$

a number in the range $0\le m\le n$.

Proof. Recall from (2.1) and (3.8) that Chebyshev polynomials on $[-1,1]$ are related to monomials on the unit circle by $T_m(x)=(z^m+z^{-m})/2$, and Chebyshev points are related to $(2n)$th roots of unity by $x_m=(z_m+z_m^{-1})/2$. It follows that the first assertion of the theorem is equivalent to the statement that the following functions take the same values at the $(2n)$th roots of unity:

$$z^m+z^{-m},\ z^{2n-m}+z^{m-2n},\ z^{2n+m}+z^{-2n-m},\ldots.$$

Inspection of the exponents shows that in every case, modulo $2n$, we have one exponent equal to $+m$ and the other to $-m$. The conclusion now follows from the elementary phenomenon of aliasing of monomials on the unit circle: at the $(2n)$th roots of unity, $z^{2\nu n}=1$ for any integer ν.

For the second assertion (4.4), suppose first that $0\le k\,(\text{mod} 2n)\le n$. Then $n-1\le(k+n-1)(\text{mod} 2n)\le 2n-1$, so (4.4) reduces to $m=k\,(\text{mod} 2n)$, with $0\le m\le n$, and we have just shown that this implies that T_k and T_m take the same values on the grid. On the other hand, suppose that $n+1\le k\,(\text{mod} 2n)\le 2n-1$. Then $0\le(k+n-1)(\text{mod} 2n)\le n-2$, so the absolute value becomes a negation and (4.4) reduces to $m=-k\,(\text{mod} 2n)$, with $1 \leq m \leq n$. Again we have just shown that this implies that T_k and T_m take the same values on the grid. ▮

Here is a numerical illustration of Theorem 4.1. Taking $n = 4$, let `X` be the Chebyshev grid with $n+1$ points, and let $T\{1\}, \ldots, T\{10\}$ be the first ten Chebyshev polynomials:

```
n = 4; X = chebpts(n+1);
for k = 1:10, T{k} = chebpoly(k); end
```

Then T_3 and T_5 are the same on the grid:

```
disp([T{3}(X) T{5}(X)])
```

```
  -1.000000000000000  -1.000000000000000
   0.707106781186548   0.707106781186547
                   0                   0
  -0.707106781186548  -0.707106781186547
   1.000000000000000   1.000000000000000
```

So are T_1, T_7, and T_9:

```
disp([T{1}(X) T{7}(X) T{9}(X)])
```

```
  -1.000000000000000  -1.000000000000000  -1.000000000000000
  -0.707106781186547  -0.707106781186548  -0.707106781186547
                   0                   0                   0
   0.707106781186547   0.707106781186548   0.707106781186547
   1.000000000000000   1.000000000000000   1.000000000000000
```

As a corollary of Theorem 4.1, we can now derive the connection between $\{a_k\}$ and $\{c_k\}$. The following result can be found in [Clenshaw & Curtis 1960].

Theorem 4.2. Aliasing formula for Chebyshev coefficients. *Let f be Lipschitz continuous on $[-1, 1]$, and let p_n be its Chebyshev interpolant in $\mathcal{P}_n$, $n \geq 1$. Let $\{a_k\}$ and $\{c_k\}$ be the Chebyshev coefficients of f and p_n, respectively. Then*

$$c_0 = a_0 + a_{2n} + a_{4n} + \cdots, \tag{4.5}$$

$$c_n = a_n + a_{3n} + a_{5n} + \cdots, \tag{4.6}$$

and for $1 \le k \le n-1$,

$$c_k = a_k + (a_{k+2n} + a_{k+4n} + \cdots) + (a_{-k+2n} + a_{-k+4n} + \cdots). \tag{4.7}$$

Proof. By Theorem 3.1, f has a unique Chebyshev series (3.11), and it converges absolutely. Thus we can rearrange the terms of the series without affecting convergence, and in particular, each of the three series expansions written above converges since they correspond to the Chebyshev series (3.11) evaluated at $x = 1$, So the formulas (4.5)–(4.7) do indeed define certain numbers $c_0, \ldots, c_n$. Taking these numbers as coefficients multiplied by the corresponding Chebyshev polynomials $T_0, \ldots, T_n$ gives us a polynomial of degree n. By Theorem 4.1, this polynomial takes the same values as f at each point of the Chebyshev grid. Thus it is the unique interpolant $p_n \in \mathcal{P}_n$. ∎

We can summarize Theorem 4.2 as follows. On the $(n+1)$-point grid, any function f is indistinguishable from a polynomial of degree n. In particular, the Chebyshev series of the polynomial interpolant to f is obtained by reassigning all the Chebyshev coefficients in the infinite series for f to their aliases of degrees 0 through n.

As a corollary, Theorems 4.1 and 4.2 give us absolutely convergent series for $f - f_n$ and $f - p_n$, which we shall exploit in Chapters 7 and 8:

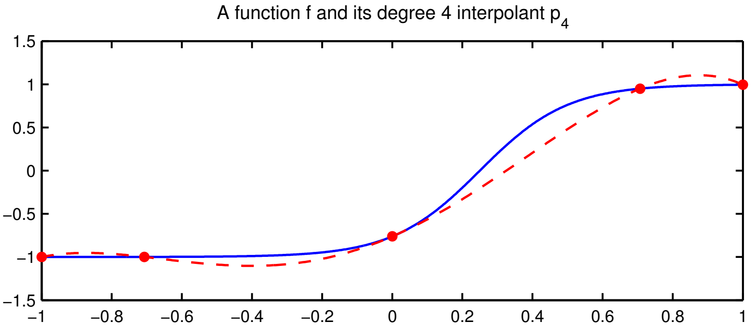
$$f(x) - f_n(x) = \sum_{k=n+1}^{\infty} a_k T_k(x), \tag{4.8}$$

$$f(x) - p_n(x) = \sum_{k=n+1}^{\infty} a_k (T_k(x) - T_m(x)), \tag{4.9}$$

where $m = m(k,n)$ is given by (4.4).

To illustrate Theorem 4.2, here is the function $f(x) = \tanh(4x-1)$ (solid) and its degree 4 Chebyshev interpolant $p_4(x)$ (dashed):

```
x = chebfun('x');
f = tanh(4*x-1);
n = 4; pn = chebfun(f,n+1);
hold off, plot(f), hold on, plot(pn,'.--r')
FS = 'fontsize';
title('A function f and its degree 4 interpolant p_4',FS,9)
```



The first 5 Chebyshev coefficients of f,

```
a = chebpoly(f); a = a(end:-1:1)'; a(1:n+1)
```

```
ans =
  -0.166584582703135
   1.193005991160944
   0.278438064117869
  -0.239362401056012
  -0.176961398392888
```

are different from the Chebyshev coefficients of p_n,

```
c = chebpoly(pn); c = c(end:-1:1)'
```

```
c =
  -0.203351068209675
   1.187719968517890
   0.379583465333916
  -0.190237989543227
  -0.178659622412174
```

As asserted in (4.5) and (4.6), the coefficients c_0 and c_n are given by sums of coefficients a_k with a stride of $2n$:

```
c0 = sum(a(1:2*n:end)), cn = sum(a(n+1:2*n:end))
```

```
c0 =
  -0.203351068209675
cn =
  -0.178659622412174
```

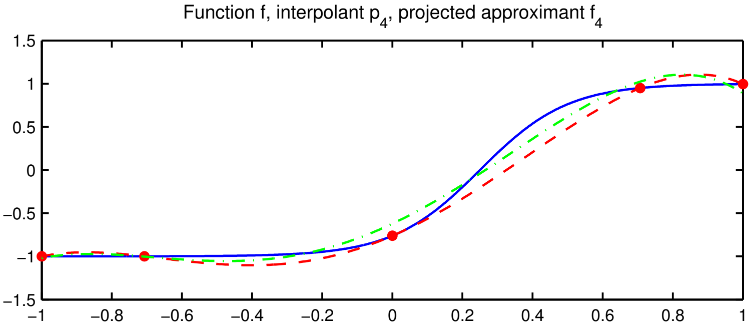
As asserted in (4.7), the coefficients c_1 through c_{n-1} involve two sums of this kind:

```
for k = 1:n-1
  ck = sum(a(1+k:2*n:end)) + sum(a(1-k+2*n:2*n:end))
end
```

```
ck =
   1.187719968517890
ck =
   0.379583465333916
ck =
  -0.190237989543227
```

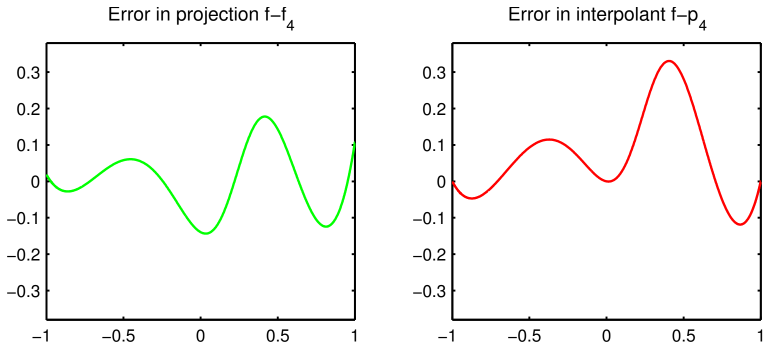
Following up on the last figure, how does the truncated series f_n compare with the interpolant p_n as an approximation to f? Chebfun includes a `'trunc'` option for computing f_n, which we now add to the plot as a dot-dash line:

```
fn = chebfun(f,'trunc',n+1);
plot(fn,'-.g')
title('Function f, interpolant p_4, projected approximant f_4',FS,9)
```



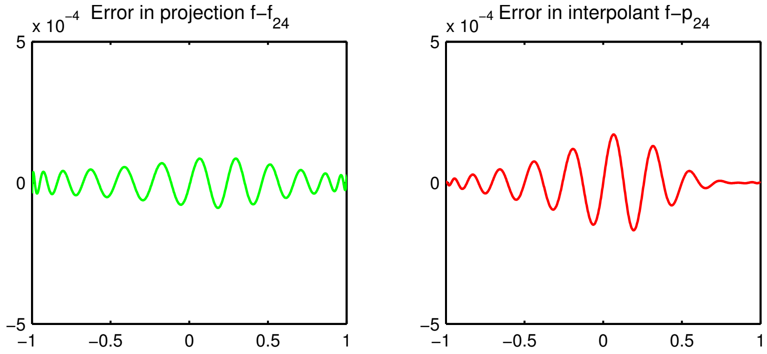
Here are the errors $f - f_n$ and $f - p_n$:

```
hold off
subplot(1,2,1), plot(f-fn,'g'), ylim(.38*[-1 1])
title('Error in projection f-f_4',FS,9)
subplot(1,2,2), plot(f-pn,'r'), ylim(.38*[-1 1])
title('Error in interpolant f-p_4',FS,9)
```



Here is the analogous plot with $n = 4$ increased to 24:

```
n = 24; pn = chebfun(f,n+1);
fn = chebfun(f,'trunc',n+1);
subplot(1,2,1), plot(f-fn,'g'), ylim(.0005*[-1 1])
title('Error in projection f-f_{24}',FS,9)
subplot(1,2,2), plot(f-pn,'r'), ylim(.0005*[-1 1])
title('Error in interpolant f-p_{24}',FS,9)
```



On the basis of plots like these, one might speculate that f_n may often be a better approximation than p_n, but that the difference is small. This is indeed the case, as we shall confirm in Theorems 7.2 and 8.2, both of which suggest a difference of a factor of 2, and Theorem 16.1, which suggests a factor of $\pi/2$.

Let us review where we stand. We have considered Chebyshev interpolants (Chapter 2) and Chebyshev expansions (Chapter 3) for a Lipschitz continuous function $f(x)$ defined on $[-1,1]$. Mathematically speaking, each coefficient of a Chebyshev expansion is equal to the value of the integral (3.12). This formula, however, is not needed for effective polynomial approximation, since Chebyshev interpolants are nearly as accurate as projections. Chebfun readily computes Chebyshev coefficients of polynomial interpolants, and this is done not by evaluating the integral but by taking the FFT of the sample values in Chebyshev points. If the degree of the interpolant is high enough that the polynomial matches f to machine precision, then the Chebyshev coefficients will match too.

> SUMMARY OF CHAPTER 4. *Two excellent methods of approximating a function f on $[-1,1]$ by a polynomial are truncation of its Chebyshev series, also known as projection, and interpolation in Chebyshev points. The Chebyshev interpolant is the polynomial obtained by reassigning contributions of degree $> n$ in the Chebyshev series to their aliases of degree $\le n$. The two approximations are typically within a factor of 2 of each other in accuracy.*

Exercise 4.1. Node polynomial for Chebyshev points. Show using Theorem 4.1 that $p(x) = 2^{-n}(T_{n+1}(x) - T_{n-1}(x))$ is the unique monic polynomial in $\mathcal{P}_{n+1}$ with zeros at the $n+1$ Chebyshev points (2.2).

Exercise 4.2. Examples of aliasing. (a) On the $(n+1)$-point Chebyshev grid with $n = 20$, which Chebyshev polynomials T_k take the same values as T_5? (b) Use Chebfun to draw plots illustrating some of these intersections.

Exercise 4.3. Aliasing in roots of unity. For each $n \ge 0$, let $p_n \in \mathcal{P}_n$ be the degree n polynomial interpolant to the function $f(z) = z^{-1}$ at the $(n+1)$st roots of unity on the unit circle in the z-plane. Use the aliasing observation of the proof of Theorem 4.1 to prove that in the closed unit disk of complex numbers z with $|z| \le 1$, there is one and only one value z for which p_n converges to f as $n \to \infty$. (This example comes from [Méray 1884].)

Exercise 4.4. Fooling the Chebfun constructor. (a) Construct the Matlab anonymous function `f = @(M) chebfun(@(x) 1+exp(-(M*(x-0.4)).^4))` and plot `f(10)` and `f(100)`. This function has a narrow spike of width proportional to $1/M$. Confirm this by comparing `sum(f(10))` and `sum(f(100))`. (b) Plot `length(f(M))` as a function of M for $M = 1, 2, 3, \ldots$, going into the region where the length becomes 1. What do you think is happening? (c) Let `Mmax` be the largest integer for which the constructor behaves normally and execute `semilogy(f(Mmax)-1,'interval',[.3 .5])`. Superimpose on this plot information to show the locations of the points returned by `chebpts(9)`, which is the default initial grid on which Chebfun samples a function. Explain how this result fits with (b). (d) Now for `np` taking values 17, 33, 65, 129, execute `chebfunpref('minsamples',np)` and `length(f(np))`, and plot the Chebyshev points on your semilog plot of (c). The `minsamples` flag forces Chebfun to sample the function at the indicated number of points. How do these results match your observations of (b) and (c)? When you're done, be sure to return Chebfun to its default state with `chebfunpref('factory')`.

Exercise 4.5. Relative precision. Try Exercise 4.4 again but without the "1+" in the definition of `f`. The value of `Mmax` will be different, and the reason has to do with Chebfun's aim of constructing each function to about 15 digits of relative precision, not absolute. Can you figure out what is happening and explain it quantitatively?

Exercise 4.6. Chebfun computation of truncations. In the text we computed Chebyshev truncations of $f(x) = \tanh(4x-1)$ using the `'trunc'` flag in the Chebfun constructor. Another method is to compute all the Chebyshev coefficients of f and then truncate the series. Compute f_4 by this method and verify that the results agree to machine precision.

Exercise 4.7. When projection equals interpolation. Sometimes the projection f_n and the interpolant p_n are identical, even though both differ from f. Characterize exactly when this occurs, and give an example with $n = 3$.

Chapter 5

Barycentric interpolation formula

`ATAPformats`

How does one evaluate a Chebyshev interpolant? One good approach, involving $O(n \log n)$ work for a single point evaluation, is to compute Chebyshev coefficients and use the Chebyshev series. However, there is a direct method requiring just $O(n)$ work, not based on the series expansion, that is both elegant and numerically stable. It also has the advantage of generalizing to sets of points other than Chebyshev. It is called the *barycentric interpolation formula,* introduced by Salzer [1972], with an earlier closely related formula by Marcel Riesz [1916]. The more general barycentric formula for arbitrary interpolation points, of which Salzer's formula is an exceptionally simple special case, was developed earlier by Dupuy [1948], with origins at least as early as Jacobi [1825]. Taylor [1945] introduced the barycentric formula for equispaced grid points. For a survey of barycentric formulas, see [Berrut & Trefethen 2004].

The study of polynomial interpolation goes back a long time; the word "interpolation" may be due to Wallis in 1656 (see [Pearson 1920] for an early account of some of the history.) In particular, Newton addressed the topic and devised a method based on divided differences. Many textbooks claim that it is important to use Newton's formulation for reasons of numerical stability, but this is not true, and we shall not discuss Newton's approach here.

Instead, the barycentric formula is of the alternative *Lagrange form,* where the interpolant is written as a linear combination of *Lagrange* or *cardinal* or *fundamental polynomials*:

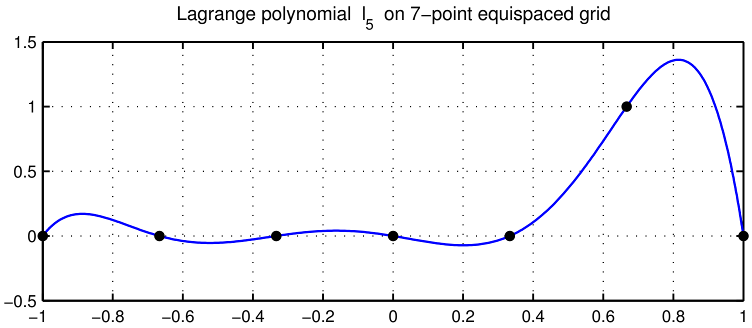
$$p(x) = \sum_{j=0}^{n} f_j \ell_j(x). \tag{5.1}$$

Here we have a set of distinct interpolation points $x_0, \dots, x_n$, which could be real or complex, and $\ell_j(x)$, the jth Lagrange polynomial, is the unique polynomial in $\mathcal{P}_n$ that takes the value 1 at x_j and 0 at the other points x_k:

$$\ell_j(x_k) = \begin{cases} 1 & k = j, \\ 0 & k \neq j. \end{cases} \tag{5.2}$$

For example, here is a plot of ℓ_5 on the equispaced 7-point grid (i.e., $n = 6$):

```
d = domain(-1,1); s = linspace(-1,1,7); y = [0 0 0 0 0 1 0];
p = interp1(s,y,d);
plot(p), hold on, plot(s,p(s),'.k'), grid on, FS = 'fontsize';
title('Lagrange polynomial  l_5  on 7-point equispaced grid',FS,9)
```



It is easy to write down an explicit expression for ℓ_j:

$$\ell_j(x) = \frac{\prod_{k\neq j}(x - x_k)}{\prod_{k\neq j}(x_j - x_k)}. \tag{5.3}$$

Since the denominator is a constant, this function is a polynomial of degree n with zeros at the right places, and clearly it takes the value 1 when $x = x_j$. Equation (5.3) is very well known and can be found in many textbooks as a standard representation for Lagrange interpolants. Lagrange worked with (5.1) and (5.3) in 1795 [Lagrange 1795], and his name is firmly attached to these ideas,[1] but the same formulas were published earlier by Waring

[1] Perhaps Cauchy did some of the attaching, since he wrote in his *Cours d'analyse,* "Cette formule, donnée pour la première fois par Lagrange, . . ." [Cauchy 1821].

[1779] and Euler [1783], who had been Lagrange's predecessor at the Berlin Academy.

Computationally speaking, (5.1) is excellent but (5.3) is not so good. It requires $O(n)$ operations to evaluate $\ell_j(x)$ for each value of x, and then $O(n)$ such evaluations must be added up in (5.1), giving a total operation count of $O(n^2)$ for evaluating $p(x)$ at a single value of x.

By a little rearrangement we can improve the operation count. The key observation is that for the various values of j, the numerators in (5.3) are the same except that they are missing different factors $x - x_j$. To take advantage of this commonality, we define the *node polynomial* $\ell \in \mathcal{P}_{n+1}$ for the given grid by

$$\ell(x) = \prod_{k=0}^{n} (x - x_k). \tag{5.4}$$

Then (5.3) becomes the elementary but extremely important identity

$$\ell_j(x) = \frac{\ell(x)}{\ell'(x_j)(x - x_j)}. \tag{5.5}$$

(We shall use this equation to derive the Hermite integral formula in Chapter 11.) Equivalently, let us define

$$\lambda_j = \frac{1}{\prod_{k \neq j}(x_j - x_k)}, \tag{5.6}$$

that is,

$$\lambda_j = \frac{1}{\ell'(x_j)}. \tag{5.7}$$

Then (5.3) becomes

$$\ell_j(x) = \ell(x) \frac{\lambda_j}{x - x_j}, \tag{5.8}$$

and the Lagrange formula (5.1) becomes

$$p(x) = \ell(x) \sum_{j=0}^{n} \frac{\lambda_j}{x - x_j} f_j. \tag{5.9}$$

These formulas were derived by Jacobi in his PhD thesis in Berlin [Jacobi 1825], and they appeared in 19th century textbooks.[2]

[2] I am grateful to Folkmar Bornemann for drawing this history to my attention.

Equation (5.9) has been called the "modified Lagrange formula" (by Higham) and the "first form of the barycentric interpolation formula" or the "type 1 barycentric formula" (starting with Rutishauser). What is valuable here is that the dependence on x inside the sum is so simple. If the weights $\{\lambda_j\}$ are known, (5.9) produces each value $p(x)$ with just $O(n)$ operations. Computing the weights from (5.6) requires $O(n^2)$ operations, but this computation only needs to be done once and for all, independently of x; and for special grids $\{x_j\}$ such as Chebyshev, as we shall see in a moment, the weights are known analytically and don't need to be computed at all. (For Legendre and other grids associated with orthogonal polynomials, the necessary computations can be carried out very fast; see Exercise 5.11 and Theorem 19.6.)

However, there is another barycentric formula that is more elegant. If we add up all the Lagrange polynomials ℓ_j, we get a polynomial in $\mathcal{P}_n$ that takes the value 1 at every point of the grid. Since polynomial interpolants are unique, this must be the constant polynomial 1:

$$\sum_{j=0}^{n} \ell_j(x) = 1.$$

Dividing (5.8) by this expression enables us to cancel the factor $\ell(x)$, giving

$$\ell_j(x) = \frac{\lambda_j}{x - x_j} \bigg/ \sum_{k=0}^{n} \frac{\lambda_k}{x - x_k}. \tag{5.10}$$

By inserting these representations in (5.1), we get the "second form of the barycentric interpolation formula" or "true barycentric formula" for polynomial interpolation in an arbitrary set of $n+1$ points $\{x_j\}$.

Theorem 5.1. Barycentric interpolation formula. *The polynomial interpolant through data $\{f_j\}$ at $n+1$ points $\{x_j\}$ is given by*

$$p(x) = \sum_{j=0}^{n} \frac{\lambda_j f_j}{x - x_j} \bigg/ \sum_{j=0}^{n} \frac{\lambda_j}{x - x_j}, \tag{5.11}$$

with the special case $p(x) = f_j$ if $x = x_j$ for some j, where the weights $\{\lambda_j\}$ are defined by

$$\lambda_j = \frac{1}{\prod_{k \neq j}(x_j - x_k)}. \tag{5.12}$$

Proof. Given in the discussion above. ∎

It is obvious that the function defined by (5.11) interpolates the data. As x approaches one of the values x_j, one term in the numerator blows up and so does one term in the denominator. Their ratio is f_j, so this is clearly the value approached as x approaches x_j. On the other hand if x is equal to x_j, we can't use the formula: that would be a division of ∞ by ∞. This is why the theorem is stated with the qualification for the special case $x = x_j$.

What is not obvious is that the function defined by (5.11) is a polynomial, let alone a polynomial of degree n: it looks like a rational function. The fact that it is a polynomial depends on the special values (5.12) of the weights. For choices of nonzero weights that differ from (5.12), (5.11) will still interpolate the data, but in general it will be a rational function that is not a polynomial. These rational barycentric interpolants can be very useful in some applications, and they are likely to get more attention in the future [Berrut, Baltensperger & Mittelmann 2005, Tee & Trefethen 2006, Floater & Hormann 2007, Berrut, Floater & Klein 2011].

Chebfun's overload of the Matlab `interp1` command, which was illustrated at the beginning of this chapter, incorporates an implementation of (5.11)–(5.12). We shall make use of `interp1` again in Exercise 5.7 and in Chapters 13 and 15. Now, however, let us turn to the special case that is so important in practice.

For Chebyshev points, the weights $\{\lambda_j\}$ are wonderfully simple: they are equal to $(-1)^j$ times the constant $2^{n-1}/n$, or half this value for $j = 0$ and n. These numbers were worked out by Marcel Riesz in 1916 [Riesz 1916]. The constant cancels in the numerator and denominator when we divide by the formula for 1 in (5.11), giving Salzer's amazingly simple result from 1972 [Salzer 1972]:

Theorem 5.2. Barycentric interpolation in Chebyshev points. *The polynomial interpolant through data $\{f_j\}$ at the Chebyshev points* (2.2) *is*

$$p(x) = {\sum_{j=0}^{n}}' \frac{(-1)^j f_j}{x - x_j} \bigg/ {\sum_{j=0}^{n}}' \frac{(-1)^j}{x - x_j}, \qquad (5.13)$$

with the special case $p(x) = f_j$ if $x = x_j$. The primes on the summation signs signify that the terms $j = 0$ and $j = n$ are multiplied by $1/2$.

Equation (5.13) is scale-invariant: for interpolation in Chebyshev points scaled to any interval $[a, b]$, the formula is exactly the same. This is a big advantage on the computer when n is in the thousands or higher, because it means that we need not worry about underflow or overflow.

Proof. Equation (5.13) is a special case of (5.11). To prove it, we will show that for Chebyshev points, the weights (5.12) reduce to $(-1)^j$ times the constant $2^{n-1}/n$, and half this value for $j = 0$ or n. To do this, we begin by noting that for Chebyshev points, the node polynomial (5.4) can be written as $\ell(x) = 2^{-n}(T_{n+1}(x) - T_{n-1}(x))$ (Exercise 4.1). Together with (5.8), this implies

$$\ell_j(x) = 2^{-n}\lambda_j \frac{T_{n+1}(x) - T_{n-1}(x)}{x - x_j},$$

and from (5.7) we have

$$\lambda_j = \frac{1}{\ell'(x_j)} = \frac{2^n}{T'_{n+1}(x_j) - T'_{n-1}(x_j)}.$$

Now it can be shown that

$$T'_{n+1}(x_j) - T'_{n-1}(x_j) = 2n(-1)^j, \quad 1 \le j \le n-1,$$

with twice this value for $j = 0$ and n (Exercise 5.3). So we have

$$\lambda_j = \frac{2^{n-1}}{n}(-1)^j, \quad 1 \le j \le n-1, \tag{5.14}$$

with half this value for $j = 0$ and n, as claimed. ∎

The formula (5.13) is extraordinarily effective, even if n is in the thousands or millions, even if p must be evaluated at thousands or millions of points. As a first example, let us construct a rather wiggly chebfun:

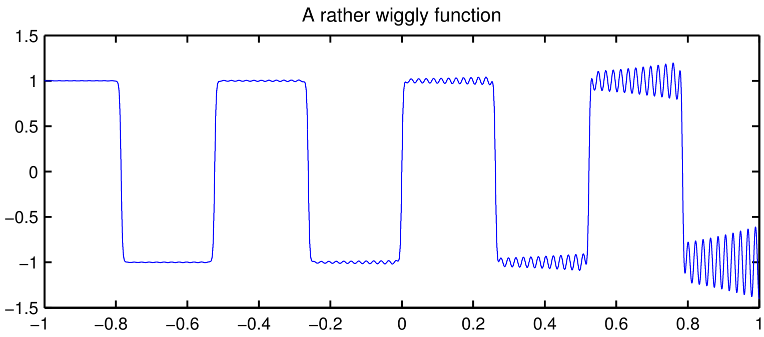
```
x = chebfun('x');
f = tanh(20*sin(12*x)) + .02*exp(3*x).*sin(300*x);
length(f)
```

```
ans =
        5282
```

We now plot `f` using 10000 sample points and note the time required:

```
hold off
tic, plot(f,'linewidth',.5,'numpts',10000), toc
title('A rather wiggly function',FS,9)
```

```
Elapsed time is 0.462332 seconds.
```



In this short time, Chebfun has evaluated a polynomial interpolant of degree about 5000 at 10000 sample points.

Raising the degree further, let p be the Chebyshev interpolant of degree 10^6 to the function $\sin(10^5 x)$ on $[-1,1]$:

```
ff = @(x) sin(1e5*x); p = chebfun(ff,1000001);
```

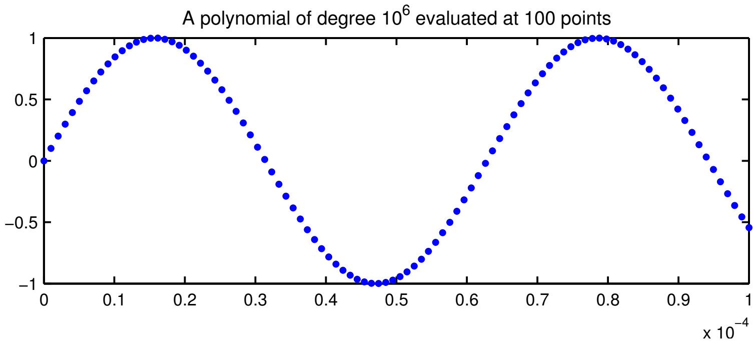
How long does it take to evaluate this interpolant at 100 points?

```
xx = linspace(0,0.0001); tic, pp = p(xx); toc
```

```
Elapsed time is 1.860470 seconds.
```

Not bad for a million-degree polynomial! The result looks fine,

```
clf, plot(xx,pp,'.','markersize',10), axis([0 0.0001 -1 1])
title('A polynomial of degree 10^6 evaluated at 100 points',FS,9)
```



and it matches the target function closely:

```
format long
for j = 1:5
    r = rand; disp([ff(r) p(r) ff(r)-p(r)])
end
```

```
  -0.512936261701570  -0.512936261699343  -0.000000000002227
   0.846686492477507   0.846686492479246  -0.000000000001739
   0.999735648990453   0.999735648991609  -0.000000000001156
  -0.027783734249832  -0.027783734250681   0.000000000000849
  -0.174873098288638  -0.174873098285456  -0.000000000003182
```

The apparent loss of 4 or 5 digits of accuracy is to be expected since the derivative of this function is of order 10^5: each evaluation is the correct result for a value of x within about 10^{-16} of the correct one (Exercise 5.5).

Experiments like these show that barycentric interpolation in Chebyshev points is a robust process: it is numerically stable, untroubled by rounding errors on a computer. This may seem surprising if you look at (5.9) or (5.13)—shouldn't cancellation errors on a computer cause trouble if x is close to one of the Chebyshev points x_j? In fact they do not, and these formulas have been proved stable in floating point arithmetic for all $x \in [-1, 1]$ [Rack & Reimer 1982, Higham 2004]. This is in marked contrast to the more familiar algorithm of polynomial interpolation via solution of a Vandermonde linear system of equations, which is exponentially unstable (Exercise 5.2).

We must emphasize that whereas (5.13) is stable for interpolation, it is unstable for extrapolation, that is, the evaluation of $p(x)$ for $x \notin [-1, 1]$. The more general formula (5.11) is unstable for extrapolation too and is unstable even for interpolation when used with arbitrary points rather than points suitably clustered like Chebyshev points. In these cases it is important to use the "type 1" barycentric formula (5.9) instead, which Higham proved stable in all cases. The disadvantage of (5.9) is that when n is larger than about a thousand, it is susceptible to troubles of underflow or overflow, which must be countered by rescaling $[-1, 1]$ to $[-2, 2]$ or by computing products by addition of logarithms.

More precisely, Higham [2004] showed that when they are used to evaluate $p(x)$ for $x \in [-1, 1]$ with data at Chebyshev points, both (5.9) and (5.11)–(5.13) have a certain property that numerical analysts call *forward stability.* If you want to evaluate $p(x)$ for values of x outside $[-1, 1]$, however, (5.11)–(5.13) lose their stability and it is important to use (5.9), which has the stronger property known as *backward stability* [Webb, Trefethen & Gonnet 2011]. It is also important to use (5.9) rather than (5.11) for computing interpolants through equispaced points or other point sets that are far from the Chebyshev distribution. (As we shall discuss in Chapters 13–14, in these cases the problem is probably so ill-conditioned that one should not be doing polynomial interpolation in the first place.)

These observations show that (5.9) has advantages over (5.11) and (5.13), but it also has an important disadvantage: it is not scale-invariant, and the weights grow exponentially as functions of the inverse of the length of the interval of interpolation. We see this in (5.14), where the weights have size 2^n, and would in fact overflow on a computer in standard IEEE double precision arithmetic for n bigger than about 1000. (Higham's analysis ignores overflow and underflow.) We shall have more to say about this exponential dependence in Chapters 11–15. So (5.11) and (5.13) remain a good choice for most applications, so long as the interpolation points are Chebyshev or similar and the evaluation points lie in $[-1, 1]$.

Summary of Chapter 5. *Polynomial interpolants can be evaluated fast and stably by the barycentric formula, even for thousands or millions of interpolation points. The barycentric formula has the form of a rational function, but reduces to a polynomial because of the use of specially determined weights.*

Exercise 5.1. Barycentric coefficients by hand. (a) Work out on paper the barycentric interpolation coefficients $\{\lambda_j\}$ for the case $n = 3$ and $x_0 = -1$, $x_1 = 0$, $x_2 = 1/2$, $x_3 = 1$. (b) Confirm that (5.9) gives the right value $p(-1/2)$ for the polynomial interpolant to data $1, 2, 3, 4$ in these points.

Exercise 5.2. Instability of Vandermonde interpolation. The best-known numerical algorithm for polynomial interpolation, unlike the barycentric formula, is unstable. This is the method implemented in the Matlab `polyfit` command, which forms a Vandermonde matrix of sampled powers of x and solves a corresponding linear system of equations. (In [Trefethen 2000], to my embarrassment, this unstable method is used throughout, forcing the values of n used for plots in that book to be kept small.) (a) Explore this instability by comparing a Chebfun evaluation of $p(0)$ with the result of `polyval(polyfit(xx,f(xx),n),0)` where `f = @(x) cos(k*x)` for $k = 10, 20, \dots, 90, 100$, n is the degree of the corresponding chebfun, and `xx` is a fine grid. (b) Examining the Matlab `polyfit` code as appropriate, construct the Vandermonde matrices V for each of these ten problems and compute their condition numbers. (You can also use the Matlab `vander` command.) By contrast, the underlying Chebyshev interpolation problem is well-conditioned.

Exercise 5.3. Calculating derivatives for the proof of Theorem 5.2. Derive the following identities used in the proof of Theorem 5.2. (a) For $1 \le j \le n-1$, $T'_{n+1}(x_j) - T'_{n-1}(x_j) = 2n(-1)^j$. (b) For $j = 0$ and $j = n$, $T'_{n+1}(x_j) - T'_{n-1}(x_j) = 4n(-1)^j$. One can derive this formula directly, or indirectly by a symmetry argument.

Exercise 5.4. Interpolating the sign function. Use `x = chebfun('x')`, `f = sign(x)` to construct the sign function on $[-1,1]$ and `p = chebfun('sign(x)',10000)` to construct its interpolant in 10000 Chebyshev points. Explore the difference in the interesting region by defining `d = f-p`, `d = d{-0.002,0.002}`. What is the maximum value of `p`? In what subset of $[-1,1]$ is `p` smaller than 0.5 in absolute value?

Exercise 5.5. Accuracy of point evaluations. (a) Construct the chebfun g corresponding to $f(x) = \sin(\exp(10x))$ on $[-1,1]$. What is the degree of this polynomial? (b) Let `xx` be the vector of 1000 linearly spaced points from -1 to 1. How long does it take on your computer to evaluate $f($`xx`$)$? $g($`xx`$)$? (c) Draw a loglog plot of the vector of errors $|f(\texttt{xx}) - g(\texttt{xx})|$ against the vector of derivatives $|f'(\texttt{xx})|$. Comment on why the dots line up as they do.

Exercise 5.6. Equispaced points. Show that for equispaced points in $[-1,1]$ with spacing h, the barycentric weights are $\lambda_j = (-1)^{n-j}/(j!(n-j)!h^n)$, or after canceling common factors, $\lambda_j = (-1)^j\binom{n}{j}$ [Taylor 1945].

Exercise 5.7. A greedy algorithm for choosing interpolation grids. Write a program using Chebfun's `interp1` command to compute a sequence of polynomial interpolants to a function f on $[-1,1]$ in points selected by a greedy algorithm: take x_0 to be a point where $|f(x)|$ achieves its maximum, then x_1 to be a point where $|(f-p_0)(x)|$ achieves its maximum, then x_2 to be a point where $|(f-p_1)(x)|$ achieves its maximum, and so on. Plot the error curves $(f-p_n)(x)$, $x \in [-1,1]$ computed by this algorithm for $f(x) = |x|$ and $0 \le n \le 25$. Comment on the spacing of the grid $\{x_0, \ldots, x_{25}\}$.

Exercise 5.8. Barycentric formula for Chebyshev polynomials. Derive an elegant formula for $T_n(x)$ from (5.13) [Salzer 1972].

Exercise 5.9. Barycentric interpolation in roots of unity. Derive the barycentric weights $\{\lambda_j\}$ for polynomial interpolation in (a) $\{\pm 1\}$, (b) $\{1, i, -1, -i\}$, (c) The $(n+1)$st roots of unity for arbitrary $n \ge 0$.

Exercise 5.10. Barycentric weights for a general interval. (a) How does the formula (5.14) for Chebyshev barycentric weights on $[-1,1]$ change for weights on an interval $[a,b]$? (b) The *capacity* of $[a,b]$ (see Chapter 12) is equal to $c = (b-a)/4$. How do the barycentric weights behave as $n \to \infty$ for an interval of capacity c? As a function of c, what is the maximal value of n for which they can be represented in IEEE double precision arithmetic without overflow or underflow? (You may assume the overflow and underflow limits are 10^{308} and 10^{-308}. The overflow/underflow problem goes away with the use of the divided form (5.13).)

Exercise 5.11. Barycentric interpolation in Legendre points. Chebfun includes fast algorithms for computing barycentric weights for various distributions of points other than Chebyshev, such as Legendre points, the zeros of Legendre polynomials (see Chapter 17 and Theorem 19.6). Perform a numerical experiment to compare the accuracy of interpolants in Chebyshev and Legendre points to $f(x) = e^x \sin(300x)$ at $x = 0.99$. Specifically, compute `[s,w,lambda] = legpts(n+1)` and `bary(0.99,f(s),s,lambda)` for $1 \le n \le 500$ and make a semilog plot of the absolute value of the error as a function of n; compare this with the analogous plot for Chebyshev points.

Exercise 5.12. Barycentric rational interpolation. (a) If the formula (5.13) is used with points $\{x_j\}$ other than Chebyshev with maximum spacing h, it produces a rational interpolant of accuracy $O(h^2)$ as $h \to 0$ [Berrut 1988]. Confirm this numerically for $f(x) = e^x$ and equispaced points in $[-1, 1]$. (b) Show numerically that the accuracy improves to $O(h^3)$ if the pattern of coefficients near the left end is changed from $\frac{1}{2}, -1, 1, -1, \ldots$ to $\frac{1}{4}, -\frac{3}{4}, 1, -1, \ldots$ and analogously at the right end [Floater & Hormann 2007].

Exercise 5.13. Barycentric weights and geometric mean distances. (a) Give an interpretation of (5.6) in terms of geometric mean distances between grid points. (b) Explain how one of the theorems of this chapter explains the result of Exercise 2.6.

Chapter 6

Weierstrass Approximation Theorem

`ATAPformats`

Every continuous function on a bounded interval can be approximated to arbitrary accuracy by polynomials. This is the famous Weierstrass approximation theorem, proved by Karl Weierstrass when he was 70 years old [Weierstrass 1885]. The theorem was independently discovered at about the same time, in essence, by Carl Runge: as pointed out in 1886 by Phragmén in remarks published as a footnote stretching over four pages in a paper by Mittag-Leffler [1900], it can be derived as a corollary of results Runge published in a pair of papers in 1885 [Runge 1885a & 1885b].

Here and throughout this book, unless indicated otherwise, $\|\cdot\|$ denotes the supremum norm on $[-1,1]$.

Theorem 6.1. Weierstrass approximation theorem. *Let f be a continuous function on $[-1,1]$, and let $\varepsilon > 0$ be arbitrary. Then there exists a polynomial p such that*

$$\|f-p\| < \varepsilon.$$

Outline of proof. We shall not spell out an argument in detail. However, here is an outline of the beautiful proof from Weierstrass's original paper. First, extend $f(x)$ to a continuous function $\tilde{f}$ with compact support on the whole real line. Now, take $\tilde{f}$ as initial data at $t=0$ for the diffusion equation $\partial u/\partial t = \partial^2 u/\partial x^2$ on the real line. It is known that by convolving $\tilde{f}$ with the Gaussian kernel $\phi(x) = e^{-x^2/4t}/\sqrt{4\pi t}$, we get a solution to this partial differential equation that converges uniformly to f as $t \to 0$, and thus can be made arbitrarily close to f on $[-1,1]$ by taking t small enough. On the other hand, since $\tilde{f}$ has compact support, for each $t > 0$ this solution is an integral over a bounded interval of entire functions and is thus itself an entire function, that is, analytic throughout the complex plane. Therefore it has a uniformly convergent Taylor series on $[-1, 1]$, which can be truncated to give polynomial approximations of arbitrary accuracy. ∎

For a fuller presentation of the argument just given as "one of the most amusing applications of the Gaussian kernel," where the result is stated for the more general case of a function of several variables approximated by multivariate polynomials, see Chapter 4 of [Folland 1995].

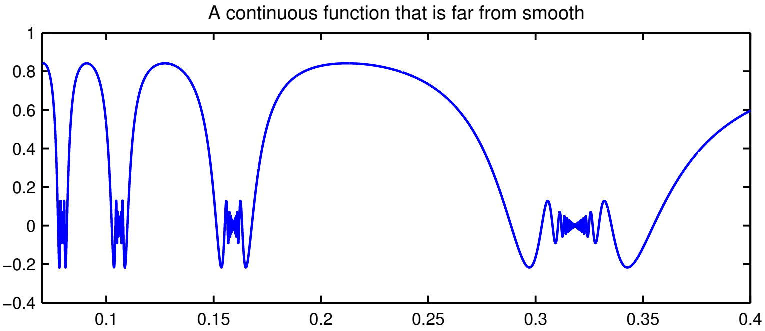
Many other proofs of the Weierstrass theorem are also known, including these early ones:

Runge (1885)
Picard (1891)
Lerch (1892 and 1903)
Volterra (1897)
Lebesgue (1898)
Mittag-Leffler (1900)
Fejér (1900 and 1916)
Landau (1908)
de la Vallée Poussin (1908)
Jackson (1911)
Sierpinski (1911)
Bernstein (1912)
Montel (1918)

For example, Bernstein's proof is a discrete analogue of the argument just given: continuous diffusion is replaced by a random walk made precise by the notion of *Bernstein polynomials* (Exercise 6.4) [Bernstein 1912D]. Lebesgue's proof, which appeared in his first paper published as a student at age 23, is based on reducing the approximation of general continuous functions to the approximation of $|x|$ (Exercise 6.5) [Lebesgue 1898]. Fejér was an even younger student, age 20, when he published his proof based on *Cesàro means* (Exercise 6.6a) [Fejér 1900], and he published a different proof years later based on *Hermite–Fejér interpolation* (Exercise 6.6b) [Fejér 1916]. This long list gives an idea of the great amount of mathematics stimulated by Weierstrass's theorem and the significant role it played in the development of analysis in the early 20th century. For a fascinating presentation of this corner of mathematical history, see [Pinkus 2000].

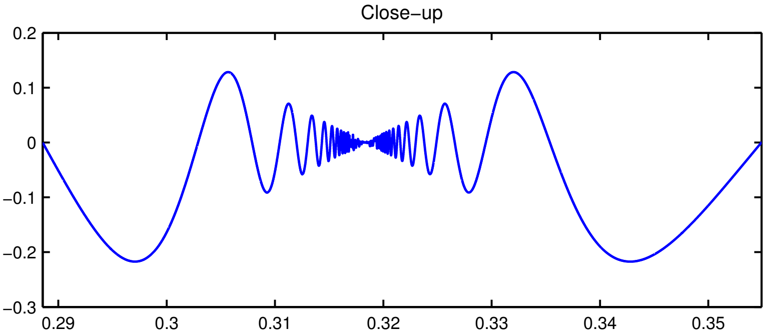
Weierstrass's theorem establishes that even extremely non-smooth functions can be approximated by polynomials, functions like $x\sin(x^{-1})$ or even $\sin(x^{-1})\sin(1/\sin(x^{-1}))$. The latter function has an infinite number of points near which it oscillates infinitely often, as we begin to see from the plot below over the range $[0.07, 0.4]$. In this calculation Chebfun is called with a user-prescribed number of interpolation points, 30,000, since the usual adaptive procedure has no chance of resolving the function to machine precision.

```
f = chebfun(@(x) sin(1./x).*sin(1./sin(1./x)),[.07 .4],30000);
plot(f), xlim([.07 .4]), FS = 'fontsize';
title('A continuous function that is far from smooth',FS,9)
```



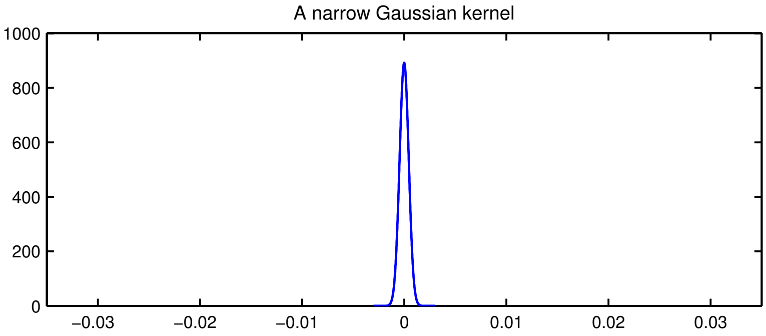
We can illustrate the idea of Weierstrass's proof by showing the convolution of this complicated function with a Gaussian. First, here is the same function f recomputed over a subinterval extending from one of its zeros to another:

```
a = 0.2885554757; b = 0.3549060246;
f2 = chebfun(@(x) sin(1./x).*sin(1./sin(1./x)),[a,b],2000);
plot(f2), xlim([a b]), title('Close-up',FS,9)
```



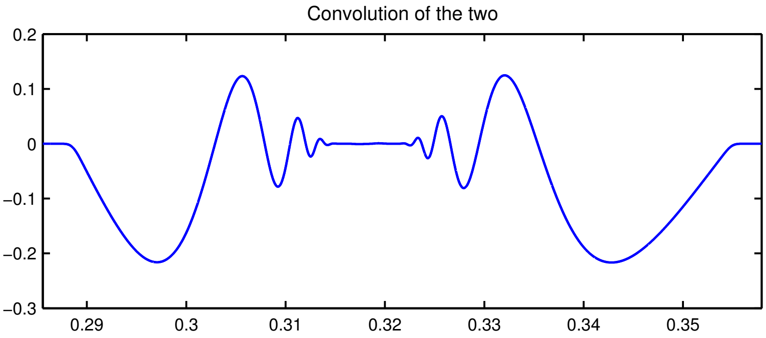
Here is a narrow Gaussian with integral 1.

```
t = 1e-7;
phi = chebfun(@(x) exp(-x.^2/(4*t))/sqrt(4*pi*t),.003*[-1 1]);
plot(phi), xlim(.035*[-1 1])
title('A narrow Gaussian kernel',FS,9)
```



Convolving the two gives a smoothed version of the close-up of f. Notice how the short wavelengths vanish while the long ones are nearly undisturbed.

```
f3 = conv(f2,phi);
plot(f3), xlim([a-.003,b+.003])
title('Convolution of the two',FS,9)
```



This is an entire function, which means it can be approximated by polynomials by truncating the Taylor series.

Weierstrass's theorem has an important generalization to complex analytic functions. Suppose a function f is defined on a compact set K in the complex plane whose complement is connected (so K cannot have any holes). *Mergelyan's theorem* asserts that if f is continuous on K and analytic in the interior, then f can be approximated on K by polynomials [Mergelyan 1951, Gaier 1987]. The earlier *Runge's theorem* is the weaker result in which f is asumed to be analytic throughout K, not just in the interior [Runge 1885A].

For all its beauty, power, and importance, the Weierstrass approximation theorem has in some respects served as an unfortunate distraction. Knowing that even troublesome functions can be approximated by polynomials, we naturally ask, how can we do it? A famous result of Faber and Bernstein asserts that there is no fixed array of grids of $1, 2, 3, \ldots$ interpolation points, Chebyshev or otherwise, that achieves convergence as $n \to \infty$ for all continuous f [Faber 1914, Bernstein 1919]. So it becomes tempting to look at approximation methods that go beyond interpolation, and to warn people that interpolation is dangerous, and to try to characterize exactly what minimal properties of f suffice to ensure that interpolation will work after all. A great deal is known about these subjects. The trouble with this line of research is that for almost all the functions encountered in practice, Chebyshev interpolation works beautifully! Weierstrass's theorem has encouraged mathematicians over the years to give too much of their attention to pathological functions at the edge of discontinuity, leading to the bizarre and unfortunate situation where many books on numerical analysis caution their readers that interpolation may fail without mentioning that for functions with a little bit of smoothness, it succeeds outstandingly. For a discussion of the history of such misrepresentations and misconceptions, see Chapter 14 and also the appendix on "Six myths of polynomial interpolation and quadrature."

SUMMARY OF CHAPTER 6. *A continuous function on a bounded interval can be approximated arbitrarily closely by polynomials.*

Exercise 6.1. A pathological function of Weierstrass. Weierstrass was one of the first to give an example of a function continuous but nowhere differentiable on $[-1, 1]$, and it is one of the early examples of a fractal [Weierstrass 1872]:

$$w(x) = \sum_{k=0}^{\infty} 2^{-k} \cos(3^k x).$$

(a) Construct a chebfun `w7` corresponding to this series truncated at $k = 7$. Plot `w7`, its derivative (use `diff`), and its indefinite integral (`cumsum`). What is the degree of the polynomial defining this chebfun? (b) Prove that w is continuous. (You can use the Weierstrass M-test.)

Exercise 6.2. Taylor series of an entire function. To illustrate the proof of the Weierstrass approximation theorem, we plotted a Gaussian kernel. The key point of the proof is that this kernel is entire, so its Taylor series converges for all x. (a) For $x = 1$ at the given time $t = 10^{-7}$, how many terms of the Taylor series about $x = 0$ would you have to take before the terms fall below 1? Estimate the answer at least to within a factor of 2. You may find Stirling's formula helpful. (b) Also for $x = 1$ and $t = 10^{-7}$, approximately how big is the biggest term in the Taylor series?

Exercise 6.3. Resolving a difficult function. Although the example function $f(x) = \sin(1/x)\sin(1/\sin(1/x))$ of this chapter is not Lipschitz continuous, its Chebyshev interpolants do in fact converge. Explore this phenomenon numerically by computing the degree n Chebyshev interpolant to f over the interval $[0.07, 0.4]$ for $n+1 = 4, 8, 16, \dots, 2^{14}$ and measuring the error in each case over a Chebyshev grid of $2n$ points. Plot the results on a loglog scale. How do you think the error depends on n as $n \to \infty$? Approximately how large would n have to be to get 16-digit accuracy for this function over this interval?

Exercise 6.4. Bernstein's proof. For $f \in C([0, 1])$, the associated degree n *Bernstein polynomial* is defined by

$$B_n(x) = \sum_{k=0}^{n} f(k/n)\binom{n}{k}x^k(1-x)^{n-k}. \tag{6.1}$$

Bernstein proved the Weierstrass approximation theorem by showing that $B_n(x) \to f(x)$ uniformly as $n \to \infty$. (a) Give an interpretation of $B_n(x)$ involving a random walk driven by a coin which comes up heads with probability x and tails with probability $1-x$. (b) Show that $\max B_n(x) \le \max f(x)$ and $\min B_n(x) \ge \min f(x)$ for $x \in [0, 1]$.

Exercise 6.5. Lebesgue's proof. (a) Show using uniform continuity that any $f \in C([-1, 1])$ can be approximated uniformly by a polygonal curve, i.e., a function $g(x)$ that is piecewise linear and continuous. (b) Show that such a function can be written in the form $g(x) = A + Bx + \sum_{k=1}^{m} C_k|x - x_k|$. (c) Show that $|x|$ can be uniformly approximated by polynomials on $[-1, 1]$ by truncating the binomial expansion

$$[1-(1-x^2)]^{1/2} = \sum_{k=0}^{\infty} \binom{\frac{1}{2}}{n}(x^2-1)^n.$$

You may use without proof the fact that these binomial coefficients are of size $O(n^{-3/2})$ as $n \to \infty$. (d) Explain how (a)–(c) combine to give a proof of the Weierstrass approximation theorem.

Exercise 6.6. Fejér's proofs. (a) In 1900 Fejér proved the Weierstrass approximation theorem via *Cesàro means.* In the Chebyshev case, define S_n to be the mean of the partial sums of the Chebyshev series (3.11)–(3.12) of orders 0 through n. Then $S_n \to f$ uniformly as $n \to \infty$ for any $f \in C([-1, 1])$. Explore such approximations for $f(x) = e^x$ with various degrees n. For this very smooth function f, how does the accuracy compare with that of ordinary Chebyshev interpolants? (b) In 1916 Fejér proved the theorem again by considering what are now known as *Hermite–Fejér interpolants*: he showed that if $p_{2n} \in \mathcal{P}_{2n}$ is obtained by interpolating $f \in C([-1, 1])$ in the zeros of $T_n(x)$ and also setting $p'(x) = 0$

at these points, then $p_{2n} \to f$ uniformly as $n \to \infty$. Explore such interpolants numerically for various n by using `interp1` to construct polynomials p_{2n} with $p_{2n}(x_j) = p_{2n}(x_j + 10^{-6}) = \exp(x_j)$. Again how does the accuracy compare with that of ordinary Chebyshev interpolants?

Exercise 6.7. Convergent series of polynomials. (a) Show that any $f \in C([-1,1])$ can be written as a uniformly convergent series

$$f(x) = \sum_{k=0}^{\infty} q_k(x),$$

where each q_k is a polynomial of some degree. (b) Show that a series of the same kind also exists for a function continuous on the whole real line, with pointwise convergence for all x and uniform convergence on any bounded subset.

Chapter 7

Convergence for differentiable functions

`ATAPformats`

The principle mentioned at the end of the last chapter might be regarded as the central dogma of approximation theory: the smoother a function, the faster its approximants converge as $n \to \infty$. Connections of this kind were explored in the early years of the 20th century by three of the founders of approximation theory: Charles de la Vallée Poussin (1866–1962), a mathematician at Leuven in Belgium, Sergei Bernstein (1880–1968), a Ukrainian mathematician who had studied with Hilbert in Göttingen, and Dunham Jackson (1888–1946), an American student of Landau's also at Göttingen. (Henri Lebesgue in France (1875–1941) also proved some of the early results. For remarks on the history see [Goncharov 2000] and [Steffens 2006].) Bernstein made the following comment concerning best approximation errors $E_n(f) = \|f - p_n^*\|_\infty$ (see Chapter 10) in his summary article for the International Congress of Mathematicians in 1912 [Bernstein 1912a]:

The general fact that emerges from this study is the existence of a most intimate connection between the differential properties of the function $f(x)$ and the asymptotic rate of decrease of the positive numbers $E_n[f(x)]$.[1]

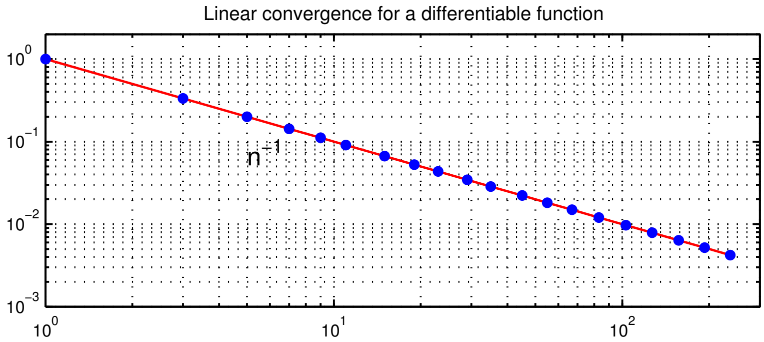
In this and the next chapter our aim is to make the smoothness–approximability link precise in the context of Chebyshev projections and interpolants. Everything here is analogous to results for Fourier analysis of

[1] "Le fait général qui se dégage de cette étude est l'existence d'une liaison des plus intimes entre les propriétés différentielles de la fonction $f(x)$ et la loi asymptotique de la décroissance des nombres positifs $E_n[f(x)]$."

periodic functions, and indeed, the whole theory of Chebyshev interpolation can be regarded as a transplant to nonperiodic functions on $[-1,1]$ of the theory of trigonometric interpolation of periodic functions on $[-\pi,\pi]$.

Suppose a function f is ν times differentiable on $[-1,1]$, possibly with jumps in the νth derivative, and suppose you look at the convergence of its Chebyshev interpolants as $n \to \infty$, measuring the error in the ∞-norm. You will typically see convergence at the rate $O(n^{-\nu})$. We can explore this effect readily with Chebfun. For example, the function $f(x) = |x|$ is once differentiable with a jump in the first derivative at $x = 0$, and the convergence curve nicely matches n^{-1} (shown as a straight line). Actually the match is more than just nice in this case—it is exact, with p_n taking its maximal error at the value $p(0) = 1/n$ for odd n. (For even n the error is somewhat smaller.)

```
x = chebfun('x'); f = abs(x);
nn = 2*round(2.^(0:.3:7))-1;
ee = 0*nn;
for j = 1:length(nn)
    n = nn(j); fn = chebfun(f,n+1); ee(j) = norm(f-fn,inf);
end
hold off, loglog(nn,1./nn,'r'), FS = 'fontsize';
text(5,0.07,'n^{-1}',FS,12)
grid on, axis([1 300 1e-3 2])
hold on, loglog(nn,ee,'.')
title('Linear convergence for a differentiable function',FS,9)
```

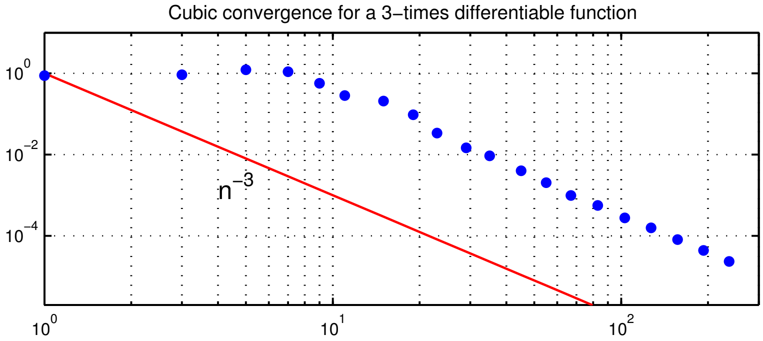


Similarly, we get cubic convergence for

$$f(x) = |\sin(5x)|^3, \tag{7.1}$$

which is three times differentiable with jumps in the third derivative at $x = 0$ and $\pm\pi/5$.

```
f = abs(sin(5*x)).^3;
for j = 1:length(nn)
    n = nn(j); fn = chebfun(f,n+1); ee(j) = norm(f-fn,inf);
end
hold off, loglog(nn,nn.^-3,’r’)
text(4,.0015,’n^{-3}’,FS,12)
grid on, axis([1 300 2e-6 10])
hold on, loglog(nn,ee,’.’)
title(’Cubic convergence for a 3-times differentiable function’,FS,9)
```



Encouraged by such experiments, you might look in a book to try to find theorems about $O(n^{-\nu})$. If you do, you'll run into two difficulties. First, it's hard to find theorems about Chebyshev interpolants, for most of the literature is about other approximations such as best approximations (see Chapters 10 and 16) or interpolants in Chebyshev polynomial roots rather than extrema. Second, you will probably fall one power of n short! In particular, the most commonly quoted of the *Jackson theorems* asserts that if f is ν times *continuously* differentiable on $[-1,1]$, then its best polynomial approximations converge at the rate $O(n^{-\nu})$ [Jackson 1911; Cheney 1966, sec. 4.6]. But the first and third derivatives of the functions we just looked at, respectively, are not continuous. Thus we must settle for the zeroth and second derivatives, respectively, if we insist on continuity, so this theorem would ensure only $O(n^0)$ and $O(n^{-2})$ convergence, not the $O(n^{-1})$ and $O(n^{-3})$ that are actually observed. And it would apply to best approximations, not Chebyshev interpolants.

We can get the result we want by recognizing that most functions encountered in applications have a property that is not assumed in the theorems just mentioned: *bounded variation.* A function, whether continuous or not, has bounded variation if its total variation is finite. The *total variation* is the 1-norm of the derivative (as defined if necessary in the distributional sense;

see [Ziemer 1989, chap. 5] or [Evans & Gariepy 1991, sec. 5.10]). We can compute this number conveniently with Chebfun by writing a function called `tv` that evaluates $\|f'\|_1$ for a given function f:

```
tv = @(f) norm(diff(f),1);
```

Here are the total variations of x and $\sin(10\pi x)$ over $[-1, 1]$:

```
disp([tv(x) tv(sin(10*pi*x))])
```

```
   2.000000000000000  40.000000000000007
```

Here is the total variation of the derivative of $|x|$:

```
tv(diff(abs(x)))
```

```
ans =
     2
```

Here is the total variation of the third derivative of the function f of (7.1):

```
tv(diff(f,3))
```

```
ans =
     2.102783663375361e+04
```

It is the finiteness of this number that allowed the Chebyshev interpolants to this function f to converge as fast as $O(n^{-3})$.

To get to a precise theorem, we begin with a bound on Chebyshev coefficients, an improvement (in the definition of the quantity V) of a similar result in [Trefethen 2008] whose proof was provided by Endre Süli. The condition of *absolute continuity* is a standard one which we shall not make detailed use of, so we will not discuss. An absolutely continuous function is equal to the integral of its derivative, which exists almost everywhere and is Lebesgue integrable.

Theorem 7.1. Chebyshev coefficients of differentiable functions. *For an integer $\nu \geq 0$, let f and its derivatives through $f^{(\nu-1)}$ be absolutely* *continuous on* $[-1,1]$ *and suppose the* ν*th derivative* $f^{(\nu)}$ *is of bounded variation* V*. Then for* $k \geq \nu+1$*, the Chebyshev coefficients of* f *satisfy*

$$|a_k| \leq \frac{2V}{\pi k(k-1)\cdots(k-\nu)} \leq \frac{2V}{\pi(k-\nu)^{\nu+1}}. \tag{7.2}$$

Proof. As in the proof of Theorem 3.1, setting $x = \frac{1}{2}(z+z^{-1})$ with z on the unit circle gives

$$a_k = \frac{1}{\pi i}\int_{|z|=1} f(\tfrac{1}{2}(z+z^{-1}))\, z^{k-1}\, dz,$$

and integrating by parts with respect to z converts this to

$$a_k = \frac{-1}{\pi i}\int_{|z|=1} f'(\tfrac{1}{2}(z+z^{-1}))\, \frac{z^k}{k}\, \frac{dx}{dz}\, dz\,; \tag{7.3}$$

the factor dx/dz appears since f' denotes the derivative with respect to x rather than z. Suppose now $\nu = 0$, so that all we are assuming about f is that it is of bounded variation $V = \|f'\|_1$. Then we note that this integral over the upper half of the unit circle is equivalent to an integral in x; the integral over the lower half gives another such integral. Combining the two gives

$$a_k = \frac{1}{\pi i}\int_{-1}^{1} f'(x)\, \frac{z^k - z^{-k}}{k}\, dx = \frac{2}{\pi}\int_{-1}^{1} f'(x)\, \mathrm{Im}\, \frac{z^k}{k}\, dx,$$

and since $|z^k/k| \leq 1/k$ for $x \in [-1,1]$ and $V = \|f'\|_1$, this implies $|a_k| \leq 2V/\pi k$, as claimed.

If $\nu > 0$, we replace dx/dz by $\frac{1}{2}(1-z^{-2})$ in (7.3), obtaining

$$a_k = -\frac{1}{\pi i}\int_{|z|=1} f'(\tfrac{1}{2}(z+z^{-1}))\left[\frac{z^k}{2k} - \frac{z^{k-2}}{2k}\right] dz.$$

Integrating by parts again with respect to z converts this to

$$a_k = \frac{1}{\pi i}\int_{|z|=1} f''(\tfrac{1}{2}(z+z^{-1}))\left[\frac{z^{k+1}}{2k(k+1)} - \frac{z^{k-1}}{2k(k-1)}\right] \frac{dx}{dz}\, dz.$$

Suppose now $\nu = 1$ so that we are assuming f' has bounded variation $V = \|f''\|_1$. Then again this integral is equivalent to an integral in x,

$$a_k = \frac{-2}{\pi}\int_{-1}^{1} f''(x)\, \mathrm{Im}\left[\frac{z^{k+1}}{2k(k+1)} - \frac{z^{k-1}}{2k(k-1)}\right] dx.$$

Since the term in square brackets is bounded by $1/k(k-1)$ for $x \in [-1,1]$ and $V = \|f''\|_1$, this implies $|a_k| \leq 2V/\pi k(k-1)$, as claimed.

If $\nu > 1$, we continue in this fashion with a total of $\nu + 1$ integrations by parts with respect to z, in each case first replacing dx/dz by $\frac{1}{2}(1 - z^{-2})$. At the next step the term that appears in square brackets is

$$\left[\frac{z^{k+2}}{4k(k+1)(k+2)} - \frac{z^k}{4k^2(k+1)} - \frac{z^k}{4k^2(k-1)} + \frac{z^{k-2}}{4k(k-1)(k-2)}\right],$$

which is bounded by $1/k(k-1)(k-2)$ for $x \in [-1, 1]$. And so on. ▮

From Theorems 3.1 and 7.1 we can derive consequences about the accuracy of Chebyshev projections and interpolants. Variations on the estimate (7.5) can be found as Corollary 2 of [Mastroianni & Szabados 1995] and Theorem 2 of [Mastroianni & Russo 2010]. The analogous result for best approximations as opposed to Chebyshev interpolants or projections was announced in [Bernstein 1911] and proved in [Bernstein 1912b].

Theorem 7.2. Convergence for differentiable functions. *If f satisfies the conditions of Theorem 7.1, with V again denoting the total variation of $f^{(\nu)}$ for some $\nu \ge 1$, then for any $n > \nu$, its Chebyshev projections satisfy*

$$\|f - f_n\| \le \frac{2V}{\pi\nu(n-\nu)^\nu} \tag{7.4}$$

and its Chebyshev interpolants satisfy

$$\|f - p_n\| \le \frac{4V}{\pi\nu(n-\nu)^\nu}. \tag{7.5}$$

Proof. For (7.4), Theorem 7.1 applied to equation (4.8) gives us

$$\|f - f_n\| \le \sum_{k=n+1}^{\infty} |a_k| \le \frac{2V}{\pi} \sum_{k=n+1}^{\infty} (k-\nu)^{-\nu-1}$$

and this sum can in turn be bounded by

$$\int_n^\infty (s-\nu)^{-\nu-1} ds = \frac{1}{\nu(n-\nu)^\nu}.$$

For (7.5), we use (4.9) instead of (4.8) and get the same bound except with coefficients $2|a_k|$ rather than $|a_k|$. ▮

In a nutshell: a νth derivative of bounded variation implies convergence at the algebraic rate $O(n^{-\nu})$. Here is a way to remember this message. Suppose we try to approximate the step function $\operatorname{sign}(x)$ by polynomials. There is no hope of convergence, since polynomials are continuous and $\text{sign}(x)$ is not, so all we can achieve is accuracy $O(1)$ as $n \to \infty$. That's the case $\nu = 0$. But now, each time we make the function "one derivative smoother," ν increases by 1 and so does the order of convergence.

How sharp is Theorem 7.2 for our example functions? In the case of $f(x) = |x|$, with $\nu = 1$ and $V = 2$, it predicts $\|f - f_n\| \le 4/\pi(n-1)$ and $\|f - p_n\| \le 8/\pi(n-1) \approx 2.55/(n-1)$. As mentioned above, the actual value for Chebyshev interpolation is $\|f - p_n\| = 1/n$ for odd n. The minimal possible error in polynomial approximation, with p_n replaced by the best approximation p_n^* (Chapter 10), is $\|f - p_n^*\| \sim 0.280169\ldots n^{-1}$ as $n \to \infty$ [Varga & Carpenter 1985]. So we see that the range from best approximant, to Chebyshev interpolant, to bound on Chebyshev interpolant is less than a factor of 10. The approximation of $|x|$ was a central problem studied by Lebesgue, de la Vallée Poussin, Bernstein, and Jackson a century ago, and we shall consider it further in Chapter 25.

The results are similar for the other example, $f(x) = |\sin(5x)|^3$, whose third derivative, we saw, has variation $V \approx 16528$. Equation (7.5) implies that the Chebyshev interpolants satisfy $\|f - p_n\| < 7020/(n-1)^3$, whereas in fact, we have $\|f - p_n\| \approx 309/n^3$ for large odd n and $\|f - p_n^*\| \approx 80/n^3$.

We close with a comment about Theorem 7.2. We have assumed in this theorem that $f^{(\nu)}$ is of bounded variation. A similar but weaker condition would be that $f^{(\nu-1)}$ is Lipschitz continuous (Exercise 7.2). This weaker assumption is enough to ensure $\|f - p_n^*\| = O(n^{-\nu})$ for the best approximations $\{p_n^*\}$; this is one of the Jackson theorems. On the other hand it is not enough to ensure $O(n^{-\nu})$ convergence of Chebyshev projections and interpolants. The reason we emphasize the stronger implication with the stronger conclusion is that in practice, one rarely deals with a function that is Lipschitz continuous while lacking a derivative of bounded variation, whereas one constantly deals with projections and interpolants rather than best approximations.

Incidentally, it was de la Vallée Poussin [1908] who first showed that the strong hypothesis is enough to reach the weak conclusion: if $f^{(\nu)}$ is of bounded variation, then $\|f - p_n^*\| = O(n^{-\nu})$ for the best approximation p_n^*. Three years later Jackson [1911] sharpened the result by weakening the hypothesis as just indicated.

Summary of Chapter 7. *The smoother a function f defined on $[-1,1]$ is, the faster its approximants converge. In particular, if the νth derivative of f is of bounded variation V, then the Chebyshev coefficients $\{a_k\}$ of f satisfy $|a_k| \le 2\pi^{-1}V(k-\nu)^{-\nu-1}$. For $\nu \ge 1$, it follows that the degree n Chebyshev projection and interpolant of f both have accuracy $O(Vn^{-\nu})$.*

Exercise 7.1. Total variation. (a) Determine numerically the total variation of $f(x) = \sin(100x)/(1+x^2)$ on $[-1,1]$. (b) It is no coincidence that the answer is close to 100, and indeed the total variation of $\sin(Mx)/(1+x^2)$ on $[-1,1]$ is asymptotic to M as $M \to \infty$. Explain why.

Exercise 7.2. Lipschitz continuous vs. derivative of bounded variation. (a) Prove that if the derivative f' of a function f has bounded variation, then f is Lipschitz continuous. (b) Give an example to show that the converse does not hold.

Exercise 7.3. Convergence for Weierstrass's function. Exercise 6.1 considered a "pathological function of Weierstrass" $w(x)$ that is continuous but nowhere differentiable on $[-1,1]$. (a) Make an anonymous function in Matlab that evaluates $w($`xx`$)$ for a vector `xx` to machine precision by taking the sum to 53 terms. (b) Use Chebfun to produce a plot of $\|w - p_n\|$ accurate enough and for high enough values of n to confirm that convergence appears to take place as $n \to \infty$. Thus w is not one of the functions for which interpolants fail to converge, a fact we can prove with the techniques of Chapter 15 (Exercise 15.9).

Exercise 7.4. Sharpness of Theorem 7.2. Consider the functions (a) $f(x) = |x|$, (b) $f(x) = |x|^5$, (c) $f(x) = \sin(100x)$. In each case plot, as functions of n, the error $\|f - p_n\|$ in Chebyshev interpolation on $[-1,1]$ and the bound on this quantity from (7.5). How close is the bound to the actuality? In cases (a) and (b) take ν as large as possible, and in case (c), take $\nu = 2$, 4, and 8.

Exercise 7.5. Total variation. Let f be a smooth function defined on $[0,1]$ and let $t(x)$ be its total variation over the interval $[0,x]$. What is the total variation of t over $[0,1]$?

Exercise 7.6. Chebyshev coefficients of a spline. A cubic spline is a piecewise cubic polynomial with two continuous derivatives. (a) How fast must the Chebyshev series coefficients of a cubic spline decay? (b) Test this prediction with the Chebfun commands `f=chebfun('exp(x)')`, `s=spline(linspace(-1,1,10),f)`, `p=chebfun(s)`, `chebpolyplot(p,'loglog')`.

Chapter 8

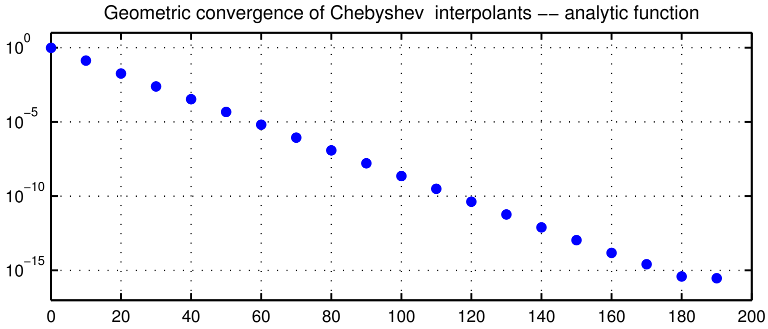
Convergence for analytic functions

```
ATAPformats
```

Suppose f is not just k times differentiable but infinitely differentiable and in fact analytic on $[-1,1]$. (Recall that this means that for any $s \in [-1,1]$, f has a Taylor series about s that converges to f in a neighborhood of s.) Then without any further assumptions we may conclude that the Chebyshev projections and interpolants converge **geometrically**, that is, at the rate $O(C^{-n})$ for some constant $C > 1$. This means the errors will look like straight lines (or better) on a semilog scale rather than a loglog scale. This kind of connection was first announced by Bernstein in 1911, who showed that the best approximations to a function f on $[-1,1]$ converge geometrically as $n \to \infty$ if and only if f is analytic [Bernstein 1911 & 1912B].

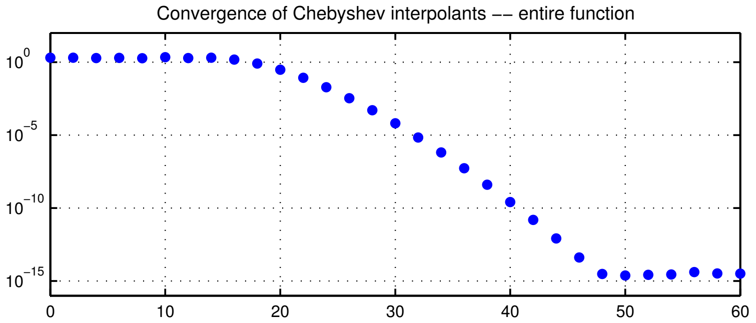
For example, for Chebyshev interpolants of the function $(1+25x^2)^{-1}$, known as the *Runge function* (Chapter 13), we get steady geometric convergence down to the level of rounding errors:

```
x = chebfun('x'); f = 1./(1+25*x.^2); nn = 0:10:200; ee = 0*nn;
for j = 1:length(nn)
    n = nn(j); fn = chebfun(f,n+1); ee(j) = norm(f-fn,inf);
end
hold off, semilogy(nn,ee,'.'), grid on, axis([0 200 1e-17 10])
FS = 'fontsize';
title(['Geometric convergence of Chebyshev ' ...
      ' interpolants -- analytic function'],FS,9)
```



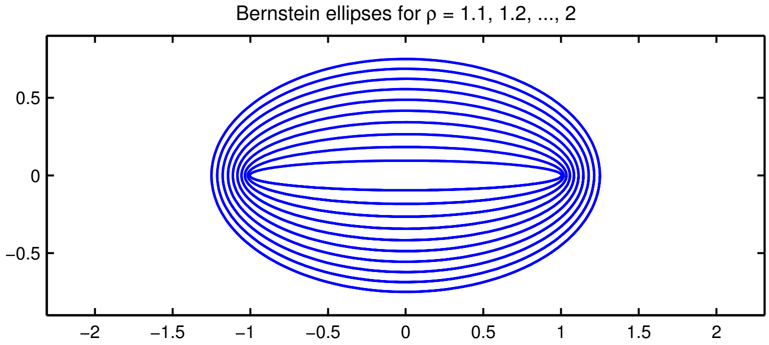
If f is analytic not just on $[-1, 1]$ but in the whole complex plane—such a function is said to be *entire*—then the convergence is even faster than geometric. Here, for the function $\cos(20x)$, the dots are not approaching a fixed straight line but a curve that gets steeper as n increases, until rounding errors cut off the progress.

```
f = cos(20*x); nn = 0:2:60; ee = 0*nn;
for j = 1:length(nn)
    n = nn(j); fn = chebfun(f,n+1); ee(j) = norm(f-fn,inf);
end
semilogy(nn,ee,'.'), grid on, axis([0 60 1e-16 100])
title('Convergence of Chebyshev interpolants -- entire function',FS,9)
```



There are elegant theorems that explain these effects. If f is analytic on $[-1, 1]$, then it can be analytically continued to a neighborhood of $[-1, 1]$ in the complex plane. (The idea of analytic continuation is explained in complex variables textbooks; see also Chapter 28.) The bigger the neighborhood, the faster the convergence. In particular, for polynomial approximations, the neighborhoods that matter are the regions in the complex plane bounded by ellipses with foci at -1 and 1, known as *Bernstein ellipses* [Bernstein 1912b, 1912c & 1914a]. It is easy to plot these curves: pick a number $\rho > 1$ and plot the image in the complex x-plane of the circle of radius ρ in the z-plane under the Joukowsky map $x = (z + z^{-1})/2$. We let E_ρ denote the open region bounded by this ellipse. Here, for example, are the Bernstein ellipses corresponding to the parameters $\rho = 1.1, 1.2, \ldots, 2$:

```
z = exp(2i*pi*x);
for rho = 1.1:0.1:2
    e = (rho*z+(rho*z).^(-1))/2; plot(e), hold on
end
ylim([-.9 .9]), axis equal
title('Bernstein ellipses for \rho = 1.1, 1.2, ..., 2',FS,9)
```



It is not hard to verify that the length of the semimajor axis of E_ρ plus the length of the semiminor axis is equal to ρ (Exercise 8.1).

Here is the basic bound on Chebyshev coefficients of analytic functions from which many other things follow. It first appeared in Section 61 of [Bernstein 1912b].

Theorem 8.1. Chebyshev coefficients of analytic functions. *Let a function f analytic in $[-1,1]$ be analytically continuable to the open Bernstein ellipse E_ρ, where it satisfies $|f(x)| \le M$ for some M. Then its Chebyshev coefficients satisfy $|a_0| \le M$ and*

$$|a_k| \le 2M\rho^{-k}, \quad k \ge 1. \tag{8.1}$$

Proof. As in the proofs of Theorems 3.1, 4.1, and 7.1, we make use of the transplantation from $f(x)$ and $T_k(x)$ on $[-1,1]$ in the x-plane to $F(z)$ and $(z^k + z^{-k})/2$ on the unit circle in the z-plane, with $x = (z + z^{-1})/2$ and $F(z) = F(z^{-1}) = f(x)$. The ellipse E_ρ in the x-plane corresponds under this formula in a 1-to-2 fashion to the annulus $\rho^{-1} < |z| < \rho$ in the z-plane.

By this we mean that for each x in $E_\rho \backslash [-1, 1]$ there are two corresponding values of z which are inverses of one another, and both the circles $|z| = \rho$ and $|z| = \rho^{-1}$ map onto the ellipse itself. (We can no longer use the formula $x = \operatorname{Re} z$, which is valid only for $|z| = 1$.) The first thing to note is that if f is analytic in the ellipse, then F is analytic in the annulus since it is the composition of the two analytic functions $z \mapsto (z + z^{-1})/2$ and $x \mapsto f(x)$. Now we make use of the contour integral formula (3.12),

$$a_k = \frac{1}{\pi i} \int_{|z|=1} z^{-1-k} F(z)\, dz,$$

with πi replaced by $2\pi i$ for $k = 0$. Suppose for a moment that F is analytic not just in the annulus but in its closure $\rho^{-1} \le |z| \le \rho$. Then we can expand the contour to $|z| = \rho$ without changing the value of the integral, giving

$$a_k = \frac{1}{\pi i} \int_{|z|=\rho} z^{-1-k} F(z)\, dz,$$

again with πi replaced by $2\pi i$ for $k = 0$. Since the circumference is $2\pi\rho$ and $|F(z)| \le M$, the required bound now follows from an elementary estimate. If F is analytic only in the open annulus, we can move the contour to $|z| = s$ for any $s < \rho$, leading to the same bound for any $s < \rho$ and hence also for $s = \rho$. ▮

Here are two of the consequences of Theorem 8.1. Equation (8.2) first appeared in [Bernstein 1912b, Sec. 61]. I do not know where equation (8.3) may have appeared, though similar slightly weaker bounds can be found in (4.13) and (4.16) of [Tadmor 1986]. For a generalization of (8.3) to interpolation in other point sets with the same asymptotic distribution as Chebyshev points, see Theorem 12.1.

Theorem 8.2. Convergence for analytic functions. *If f has the properties of Theorem 8.1, then for each $n \ge 0$ its Chebyshev projections satisfy*

$$\|f - f_n\| \le \frac{2M\rho^{-n}}{\rho - 1} \tag{8.2}$$

and its Chebyshev interpolants satisfy

$$\|f - p_n\| \le \frac{4M\rho^{-n}}{\rho - 1}. \tag{8.3}$$

Proof. Equation (8.2) follows from Theorem 8.1 and (4.8), and (8.3) follows from Theorem 8.1 and (4.9). ▮

We can apply Theorem 8.2 directly if f is analytic and bounded in E_ρ. If it is analytic but unbounded in E_ρ, then it will be analytic and bounded in E_s for any $s < \rho$, so we still get convergence at the rate $O(s^{-n})$ for any $s < \rho$. If it is unbounded in E_ρ but the only singularities on the ellipse are simple poles, then we get convergence at the rate $O(\rho^{-n})$ after all (Exercise 8.15).

Before applying Theorem 8.2 to a couple of examples, it will be convenient to note formulas for ρ in two common special cases. Suppose f has its first singularity at a real value $x_0 = \pm\alpha$ for some $\alpha > 1$. Then the corresponding ellipse parameter is

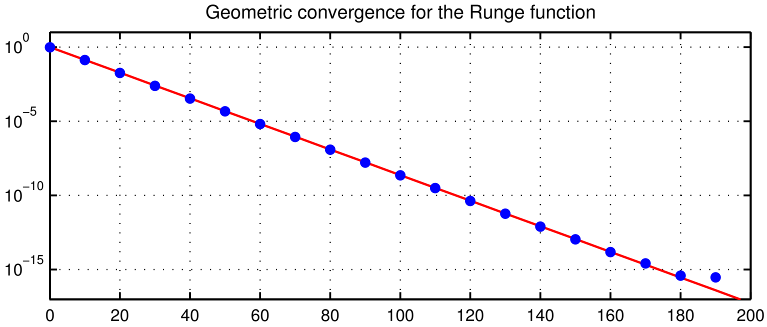
$$\rho = \alpha + \sqrt{\alpha^2 - 1} \quad \text{(real singularity at } x = \pm\alpha\text{)}. \tag{8.4}$$

Or suppose that the first singularity is at the pure imaginary value $x_0 = \pm i\beta$ for some $\beta > 0$. Then we have

$$\rho = \beta + \sqrt{\beta^2 + 1} \quad \text{(imaginary singularity at } x = \pm i\beta\text{)}. \tag{8.5}$$

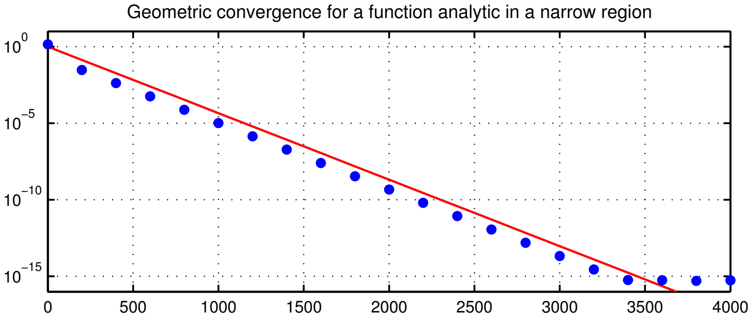
For example, the Runge function $(1 + 25x^2)^{-1}$ considered above has poles at $\pm i/5$. By (8.5), the corresponding value of ρ is $(1+\sqrt{26})/5 \approx 1.220$, and the errors in Chebyshev interpolation match this rate beautifully:

```
f = 1./(1+25*x.^2); nn = 0:10:200; ee = 0*nn;
for j = 1:length(nn)
    n = nn(j); fn = chebfun(f,n+1); ee(j) = norm(f-fn,inf);
end
rho = (1+sqrt(26))/5;
hold off, semilogy(nn,rho.^(-nn),'-r')
hold on, semilogy(nn,ee,'.'), grid on, axis([0 200 1e-17 10])
title('Geometric convergence for the Runge function',FS,9)
```



Here is a more extreme but entirely analogous example: $\tanh(50\pi x)$, with poles at $\pm 0.01i$. These poles are so close to $[-1, 1]$ that the convergence is much slower, but it is still robust. The only difference in this code segment is that `norm(f-fn,inf)`, a relatively slow Chebfun operation that depends on finding zeros of the derivative of `f-fn`, has been replaced by the default 2-norm `norm(f-fn)`, which is quick. This makes little difference to the figure, as the exponential decay rates are the same. (In the ∞-norm, the dots in the figure would appear just above the red line instead of just below it.)

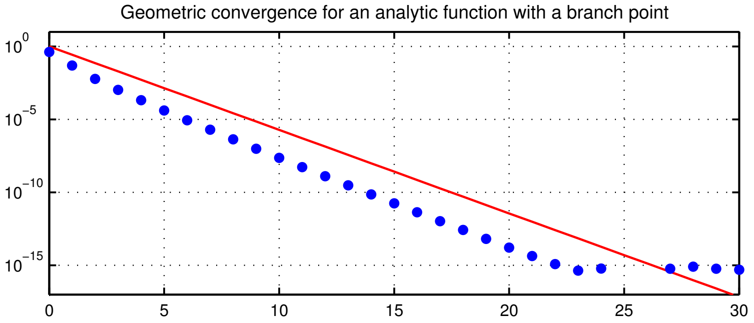
```
f = tanh(50*pi*x); nn = 0:200:4000; ee = 0*nn;
for j = 1:length(nn)
    n = nn(j); fn = chebfun(f,n+1); ee(j) = norm(f-fn);
end
rho = (1+sqrt(10001))/100;
hold off, semilogy(nn,rho.^(-nn),'-r')
hold on, semilogy(nn,ee,'.')
grid on, axis([0 4000 1e-16 10])
title(['Geometric convergence for a function ' ...
       'analytic in a narrow region'],FS,9)
```



For an example with a real singularity, the function $\sqrt{2-x}$ has a branch point at $x = 2$, corresponding by (8.4) to $\rho = 2 + \sqrt{3}$. Again we see a good match, with the curve gradually bending over to the expected slope as $n \to \infty$.

```
f = sqrt(2-x);
nn = 0:30; ee = 0*nn;
for j = 1:length(nn)
    n = nn(j); fn = chebfun(f,n+1); ee(j) = norm(f-fn,inf);
end
rho = 2+sqrt(3);
hold off, semilogy(nn,rho.^(-nn),'-r')
hold on, semilogy(nn,ee,'.')
grid on, axis([0 30 1e-17 10])
```

```
title(['Geometric convergence for an analytic ' ...
       'function with a branch point'],FS,9)
```



We now derive an elegant converse of Theorem 8.2, also due to Bernstein [1912b, Section 9]. The converse is not quite exact: Theorem 8.2 assumes analyticity and boundedness in E_ρ, whereas the conclusion of Theorem 8.3 is analyticity but not necessarily boundedness (Exercise 8.15).

Theorem 8.3. Converse of Theorem 8.2. *Suppose f is a function on $[-1,1]$ for which there exist polynomial approximations $\{q_n\}$ satisfying*

$$\|f - q_n\| \le C\rho^{-n}, \quad n \ge 0$$

for some constants $\rho > 1$ and $C > 0$. Then f can be analytically continued to an analytic function in the open Bernstein ellipse E_ρ.

Proof. The assumption implies that the polynomials $\{q_n\}$ satisfy $\|q_n - q_{n-1}\| \le 2C\rho^{1-n}$ on $[-1,1]$. Since $q_n - q_{n-1} \in \mathcal{P}_n$, it can be shown that this implies $\|q_n - q_{n-1}\|_{E_s} \le 2\,Cs^n\rho^{1-n}$ for any $s > 1$, where $\|\cdot\|_{E_s}$ is the supremum norm on the s-ellipse E_s. (This estimate is one of *Bernstein's inequalities,* from Section 9 of [Bernstein 1912b]; see Exercise 8.6.) For $s < \rho$, this gives us a representation for f in E_s as a series of analytic functions,

$$f = q_0 + (q_1 - q_0) + (q_2 - q_1) + \cdots,$$

which is uniformly convergent according to the Weierstrass M-test. According to another well-known theorem of Weierstrass, this implies that the limit is a bounded analytic function [Ahlfors 1953, Markushevich 1985]. Since this is true for any $s < \rho$, the analyticity applies throughout E_ρ. ∎

Note that Theorem 8.2 and 8.3 together establish a simple fact, sometimes known as *Bernstein's theorem*: a function defined on $[-1,1]$ can be approximated by polynomials with geometric accuracy if and only if it is analytic. (See also Exercise 8.11 and [Bagby & Levenberg 1993].)

The term “Bernstein ellipse” refers to any ellipse in the complex plane with foci $\{-1, 1\}$, and if f is a function analytic on $[-1, 1]$, the bounds of Theorems 8.1 and 8.2 apply for any Bernstein ellipse inside which f is analytic and bounded. If there is a largest ellipse inside which f is analytic, then one might choose to say that this was “the” Bernstein ellipse for f, but this might not always be the ellipse that gives the most useful bound, and if f is entire, then there is no largest ellipse at all (Exercise 8.3).

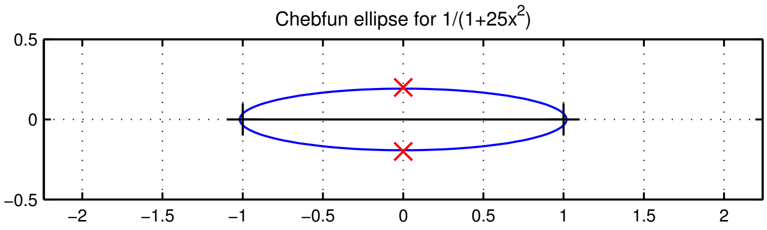
Chebfun computations, however, suggest a practical way to single out a special Bernstein ellipse associated with a given function f. The *Chebfun ellipse* for f is the Bernstein ellipse whose parameter ρ satisfies the condition

$$\rho^{-n} = \varepsilon, \tag{8.6}$$

where ε is the tolerance used by the Chebfun constructor (normally 2^{-52}) and n is the degree of the polynomial chosen by Chebfun to resolve f. The command `chebellipseplot` plots these Chebfun ellipses. Thus for $f(x) = 1/(1+25x^2)$, for example, the Chebfun ellipse comes very close to passing through the poles at $\pm 0.2i$:

```
f = chebfun('1./(1+25*x.^2)');
clf, chebellipseplot(f,'linewidth',1)
hold on, plot([.2i -.2i],'xr','markersize',12)
axis equal, ylim(.5*[-1 1]), grid on
title('Chebfun ellipse for 1/(1+25x^2)',FS,9)
```

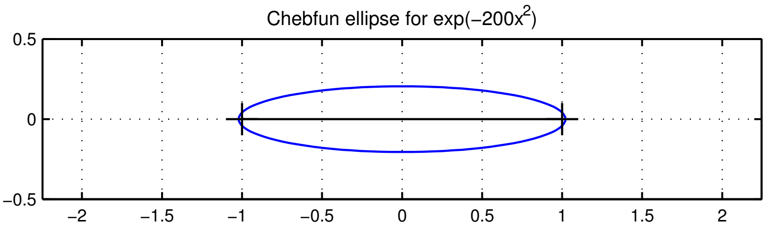
```
Warning: CHEBELLIPSEPLOT is deprecated.  Please use PLOTREGION instead.
```



For the entire function $f(x) = \exp(-200x^2)$, the Chebfun ellipse has much the same shape although now f has no singularities:

```
f = chebfun('exp(-200*x.^2)');
hold off, chebellipseplot(f,'linewidth',1)
axis equal, ylim(.5*[-1 1]), grid on
title('Chebfun ellipse for exp(-200x^2)',FS,9)
```

```
Warning: CHEBELLIPSEPLOT is deprecated.  Please use PLOTREGION instead.
```



Summary of Chapter 8. *If f is analytic, its Chebyshev coefficients $\{a_k\}$ decrease geometrically. In particular, if f is analytic with $|f(x)| \leq M$ in the Bernstein ρ-ellipse about $[-1,1]$, then $|a_k| \leq 2M\rho^{-k}$. It follows that the degree n Chebyshev projection and interpolant of f have accuracy $O(M\rho^{-n})$.*

Exercise 8.1. Bernstein ellipses. Verify that for any $\rho > 1$, the length of the semiminor axis plus the length of the semimajor axis of the Bernstein ellipse E_ρ is equal to ρ.

Exercise 8.2. A Chebyshev series. With `x = chebfun('x')`, execute the command `chebpolyplot(sin(100*(x-.1))+.01*tanh(20*x))`. Explain the various features of the resulting plot as quantitatively as you can.

Exercise 8.3. Interpolation of an entire function. The function $f(x) = \exp(-x^2)$ is analytic throughout the complex x-plane, so Theorem 8.2 can be applied for any value of the parameter $\rho > 1$. Produce a semilog plot of $\|f - p_n\|$ as a function of n together with lines corresponding to the upper bound of the theorem for $\rho = 1.1, 1.2, 1.4, 2, 3, 5, 8$. Be sure to use the right value of M in each case. How well do your bounds fit the data?

Exercise 8.4. Convergence rates for different functions. Based on the theorems of this chapter, what can you say about the convergence as $n \to \infty$ of the Chebyshev interpolants to (a) $\tan(x)$, (b) $\tanh(x)$, (c) $\log((x+3)/4)/(x-1)$, (d) $\int_{-1}^{x} \cos(t^2)\,dt$, (e) $\tan(\tan(x))$, (f) $(1+x)\log(1+x)$? In each case compare theoretical bounds with numerically computed results. Which is the case that converges much faster than the theorems predict? Can you speculate as to why?

Exercise 8.5. Accuracy of approximations in the complex plane. Let p be the chebfun for $f(x) = \exp(-200x^2)$ and plot contour lines in the complex x-plane corresponding to $|f(x) - p(x)| = 10^{-2}, 10^{-4}, \ldots, 10^{-14}$. How do these curves compare to the Bernstein ellipses corresponding to parameters ρ satisfying $\rho^{-n} = \varepsilon \times \{10^2, 10^4, \ldots, 10^{14}\}$, where ε is the Chebfun constructor tolerance 2^{-52}?

Exercise 8.6. Proof of Bernstein inequality. Prove Bernstein's inequality used in the proof of Theorem 8.3: if p is a polynomial of degree d, then $\|p\|_{E_\rho} \leq$

$\rho^d\|p\|$, where $\|\cdot\|_{E_\rho}$ is the ∞-norm over the ρ-ellipse and $\|\cdot\|$ is the ∞-norm over $[-1,1]$. (Hint: Show that if the branch cut is taken to be the unit interval $[-1,1]$, the function $q(z) = p(z)/(z+(z^2-1)^{1/2})^d$ is analytic throughout the region consisting of the complex plane plus the point $z=\infty$ minus $[-1,1]$. Apply the maximum modulus principle.)

Exercise 8.7. Absolute value function. The function $|x-i|$ is analytic for $x \in [-1,1]$. This means it can be analytically continued to an analytic function $f(x)$ in a neighborhood of $[-1,1]$ in the complex x-plane. The formula $|x-i|$ itself does not define an analytic function in any complex neighborhood. Find another formula for f that does, and use it to explain what singularities f has in the complex plane.

Exercise 8.8. Chebyshev polynomials on the Bernstein ellipse. Show that for any $\rho > 1$ and any z on the boundary of the ellipse E_ρ in the complex x-plane, $\lim_{n\to\infty} |T_n(x)|^{1/n} = \rho$.

Exercise 8.9. You can't judge smoothness by eye. Define $f(x) = 2+\sin(50x)$ and $g(x) = f(x)^{1.0001}$ and construct chebfuns for these functions on $[-1,1]$. What are their lengths? Explain this effect quantitatively using the theorems of this chapter.

Exercise 8.10. Convergence of conjugate gradient iteration. Suppose we wish to approximate $f(x) = x^{-1}$ on the interval $[m, M]$ with $0 < m < M$. Show that for any $\kappa < M/m$, there exist polynomials $p_n \in \mathcal{P}_n$ such that $\|f - p_n\| = O((1+2/\sqrt{\kappa})^{-n})$ as $n \to \infty$, where $\|\cdot\|$ is the ∞-norm on $[m, M]$. This result is famous in numerical linear algebra as providing an upper bound for the convergence of the conjugate gradient iteration applied to a symmetric positive definite system of equations $Ax = b$ with condition number κ. See Theorem 38.5 of [Trefethen & Bau 1997].

Exercise 8.11. Bernstein's theorem. Show that the conclusion of Theorem 8.3 also holds if the hypothesis is weakened to $\limsup_{n\to\infty} \|f - q_n\|^{1/n} \le \rho^{-1}$.

Exercise 8.12. Resolution power of Chebyshev interpolants. The function $f_M(x) = \exp(-M^2x^2/2)$ has a spike of width $O(1/M)$ at $x = 0$. Let $n(M)$ be the degree of a chebfun for f_M. (a) Determine the asymptotic behavior of $n(M)$ as $M \to \infty$ by numerical experiments. (b) Explain this result based on the theorems of this chapter.

Exercise 8.13. Resolution power of Bernstein polynomials. Continuing the last exercise, now let $n(M)$ be the degree of a Bernstein polynomial (6.1) needed to approximate f_M to machine precision. (For this discussion rescale (6.1) from $[0,1]$ to $[-1,1]$.) (a) Determine the asymptotic behavior of $n(M)$ as $M \to \infty$ by numerical experiments. (b) Explain this result, not necessarily rigorously.

Exercise 8.14. Formulas for ellipse parameter. Derive (8.4) and (8.5).

Exercise 8.15. Simple poles on the Bernstein ellipse. (a) Explain how equation (3.16) illustrates that Theorem 8.3 is not an exact converse of Theorem 8.2. (b) Let f be analytic in the open Bernstein ellipse region E_ρ for some $\rho > 1$

with the only singularities on the ellipse itself being simple poles. Show that $\|f - f_n\|$ and $\|f - p_n\|$ are of size $O(\rho^{-n})$ as $n \to \infty$.

Chapter 9

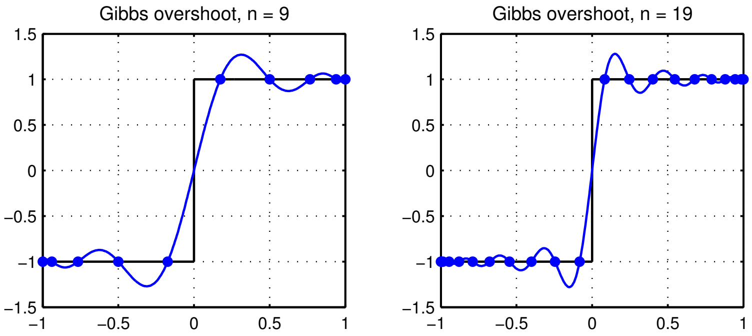
Gibbs phenomenon

```
ATAPformats
```

Polynomial interpolants and projections oscillate and overshoot near discontinuities. We have observed this *Gibbs phenomenon* already in Chapter 2, and now we shall look at it more carefully. We shall see that the Gibbs effect for interpolants can be regarded as a consequence of the oscillating inverse-linear tails of Lagrange polynomials, i.e., interpolants of Kronecker delta functions. Chapter 15 will show that these same tails, combined together in a different manner, are also the origin of Lebesgue constants of size $O(\log n)$, with implications throughout approximation theory.

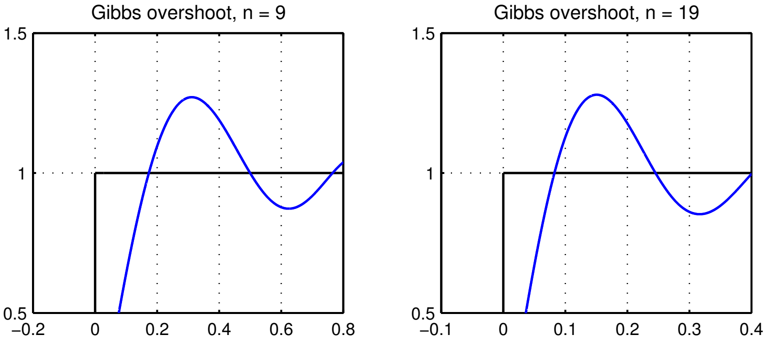
To start, let us consider the function `sign(x)`, which we interpolate in $n+1 =$ 10 and 20 Chebyshev points. We take n to be odd to avoid having a gridpoint at the middle of the step.

```
x = chebfun('x'); f = sign(x);
subplot(1,2,1), hold off, plot(f,'k','jumpline','-k'), hold on, grid on
f9 = chebfun(f,10); plot(f9,'.-'); FS = 'fontsize';
title('Gibbs overshoot, n = 9',FS,9)
subplot(1,2,2), hold off, plot(f,'k','jumpline','-k'), hold on, grid on
f19 = chebfun(f,20); plot(f19,'.-')
title('Gibbs overshoot, n = 19',FS,9)
```



Both of these figures show a substantial overshoot near the jump. As n increases from 9 to 19, the overshoot gets narrower, but not shorter, and it will not go away as $n \to \infty$. Let us zoom in and look at the plot on subintervals:

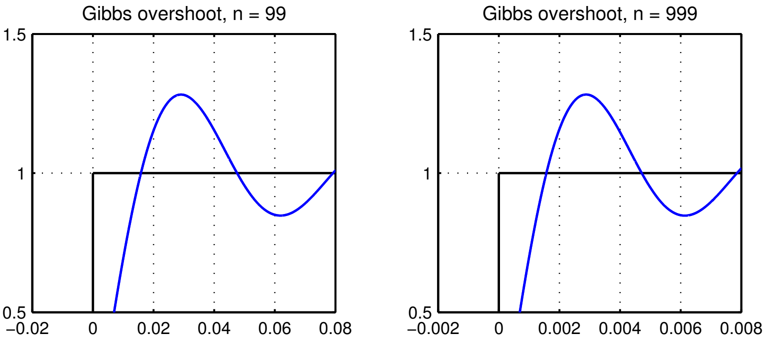
```
subplot(1,2,1), hold off, plot(f,’k’,’jumpline’,’-k’), hold on, grid on
plot(f9,’.-’,’interval’,[0 0.8]), axis([-.2 .8 .5 1.5])
title(’Gibbs overshoot, n = 9’,FS,9)
subplot(1,2,2), hold off, plot(f,’k’,’jumpline’,’-k’), hold on, grid on
plot(f19,’.-’,’interval’,[0 0.4]), axis([-.1 .4 .5 1.5])
title(’Gibbs overshoot, n = 19’,FS,9)
```



We now zoom in further with analogous plots for $n = 99$ and 999.

```
subplot(1,2,1), hold off, plot(f,’k’,’jumpline’,’-k’), hold on
f99 = chebfun(f,100); plot(f99,’.-’,’interval’,[0 0.08])
title(’Gibbs overshoot, n = 99’,FS,9)
grid on, axis([-.02 .08 .5 1.5])
subplot(1,2,2), hold off, plot(f,’k’,’jumpline’,’-k’), hold on
f999 = chebfun(f,1000); plot(f999,’.-’,’interval’,[0 0.008])
set(gca,’xtick’,-.002:.002:.01)
set(gca,’xticklabel’,{’-0.002’,’0’,’0.002’,’0.004’,’0.006’,’0.008’})
```

```
title('Gibbs overshoot, n = 999',FS,9)
grid on, axis([-.002 .008 .5 1.5])
```



Notice that in these figures, the vertical scale is always fixed while the horizontal scale is adjusted proportionally, confirming that the Gibbs overshoot gets narrower but approaches a constant height in the limit $n \to \infty$.

What is this height? We can measure it numerically with the `max` command:

```
disp('     n        Gibbs amplitude')
for n = 2.^(1:8)-1
   gibbs = max(chebfun(f,n+1));
   fprintf('%7d  %17.8f\n', n, gibbs)
end
```

```
     n        Gibbs amplitude
      1         1.00000000
      3         1.18807518
      7         1.26355125
     15         1.27816423
     31         1.28131717
     63         1.28204939
    127         1.28222585
    255         1.28226917
```

Clearly as $n \to \infty$, the maximum of the Chebyshev interpolant to the sign function converges to a number bigger than 1. The total variation of the interpolant, meanwhile, diverges slowly to ∞, at a rate proportional to $\log n$, and this is the effect we shall examine further in Chapter 15.

```
disp('     n          variation')
for n = 2.^(1:8)-1
   tv = norm(diff(chebfun(f,n+1)),1);
   fprintf('%7d  %14.2f\n', n, tv)
end
```

```
     n          variation
      1            2.00
      3            2.75
      7            3.64
     15            4.56
     31            5.47
     63            6.37
    127            7.26
    255            8.15
```

The following theorem summarizes the Gibbs phenomenon for Chebyshev interpolants. Well, perhaps it is a little bold to call it a "theorem", since it is not clear that a proof has ever been written down. The formulas necessary to represent the interpolant (in the equivalent trigonometric case—see Exercise 9.4) can be found in various forms in [Runck 1962] and [Helmberg & Wagner 1997], which relates the interpolating polynomial to the beta function and reports the numbers 1.282 and 1.066 to three digits of accuracy. The more precise results presented here have been privately communicated to me by Wagner based on calculations to more than 500 digits.

Theorem 9.1. Gibbs phenomenon for Chebyshev interpolants. *Let p_n be the degree n Chebyshev interpolant of the function $f(x) = \mathrm{sign}(x)$ on $[-1,1]$. Then as $n \to \infty$,*

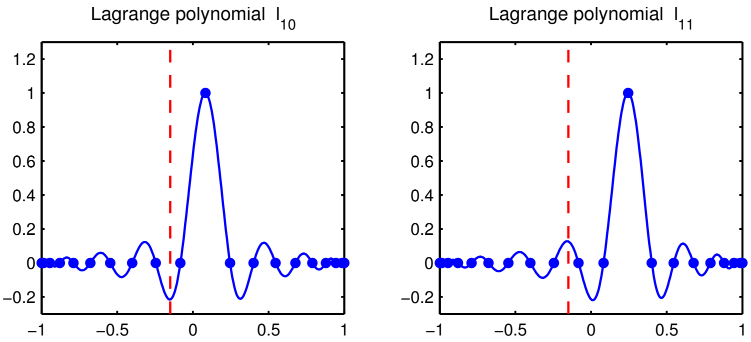
$$\lim_{n\to\infty,\, n\ odd} \|p_n\| \;=\; c_1 \;=\; 1.28228345577542854813\ldots, \tag{9.1}$$

$$\lim_{n\to\infty,\, n\ even} \|p_n\| \;=\; c_2 \;=\; 1.06578388826644809905\ldots. \tag{9.2}$$

(The case of n even differs in having a gridpoint at the middle of the jump.)

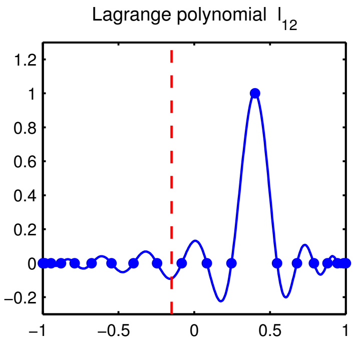
Although we are not going to prove Theorem 9.1, we do want to indicate where the fixed-overshoot effect comes from. Everything falls into place when we consider the Lagrange polynomials introduced in Chapter 5. Recall from (5.2) that the jth Lagrange polynomial $\ell_j(x)$ for the $(n+1)$-point Chebyshev grid is the unique polynomial in $\mathcal{P}_n$ that takes the values 1 at x_j and 0 at the other grid points x_k. On the 20-point grid, i.e. $n = 19$, here are the Lagrange polynomials ℓ_{10} and ℓ_{11} with a dashed line marked at $x = -0.15$, which we will take as our point of special interest.

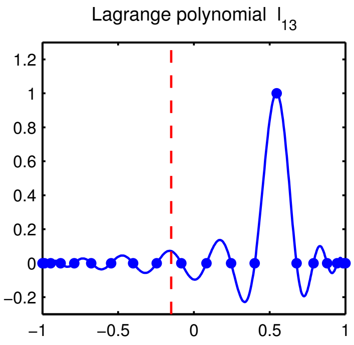
```
clf, yl = [-0.3 1.3];
xc = -0.15*[1 1];
p10 = chebfun([zeros(1,10) 1 zeros(1,9)]');
p11 = chebfun([zeros(1,11) 1 zeros(1,8)]');
subplot(1,2,1), plot(p10,'.-')
hold on, plot(xc,yl,'--r'), ylim(yl)
title('Lagrange polynomial  l_{10}',FS,9)
subplot(1,2,2), plot(p11,'.-')
hold on, plot(xc,yl,'--r'), ylim(yl)
title('Lagrange polynomial  l_{11}',FS,9)
```



Here are ℓ_{12} and ℓ_{13}:

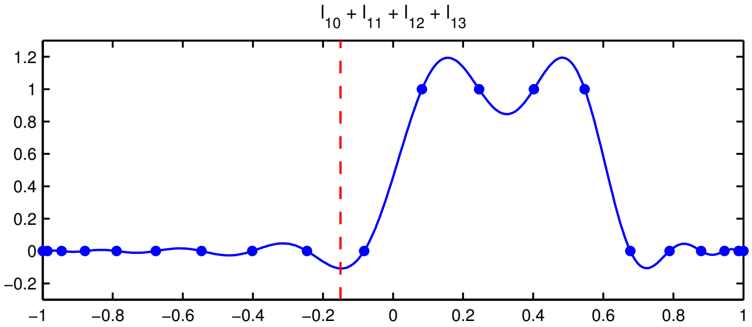
```
p12 = chebfun([zeros(1,12) 1 zeros(1,7)]');
p13 = chebfun([zeros(1,13) 1 zeros(1,6)]');
subplot(1,2,1), hold off, plot(p12,'.-')
hold on, plot(xc,yl,'--r'), ylim(yl)
title('Lagrange polynomial  l_{12}',FS,9)
subplot(1,2,2), hold off, plot(p13,'.-')
hold on, plot(xc,yl,'--r'), ylim(yl)
title('Lagrange polynomial  l_{13}',FS,9)
```





Following (5.1), we note that by taking the sum of a sequence of such Lagrange functions, we get the interpolant to the function that jumps from 0 for $x < 0$ to 1 for $x > 0$. Here is the sum of the four just plotted, which is beginning to look like a square wave:

```
clf, plot(p10+p11+p12+p13,'.-')
hold on, plot(xc,yl,'--r'), ylim(yl)
title('l_{10} + l_{11} + l_{12} + l_{13}',FS,9)
```



If we went all the way to the last grid point we would get the interpolant

$$p(x) = \sum_{j=(n+1)/2}^{n} \ell_j(x).$$

Note that for any fixed $x < x_{(n-1)/2}$, this is an *alternating series* of small terms whose amplitudes decrease inverse-linearly to zero. The finite but nonzero sum of such a series in the limit $n \to \infty$ is what gives rise to the fixed overshoot Gibbs effect in polynomial interpolation.

In particular, suppose we focus on the dashed line at $x = -0.15$ in the figures. Notice the alternating signs of the values of $\ell_{10}, \ell_{11}, \ell_{12}, \ell_{13}$ at this value of x. In the figure for $\ell_{10} + \ell_{11} + \ell_{12} + \ell_{13}$ we accordingly see the Gibbs overshoot beginning to converge to its asymptotic amplitude ≈ 0.141. This number is half of the value $0.282\ldots$ of Theorem 9.1, since the jump for this function is of amplitude 1 instead of 2.

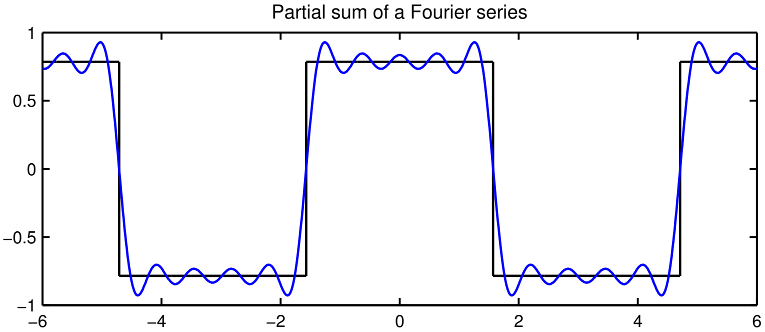
In Chapter 15 we shall consider the same alternating series but with signs multiplied by $(-1)^j$. This eliminates the alternation, so that we have approximately a harmonic series of inverse-linear terms. The partial sums of such a series grow at a logarithmic rate, as we saw above in the calculation of the variation.

Our discussion so far has concerned interpolants, but there is a parallel theory of the Gibbs phenomenon for projections—in the notation of this book, polynomials f_n rather than p_n. (The required Chebyshev coefficients are defined by the same integral (3.12) of Theorem 3.1, even though we are now dealing with functions f that are not Lipschitz continuous as in the assumption stated for that theorem.) As always, though the interpolants are closer to practical computation, the projections may appear to be more fundamental mathematically. Historically speaking, it was the case of Fourier (trigonometric) projection that was analyzed first. The original discoverer was not Gibbs but Henry Wilbraham, a 22-year-old fellow of Trinity College, Cambridge, in 1848, who unfortunately made the mistake of publishing his fine paper in the short-lived *Cambridge and Dublin Journal of Mathematics* [Wilbraham 1848]. Fourier series for certain functions with jumps were already long known in Wilbraham's day—in fact they go back to Euler, half a century before Fourier. The particular series studied by Wilbraham, originally due to Euler in 1772, is

$$\cos(t) - \frac{1}{3}\cos(3t) + \frac{1}{5}\cos(5t) - \cdots, \tag{9.3}$$

which approximates a square wave of height $\pm\pi/4$ (compare Exercise 3.6(a)):

```
t = chebfun('t',[-6,6]);
f = (pi/4)*sign(cos(t));
clf, plot(f,'k','jumpline','k')
f9 = cos(t) - cos(3*t)/3 + cos(5*t)/5 - cos(7*t)/7 + cos(9*t)/9;
hold on, plot(f9), xlim([-6 6])
title('Partial sum of a Fourier series',FS,9)
```



Wilbraham worked out the magnitude of the overshoot, and thus the following analogue of Theorem 9.1 is due to him.

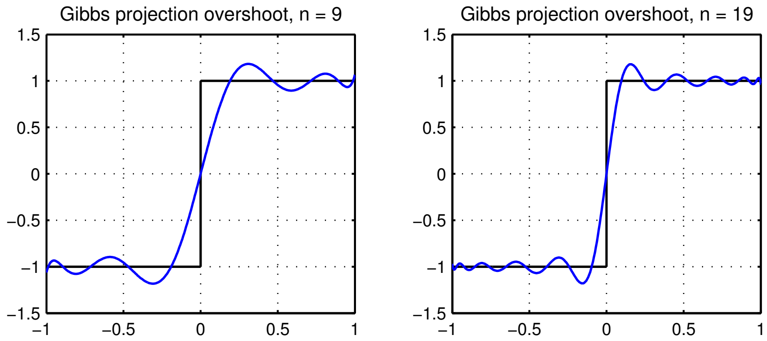
Theorem 9.2. Gibbs phenomenon for Chebyshev projections. *Let f_n be the degree n Chebyshev projection of the sign function $f(x) = \operatorname{sign}(x)$ on $[-1,1]$. Then as $n \to \infty$,*

$$\lim_{n\to\infty} \|f_n\| = \frac{2}{\pi}\int_0^{\pi} \frac{\sin x}{x}\, dx = 1.178979744472167\ldots. \qquad (9.4)$$

(The function $\operatorname{Si}(x) = \int_0^x t^{-1}\sin t\, dt$ is known as the *sine integral*; see Exercise 9.6.) To see this number experimentally we can use the `'trunc'` option in the Chebfun constructor. The overshoots look similar to what we saw before, but with smaller amplitude.

```
f = sign(x);
warnState = warning('off', 'CHEBFUN:constructor')
subplot(1,2,1), hold off, plot(f,'k','jumpline','k'), hold on, grid on
f9 = chebfun(f,'trunc',10); plot(f9,'-')
title('Gibbs projection overshoot, n = 9',FS,9)
subplot(1,2,2), hold off, plot(f,'k','jumpline','k'), hold on, grid on
f19 = chebfun(f,'trunc',20); plot(f19,'-')
title('Gibbs projection overshoot, n = 19',FS,9);
```

```
warnState =
    identifier: 'CHEBFUN:constructor'
         state: 'on'
```



The numbers behave as predicted:

```
disp('     n       Gibbs amplitude')
for np = 2.^(4:7)
  g = chebfun(f,'trunc',np);
  fprintf('%7d  %17.8f\n', np, max(g{0,5/np}))
end
limit = (2/pi)*sum(chebfun('sin(x)./x',[0 pi]))
warning(warnState)
```

```
     n       Gibbs amplitude
    16          1.18028413
    32          1.17930541
    64          1.17906113
   128          1.17900009
limit =
   1.178979744472167
```

In all the experiments of this chapter we have worked with polynomials rather than trigonometric series, but the effects are the same (Exercise 9.4).

It is worth commenting on a particular property of series such as (9.3) that we have taken for granted throughout this discussion: even though each partial sum is continuous, a series may converge pointwise to a discontinuous limit, everywhere except at the points of discontinuity themselves. This kind of behavior seems familiar enough nowadays, but in the century beginning with Fourier's work in 1807, it often seemed paradoxical and confusing to mathematicians. The same pointwise convergence to discontinuous functions can also occur with interpolants, as in Theorem 9.1.

In this chapter we have focussed on the height of the overshoot of a Gibbs oscillation, because this is the effect so readily seen in plots. Perhaps the most important property of Gibbs oscillations for practical applications, however, is not their height but their slow decay as one moves away from the point of discontinuity. If f has a jump, the oscillations at a distance k gridpoints away must be expected to be of size $O(k^{-1})$; if f' has a jump we expect oscillations of size $O(k^{-2})$, and so on. (Exercise 26.5 will look at the analogous exponents for interpolation by rational functions rather than polynomials.) This algebraic rate of decay of information in polynomial interpolants can be contrasted with the exponential decay that one gets with spline approximations, which is the key advantage of splines for certain applications. Chebfun responds to this problem by representing functions with discontinuities by piecewise polynomials rather than global ones, with breakpoints at the discontinuities. For example, the location of the discontinuity in the function $\exp(|x-0.1|)$ will be determined automatically in response to the command

```
f = chebfun(@(x) exp(abs(x-0.1)),'splitting','on');
```

The result is a chebfun consisting of two pieces each of degree 3, and the break in the middle appears at the right place:

```
f.ends(2)
```

```
ans =
   0.100000000000000
```

Let us return to 22-year-old Mr. Wilbraham. Unfortunately, his published paper had little impact, and the effect was rediscovered and discussed in the pages of *Nature* during 1898–1899 by James Michelson, A. E. H. Love, and J. Willard Gibbs. These authors got more attention for a number of reasons. First, they were leading scientists. Second, their problem arose at a time when applied mathematics had advanced much further and in a practical application (a mechanical graphing machine called a “harmonic analyser” used by Michelson and Stratton). Third, they published their observations in a major journal. Fourth, they failed to get it right at first, so several publications appeared in succession! Other mathematicians got involved too, notably Poincaré. Finally, they were lucky enough to have “Gibbs’s phenomenon” named and highlighted a few years later in a major research article on Fourier analysis by the mathematician Maxime Bôcher [1906]. For a fascinating discussion of the history of the Gibbs phenomenon (for projection, not interpolation), which they more properly call the *Gibbs–Wilbraham phenomenon*, see [Hewitt & Hewitt 1979].

Summary of Chapter 9. *Chebyshev projections and interpolants, as well as other polynomial and trigonometric approximations, tend to oscillate near discontinuities. The oscillations decay algebraically, not exponentially, with distance from the discontinuity.*

Exercise 9.1. Calculations for larger n. We measured the height of the Gibbs overshoot for a step function for $n = 1, 3, 7, \ldots, 255$. Larger values of n get a bit slow, but knowing that the maximum occurs around $x = 3/n$, compute these numbers up to $n = 4095$ using a command of the form `max(g{0,5/n})`. How great a speedup does this trick produce?

Exercise 9.2. A function with many jumps. Use Chebfun to produce a plot of the degree 200 Chebyshev interpolant to the function `round(exp(sin(2*pi*x)))` on $[-1, 1]$.

Exercise 9.3. Lagrange polynomials. Take $n \geq 2$ to be even and let p be the degree n Chebyshev interpolant to the Kronecker delta function at $x = x_{n/2} = 0$. (a) Use the barycentric formula of Theorem 5.2 to obtain a simple formula for p. (b) Derive a formula for the values of p at the "Chebyshev midpoints" defined by the usual formula $x_j = \cos(j\pi/n)$ of Chapter 2 except with half-integer values of j. (c) For $n = 100$, use Chebfun to produce an elegant plot showing the inverse-linear amplitudes of these values. (You can get the Chebyshev midpoints from `chebpts(n,1)` or from `x=chebpts(2*n+1), x=x(2:2:end)`.)

Exercise 9.4. Fourier and Chebyshev Gibbs phenomena. We have repeatedly made the connection between Chebyshev polynomials $T_n(x)$ on the unit interval, Laurent polynomials $(z^n + z^{-n})/2$ on the unit circle, and trigonometric polynomials $\cos(n\theta)$ on $[-\pi, \pi]$. Use these connections to show that the Gibbs overshoot in Chebyshev interpolation of $\text{sign}(x)$ on $[-1, 1]$, with n even, is identical to the overshoot for a certain problem of trigonometric interpolation in θ.

Exercise 9.5. Local minima of a truncated sine series. (a) Plot ϕ_n with $n = 10, 100$, and 1000 for a sum going back to Euler in 1755,

$$\phi_n(x) = \sum_{k=1}^{n} \frac{\sin(kx)}{k}.$$

What function does the sum evidently converge to? Is the Gibbs overshoot of the same relative magnitude as for (9.3)? (b) For each case, determine the first four local minimum values of $\phi_n(x)$ in $(0, \pi)$. (c) Write an elegant Chebfun program that determines the smallest value of n for which these minima are not monotonically decreasing. (This effect was investigated by Gronwall [1912].)

Exercise 9.6. Sine integral. (a) Construct and plot a chebfun for the sine integral $\mathrm{Si}(x) = \int_0^x t^{-1}\sin t$ for $x \in [0,10]$. What is its length? (b) Same for $x \in [0,100]$. (c) Same for $x \in [0,1000]$.

Exercise 9.7. An unresolvable function. The command `f = chebfun( 'sin(1./x)',100000)` produces a polynomial interpolant to $\sin(1/x)$ through 100,000 Chebyshev points. The plot produced by `plot(f)` looks as if there is a bug in the computation somewhere. Produce similar plots for 10000, 1000, and smaller even numbers of points and explain why in fact, there is no bug.

Exercise 9.8. Decay away from discontinuity. Plot the function $f(x) = \cos(7x)\sin(3x) + \mathrm{sign}(\sin(x/2))e^x$ on $[-1,1]$ as well as its interpolating polynomial $p_n(x)$ in $n+1 = 100$ Chebyshev points. Confirm the algebraic rate of decay away from the discontinuity by plotting $|f(x) - p_n(x)|$ together with the function $c/|x|$ for a suitable value of c.

Chapter 10

Best approximation

```
ATAPformats
```

An old idea, going back to Chebyshev himself and earlier to Poncelet, is to look for a polynomial p^* of specified degree n that is the *best approximation* to a given continuous function f in the sense of minimizing the ∞-norm of the difference on an interval [Poncelet 1835, Chebyshev 1854 & 1859]. (A best approximation is also called a *Chebyshev approximation,* but we shall avoid this usage to minimize confusion. Other terms for the same idea include *minimax* and *equiripple.*) It is known that p^* exists and is unique, as we shall prove below. There is a Chebfun command `remez` for computing these approximants, due to Ricardo Pachón: if f is a chebfun, then `remez(f,n)` is the chebfun corresponding to its best approximation of degree n. For details see [Pachón & Trefethen 2009].

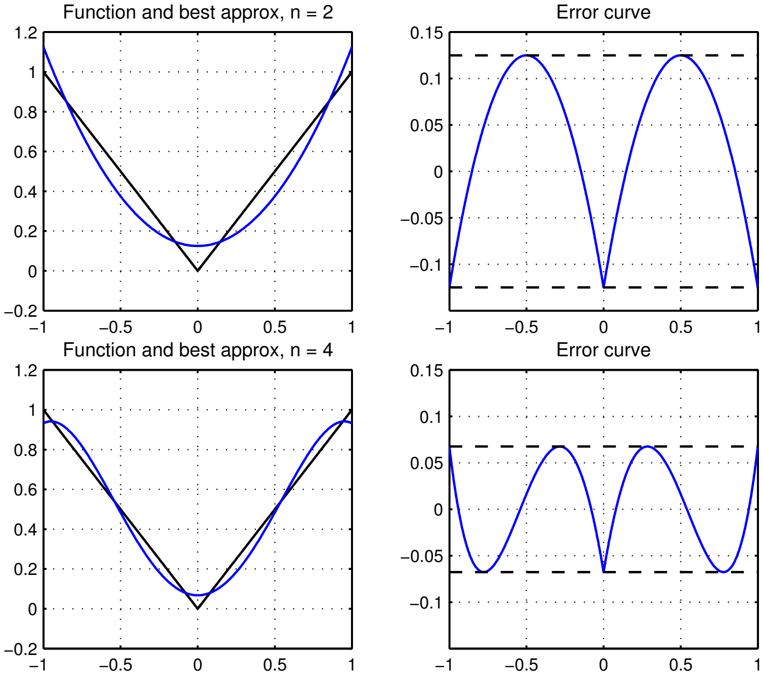
We shall argue in Chapter 16 that best approximations in the ∞-norm are not always as useful as one might imagine; Chebyshev interpolants are often as good or even better. Nevertheless, they represent an elegant and fundamental idea and a line of investigation going back more than 150 years. So for the moment, let us enjoy them.

For example, here are the best approximants of degree 2 and 4 to $|x|$, together with their *error curves* $(f-p^*)([-1,1])$:

```
x = chebfun('x'); f = abs(x);
for n = 2:2:4
  subplot(1,2,1), hold off, plot(f,'k'), grid on
  [p,err] = remez(f,n); hold on, plot(p,'b'), axis([-1 1 -.2 1.2])
  FS = 'fontsize';
  title(['Function and best approx, n = ' int2str(n)],FS,9)
  subplot(1,2,2), hold off, plot(f-p), grid on, hold on
  axis([-1 1 -.15 .15]), title('Error curve',FS,9)
  plot([-1 1],err*[1 1],'--k'), plot([-1 1],-err*[1 1],'--k')
  snapnow
end
```

```
Warning:  This command is deprecated. Use minimax instead.
```

```
Warning:  This command is deprecated. Use minimax instead.
```



Notice the *equioscillation* property: the error curve attains its extreme magnitude with alternating signs at a succession of values of x. Chebyshev appears to have known this in the 1850s, and indeed suggested he was not the first to know it ("comme on le sait", p. 114 of [Chebyshev 1854]), but he did not explicitly address questions of existence, uniqueness, or even alternation of signs. More systematic treatments came at the beginning of the 20th century with work by Blichfeldt [1901], Kirchberger [1902], and Borel [1905]. It seems to have been Kirchberger, in his PhD thesis written under Hilbert, who first stated and proved the characterization theorem that is now so well known [Kirchberger 1902], proving in particular that a best approximation p^* exists. Note that in the characterization part of this theorem, f is assumed to be real, whereas most of the discussion in this book allows f to be real or complex. Existence and uniqueness in the complex case were established by Tonelli [1908]. Complex generalizations of the characterization originate with [Kolmogorov 1948] and [Remez 1951]. Many further generalizations can also be found in the approximation theory literature, for example with the set of polynomials on an interval replaced by a more general set of functions satisfying a property known as the *Haar condition.*

Theorem 10.1. Equioscillation characterization of best approximants. *A continuous function f on $[-1,1]$ has a unique best approximation $p^* \in \mathcal{P}_n$. If f is real, then p^* is real too, and in this case a polynomial $p \in \mathcal{P}_n$ is equal to p^* if and only if $f-p$ equioscillates in at least $n+2$ extreme points.*

Proof. A set of $n+2$ points of equioscillation of this kind is called an *alternant,* though we shall not make much use of this term.

To prove existence of a best approximation, we note that $\|f-p\|$ is a continuous function of $p \in \mathcal{P}_n$. Since one candidate approximation is the zero function, we know that if p^* exists, it lies in $\{p \in \mathcal{P}_n : \|f-p\| \le \|f\|\}$. This is a closed and bounded subset of a finite-dimensional space, hence compact (the Bolzano–Weierstrass property), and thus the minimum is attained. (This argument originates with F. Riesz [1918].)

Next we show that equioscillation implies optimality. Suppose f and p are real and $(f-p)(x)$ takes equal extreme values with alternating signs at $n+2$ points $x_0 < x_1 < \cdots < x_{n+1}$, and suppose $\|f-q\| < \|f-p\|$ for some real polynomial $q \in \mathcal{P}_n$. Then $p-q$ must take nonzero values with alternating signs at the equioscillation points, implying that it takes the value zero in at least $n+1$ points in-between. This implies that $p-q$ is identically zero, which is a contradiction.

The third step is to show that optimality implies equioscillation (this part of the argument was given in [Blichfeldt 1901]). Suppose $f-p$ equioscillates at fewer than $n+2$ points, and set $E = \|f-p\|$. Without loss of generality suppose the leftmost extremum is one where $f-p$ takes the value $-E$. Then there are numbers $-1 < x_1 < \cdots < x_k < 1$ with $k \le n$ such that $(f-p)(x) < E$ for $x \in [-1,x_1] \cup [x_2,x_3] \cup [x_4,x_5] \cup \cdots$ and $(f-p)(x) > -E$ for $x \in [x_1,x_2] \cup [x_3,x_4] \cup \cdots$. If we define $\delta p(x) = (x_1-x)(x_2-x)\cdots(x_k-x)$, then $(p-\varepsilon\delta p)(x)$ will be a better approximation than p to f for all sufficiently small $\varepsilon > 0$.

Finally, to prove uniqueness of best approximations—we treat the real case only—we refine the argument that equioscillation implies optimality. Suppose p is a best approximation with equioscillation extreme points $x_0 < x_1 < \cdots < x_{n+1}$, and suppose $\|f-q\| \le \|f-p\|$ for some real polynomial $q \in \mathcal{P}_n$. Then (without loss of generality) $(p-q)(x)$ must be ≤ 0 at $x_0, x_2, x_4, \ldots$ and ≥ 0 at $x_1, x_3, x_5, \ldots$. This implies that $p-q$ has roots in each of the $n+1$ closed intervals $[x_0, x_1], [x_1, x_2], \ldots, [x_n, x_{n+1}]$. We wish to conclude that $p-q$ has at least $n+1$ roots in total, counted with multiplicity, implying that $p=q$. To make the argument we prove by induction that $p-q$ has at least k roots in $[x_0, x_k]$ for each k. The case $k=1$ is immediate. For the general case, suppose $p-q$ has at least j roots in $[x_0, x_j]$ for each $j \le k-1$ but only $k-1$ roots in $[x_0, x_k]$. Then there must be a simple root at x_{k-1}. By the induction hypothesis, $p-q$ must have exactly $k-2$ roots in $[x_0, x_{k-2}]$ with a simple root at x_{k-2}, $k-3$ roots in $[x_0, x_{k-3}]$ with a simple root at x_{k-3}, and so on down to 1 root in $[x_0, x_1]$, with a simple root at x_1. It follows that $p-q$ must be nonzero at x_0 and at x_k, and since the sign of $p-q$ changes at each of the simple roots $x_1, \ldots x_{k-1}$, the signs at x_0 and x_k must be the same if k is odd and opposite if k is even. On the other hand from the original alternation condition we know that $p-q$ must take the same signs at x_0 and x_k if k is even and opposite signs if k is odd.

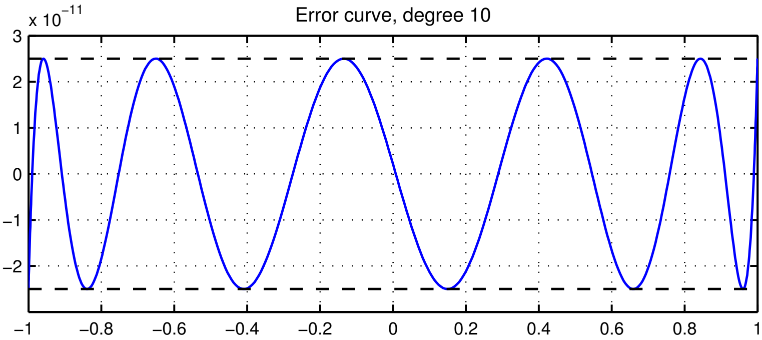
There is a simpler proof of uniqueness than the one just given, in which one supposes p and q are distinct best approximations and considers $(p+q)/2$ (Exercise 10.10). However, that proof does not generalize to the problem of rational approximation (Theorem 24.1). ▮

Note that the error curve for a best approximation may have more than $n+2$ points of equioscillation, and indeed this will always happen if f and n are both even or both odd (Exercise 10.4). For example, for the function $f(x) = |x|$ considered above, the degree 2 approximation equioscillates at 5 points, not 4, and the degree 4 approximation equioscillates at 7 points, not 6. This phenomenon of "extra" points of equioscillation will become important in the generalization to rational approximation in Chapter 24.

Here is another example, the degree 10 best approximation to $\exp(x)$. There are 12 points of equioscillation.

```
f = exp(x);
[p,err] = remez(f,10);
clf, plot(f-p), grid on, hold on
title('Error curve, degree 10',FS,9)
plot([-1 1],err*[1 1],'--k'), plot([-1 1],-err*[1 1],'--k')
```

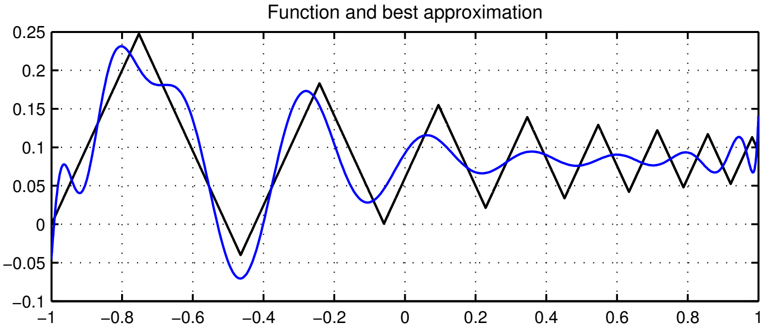
```
Warning:  This command is deprecated. Use minimax instead.
```



And here is another example. The Chebfun `cumsum` command returns the indefinite integral, producing in this case a zigzag function.

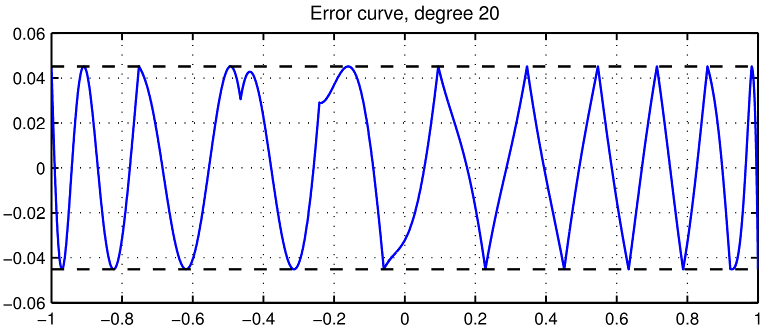
```
f = cumsum(sign(sin(20*exp(x))));
clf, plot(f,'k'), hold on
[p,err] = remez(f,20);
plot(p), grid on, title('Function and best approximation',FS,9)
```

```
Warning:  This command is deprecated. Use minimax instead.
```



The corresponding error curve reveals $20 + 2 = 22$ points of equioscillation:

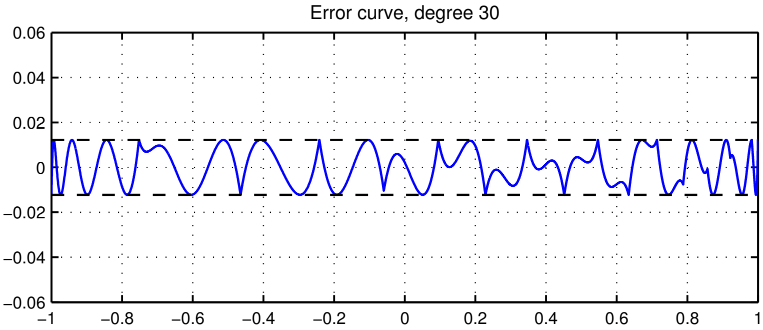
```
hold off, plot(f-p), grid on, hold on, axis([-1 1 -.06 .06])
plot([-1 1],err*[1 1],'--k'), plot([-1 1],-err*[1 1],'--k')
title('Error curve, degree 20',FS,9)
```



Here's the analogous curve for degree 30, plotted on the same scale.

```
[p,err] = remez(f,30);
hold off, plot(f-p), grid on, hold on, axis([-1 1 -.06 .06])
plot([-1 1],err*[1 1],'--k'), plot([-1 1],-err*[1 1],'--k')
title('Error curve, degree 30',FS,9)
```

```
Warning:  This command is deprecated. Use minimax instead.
```



The algorithm underlying `remez`, known as the *Remez algorithm* or the *exchange algorithm,* goes back to the Soviet mathematician Evgeny Remez in 1934, and is based on iteratively adjusting a trial alternant until it converges to a correct one [Remez 1934a, 1934b, 1957; Powell 1981]. We shall not give details here, but in fact, Chebfun is an excellent platform for such computations since the algorithm depends on repeatedly finding the local extrema of trial error curves, an operation carried out easily via the `roots` command (see Chapter 18). Also crucial to the success of `remez` is the use of the barycentric representation (5.11) for all polynomials, based not at Chebyshev points but at the points of each trial alternant [Pachón & Trefethen 2009]. (The observations of [Webb, Gonnet & Trefethen 2011] suggest that it might be better to use the "first barycentric formula" (5.9).)

The history of the Remez algorithm is interesting, or perhaps we should say the sociology. It stands out as one of the preeminent examples of a nontrivial algorithm for a nonlinear computational problem that was developed before the invention of computers. Perhaps in part because of this early appearance, it became remarkably well known, a fixture in numerical analysis courses worldwide. One might imagine, based on its fame, that the Remez algorithm must be very important in practice, but in fact it seems there is not much software and just a moderate amount of use of these ideas. One application has been in the design of routines for computing special functions [Cody, Fraser & Hart 1968, Cody 1993, Muller 2006]. Another is in the field of digital signal processing, where variants of the Remez ideas were developed by Parks and McClellan beginning in 1971 with tremendous success for designing low-pass, high-pass, and other digital filters [Parks & McClellan 1972]. Parks and McClellan too found that the use of a barycentric representation was crucial, as they describe memorably in [McClellan & Parks 2005].

Chapter 16 will show that Chebyshev interpolants are often as good as best approximations in practice, and this fact may have something to do with why the Remez algorithm is used rather little. Chapter 20 will show that if you really want a best approximation, it may be more practical to compute it by CF approximation than by the Remez algorithm, at least if f is smooth. There are also other algorithms for computing best approximations, based for example on linear programming, which we shall not discuss.

Summary of Chapter 10. *Any $f \in C([-1,1])$ has a unique best approximation $p^* \in \mathcal{P}_n$ with respect to the ∞-norm. If f is real, p^* is characterized by having an error curve that equioscillates between at least $n+2$ extreme points.*

Exercise 10.1. A function with spikes. Compute numerically the degree 10 polynomial best approximation to $\text{sech}^2(5(x+0.6)) + \text{sech}^4(50(x+0.2)) + \text{sech}^6(500(x-0.2))$ on $[-1,1]$ and plot f together with p^* as well as the error curve. What is the error? How does this compare with the error in 11-point Chebyshev interpolation? (For these Chebfun computations to be practical, use `'splitting','on'`.)

Exercise 10.2. Best approximation of $|x|$. (a) Use Chebfun to determine the errors $E_n = \|f - p_n\|$ in degree n best approximation of $f(x) = |x|$ on $[-1,1]$ for $n = 2, 4, 8, \dots, 256$, and make a table of the values $\beta_n = nE_n$ as a function of n. (b) Use Richardson extrapolation to improve your data. How many digits can you estimate for the limiting number $\beta = \lim_{n\to\infty} \beta_n$? (We shall discuss this problem in detail in Chapter 25.)

Exercise 10.3. de la Vallée Poussin lower bound. Suppose an approximation $p \in \mathcal{P}_n$ to $f \in C([-1,1])$ approximately equioscillates in the sense that there are points $-1 \le s_0 < s_1 < \cdots < s_{n+1} \le 1$ at which $f - p$ alternates in sign with $|f(s_j) - p(s_j)| \ge \varepsilon$ for some $\varepsilon > 0$. Show that $\|f - p^*\| \ge \varepsilon$. (This estimate originates in [de la Vallée Poussin 1910].)

Exercise 10.4. Best approximation of even functions. Let $f \in C([-1,1])$ be an even function, i.e., $f(-x) = f(x)$ for all x. (a) Prove as a corollary of Theorem 10.1 that for any $n \ge 0$, the best approximation p_n^* is even. (b) Prove that for any $n \ge 0$, $p_{2n}^* = p_{2n+1}^*$. (c) Conversely, suppose $f \in C([-1,1])$ is not even. Prove that for all sufficiently large n, its best approximations p_n^* are not even.

Exercise 10.5. An invalid theorem. The first two figures of this chapter suggest the following "theorem": *if f is an even function on $[-1,1]$ and p^* is its best approximation of some degree n, then one of the extreme points of $|(f-p^*)(x)|$ occurs at $x = 0$.* Pinpoint the flaw in the following "proof". By the argument of Exercise 10.4(b), p^* is the best approximation to f for all n in some range of the form even $\le n \le$ odd, such as $4 \le n \le 5$ or $10 \le n \le 13$. By Theorem 10.1, the number of equioscillation points of $f - p^*$ must accordingly be of the form odd $+ 2$, that is, odd. By symmetry, 0 must be one of these points.

Exercise 10.6. Nonlinearity of best approximation operator. We have mentioned that for given n, the operator that maps a function $f \in C([-1,1])$ to its best degree n approximation p_n^* is nonlinear. Prove this (on paper, not numerically) by finding two functions f_1 and f_2 and an integer $n \ge 0$ such that the best approximation of the sum in $\mathcal{P}_n$ is not the sum of the best approximations.

Exercise 10.7. Bernstein's lethargy theorem. Exercise 6.1 considered a function of Weierstrass, continuous but nowhere differentiable. A variant of the same function based on Chebyshev polynomials would be

$$f(x) = \sum_{k=0}^{\infty} 2^{-k} T_{3^k}(x). \tag{10.1}$$

(a) Show that the polynomial f_{3^k} obtained by truncating (10.1) to degree 3^k is the best approximation to f in the spaces $\mathcal{P}_n$ for certain n. What is the complete set of n for which this is true? What is the error? (b) Let $\{\varepsilon_n\}$ be a sequence decreasing monotonically to 0. Prove that there is a function $f \in C([-1,1])$ such that $\|f - p_n^*\| \ge \varepsilon_n$ for all n. (Hint: change the coefficients 2^{-k} of (10.1) to values related to $\{\varepsilon_n\}$.)

Exercise 10.8. Continuity of best approximation operator. For any $n \ge 0$, the mapping from functions $f \in C([-1,1])$ to their best approximants $p^* \in \mathcal{P}_n$ is continuous with respect to the ∞-norm in $C([-1,1])$. Prove this by an argument combining the uniqueness of best approximations with compactness. (This continuity result appears in Section I.5 of [Kirchberger 1902]. In fact, the mapping is not just continuous but Lipschitz continuous, a property known as *strong uniqueness,* but this is harder to prove.)

Exercise 10.9. Approximation of e^x. Truncating the Taylor series for e^x gives polynomial approximations with maximum error $E_n \sim 1/(n+1)!$ on $[-1,1]$, but the best approximations do better by a factor of 2^n:

$$E_n \sim \frac{1}{2^n(n+1)!}, \quad n \to \infty. \tag{10.2}$$

(a) Derive (10.2) by combining Exercises 3.15 and 10.3 with the asymptotic formula $I_k(1) \sim 1/(2^n n!)$. (b) Make a table comparing this estimate with the actual values E_n computed numerically for $0 \le n \le 10$.

Exercise 10.10. Alternative proof of uniqueness. Prove uniqueness of the degree n best approximant to a real continuous function f by a simpler argument than the one given in the proof of Theorem 10.1: suppose p and q are best approximants, and apply the equioscillation characterization to $r = (p+q)/2$.

Exercise 10.11. Chebyshev polynomials and best approximations. (a) What is the best degree n polynomial approximation to x^{n+1} on $[-1,1]$? What is the error? Derive the answers from Theorem 10.1, using the fact that T_{n+1} oscillates between values ± 1 at $n+2$ points in $[-1,1]$. (b) What is the best approximation to 0 among monic polynomials of degree $n+1$? What is the error?

Exercise 10.12. Every best approximant is an interpolant. Let p be the best approximation in $\mathcal{P}_n$ to a real function $f \in C([-1,1])$. Show that there exist $n+1$ distinct points $-1 < x_0 < x_1 < \cdots < x_n < 1$ such that p is the interpolant in $\mathcal{P}_n$ to f in the points $\{x_j\}$.

Exercise 10.13. A contrast to Faber's theorem. Although Faber showed that there does not exist an array of nodes in $[-1,1]$ whose polynomial interpolants converge for every $f \in C[-1,1]$, for any *fixed* f there exists an array whose interpolants converge to f [Marcinkiewicz 1936/7]. Prove this by combining the Weierstrass approximation theorem with the result of the previous exercise.

Exercise 10.14. Asymptotics of the leading coefficient. Let $\{p_n^*\}$ be the sequence of best approximations of a function $f \in C([-1,1])$, and let p_n^* have leading Chebyshev coefficient a_n^*. It is known that $\limsup_{n\to\infty} |a_n^*|^{1/n} \le 1$, with strict inequality if and only if f is analytic on $[-1,1]$ [Blatt & Saff 1986, Thm. 2.1]. Verify this result numerically by estimating $\limsup_{n\to\infty} |a_n^*|^{1/n}$ for $f(x) = |x|$ and $f(x) = 1/(1+25x^2)$.

Chapter 11

Hermite integral formula

`ATAPformats`

If there is a single most valuable mathematical tool in the analysis of accuracy of polynomial approximations, it is contour integrals in the complex plane.[1] From a contour integral one can see why some approximations are extraordinarily good, like interpolation in Chebyshev points, and others are impossibly bad, like interpolation in equispaced points. This chapter presents the basics of the contour integrals, and the next applies them to take some first steps toward the subject of potential theory, which relates the accuracy of approximations to equipotential or minimal-energy problems for electrostatic charge distributions in the plane.

The starting ingredients have already appeared in Chapter 5. Following the formulation there, let $x_0, \dots, x_n$ be a set of $n+1$ distinct interpolation or "grid" points, which may be real or complex, and define the node polynomial $\ell \in \mathcal{P}_{n+1}$ as in (5.4) by

$$\ell(x) = \prod_{k=0}^{n} (x - x_k). \tag{11.1}$$

Repeating (5.5), the function

$$\ell_j(x) = \frac{\ell(x)}{\ell'(x_j)\,(x - x_j)} \tag{11.2}$$

is the Lagrange polynomial associated with x_j, that is, the unique polynomial in $\mathcal{P}_n$ that takes the value 1 at x_j and 0 at the other points x_k. Following

[1] This and the next chapter, together with Chapter 20, are possibly the hardest in the book, with a good deal of mathematics presented in a few pages and heavy use of complex variables.

(5.1), a linear combination of these functions gives the interpolant in $\mathcal{P}_n$ to an arbitrary function f defined on the grid:

$$p(x) = \sum_{j=0}^{n} f(x_j)\ell_j(x). \tag{11.3}$$

We now make a crucial observation. Let Γ_j be a contour in the complex x-plane that encloses x_j but none of the other grid points, nor the point x. (By "encloses" we always mean that it winds around the specified set once in the counterclockwise direction, in the usual sense of complex variables.) Then the expression on the right in (11.2) can be written

$$\frac{\ell(x)}{\ell'(x_j)\,(x-x_j)} = \frac{1}{2\pi i}\int_{\Gamma_j} \frac{\ell(x)}{\ell(t)\,(x-t)}\,dt. \tag{11.4}$$

To verify this formula we ignore the $\ell(x)$ term on both sides, which has nothing to do with the integral, and use the fact that $1/(\ell'(x_j)(x-x_j))$ is the *residue* of the function $1/(\ell(t)(x-t))$ at the pole $t = x_j$.

From (11.2) and (11.4) we thus have an expression for $\ell_j(x)$ as a contour integral:

$$\ell_j(x) \;= \frac{1}{2\pi i}\int_{\Gamma_j} \frac{\ell(x)}{\ell(t)\,(x-t)}\,dt, \tag{11.5}$$

where Γ_j encloses x_j. Now let Γ' be a contour that encloses all of the grid points $\{x_j\}$, but still not the point x, and let f be a function analytic on and interior to Γ'. Then we can combine together these integrals to get an expression for the interpolant p to f in $\{x_j\}$:

$$p(x) = \frac{1}{2\pi i}\int_{\Gamma'} \frac{\ell(x)f(t)}{\ell(t)\,(x-t)}\,dt. \tag{11.6}$$

Note how neatly this formula replaces the sum of (11.3) by a contour integral with contributions from the same points x_j.

Now suppose we enlarge the contour of integration to a new contour Γ that encloses x as well as $\{x_j\}$, and we assume f is analytic on and inside Γ. The residue of the integrand of (11.6) at $t = x$ is $-f(x)$, so this brings in a new contribution $-f(x)$ to the integral, yielding an equation for the error in polynomial interpolation:

$$p(x) - f(x) = \frac{1}{2\pi i}\int_{\Gamma} \frac{\ell(x)f(t)}{\ell(t)\,(x-t)}\,dt. \tag{11.7}$$

And thus we have derived one of the most powerful formulas in all of approximation theory, the *Hermite interpolation formula.* This name comes from Hermite [1878], but the same result had been stated 52 years earlier by Cauchy [1826]. (Hermite, however, generalized the formulation significantly to non-distinct or "confluent" interpolation points and corresponding interpolation of derivatives as well as function values; see Exercise 11.2.)

Theorem 11.1. Hermite interpolation formula. *Let f be analytic in a region Ω containing distinct points $x_0, \dots, x_n$, and let Γ be a contour in Ω enclosing these points in the positive direction. The polynomial interpolant $p \in \mathcal{P}_n$ to f at $\{x_j\}$ is*

$$p(x) = \frac{1}{2\pi i} \int_\Gamma \frac{f(t)\,(\ell(t) - \ell(x))}{\ell(t)\,(t-x)}\, dt, \tag{11.8}$$

and if x is enclosed by Γ, the error in the interpolant is

$$f(x) - p(x) = \frac{1}{2\pi i} \int_\Gamma \frac{\ell(x)}{\ell(t)} \frac{f(t)}{(t-x)}\, dt. \tag{11.9}$$

Proof. Equation (11.9) is the same as (11.7). For (11.8), we note that if Γ encloses x, then $f(x)$ can be written

$$f(x) = \frac{1}{2\pi i} \int_\Gamma \frac{\ell(t)\,f(t)}{\ell(t)\,(t-x)}\, dt,$$

and combining this with (11.7) gives the result. But the integrand of (11.8) has no pole at $t = x$, so the same result also applies if Γ does not enclose x.

∎

It is perhaps interesting to sketch Cauchy's slightly different derivation from 1826, outlined in [Smithies 1997, p. 117], which may have been influenced by Jacobi's thesis a year earlier [Jacobi 1825]. Cauchy started from the observation that $p(x)/\ell(x)$ is a rational function with denominator degree greater than the numerator degree. This implies that it must be equal to the sum of the $n+1$ inverse-linear functions $r_j/(x-x_j)$, where r_j is the residue of $p(t)/\ell(t)$ at $t = x_j$ (a partial fraction decomposition, to be discussed further in Chapter 23). Since p interpolates f at $\{x_j\}$, r_j is also the residue of $f(t)/\ell(t)$ at $t = x_j$. By residue calculus we therefore have

$$\frac{p(x)}{\ell(x)} = \frac{1}{2\pi i} \int_{\Gamma'} \frac{f(t)}{\ell(t)\,(x-t)}\, dt$$

if Γ' is again a contour that encloses the points $\{x_k\}$ but not x itself, or equivalently, (11.6).

Now let us see how Theorem 11.1 can be used to estimate the accuracy of polynomial interpolants.

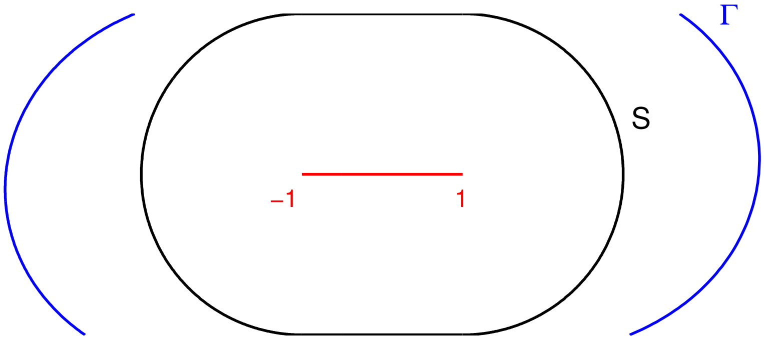
Suppose f and x are fixed and we want to estimate $f(x) - p(x)$ for various degrees n and corresponding sets of $n+1$ points $\{x_j\}$. On a fixed contour Γ, the quantities $f(t)$ and $t - x$ in (11.9) are independent of n. The ratio

$$\frac{\ell(x)}{\ell(t)} = \frac{\prod_{j=0}^{n}(x - x_j)}{\prod_{j=0}^{n}(t - x_j)}, \tag{11.10}$$

however, is another matter. If Γ is far enough from $\{x_j\}$, then for each $t \in \Gamma$, this ratio will shrink exponentially as $n \to \infty$, and if this happens, we may conclude from (11.9) that $p(x)$ converges exponentially to $f(x)$ as $n \to \infty$. The crucial condition for this argument is that it must be possible for f to be analytically continued as far out as Γ.

Here is a warm-up example mentioned in [Gaier 1987, p. 63]. Suppose the interpolation points $\{x_j\}$ lie in $[-1, 1]$ for each n and $x \in [-1, 1]$ also. Let S be the "stadium" in the complex plane consisting of all numbers lying at a distance ≤ 2 from $[-1, 1]$, and suppose f is analytic in a larger region Ω that includes a contour Γ enclosing S. We can sketch the situation like this:

```
x = chebfun('x');
hold off, plot(real(x),imag(x),'r')
semi = 2*exp(0.5i*pi*x);
S = join(x-2i, 1+semi, 2i-x, -1-semi);
hold on, plot(S,'k'), axis equal off
z = exp(1i*pi*x);
Gamma = (2.8+.2i)*(sinh(z)+.5*real(z));
plot(Gamma,'b')
text(4.2,2,'\Gamma','color','b','fontsize',12)
text(3.1,.7,'S','fontsize',12)
text(.9,-.3,'1','color','r')
text(-1.4,-.3,'-1','color','r')
```



Under these assumptions, there is a constant $\gamma > 1$ such that for every $t \in \Gamma$ and every x_j, $|t - x_j| \ge \gamma |x - x_j|$. This implies,

$$|\ell(x)/\ell(t)| \le \gamma^{-n-1}$$

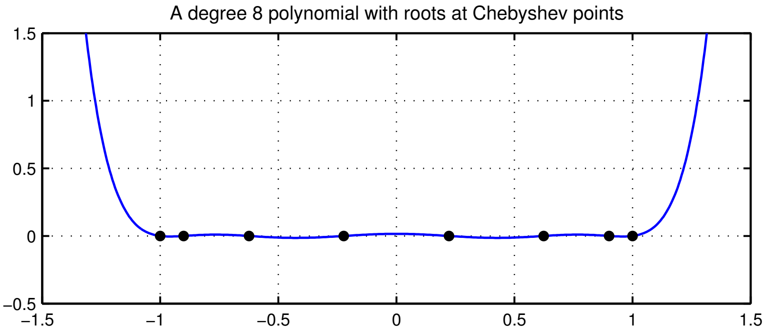
and thus by (11.9),

$$\|f - p\| = O(\gamma^{-n}).$$

Note that this conclusion applies regardless of the distribution of the interpolation points in $[-1,1]$. They could be equally spaced or random, for example. (At least that is true in theory. In practice, such choices would be undone by rounding errors on a computer, as we shall see in the next chapter.)

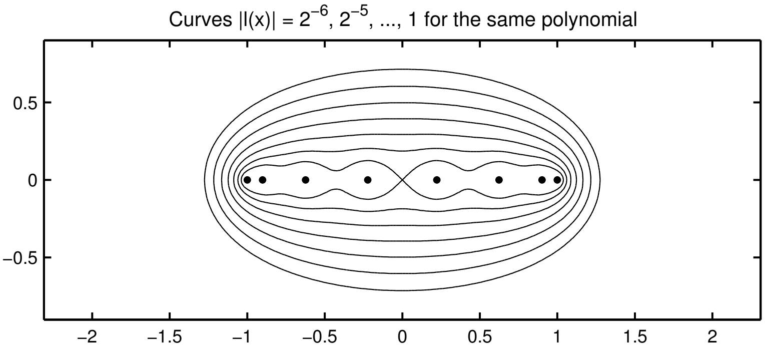
So convergence of polynomial interpolants to analytic functions on $[-1,1]$ is all about how small $\ell(x)$ is on $[-1,1]$, compared with how big it is on a contour Γ inside which f is analytic. From this point of view we can begin to see why Chebyshev points are so good: because a polynomial with roots at the Chebyshev points has approximately uniform magnitude on $[-1,1]$. Suppose for example we consider the polynomial $\ell \in \mathcal{P}_8$ with roots at 8 Chebyshev points. On $[-1,1]$ it has size $O(2^{-8})$, roughly speaking, but it grows rapidly for x outside this interval. Here is a plot for $x \in [-1.5, 1.5]$:

```
np = 8; xj = chebpts(np); FS = 'fontsize';
d = domain(-1.5,1.5);
ell = poly(xj,d);
hold off, plot(ell), grid on
hold on, plot(xj,ell(xj),'.k'), ylim([-.5 1.5])
title('A degree 8 polynomial with roots at Chebyshev points',FS,9)
```



With Matlab's `contour` command we can examine the size of $\ell(x)$ for complex values of x. The following code plots contours at $|\ell(x)| = 2^{-6}, 2^{-5}, \ldots, 1$.

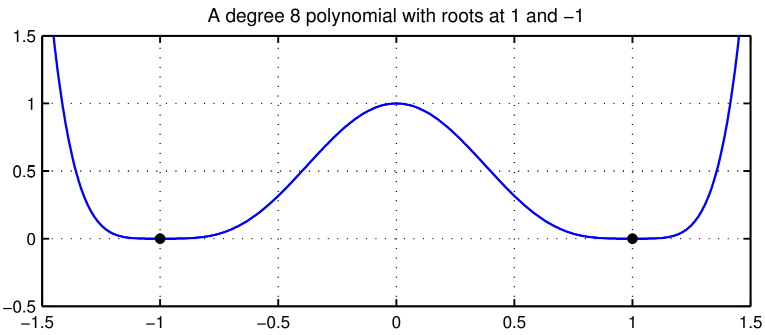
```
hold off, plot(xj,ell(xj),'.k','markersize',10)
hold on, ylim([-0.9,0.9]), axis equal
xgrid = -1.5:.02:1.5; ygrid = -0.9:.02:0.9;
[xx,yy] = meshgrid(xgrid,ygrid); zz = xx+1i*yy;
ellzz = ell(zz); levels = 2.^(-6:0);
contour(xx,yy,abs(ellzz),levels,'k')
title(['Curves |l(x)| = 2^{-6}, 2^{-5}, ..., 1 '...
    'for the same polynomial'],FS,9)
```



We can see a great deal in this figure. On $[-1,1]$, it confirms that $\ell(x)$ is small, with maximum value $|\ell(x)| = 2^{-6}$ at $x = 0$. Away from $[-1,1]$, $|\ell(x)|$ grows rapidly and takes constant values on curves that look close to ellipses. For t on the outermost of the curves plotted, the ratio $|\ell(x)/\ell(t)|$ will be bounded by 2^{-6} for any $x \in [-1,1]$.

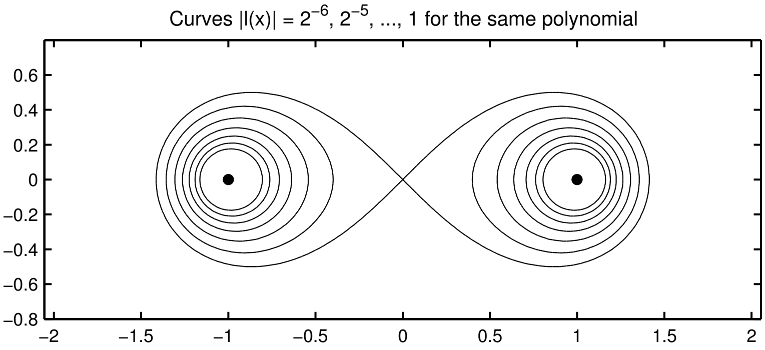
Let us compare this to the very different behavior if we take points that are not close to the Chebyshev distribution. To make a specific and quite arbitrary choice, let us again take 8 points, four of them at -1 and four at 1. Here is the plot on the real axis.

```
xj = [-1 -1 -1 -1 1 1 1 1];
ell = poly(xj,d);
hold off, plot(ell), grid on
hold on, plot(xj,ell(xj),'.k'), ylim([-.5 1.5])
title('A degree 8 polynomial with roots at 1 and -1',FS,9)
```



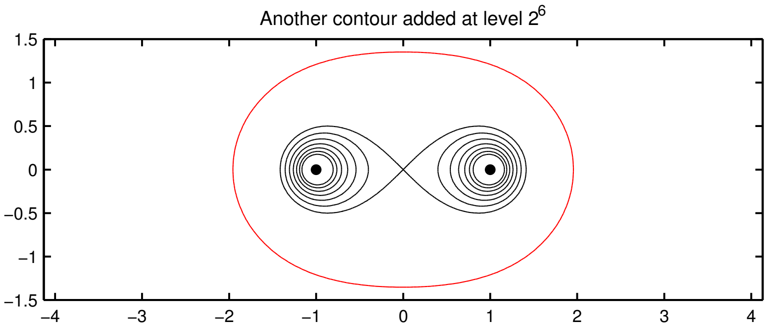
And here are the contours in the complex plane.

```
hold off, plot(xj,ell(xj),'.k'), hold on
ylim([-0.8,0.8]), axis equal, ellzz = ell(zz);
contour(xgrid,ygrid,abs(ellzz),levels,'k')
title(['Curves |l(x)| = 2^{-6}, 2^{-5}, ..., 1 '...
    'for the same polynomial'],FS,9)
```



These figures show that the size of $\ell(x)$ on $[-1,1]$ is not at all uniform: it is far smaller than 2^{-6} for $x \approx \pm 1$, but as big as 1 at $x = 0$. Now, for $x \in [-1,1]$ and t on the outermost curve shown, the maximum of the ratio $|\ell(x)/\ell(t)|$ is no better than 1 since that curve touches $[-1,1]$. If we wanted to achieve $|\ell(x)/\ell(t)| \leq 2^{-6}$ as in the last example, Γ would have to be a much bigger curve—closer to the “stadium”:

```
xgrid = -2:.04:2; ygrid = -1.5:.04:1.5;
[xx,yy] = meshgrid(xgrid,ygrid); zz = xx+1i*yy;
ellzz = ell(zz); levels = 2.^(-6:0); levels = [2^6,2^6];
hold on, contour(xgrid,ygrid,abs(ellzz),levels,’r’)
ylim([-1.5 1.5]), axis equal
title(’Another contour added at level 2^6’,FS,9)
```



The function f would have to be analytic within this much larger region for the bound (11.9) to apply with a ratio $|\ell(x)/\ell(t)|$ as favorable as 2^{-6}.

Summary of Chapter 11. *The error of a polynomial interpolant can be represented by a contour integral in the complex plane, the Hermite integral formula. This provides the standard method for showing geometric convergence for certain approximations of analytic functions.*

Exercise 11.1. Chebfun computation of Cauchy integrals. (a) Figure out (on paper) the polynomial $p \in \mathcal{P}_2$ that takes the values $p(-1) = 1$, $p(1/2) = 2$, and $p(1) = 2$. What is $p(2)$? (b) Read about the numerical computation of Cauchy integrals in Chapter 5 of the online *Chebfun Guide*. Write a program to confirm Theorem 11.1 by computing $p(2)$ numerically by a Cauchy integral for the function $f(x) = (x+1)(x-0.5)(x-1)e^x + 11/6 + x/2 - x^2/3$. Take both $|x| = 3/2$ and $|x| = 3$ as contours to confirm that it does not matter whether or not Γ encloses x. (c) Write an anonymous function `p = @(x) ...` to apply the above calculation not just for $x = 2$ but for arbitrary x, and construct a chebfun on $[-1, 1]$ from this anonymous function. Do its coefficients as reported by `poly` match your expectations?

Exercise 11.2. Confluent interpolation points. Modify the above problem to require $p(-1) = 1$, $p(1) = 2$, and $p'(1) = 0$. This is a *Hermite interpolation problem,* in which some interpolation points are specified multiply with corresponding values specified for derivatives. What is the analytic solution to this interpolation problem? Do the computations involving contour integrals and anonymous functions deliver the right result?

Exercise 11.3. Interpolation in a disk. Suppose a function f is interpolated by polynomials in arbitrary points of the disk $|x| \le r'$ and we measure the accuracy $f(x) - p(x)$ for x in the disk $|x| \le r$. Show that geometric convergence is assured (in exact arithmetic, ignoring rounding errors) if f is analytic in the disk $|x| \le r + 2r'$. Give the constant ρ for convergence at the rate $O(\rho^{-n})$. (This result originates with [Méray 1884].)

Exercise 11.4. Working around a simple pole. Let f be analytic on the closed Bernstein ellipse region $\overline{E}_\rho$ for some $\rho > 1$. It can be shown that $|\ell(x)/\ell(t)| = O(\rho^{-n})$ uniformly as $n \to \infty$ for $x \in [-1, 1]$ and t on the ellipse, and thus Theorem 11.1 can be used to show that $\|f - p_n\| = O(\rho^{-n})$ as asserted by Theorem 8.2. Now suppose that f has one or more singularities on the ellipse but these are just simple poles. Explain how the contour integral argument can be modified to show that the rate of convergence will still be $\|f - p_n\| = O(\rho^{-n})$, as was established by another method in Exercise 8.15.

Chapter 12

Potential theory and approximation

`ATAPformats`

The explorations of the last chapter are glimmerings of potential theory in the complex plane, a subject that has been connected with approximation of functions since the work of Walsh early in the 20th century [Walsh 1969]. In this chapter we shall outline this connection. Potential theory in the complex plane is presented in [Ransford 1995] and [Finkelshtein 2006], and a survey of applications in approximation theory can be found in [Levin & Saff 2006].

We begin by looking again at (11.10), the formula giving the ratio of the size of the node polynomial ℓ at an approximation point x to its size at a point t on a contour Γ. Notice that the numerator and the denominator of this formula each contain a product of $n+1$ terms. With this in mind, let us define $\gamma_n(x,t)$ as the following $(n+1)$st root:

$$\gamma_n(x,t) = \frac{\left(\prod_{j=0}^{n} |t - x_j|\right)^{1/(n+1)}}{\left(\prod_{j=0}^{n} |x - x_j|\right)^{1/(n+1)}}. \qquad (12.1)$$

Then the magnitude of the quotient in (11.10) becomes

$$\left|\frac{\ell(x)}{\ell(t)}\right| = \gamma_n(x,t)^{-n-1}. \qquad (12.2)$$

This way of writing things brings out a key point: if $\gamma_n(x,t)$ is bounded above 1, we will get exponential convergence as $n \to \infty$. With this in mind, let us define α_n to be the scalar

$$\alpha_n = \min_{x\in X,\, t\in\Gamma} \gamma_n(x,t), \tag{12.3}$$

where x ranges over a domain X where we wish to approximate f (say, $X = [-1,1]$) and t ranges over a contour Γ enclosing that domain. If $\alpha_n \geq \alpha$ for some $\alpha > 1$ for all sufficiently large n, and if f is analytic in the region bounded by Γ, then (11.9) tells us that $p(x)$ must converge to $f(x)$ at the rate $O(\alpha^{-n})$.

The condition $\alpha_n > 1$ has a geometric interpretation. The numerator of (12.1) is the geometric mean distance of t to the grid points $\{x_j\}$, and the denominator is the geometric mean distance of x to the same points. If $\alpha_n > 1$, then every point $t \in \Gamma$ is at least α_n times farther from the grid points, in the geometric mean sense, than every point x in the approximation domain. It is this property that allows the Hermite integral formula to show exponential convergence.

To bring these observations into potential theory, we linearize the products by taking logarithms. From (12.1) we find

$$\log \gamma_n(x,t) = \frac{1}{n+1}\sum_{j=0}^{n} \log|t - x_j| - \frac{1}{n+1}\sum_{j=0}^{n} \log|x - x_j|. \tag{12.4}$$

Let us define the **discrete potential function** associated with the points $x_0, \dots, x_n$ by

$$u_n(s) = \frac{1}{n+1}\sum_{j=0}^{n} \log|s - x_j|. \tag{12.5}$$

Note that u_n is a harmonic function throughout the complex s-plane away from the gridpoints, that is, a solution of the Laplace equation $\Delta u_n = 0$. We may think of each x_j as a point charge of strength $1/(n+1)$, like an electron, and of u_n as the potential generated by all these charges, whose gradient defines an "electric" field. A difference from the electrical case is that whereas electrons repel one another with an inverse-square force, whose potential function is inverse-linear, here in the two-dimensional plane the repulsion is inverse-linear and the potential is logarithmic. (Some authors put a minus sign in front of (12.5), so that the potential approaches ∞ rather than $-\infty$ as $s \to x_j$, making u_n an energy rather than the negative of an energy.)

From (12.4) and (12.5) we find

$$\log \gamma_n(x,t) = u_n(t) - u_n(x),$$

and hence by (12.2),

$$\left|\frac{\ell(x)}{\ell(t)}\right| = e^{(n+1)[u_n(x)-u_n(t)]}. \tag{12.6}$$

If $\alpha_n \geq \alpha > 1$ for all sufficiently large n, as considered above, then $\log \gamma_n(x,t) \geq \log \alpha_n \geq \log \alpha > 0$, so we have

$$\min_{t\in\Gamma} u_n(t) - \max_{x\in X} u_n(x) \geq \log \alpha.$$

Together with (11.9) this implies

$$\|f-p\| = O(e^{-n\log\alpha}).$$

Notice the flavor of this result: the interpolants converge exponentially, with a convergence constant that depends on the difference of the values taken by the potential function on the set of points where the interpolant is to be evaluated and on a contour inside which f is analytic.

We now take the step from discrete to continuous potentials. Another way to write (12.5) is as a Lebesgue–Stieltjes integral [Stein & Shakarchi 2005],

$$u(s) = \int_{-1}^{1} \log|s-\tau|\,d\mu(\tau), \tag{12.7}$$

where μ is a measure consisting of a sum of Dirac delta functions, each of strength $1/(n+1)$,

$$\mu(\tau) = \frac{1}{n+1}\sum_{j=0}^{n} \delta(\tau - x_j). \tag{12.8}$$

This is the *potential* or *logarithmic potential* associated with the measure μ. The same formula (12.7) also applies if μ is a continuous measure, which will typically be obtained as the limit of a family of discrete measures as $n \to \infty$. (The precise notion of convergence appropriate for this limit is known as weak* convergence, pronounced "weak-star.") Equally spaced grids in $[-1,1]$ converge to the limiting measure

$$\mu(\tau) = \frac{1}{2}. \tag{12.9}$$

Chebyshev grids in $[-1,1]$ converge to the *Chebyshev measure* identified in Exercise 2.2,

$$\mu(\tau) = \frac{1}{\pi\sqrt{1-\tau^2}}, \tag{12.10}$$

and so do other grids associated with zeros or extrema of orthogonal polynomials on $[-1,1]$, such as Legendre, Jacobi, or Gegenbauer polynomials (see Chapter 17).

And now we can identify the crucial property of the Chebyshev measure (12.10): *The potential* (12.7) *it generates is constant on* $[-1,1]$. The measure is known as the *equilibrium measure* for $[-1,1]$, and physically, it corresponds to one unit of charge adjusting itself into an equilibrium, minimal-energy distribution. Given a unit charge distribution μ with support on $[-1,1]$, the associated energy is the integral

$$I(\mu) = -\int_{-1}^{1} u(s)\,d\mu(s) = -\int_{-1}^{1}\int_{-1}^{1} \log|s-\tau|\,d\mu(\tau)\,d\mu(s). \tag{12.11}$$

It is clear physically, and can be proved mathematically, that for $I(\mu)$ to be minimized, $u(s)$ must be constant, so the gradient of the potential is zero and there are no net forces on the points in $(-1,1)$ [Ransford 1995].

This discussion has gone by speedily, and the reader may have to study these matters several times to appreciate how naturally ideas associated with electric charges connect with the accuracy of polynomial approximations. Potential theory is also of central importance in the study of approximation by rational functions; see [Levin & Saff 2006] and [Stahl & Schmelzer 2009].

We have just characterized the equilibrium measure μ for interpolation on $[-1,1]$ as the unit measure on $[-1,1]$ that generates a potential u that takes a constant value on $[-1,1]$. To be precise, u is the solution to the following problem involving a Green's function: find a function $u(s)$ in the complex s-plane that is harmonic outside $[-1,1]$, approaches a constant value as $s \to [-1,1]$, and is equal to $\log|s| + O(s^{-1})$ as $s \to \infty$. (This last condition comes from the property that the total amount of charge is 1.) Quite apart from the motivation from approximation theory, suppose we are given this Green's function problem to solve. Since Laplace's equation is invariant under conformal maps, the solution can be derived by introducing a conformal map that transplants the exterior of the interval to the exterior of a disk, taking advantage of the fact that the Green's function problem is trivial on a disk. Such a mapping is the function

$$z = \phi(s) = \frac{1}{2}(s + i\sqrt{1-s^2}), \tag{12.12}$$

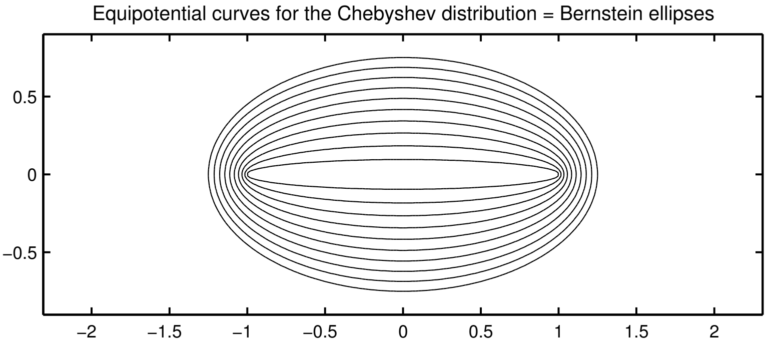
which maps the exterior of $[-1,1]$ in the s-plane onto the exterior of the disk $|z| \le 1/2$ in the z-plane. There, the solution of the potential problem is $\log|z|$. Mapping back to s, we find that the Chebyshev potential is given by $u(s) = \log|\phi(s)|$, that is,

$$u(s) = \log|s + i\sqrt{1-s^2}\,| - \log 2, \tag{12.13}$$

with constant value $u(s) = -\log 2$ on $[-1,1]$.

By definition, the Green's function has a constant value on $[-1,1]$, namely $u(s) = -\log 2$. For values $u_0 > -\log 2$, the equation $u(s) = u_0$ defines an *equipotential curve* enclosing $[-1,1]$ that is exactly the Bernstein ellipse E_ρ with $\rho = 2\exp(u_0)$, as defined in Chapter 8. Here is a contour plot of (12.13), confirming that the contours look the same as the ellipses plotted there. The factor `sign(imag(s))` is included to make `u` return the correct branch of the square root for $\operatorname{Im} s < 0$.

```
u = @(s) log(abs(s+1i*sign(imag(s)).*sqrt(1-s.^2))) - log(2);
xgrid = -1.5:.02:1.5; ygrid = -0.91:.02:0.91;
[xx,yy] = meshgrid(xgrid,ygrid); ss = xx+1i*yy; uss = u(ss);
levels = -log(2) + log(1.1:0.1:2);
hold off, contour(xgrid,ygrid,uss,levels,'k')
ylim([-0.9,0.9]), axis equal, FS = 'fontsize';
title(['Equipotential curves for the Chebyshev ' ...
            'distribution = Bernstein ellipses'],FS,9)
```



The constant $-\log 2$ in (12.13) is a reflection of the length of the interval $[-1,1]$. Specifically, this constant is the logarithm of the *capacity* (or *logarithmic capacity* or *transfinite diameter*) of $[-1,1]$,

$$c = \frac{1}{2}.$$

The capacity is a standard notion of potential theory, and in a simply connected 2D case like this one, it can be defined as the radius of the equivalent disk. The associated minimal energy is the *Robin constant* of $[-1,1]$:

$$\min_{\mu} I(\mu) = -\log(c) = \log 2.$$

The fact that the capacity of $[-1,1]$ is $1/2$ has the following interpretation, explored earlier in Exercise 2.6. For Chebyshev or other asymptotically optimal grids on $[-1,1]$, in the limit $n\to\infty$, each grid point lies at a distance $1/2$ from the others in the geometric mean sense.

This is a book about approximation on intervals, but it is worth noting that all these ideas of equilibrium measure, minimal energy, Robin constant and capacity generalize to other compact sets E in the complex plane. If E is connected, then μ and u can be obtained from a conformal map of its exterior onto the exterior of a disk, whereas if it is disconnected, a more general Green's function problem must be solved. In any case, the equilibrium measure, which is supported on the outer boundary of E, describes a good asymptotic distribution of interpolation points as $n\to\infty$, and the limiting geometric mean distance from one point to the others is equal to the capacity, which is related to the Robin constant by $c(E)=\exp(-\min_\mu I(\mu))$.

Having discussed the continuous limit, let us return to the finite problem of finding good sets of $n+1$ points $\{x_j\}$ for interpolation by a polynomial $p\in\mathcal{P}_n$ on a compact set E in the complex plane. Three particular families of points have received special attention. We say that $\{x_j\}$ is a set of *Fekete points* for the given n and E if the quantity

$$\left(\prod_{j\neq k}|x_j-x_k|\right)^{2/n(n+1)}, \qquad (12.14)$$

which is the geometric mean of the distances between the points, is as large as possible, that is, the points are exactly in a minimal-energy configuration. As $n\to\infty$, these maximal quantities decrease monotonically to $c(E)$, the fact which gives rise to the expression "transfinite diameter". As a rule Fekete points have some of the cleanest mathematical properties for a given set E but are the hardest to compute numerically. Next, if E is connected and $\phi(x)$ is a map of its exterior to the exterior of a disk in the z-plane centered at the origin, a set of *Fejér points* is a set $\phi^{-1}(\{z_j\})$, where $\{z_j\}$ consists of any $n+1$ points spaced equally around the boundary circle. Fejér points are more readily computable since it is often possible to get one's hands on a suitable mapping ϕ. Finally, *Leja points* are approximations to Fekete points obtained by a "greedy algorithm." Here, one starts with an arbitrary first point $x_0\in E$ and then computes successive points $x_1,x_2,\dots$ by an incremental version of the Fekete condition: with $x_0,\dots,x_{n-1}$ known, x_n is chosen to maximize the same quantity (12.14), or equivalently, to maximize

$$\prod_{j=0}^{n-1}|x_j-x_n|. \qquad (12.15)$$

All three of these families of points can be shown, under reasonable assumptions, to converge to the equilibrium measure as $n \to \infty$, and all work well in practice for interpolation. A result showing near-optimality of Leja points for interpolation on general sets in the complex plane can be found in [Taylor & Totik 2010].

In Chapter 8 we proved a precise theorem (Theorem 8.2): if f is analytic and bounded by M in the Bernstein ellipse E_ρ, then $\|f - p_n\| \le 4M\rho^{-n}/(\rho - 1)$, where $p_n \in \mathcal{P}_n$ is the interpolant in $n+1$ Chebyshev points. The proof made use of the Chebyshev expansion of f and the aliasing properties of Chebyshev polynomials at Chebyshev points. By the methods of potential theory and the Hermite integral formula discussed in this chapter one can derive a much more general theorem to similar effect. For any set of $n+1$ nodes in $[-1,1]$, let $\ell \in \mathcal{P}_{n+1}$ be the node polynomial (5.4), and let $M_n = \sup_{x\in[-1,1]} |\ell(x)|$. A sequence of grids of $1, 2, 3, \dots$ interpolation nodes is said to be *uniformly distributed* on $[-1,1]$ if it satisfies

$$\lim_{n\to\infty} M_n^{1/n} = \frac{1}{2}.$$

(On a general set E, the number $1/2$ becomes the capacity.)

Theorem 12.1. Interpolation in uniformly distributed points. *Given $f \in C([-1,1])$, let ρ $(1 \le \rho \le \infty)$ be the parameter of the largest Bernstein ellipse E_ρ to which f can be analytically continued, and let $\{p_n\}$ be the interpolants to f in any sequence of grids $\{x_n\}$ of $n+1$ points in $[-1,1]$ uniformly distributed as defined above. Then the errors satisfy*

$$\lim_{n\to\infty} \|f - p_n\|^{1/n} = \rho^{-1}. \tag{12.16}$$

Proof. See Chapter 2 of [Gaier 1987]. ∎

A set of polynomials satisfying (12.16) is said to be *maximally convergent.* Examples of such polynomials are interpolants through most systems of roots or extrema of Legendre, Chebyshev, or Gauss–Jacobi points; the convergence rates of such systems differ only at the margins, in possible algebraic factors like n or $\log n$.

Summary of Chapter 12. *Polynomial interpolants to analytic functions on $[-1,1]$ converge geometrically if the grids are asymptotically distributed according to the Chebyshev distribution.*

Exercise 12.1. Fekete points in an interval. It can be shown that the equilibrium configuration for $n+1$ points in $[-1,1]$ consists of the roots of $(x^2-1)P_{n-1}^{(1,1)}(x)$, where $P_{n-1}^{(1,1)}$ is the degree $n-1$ Jacobi polynomial with parameters $(1,1)$ [Stieltjes 1885] (see Chapter 17). (An equivalent statement is that the points lie at the local extrema in $[-1,1]$ of the Legendre polynomial of degree $n+1$.) Thus $(x^2-1)P_{n-1}^{(1,1)}(x)$ is the degree $n-1$ Fekete polynomial in $[-1,1]$. Verify numerically using the Chebfun `jacpts` command that in the case $n=10$, the net forces on the 9 interior points are zero.

Exercise 12.2. Capacity of an ellipse. Let E be an ellipse in the complex plane of semiaxis lengths a and b. Show that $c(E)=(a+b)/2$.

Exercise 12.3. Leja points and capacity. Let E be the "half-moon" set consisting of the boundary of the right half of the unit disk. Write a code to compute a sequence of 100 Leja points for this set. To keep things simple, approximate the boundary by a discrete set of 1000 points. What approximation of the capacity of E do your points provide? (The exact answer is $4/3^{3/2}$, as discussed with other examples and algorithms in [Ransford 2010].)

Chapter 13

Equispaced points, Runge phenomenon

`ATAPformats`

So far in this book, we have considered three good methods for approximating functions by polynomials: Chebyshev interpolation, Chebyshev projection, and best approximation. Now we shall look at a catastrophically bad method!—interpolation in equally spaced points. This procedure is so unreliable that for generations, it has tainted people's views of the whole subject of polynomial interpolation. The mathematical tools we will need to understand what is going on are the Hermite integral formula and potential theory, as discussed in the last two chapters.

As mentioned in Chapter 5, polynomial interpolation was an established tool by the 19th century. The question of whether or not polynomial interpolants would converge to an underlying function as $n \to \infty$ was not given much attention. Presumably many mathematicians would have supposed that if the function was analytic, the answer would be yes. In 1884 and 1896, Méray published a pair of papers in which he identified the fact that certain interpolation schemes do not converge [Méray 1884 & 1896]. In the first paper he writes,

It is rather astonishing that practical applications have not yet turned up any cases in which the interpolation is illusory.[1]

Méray's derivations had the key idea of making use of the Hermite integral formula. However, the examples he devised were rather contrived, and his

[1] "Il est assez étonnant que les hasards de la pratique n'aient encore fait connaître aucun cas dans lequel l'interpolation soit illusoire." By illusory, Méray means nonconvergent.

idiosyncratically written papers had little impact. It was Runge in 1901 who made the possibility of divergence famous by showing that divergence of interpolants occurs in general even for equispaced points in an real interval and evaluation points in the interior of that interval [Runge 1901].

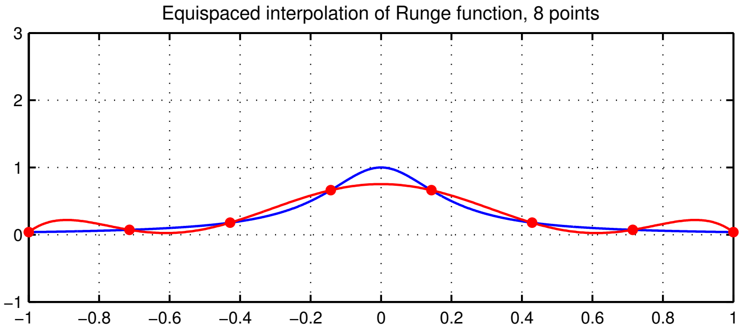
Runge illustrated his discovery with an example that has become known as the *Runge function*: $1/(1+x^2)$ on $[-5,5]$, or equivalently, $1/(1+25x^2)$ on $[-1,1]$:

```
x = chebfun('x'); f = 1./(1+25*x.^2);
```

We already know from Chapter 8 that there is nothing wrong with this function for polynomial interpolation in Chebyshev points: f is analytic, and the polynomial interpolants converge geometrically. Now, however, let us follow Runge and look at interpolants in equally spaced points, which we can compute using the Chebfun overload of Matlab's `interp1` command.

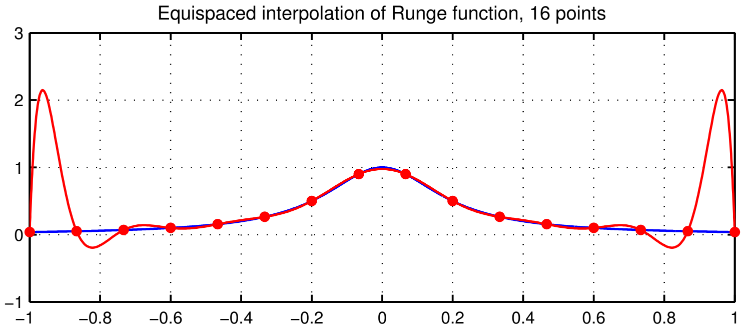
Here is what we get with 8 points:

```
s = linspace(-1,1,8); p = interp1(s,f,domain(-1,1));
hold off, plot(f), hold on, plot(p,'r'), grid on
plot(s,p(s),'.r'), axis([-1 1 -1 3]), FS = 'fontsize';
title('Equispaced interpolation of Runge function, 8 points',FS,9)
```



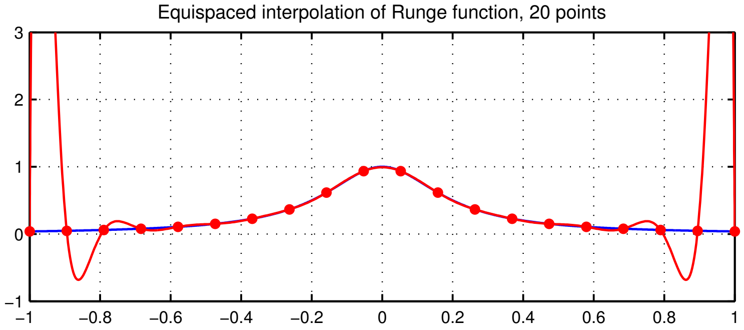
Here is the result for 16 points:

```
s = linspace(-1,1,16); p = interp1(s,f,domain(-1,1));
hold off, plot(f), hold on, plot(p,'r'), grid on
plot(s,p(s),'.r'), axis([-1 1 -1 3])
title('Equispaced interpolation of Runge function, 16 points',FS,9)
```



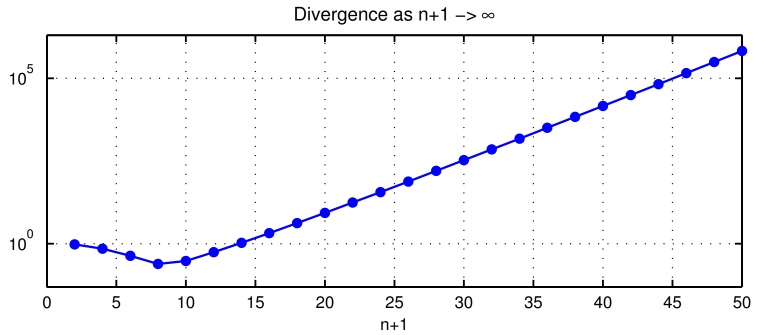
Is this going to converge as $n \to \infty$? Things look pretty good in the middle, but not so good at the edges. Here is the result for 20 points:

```
s = linspace(-1,1,20); p = interp1(s,f,domain(-1,1));
hold off, plot(f), hold on, plot(p,'r'), grid on
plot(s,p(s),'.r'), axis([-1 1 -1 3])
title('Equispaced interpolation of Runge function, 20 points',FS,9)
```



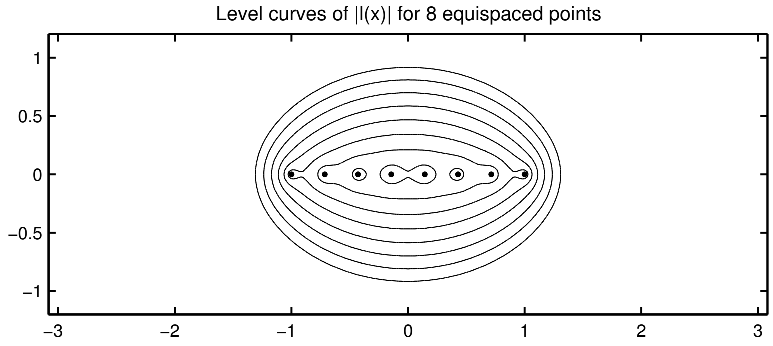
What is happening is exponential convergence in the middle of the interval but exponential divergence near the ends. The next figure shows the maximum error over $[-1, 1]$ as a function of the number of points. We get a hint of convergence at first, but after $n = 10$, things just get worse. Note the log scale.

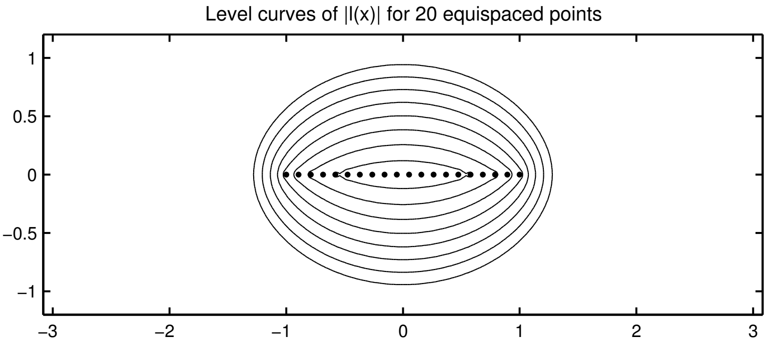
```
ee = []; nn = 2:2:50;
for np = nn
  s = linspace(-1,1,np); p = interp1(s,f,domain(-1,1));
  ee = [ee norm(f-p,inf)];
end
hold off, semilogy(nn,ee,'.-'), grid on, axis([0 50 5e-2 2e6])
xlabel n+1, title('Divergence as n+1 -> \infty',FS,9)
```



By now the reader may have suspected that what is going wrong here can be understood by looking at potentials, as in the last two chapters. Here is an adaptation of a code segment from Chapter 11 to plot equipotential curves for $n + 1 = 8$ and 20.

```
d = domain(-1.5,1.5);
xgrid = -1.5:.02:1.5; ygrid = -1:.02:1;
[xx,yy] = meshgrid(xgrid,ygrid); zz = xx+1i*yy;
for np = [8 20]
  xj = linspace(-1,1,np);
  ell = poly(xj,d);
  hold off, plot(xj,ell(xj),'.k','markersize',8)
  hold on, ylim([-1.2 1.2]), axis equal
  ellzz = ell(zz);
  levels = ((1.25:.25:3)/exp(1)).^np;
  contour(xx,yy,abs(ellzz),levels,'k')
  title(['Level curves of |l(x)| for '...
    int2str(np) ' equispaced points'],FS,9)
  snapnow
end
```





What we see here is that $[-1,1]$ is very far from being a level curve for equispaced interpolation points. From the last two chapters, we expect serious consequences of this irregularity. In the second plot just shown, for example, it is the fourth curve (from inside out) that approximately touches the endpoints ±1. For Theorem 11.1 to be of any use in such a landscape, f will have to be analytic in a region larger than the "football" enclosed by that curve. Analyticity on $[-1,1]$ is not enough for convergence; we will need analyticity in a much bigger region of the complex plane. This is what Runge discovered in 1901.

Following the method of the last chapter, we now consider the limit $n \to \infty$, where the distribution of interpolation points approaches the constant measure (12.9),

$$\mu(\tau) = \frac{1}{2}. \tag{13.1}$$

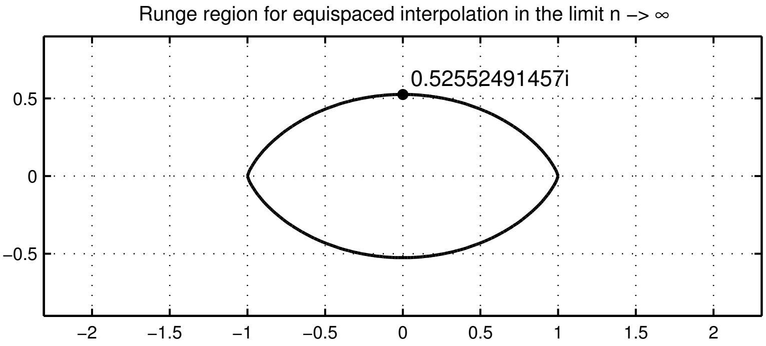
The potential (12.7) associated with this measure is

$$u(s) = -1 + \frac{1}{2}\mathrm{Re}\left[(s+1)\log(s+1) - (s-1)\log(s-1)\right]. \tag{13.2}$$

Here is a code that plots just one level curve of this potential, the one passing through ±1, where the value of the potential is $-1+\log 2$.

```
x1 = -1.65:.02:1.65; y1 = -0.7:.02:0.7;
[xx,yy] = meshgrid(x1,y1); ss = xx+1i*yy;
u = @(s) -1 + 0.5*real((s+1).*log(s+1)-(s-1).*log(s-1));
hold off
contour(x1,y1,u(ss),(-1+log(2))*[1 1],'k','linewidth',1.4)
set(gca,'xtick',-2:.5:2,'ytick',-.5:.5:.5), grid on
ylim([-.9 .9]), axis equal
hold on, plot(.5255i,'.k')
text(0.05,.63,'0.52552491457i')
```

```
title(['Runge region for equispaced interpolation ' ...
      'in the limit n -> \infty'],FS,9)
```



For the interpolants to a function f in equispaced nodes to converge as $n \to \infty$ for all $x \in [-1,1]$, f must be analytic, not just on $[-1,1]$, but throughout this *Runge region,* which crosses the real axis at ± 1 and the imaginary axis at $\pm 0.52552491457\ldots i$. Runge reports this number correctly to 4 digits, and writes

The curve has somewhat the shape of an ellipse. At the ends of the long axis, however, our curve is more pointed than an ellipse.[2]

Here are the values of (13.2) at the endpoints and the middle:

$$u(\pm 1) = -1 + \log 2, \quad u(0) = -1,$$

and thus

$$\exp(u(\pm 1)) = \frac{2}{e}, \quad \exp(u(0)) = \frac{1}{e}.$$

These numbers indicate that in the limit $n \to \infty$, the endpoints of an equispaced grid in $[-1,1]$ lie at an average distance $2/e$ from the other grid points, in the geometric mean sense, whereas the midpoint lies at an average distance of just $1/e$. As emphasized in the last chapter, notably in equation (12.6), the effect of such a discrepancy grows exponentially with n.

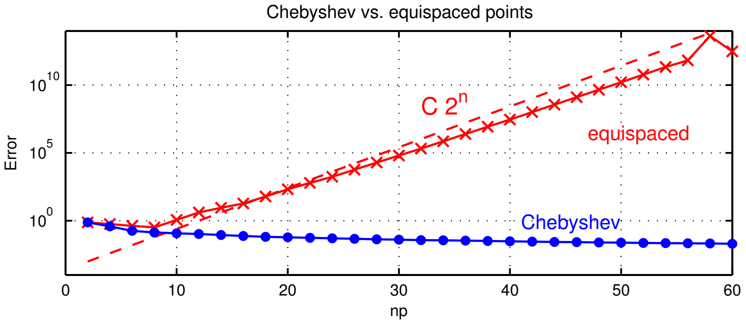
Here are some examples. Equispaced interpolation will converge throughout $[-1,1]$ for $f(x) = \exp(-x^2)$, which is analytic everywhere, and for $f(x) = (1+x^2)^{-1}$, which has poles at $\pm i$. On the other hand it will not converge for $f(x) = (1+16x^2)^{-1}$, which has poles at $\pm i/4$. It will converge slowly for

[2] "Die Kurve. . . hat etwa die Gestalt einer Ellipse. . . . An den Enden der grossen Achse ist unsere Kurve aber spitzer als eine Ellipse."

$f(x) = (1 + (x/0.53)^2)^{-1}$, and diverge slowly for $f(x) = (1 + (x/0.52)^2)^{-1}$ (Exercise 13.3).

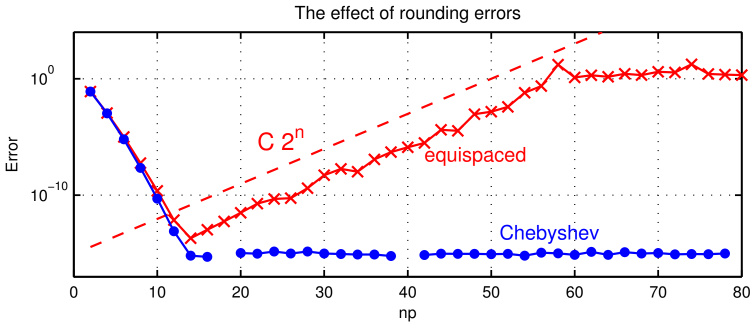
The worst-case rate of divergence is 2^n, and this rate will always appear if f is not analytic on $[-1, 1]$. To be precise, for such a function the errors will be of size $O(2^n)$ as $n \to \infty$ but not of size $O(C^n)$ for any $C < 2$. Here, for example, we take f to be a hat function, with just one derivative of bounded variation. The dots show errors in Chebyshev interpolation, converging at the rate $O(n^{-1})$ in keeping with Theorem 7.2, and the crosses show errors in equispaced interpolation, with a dashed line parallel to 2^n for comparison.

```
f = max(0,1-2*abs(x));
eequi = []; echeb = []; nn = 2:2:60;
for n = nn
  s = linspace(-1,1,n+1);
  pequi = interp1(s,f,domain(-1,1)); eequi = [eequi norm(f-pequi,inf)];
  pcheb = chebfun(f,n+1); echeb = [echeb norm(f-pcheb,inf)];
end
hold off, semilogy(nn,2.^(nn-12),'--r')
hold on, axis([0 60 1e-4 1e14]), grid on
semilogy(nn,eequi,'x-r','markersize',8), semilogy(nn,echeb,'.-b')
text(47,3e6,'equispaced','color','r')
text(41,0.8,'Chebyshev','color','b')
text(32,4e8,'C 2^n','fontsize',12,'color','r')
xlabel np, ylabel Error, title('Chebyshev vs. equispaced points',FS,9)
```



All of the above remarks about equispaced interpolation concern the ideal mathematics of the problem. On a computer in floating point arithmetic, however, a further complication arises: even if convergence ought to take place in theory, rounding errors will be amplified by $O(2^n)$, causing divergence in practice. Here, for example, are the errors in equispaced and Chebyshev interpolation of the entire function $\exp(x)$:

```
f = exp(x);
eequi = []; echeb = []; nn = 2:2:80;
for n = nn
  s = linspace(-1,1,n+1);
  pequi = interp1(s,f,domain(-1,1)); eequi = [eequi norm(f-pequi,inf)];
  pcheb = chebfun(f,n+1); echeb = [echeb norm(f-pcheb,inf)];
end
hold off, semilogy(nn,2.^(nn-50),'--r')
hold on, axis([0 80 1e-17 1e4]), grid on
semilogy(nn,eequi,'x-r','markersize',8), semilogy(nn,echeb,'.-b')
text(22,6e-6,'C 2^n','fontsize',12,'color','r')
text(42,3e-7,'equispaced','color','r')
text(51,6e-14,'Chebyshev','color','b')
xlabel np, ylabel Error, title('The effect of rounding errors',FS,9)
```



In exact arithmetic we would see convergence for both sets of points, but on the computer the divergence for equispaced points sets in early. The rate is cleanly $O(2^n)$ until we reach $O(1)$. Notice that the line of crosses, if extended backward to $n = 0$, meets the y axis at approximately 10^{-18}, i.e., a digit or two below machine precision.

The 2^n divergence of equispaced polynomial interpolants is a fascinating subject, and we must remind ourselves that one should only go into so much detail in analyzing a method that should never be used! But perhaps we should qualify that "never" in one respect. As Runge himself emphasized, though interpolants in equispaced points do not converge on the whole interval of interpolation, they may still do very well near the middle. In the numerical solution of differential equations, for example, higher order centered difference formulas are successfully used based on 5 or 7 equally spaced grid points. A less happy example would be Newton–Cotes quadrature formulas, based on polynomial interpolation in equally spaced points, where the $O(2^n)$ effect is unavoidable and causes serious problems for larger n and divergence as

$n \to \infty$, as first proved by Pólya [1933]. We shall discuss quadrature in Chapter 19.

We close this chapter with an observation that highlights the fundamental nature of the Runge phenomenon and its associated mathematics. Suppose you want to persuade somebody that it is important to know something about complex variables, even for dealing with real functions. I still remember the argument my calculus teacher explained to me: to understand why the Taylor series for $1/(1+x^2)$ only converges for $-1 < x < 1$, you need to know that Taylor series converge within disks in the complex plane, and this function has poles at $\pm i$.

Runge's observation is precisely a generalization of this fact to interpolation points equispaced in an interval rather than all at $x = 0$. The convergence or divergence of polynomial interpolants to a function f again depends on whether f is analytic in a certain region; the change is that the region is now not a disk, but elongated. Even the phenomenon of divergence in floating-point arithmetic for functions whose interpolants "ought" to converge is a generalization of familiar facts from real arithmetic. Just try to evaluate $\exp(x)$ on a computer for $x = -20$ using the Taylor series!

Summary of Chapter 13. *Polynomial interpolation in equispaced points is exponentially ill-conditioned: the interpolant p_n may have oscillations near the edge of the interval nearly 2^n times larger than the function f being interpolated, even if f is analytic. In particular, even if f is analytic and the interpolant is computed exactly without rounding errors, p_n need not converge to f as $n \to \infty$.*

Exercise 13.1. Three examples. Draw plots comparing accuracy of equispaced and Chebyshev interpolants as functions of n for $\exp(x^2)$, $\exp(-x^2)$, $\exp(-1/x^2)$.

Exercise 13.2. Computing geometric means in Chebfun. (a) What output is produced by the program below? (b) Why?

```
x = chebfun('x',[0 1]);
f = chebfun(@(y) prod(abs(x-1i*y)),[0.1 1],'vectorize');
roots(f-2/exp(1))
```

Exercise 13.3. Borderline convergence. The claim was made in the text that equispaced polynomial interpolants on $[-1,1]$ converge for $f(x) = (1+(x/0.53)^2)^{-1}$ and diverge for $f(x) = (1+(x/0.52)^2)^{-1}$. Can you observe this difference numerically? Run appropriate experiments and discuss the results.

Exercise 13.4. Approaching the sinc function. The sinc function is defined for all x by $S(x) = \sin(\pi x)/(\pi x)$ (and $S(0) = 1$). For any $n \geq 1$, an approximation to S is given by

$$S_n = \prod_{k=1}^{n} (1 - x^2/k^2).$$

Construct S_{20} in Chebfun on the interval $[-20, 20]$ and compare it with S. On what interval around $x = 0$ do you find $|S_{20}(x) - S(x)| < 0.1$? (It can be shown that for every x, $\lim_{n\to\infty} S_n(x) = S(x)$.)

Exercise 13.5. Barycentric weights and ill-conditioning. (a) Suppose a function is interpolated by a polynomial in $n+1$ equispaced points in $[-1, 1]$, with n even. From the result of Exercise 5.6, derive a formula for the ratio of the barycentric weights at the midpoint $x = 0$ and the endpoint $x = 1$. (b) With reference to the barycentric formula (5.11), explain what this implies about sensitivity of these polynomial interpolants to perturbations in the data at $x = 0$.

Chapter 14

Discussion of high-order interpolation

As mentioned at various points in this book, high-order polynomial interpolation has a bad reputation. For equispaced points this is entirely appropriate, as shown in the last chapter, but for Chebyshev points it is entirely inappropriate. Here are some illustrative quotes from fifty years of numerical analysis textbooks, which we present anonymously.

We cannot rely on a polynomial to be a good approximation if exact matching at the sample points is the criterion used to select the polynomial. The explanation of this phenomenon is, of course, that the derivatives grow too rapidly. (1962)

However, for certain functions the approximate representation of $f(x)$ by a single polynomial throughout the interval is not satisfactory. (1972)

But there are many functions which are not at all suited for approximation by a single polynomial in the entire interval which is of interest. (1974)

Polynomial interpolation has drawbacks in addition to those of global convergence. The determination and evaluation of interpolating polynomials of high degree can be too time-consuming for certain applications. Polynomials of high degree can also lead to difficult problems associated with roundoff error. (1977)

We end this section with two brief warnings, one against trusting the interpolating polynomial outside [the interval] and one against expecting too much of polynomial interpolation inside [the interval]. (1980)

Although Lagrangian interpolation is sometimes useful in theoretical investigations, it is rarely used in practical computations. (1985)

Polynomial interpolants rarely converge to a general continuous function. Polynomial interpolation is a bad idea. (1989)

While theoretically important, Lagrange's formula is, in general, not as suitable for actual calculations as some other methods to be described below, particularly for large numbers n of support points. (1993)

Unfortunately, there are functions for which interpolation at the Chebyshev points fails to converge. Moreoever, better approximations of functions like $1/(1+x^2)$ can be obtained by other interpolants—e.g., cubic splines. (1996)

In this section we consider examples which warn us of the limitations of using interpolation polynomials as approximations to functions. (1996)

Increasing the number of interpolation points, i.e., increasing the degree of the polynomials, does not always lead to an improvement in the approximation. The spline interpolation *that we will study in this section remedies this deficiency.* (1998)

The surprising state of affairs is that for most continuous functions, the quantity $\|f-p_n\|_\infty$ will not coverge to 0. (2002)

Because its derivative has $n-1$ zeros, a polynomial of degree n has $n-1$ extreme or inflection points. Thus, simply put, a high-degree polynomial necessarily has many "wiggles," which may bear no relation to the data to be fit. (2002)

By their very nature, polynomials of a very high degree do not constitute reasonable models for real-life phenomena, from the approximation and from the handling point-of-view. (2004)

The oscillatory nature of high degree polynomials, and the property that a fluctuation over a small portion of the interval can induce large fluctuations over the entire range, restricts their use. (2005)

In addition to the inherent instability of Lagrange interpolation for large n, there are also classes of functions that are not suitable for approximation by certain types of interpolation. There is a celebrated example of Runge.... (2011)

A great deal of confusion underlies remarks like these. Some of them are literally correct, but they are all misleading. In fact, polynomial interpolants in Chebyshev points are problem-free when evaluated by the the barycentric interpolation formula. They have the same behavior as discrete Fourier series for periodic functions, whose reliability nobody worries about. The introduction of splines is a red herring: the true advantage of splines, as mentioned in Chapter 9, is not that they converge where polynomials fail to do so, but that they are more easily adapted to irregular point distributions and more localized, giving errors that decay exponentially away from singularities rather than just algebraically.

It is interesting to speculate as to how the distrust of high-degree polynomials became so firmly established. I think the crucial circumstance is that not one but several combined problems affect certain computations with polynomials, a situation complex enough to have obscured the truth from easy elucidation. If working with polynomials had been the central task of scientific computing, the truth would have been worked out nonetheless, but over the years there were always bigger problems pressing upon the attention of numerical analysts, like matrix computations and differential equations. Polynomial computations were always a sideline.

At the most fundamental level there are the two issues of conditioning and stability: both crucial, and not the same. See [Trefethen & Bau 1997] for a general discussion of conditioning and stability.

(1) *Conditioning of the problem.* The interpolation points must be properly spaced (e.g., Chebyshev or Legendre) for the interpolation problem to be well-conditioned. This means that the interpolant should depend not too sensitively on the data. The Runge phenomenon for equally spaced points is the well-known consequence of extreme ill-conditioning, with sensitivities of order 2^n. The next chapter makes such statements precise through the use of Lebesgue constants.

(2) *Stability of the algorithm.* The interpolation algorithm must be stable (e.g., the barycentric interpolation formula) for a computation to be accurate, even when the problem is well-conditioned. This means that in the presence of rounding errors, the computed solution should be close to an exact solution for some interpolation data close to the exact data. In particular, the best-known algorithm of all, namely solving a Vandermonde linear system of equations to find the coefficients of the interpolant expressed as a linear combination of monomials, is explosively unstable, relying on a matrix

whose condition number grows exponentially with the dimension (Exercise 5.2).

These facts would be enough to explain a good deal of confusion, but another consideration has muddied the water further, namely crosstalk with the notoriously troublesome problem of finding roots of a polynomial from its coefficients (to be discussed in Chapter 18). The difficulties of polynomial rootfinding were widely publicized by Wilkinson beginning in the 1950s, who later wrote an article called the "The perfidious polynomial" that won the Chauvenet Prize of the Mathematical Association of America [Wilkinson 1984]. Undoubtedly this negative publicity further discouraged people from the use of polynomials, even though interpolation and rootfinding are different problems. They are related, with related widespread misconceptions about accuracy: just as interpolation on an interval is trouble-free for a stable algorithm based on Chebyshev points, rootfinding on an interval is trouble-free for a stable algorithm based on expansions in Chebyshev polynomials (Chapter 18). But very few textbooks tell readers these facts.

Summary of Chapter 14. *Generations of numerical analysis textbooks have warned readers that polynomial interpolation is dangerous. In fact, if the interpolation points are clustered and a stable algorithm is used, it is bulletproof.*

Exercise 14.1. Convergence as $n \to \infty$. The 1998 quote asserts that increasing n "does not always lead to an improvement". Based on the theorems of this book, for interpolation in Chebyshev points, for which functions f do we know that increasing n must lead to an improvement?

Exercise 14.2. Too many wiggles. Using Chebfun's `roots(f,'all')` option, plot all the roots in the complex plane of the derivative of the chebfun corresponding to $f(x) = \exp(x)\tanh(2x-1)$ on $[-1,1]$. What is the error in the argument in the second 2002 quote used to show that "a high-degree polynomial necessarily has many wiggles"?

Exercise 14.3. Your own textbook. Find a textbook of numerical analysis and examine its treatment of polynomial interpolation. (a) What does it say about behavior for large n? If it asserts that this behavior is problematic, is this conclusion based on the assumption of equally spaced points, and does the text make this clear? (b) Does it mention interpolation in Chebyshev points? Does it state that such interpolants converge exponentially for analytic functions? (c) Does it mention the barycentric formula? (d) Does it claim that one should use a Newton rather than a Lagrange interpolation formula for numerical work? (This is a myth.)

Exercise 14.4. Spline interpolants. (a) Use Chebfun's `spline` command to interpolate $f(x) = 1/(1+25x^2)$ by a cubic spline in $n+1$ equally spaced points on $[-1,1]$. Compare the ∞-norm error as $n \to \infty$ with that of polynomial interpolation in Chebyshev points. (b) Same problem for $f(x) = |x + 1/\pi|$. (c) Same problem for $f(x) = |x + 1/\pi|$, but measuring the error by the ∞-norm over the interval $[0,1]$.

Chapter 15

Lebesgue constants

`ATAPformats`

There is a well developed theory that quantifies the convergence or divergence of polynomial interpolants. A key notion is that of the *Lebesgue constant,* Λ, for interpolation in a given set of points. The Lebesgue constant is the ∞-norm of the linear mapping from data to interpolant:

$$\Lambda = \sup_f \frac{\|p\|}{\|f\|}, \tag{15.1}$$

where $\|\cdot\|$ denotes the ∞-norm in $C([-1,1])$. In words, if you have data values on an $(n+1)$-point grid, and the data come from sampling a function that is no greater than 1 in absolute value, what is the largest possible value of the interpolant p somewhere in $[-1,1]$?

In the plots of Chapter 13 for interpolation of Runge's function, for example, we saw that the interpolants grew much bigger than the data. Thus the Lebesgue constants must be large for equispaced interpolation. For example, for $n = 50$, the data are bounded by 1 for all n, yet the interpolant is bigger than 10^5. Thus the Lebesgue constant for interpolation in 50 equispaced points must be greater than 10^5. (In fact, it is about 4.2×10^{12}.)

From the basic Lagrange formula (5.1) for polynomial interpolation,

$$p(x) = \sum_{j=0}^{n} f_j \ell_j(x), \tag{15.2}$$

we can get a formula for Λ in terms of the Lagrange polynomials $\{\ell_j\}$. At any point $x \in [-1,1]$, the maximum possible value of $|p(x)|$ for grid data bounded by 1 in absolute value will be the number $\lambda(x)$ obtained if each data value is ± 1, with signs chosen to make all the signs at x coincide:

$$\lambda(x) = \sum_{j=0}^{n} |\ell_j(x)|. \tag{15.3}$$

This sum of absolute values is known as the *Lebesgue function* for the given grid, and the Lebesgue constant is equal to its maximum value,

$$\Lambda = \sup_{x\in[-1,1]} \lambda(x). \tag{15.4}$$

The reason Lebesgue constants are interesting is that interpolants are guaranteed to be good if and only if the Lebesgue constants are small. We can make this statement precise as follows. Let Λ be the Lebesgue constant for interpolation in a certain set of points. Without loss of generality (since the interpolation process is linear), suppose the largest absolute value of the samples is 1. If p is the interpolant in these points to a function f, we know that $\|p\|$ might be as great as Λ; yet $\|f\|$ might be as small as 1. Thus $\|f-p\|$ might be as great as $\Lambda - 1$, showing that a large Lebesgue constant rigorously implies the possibility of a large interpolation error.

Conversely, let p^* be the best degree n polynomial approximation to f in the ∞-norm. If p is the polynomial interpolant to f in the given points, then $p - p^*$ is the polynomial interpolant to $f - p^*$. By the definition of the Lebesgue constant, $\|p - p^*\|$ is no greater than $\Lambda\|f - p^*\|$. Since $f - p = (f - p^*) - (p - p^*)$, this implies that $\|f - p\|$ is no greater than $(\Lambda + 1)\|f - p^*\|$. Thus a small Lebesgue constant implies that interpolation will be close to best.

Actually, the discussion of the last two paragraphs is not limited to interpolation. What is really in play here is any approximation process that is a *linear projection* from $C([-1,1])$ to $\mathcal{P}_n$, of which Chebyshev projection (truncation of the Chebyshev series) would be an example as well as interpolation. Suppose we let L denote an operator that maps functions $f \in C([-1,1])$ to approximations by polynomials $p \in \mathcal{P}_n$. For L to be linear means that $L(f_1 + f_2) = Lf_1 + Lf_2$ for any $f_1, f_2 \in C([-1,1])$ and $L(\alpha f) = \alpha Lf$ for any scalar α, and for L to be a projection means that if $p \in \mathcal{P}_n$, then $Lp = p$. By the argument above we have established the following result applicable to any linear projection.

Theorem 15.1. Near-best approximation and Lebesgue constants. *Let Λ be the Lebesgue constant for a linear projection L of $C([-1,1])$ onto*

$\mathcal{P}_n$. Let f be a function in $C([-1,1])$, $p = Lf$ the corresponding polynomial approximant to f, and p^* the best approximation. Then

$$\|f - p\| \le (\Lambda + 1)\|f - p^*\|. \qquad (15.5)$$

Proof. Given in the paragraphs above. ∎

So it all comes down to the question, how big is Λ? According to the theorem of Faber mentioned in Chapter 6 [Faber 1914], no sets of interpolation points can lead to convergence for all $f \in C([-1,1])$, so it follows from Theorems 6.1 and 15.1 that

$$\limsup_{n\to\infty} \Lambda_n = \infty \qquad (15.6)$$

for interpolation in any sets of points (Excercise 15.12). However, for well chosen sets of points, the growth of Λ_n as $n \to \infty$ may be exceedingly slow. Chebyshev points are nearly optimal, whereas equispaced points are very bad.

The following theorem summarizes a great deal of knowledge accumulated over the past century about interpolation processes. At the end of the chapter an analogous theorem is stated for Chebyshev projection. As always in this book, by "Chebyshev points" we mean Chebyshev points of the second kind, defined by (2.2).

Theorem 15.2. Lebesgue constants for polynomial interpolation. *The Lebesgue constants Λ_n for degree $n \ge 0$ polynomial interpolation in any set of $n+1$ distinct points in $[-1,1]$ satisfy*

$$\Lambda_n \ge \frac{2}{\pi}\log(n+1) + 0.52125\ldots; \qquad (15.7)$$

the number $0.52125\ldots$ is $(2/\pi)(\gamma + \log(4/\pi))$, where $\gamma \approx 0.577$ is Euler's constant. For Chebyshev points, they satisfy

$$\Lambda_n \le \frac{2}{\pi}\log(n+1) + 1 \quad \textit{and} \quad \Lambda_n \sim \frac{2}{\pi}\log n, \quad n \to \infty. \qquad (15.8a, b)$$

For equispaced points they satisfy

$$\Lambda_n > \frac{2^{n-2}}{n^2} \quad \textit{and} \quad \Lambda_n \sim \frac{2^{n+1}}{e\, n \log n}, \quad n \to \infty, \qquad (15.9a, b)$$

with the inequality (15.9a) applying for $n \ge 1$.

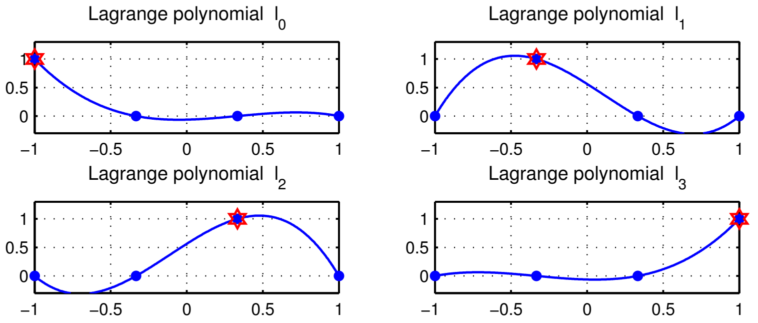
Proof. The fact that Lebesgue constants for polynomial interpolation always grow at least logarithmically goes back to Bernstein [1912b], Jackson [1913], and Faber [1914]. Bernstein knew that $(2/\pi)\log n$ was the controlling asymptotic factor for interpolation in an interval, and the proof of (15.7) in this sharp form is due to Erdős [1961], who got a constant C, and Brutman [1978], who improved the constant to $0.52125\ldots$. Equation (15.8a) is a consequence of Theorem 4 of [Ehlich & Zeller 1966]; see also [Brutman 1997] and [McCabe & Phillips 1973]. Equation (15.8b) follows from this together with Erdős's result. (Bernstein [1919] did the essential work, deriving this asymptotic result for Chebyshev points of the first kind, i.e., zeros rather than extrema of Chebyshev polynomials—see Exercise 15.2.) Equation (15.9b) is due to Turetskii [1940] and independently Schönhage [1961], and for (15.9a) and a discussion of related work, see [Trefethen & Weideman 1991]. ▮

Equations (15.8) show that Lebesgue constants for Chebyshev points grow more slowly than any polynomial: for many practical purposes they might as well be 1. It is interesting to relate this bound to the interpolant through 100 random data points plotted at the end of Chapter 2. The Lebesgue constant is the maximum amplitude this curve could possibly have attained, if the data had been as bad as possible. For 100 points this number is about 3.94. In the plot we see that random data have in fact come nowhere near even this modest limit.

On the other hand, equations (15.9) show that Lebesgue constants for equispaced points grow faster than any polynomial: for many practical purposes, unless n is very small, they might as well be ∞.

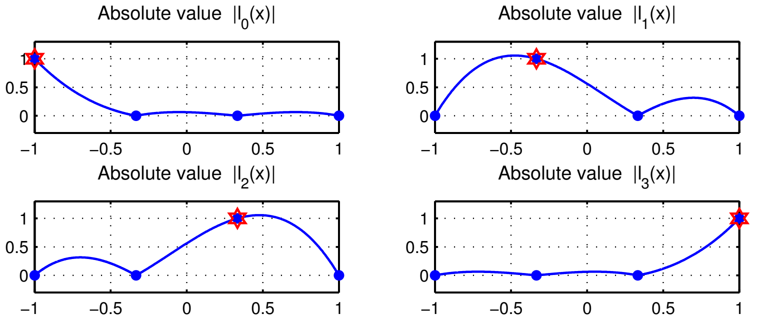
Taking advantage again of the `interp1` command, as in Chapter 13, we can use Chebfun as a laboratory in which to see how such widely different Lebesgue constants emerge. Consider for example the case of four equally spaced points. Here are plots of the four Lagrange polynomials $\{\ell_j\}$. In Chapter 9 we have already seen plots of Lagrange polynomials, but on a grid of 20 Chebyshev points instead of 4 equispaced points.

```
npts = 4; clear p
d = domain(-1,1); s = linspace(-1,1,4);
for k = 1:npts
  subplot(2,2,k)
  y = [zeros(1,k-1) 1 zeros(1,npts-k)];
  p{k} = interp1(s,y,d);
  hold off, plot(p{k}), grid on
  hold on, plot(s,p{k}(s),'.'), FS = 'fontsize';
  plot(s(k),p{k}(s(k)),'hr','markersize',9), ylim([-.3 1.3])
  title(['Lagrange polynomial  l_' int2str(k-1)],FS,9)
end
```



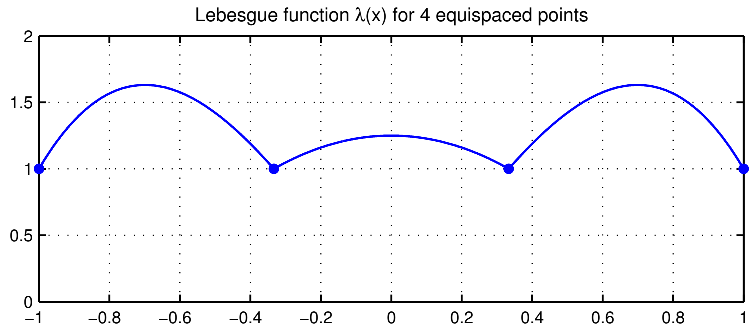
By taking the absolute values of these curves, we see the largest possible effect at each point in $[-1,1]$ of data that is nonzero at just one point of the grid:

```
for k = 1:npts
  subplot(2,2,k), absp = abs(p{k});
  hold off, plot(absp), grid on, hold on, plot(s,absp(s),'.')
  plot(s(k),absp(s(k)),'hr','markersize',9), ylim([-.3 1.3])
  title(['Absolute value  |l_' int2str(k-1) '(x)|'],FS,9)
end
```



Now let us add up these absolute values as in (15.3):

```
x = chebfun('x'); L = 0*x;
for k = 1:npts, L = L + abs(p{k}); end
clf, plot(L), grid on, hold on, plot(s,L(s),'.')
axis([-1 1 0 2])
title('Lebesgue function \lambda(x) for 4 equispaced points',FS,9)
```



This is the Lebesgue function $\lambda(x)$, a piecewise polynomial, telling us the largest possible effect at each point $x \in [-1,1]$ of interpolating data of norm 1. The Lebesgue constant (15.4) is the height of the curve:

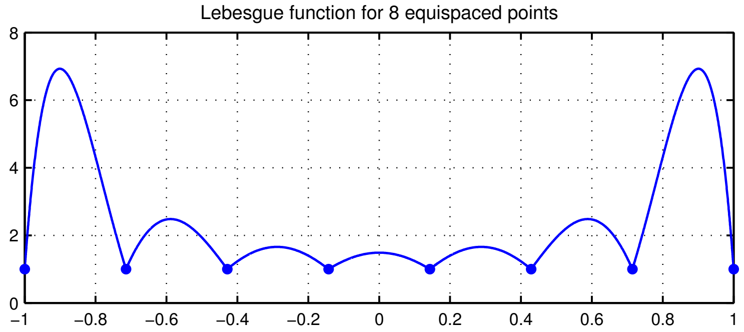
```
Lconst = norm(L,inf)
```

```
Lconst =
   1.631130309440899
```

A code `lebesgue` for automating the above computation (actually based on a more efficient method) is included in Chebfun, and it optionally returns the Lebesgue constant as well as the Lebesgue function. Here are the results for 8 equispaced points:

```
s = linspace(-1,1,8); [L,Lconst] = lebesgue(s);
hold off, plot(L), grid on, hold on, plot(s,L(s),'.')
title('Lebesgue function for 8 equispaced points',FS,9), Lconst
axis([-1 1 0 8])
```

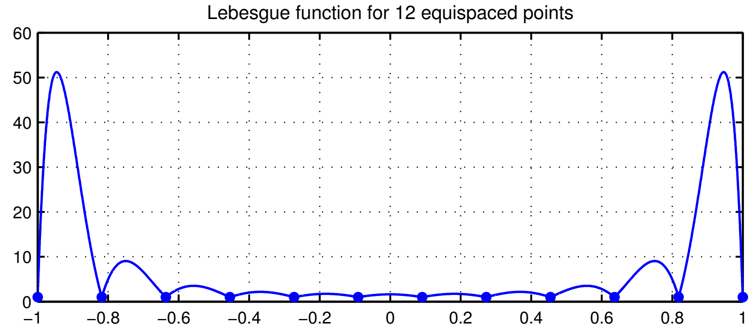
```
Lconst =
   6.929739656126463
```



And here they are for 12 points. Note that the Lebesgue constant has jumped from 7 to 51.

```
s = linspace(-1,1,12); [L,Lconst] = lebesgue(s);
hold off, plot(L), grid on, hold on, plot(s,L(s),'.')
title('Lebesgue function for 12 equispaced points',FS,9), Lconst
```

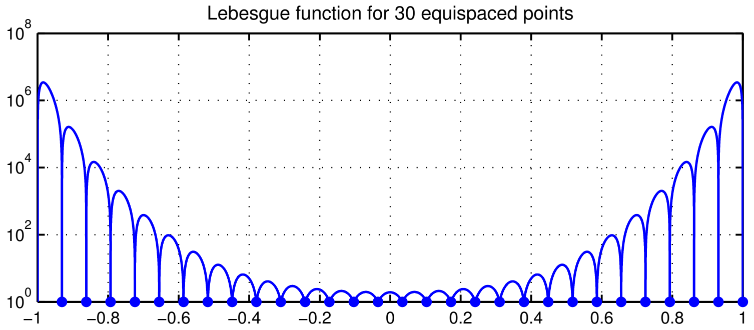
```
Lconst =
  51.214223185730248
```



The function takes large values near ±1, as we expect from Chapter 13 since the Runge phenomenon is associated with interpolants becoming very large near the endpoints. In fact the Lebesgue function for interpolation in equispaced points is more naturally displayed on a log scale. Here it is for $n = 30$:

```
s = linspace(-1,1,30); [L,Lconst] = lebesgue(s);
hold off, semilogy(L), grid on, hold on, semilogy(s,L(s),'.')
title('Lebesgue function for 30 equispaced points',FS,9)
Lconst
```

```
Lconst =
     3.447738672845219e+06
```



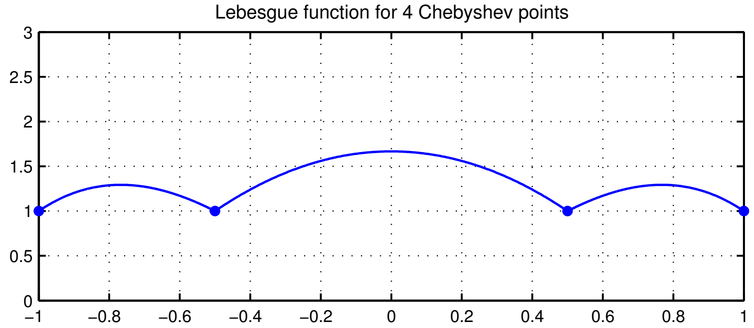
For comparison, here are the corresponding results for 4, 8, and 12 Chebyshev points, now back again on a linear scale.

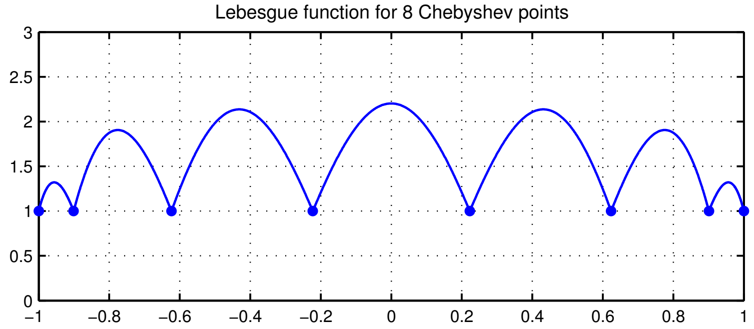
```
for npts = 4:4:12
  s = chebpts(npts); [L,Lconst] = lebesgue(s);
  hold off, plot(L), grid on, hold on, plot(s,L(s),'.')
  title(['Lebesgue function for ' int2str(npts) ' Chebyshev points'],FS,9)
  axis([-1 1 0 3])
  snapnow, Lconst
end
```

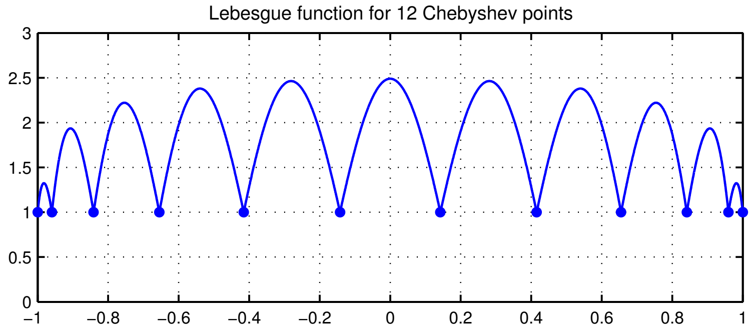
```
Lconst =
   1.666666666666667
```

```
Lconst =
   2.202214555205529
```

```
Lconst =
   2.489430376881967
```



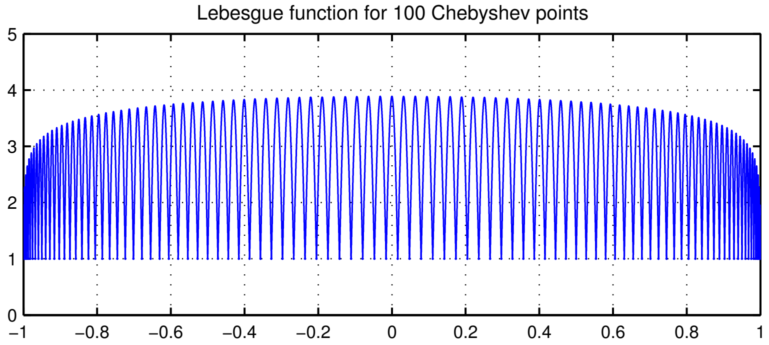




Here are 100 Chebyshev points, with a comparison of the actual Lebesgue constant with the bound from Theorem 15.2:

```
npts = 100; s = chebpts(npts); [L,Lconst] = lebesgue(s);
clf, plot(L,'linewidth',0.7), grid on, ylim([0 5])
Lconst, Lbound = 1 + (2/pi)*log(npts)
title('Lebesgue function for 100 Chebyshev points',FS,9)
```

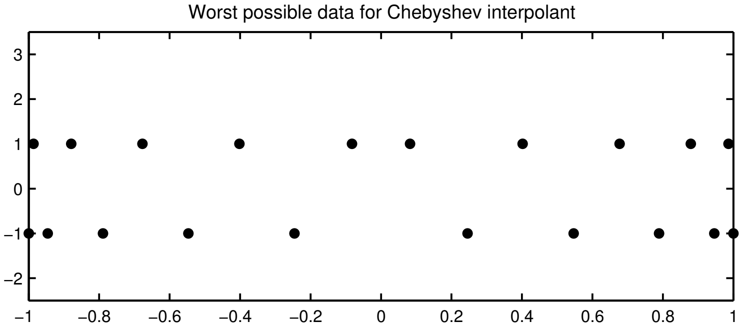
```
Lconst =
   3.887871431579912
Lbound =
   3.931742395517711
```



The low height of this curve shows how stable Chebyshev interpolation is.

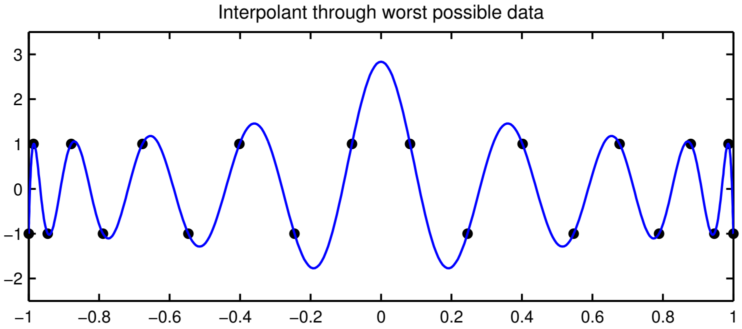
In Chapter 9 it was mentioned that combinations of Lagrange polynomials can explain both the Gibbs phenomenon and the size of Lebesgue functions. Let us now explain this remark. To analyze the Gibbs oscillations near a step, we added up a succession of Lagrange polynomials with constant amplitude 1. Since a single Lagrange polynomial has an oscillatory inverse-linear tail, the sum corresponds to an alternating series that converges as $n \to \infty$ to a constant. Lebesgue functions, on the other hand, are defined by taking a *maximum* at each point on the grid. The maximum is achieved by adding up Lagrange polynomials with equal but alternating coefficients, so as to make the combined signs all equal. For example, on the 20-point Chebyshev grid, the maximum possible value of an interpolant is achieved at $x = 0$ by taking data with this pattern:

```
s = chebpts(20); d = (-1).^[1:10 10:19]';
plot(s,d,'.k'), ylim([-2.5 3.5])
title('Worst possible data for Chebyshev interpolant',FS,9)
```



Here is the Chebyshev interpolant:

```
p = chebfun(d);
hold on, plot(p)
title('Interpolant through worst possible data',FS,9)
```



We readily confirm that the maximum of this interpolant is indeed the Lebesgue constant for this grid:

```
max(p)
[L,Lconst] = lebesgue(s);
Lconst
```

```
ans =
   2.837131699740444
Lconst =
   2.837131699740441
```

We can now summarize why Lebesgue constants for Chebyshev points, and indeed for any sets of interpolation points, must grow at least logarithmically with n. The fastest a Lagrange polynomial can decay is inverse-linearly, and the Lebesque function adds up those alternating tails with alternating coefficients, giving a harmonic series.

Our discussion in this chapter has focussed on Chebyshev interpolation rather than projection. However, as usual, there are parallel results for projection, which historically were worked out earlier (for the Fourier case, not Chebyshev). We record here a theorem analogous to Theorem 15.2.

Theorem 15.3. Lebesgue constants for Chebyshev projection. *The Lebesgue constants Λ_n for degree $n \ge 1$ Chebyshev projection in $[-1,1]$ are given by*

$$\Lambda_n = \frac{1}{2\pi}\int_{-\pi}^{\pi}\left|\frac{\sin((n+1/2)t)}{\sin(t/2)}\right| dt. \tag{15.10}$$

They satisfy

$$\Lambda_n \le \frac{4}{\pi^2}\log(n+1)+3 \quad \textit{and} \quad \Lambda_n \sim \frac{4}{\pi^2}\log n, \quad n\to\infty. \tag{15.11a,b}$$

Proof. See [Rivlin 1981]. Equation (15.11b) is due to Fejér in 1910 [Fejér 1910]. ∎

Related to Theorem 15.3 is another result concerning the norm of projection operators, proved by Landau [1913]. If f is analytic in the unit disk and continuous on the boundary, and $p \in \mathcal{P}_n$ is the *Taylor projection* of f obtained by truncating its Taylor series, how much bigger can p be than f on the unit disk? Landau showed that these norms (now known as *Landau constants*) grow at a rate asymptotic to $(1/\pi)\log n$ as $n \to \infty$, a discovery that is perhaps the starting point of all results about logarithmic growth of norms of approximation operators.

For details about Lebesgue constants, an outstanding source is the survey article by Brutman [1997].

Summary of Chapter 15. *The Lebesgue constant for interpolation or any other linear projection is the ∞-norm of the operator mapping data to approximant. For interpolation in $n+1$ Chebyshev points the Lebesgue constant is bounded by $1 + 2\pi^{-1}\log(n+1)$, whereas for $n+1$ equispaced points it is asymptotic to $2^{n+1}/en\log(n)$.*

Exercise 15.1. Plots of Lebesgue functions. Plot the Lebesgue functions for the following distributions of interpolation points. (a) -0.9, -0.8, 0, 0.1, 0.2, 0.8. (b) Same as in (a) but with additional points at a distance 0.01 to the right of the others.

Exercise 15.2. Chebyshev points of the first kind. The Lebesgue constants for degree n Chebyshev interpolation are bounded by those for degree n interpolation in Chebyshev points of the first kind, introduced in Exercise 2.4 (see also `help chebpts`), with equality when n is odd (Ehlich and Zeller [1966], McCabe and Phillips [1973]). Verify this numerically for $0 \le n \le 20$.

Exercise 15.3. Reproducing a table by Brutman. Page 698 of [Brutman 1978] gives a table of various quantities associated with the Lebesgue function for interpolation in Chebyshev points of the first kind, mentioned in the last exercise. Track down this paper and write the shortest, most elegant Chebfun program you can to duplicate this table. Are all of Brutman's digits correct?

Exercise 15.4. Omitting the endpoints. Suppose one performs polynomial interpolation in the usual Chebyshev points (2.2), but omitting the endpoints $x = \pm 1$. Perform numerical experiments to determine what happens to the Lebesgue constants in this case. Does the growth appear to still be of order $\log n$, or n^α for some α, or what?

Exercise 15.5. Optimal interpolation points. Starting from the $n+1$ Chebyshev points, one could attempt to use one of Matlab's optimization codes to adjust the points to minimize the Lebesque constant. Do this and give the Lebesgue constant and plot the Lebesgue function for (a) $n = 4$, (b) $n = 5$, (c) $n = 6$, (d) $n = 7$, and (e) $n = 8$. How much improvement do you find in the Lebesgue constants as compared with Chebyshev points?

Exercise 15.6. Improving Turetskii's estimate. For interpolation in equispaced points, Schönhage [1961] derived a more accurate estimate than (15.9b): $\Lambda_n \sim 2^{n+1}/en(\log n + \gamma)$, where $\gamma = 0.577\ldots$ is again Euler's constant. Perform a numerical study of Λ_n as a function of n and see what difference this correction makes. For example, it might be helpful to have a table showing the percentage errors in both estimates as functions of n.

Exercise 15.7. Interpolating data with a gap. (a) Consider polynomial interpolation in $n+1$ points of a function f defined on $[-1, 1]$, with half the points equally spaced from -1 to $-1/4$ and the other half equally spaced from $1/4$ to 1. Determine the Lebesgue constants for this interpolation process numerically for the cases $n+1 = 20$ and 40. (b) Suppose f is analytic and bounded by 1 in the ρ-ellipse E_ρ with $\rho = 2$. Carefully quoting theorems from this book as appropriate, give upper bounds for the error $|f(0) - p(0)|$ for these two cases.

Exercise 15.8. Smallest local minimum of the Lebesgue function. Interpolation in equispaced points is much better near the middle of an interval than at the ends. In particular, the smallest local maximum of the Lebesgue function λ is $\sim \log n/\pi$ as $n \to \infty$ [Tietze 1917]. Make a plot of these minima as a function of n to verify this behavior numerically.

Exercise 15.9. Convergence for Weierstrass's function. Exercise 7.3 promised that in Chapter 15, we would show that Chebyshev interpolants to Weierstrass's nowhere-differentiable function of Exercise 6.1 converge as $n \to \infty$. Write down such a proof based on combining various theorems of this book.

Exercise 15.10. Random interpolation points. (a) Compute Lebesgue functions and constants numerically for degree n interpolation in uniformly distributed random points in $[-1, 1]$. How does Λ appear to grow with n? (b) Same question for points randomly distributed according to the Chebyshev density (11.18).

Exercise 15.11. A wiggly function. (a) Let f be the function $T_m(x) + T_{m+1}(x) + \cdots + T_n(x)$ with $m = 20$ and $n = 40$, and let p^* be the best approximation of f of degree $m-1$. Plot f and $f - p^*$. What are their ∞-norms and 2-norms? (b) The same questions with $m = 200$ and $n = 300$.

Exercise 15.12. Divergence of Lebesgue constants. Spell out precisely the reasoning used to justify (15.6) in the text. In particular, make it clear why a "lim sup" rather than a "sup" appears in the formula.

Exercise 15.13. Confluent interpolation nodes. Let $\{x_j\}$ be a set of $n+1$ distinct interpolation nodes in $[-1, 1]$. Now change x_0 to $x_1 + \varepsilon$, where $\varepsilon > 0$ is a parameter, and let $\Lambda(\varepsilon)$ be the corresponding Lebesgue constant. Show that $\Lambda(\varepsilon)$ diverges to ∞ as $\varepsilon \to 0$. Can you quantify the rate of divergence?

Chapter 16

Best and near-best

```
ATAPformats
```

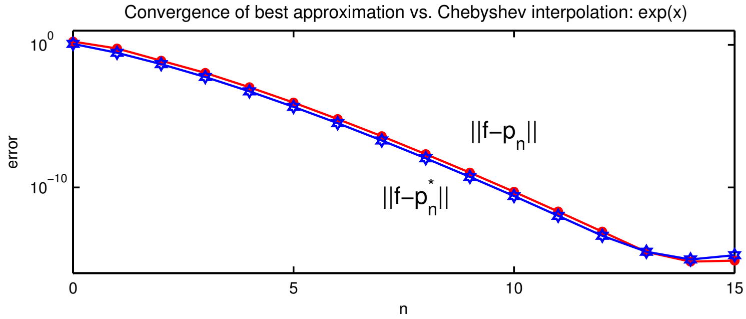
Traditionally, approximation theory has given a great deal of attention to best approximations, by which we continue to mean best approximations in the ∞-norm, and rather less to alternatives such as Chebyshev interpolants. One might think that this is because best approximations are much better than the alternatives. However, this is not true.

In a moment we shall continue with Lebesgue constants to shed some light on this matter, but first, let us do some experiments. We start with the extreme case of a very smooth function, $\exp(x)$, and compare convergence of its Chebyshev interpolants p and best approximants p^*. (The difference between n and $n+1$ in this code is intentional, since `chebfun` takes as argument the number of interpolation points whereas `remez` takes the degree of the polynomial.)

```
x = chebfun('x'); f = exp(x); nn = 0:15;
errbest = []; errcheb = []; i = 0;
for n = nn
    i = i+1;
    [p,err] = remez(f,n);
    errbest(i) = err;
    errcheb(i) = norm(f-chebfun(f,n+1),inf);
end
hold off, semilogy(nn,errcheb,'.-r')
hold on, semilogy(nn,errbest,'h-b','markersize',6)
FS = 'fontsize';
text(7,3e-12,'||f-p_n^*||',FS,12)
text(9,2e-7,'||f-p_n||',FS,12)
```

```
ylim([1e-16 10])
xlabel n, ylabel error
title(['Convergence of best approximation '...
       'vs. Chebyshev interpolation: exp(x)'],FS,9)
```

```
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
```



Clearly the stars for p^* aren't much better than the dots for p. The ratio of the two converges toward 2 until the rounding errors set in for larger degrees:

```
format short
ratio = errcheb./errbest;
disp('       n       ratio')
fprintf('%8d %12.5f\n',[nn; ratio])
```

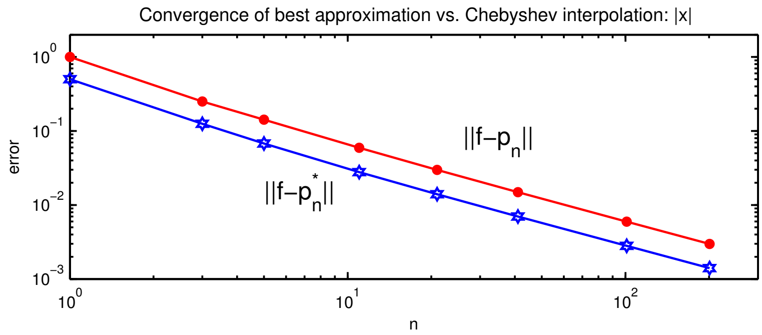
```
   n       ratio
   0      1.46212
   1      2.00000
   2      1.74436
   3      1.96807
   4      1.94991
   5      1.98188
   6      1.98182
   7      1.98861
   8      1.99105
   9      1.99222
  10      1.99473
  11      1.99177
  12      1.94923
  13      0.99246
  14      0.71220
  15      0.41466
```

At the other extreme of smoothness, consider $|x|$:

```
f = abs(x); nn = [0 2 4 10 20 40 100 200];
errbest = []; errcheb = []; i = 0;
for n = nn
    i = i+1;
    [p,err] = remez(f,n);
    errbest(i) = err;
    errcheb(i) = norm(f-chebfun(f,n+1),inf);
end
hold off, loglog(nn+1,errbest,'h-b','markersize',6)
hold on, loglog(nn+1,errcheb,'.-r')
axis([1 300 .001 2])
text(5,.01,'||f-p_n^*||',FS,12)
text(26,.06,'||f-p_n||',FS,12)
xlabel n, ylabel error
title(['Convergence of best approximation '...
       'vs. Chebyshev interpolation: |x|'],FS,9)
```

```
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
```

```
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
```



Again the stars are only a little bit better than the dots, by a constant factor of about 2.13060:

```
ratio = errcheb./errbest;
disp('       n       ratio')
fprintf('%8d %12.5f\n',[nn; ratio])
```

```
       n       ratio
       0     2.00000
       2     2.00000
       4     2.10234
      10     2.12677
      20     2.12968
      40     2.13037
     100     2.13056
     200     2.13059
```

(For odd values of n the ratio is somewhat larger, approaching a constant of about 3.57.)

So for these examples at least, you don't buy much with best approximations. And the cost in computing time is considerable. Here is the time for computing a Chebyshev interpolant p of degree 200 and evaluating it at 100 points:

```
xx = rand(100,1);
tic, p = chebfun(f,201); p(xx); toc
```

```
Elapsed time is 0.150190 seconds.
```

Here is the time for finding the best approximation p^* and evaluating it at the same points:

```
tic, p = remez(f,200); p(xx); toc
```

```
Warning:  This command is deprecated. Use minimax instead.
Elapsed time is 4.746876 seconds.
```

The reason computing p^* is more difficult is that the mapping from f to p^* is nonlinear, hence requiring iteration in a numerical implementation, whereas the mapping from f to p is linear (Exercise 10.5). It is perfectly feasible to compute p for degrees in the millions, whereas for p^* we would rarely attempt degrees higher than hundreds.

Why has p^* received so much more attention than p over the years? One reason is that in the days before fast computers, the degrees were low, so small differences in accuracy were more important. Another is that the theory of best approximations is so beautiful! Indeed, their very nonlinearity makes best approximations seemingly a richer field for research than the simpler Chebyshev interpolants. Everybody remembers Theorem 10.1, the equioscillation theorem, from the moment they first see it.

Yet in actual computation, true best approximations are not so often used, as we have mentioned earlier (Chapter 10). This is a clue that the world of practice may have its own wisdom, independent of the theorists.

Now let us see what theoretical results might tell us about the difference between p and p^*. The first such results pertain to Theorems 7.2 and 8.2 given earlier. Those theorems concerned convergence rates of p_n to f, depending on the smoothness of f. What about analogous theorems for p_n^*? Apart from constant factors, they turn out to be the same! For example, exactly the same bound (8.3) was published by de la Vallée Poussin [1919, pp. 123–124], except with the Chebyshev interpolant p_n replaced by the best approximation p_n^*. So within the two classes of functions considered in Chapters 7 and 8—f having a kth derivative of bounded variation, or f being analytic—there is no clear reason to expect p_n^* to be much better than p_n.

An observation for arbitrary functions f is the following consequence of Theorems 15.1–15.3:

Theorem 16.1. Chebyshev projections and interpolants are near-best. *Let $f \in C([-1,1])$ have degree n Chebyshev projection f_n, Chebyshev interpolant p_n, and best approximant p_n^*, $n \geq 1$. Then*

$$\|f - f_n\| \leq \left(4 + \frac{4}{\pi^2}\log(n+1)\right)\|f - p_n^*\| \tag{16.1}$$

and

$$\|f - p_n\| \leq \left(2 + \frac{2}{\pi}\log(n+1)\right)\|f - p_n^*\|. \tag{16.2}$$

Proof. Follows from Theorems 15.1, 15.2, and 15.3. ▮

So the loss of accuracy in going from p_n^* to p_n, say, can never be larger than a factor of $2 + (2/\pi)\log(n+1)$. It is interesting to examine the size of this quantity for various values of n. For $n = 10^5$, for example:

```
2 + (2/pi)*log(100001)
```

```
ans =
    9.3294
```

Since this number is less than 10, we see that in dealing with polynomials of degree up to $n = 100000$, the non-optimality of Chebyshev interpolation can never cost us more than one digit of accuracy. Here is the computation for $n = 10^{66}$:

```
2 + (2/pi)*log(1e66)
```

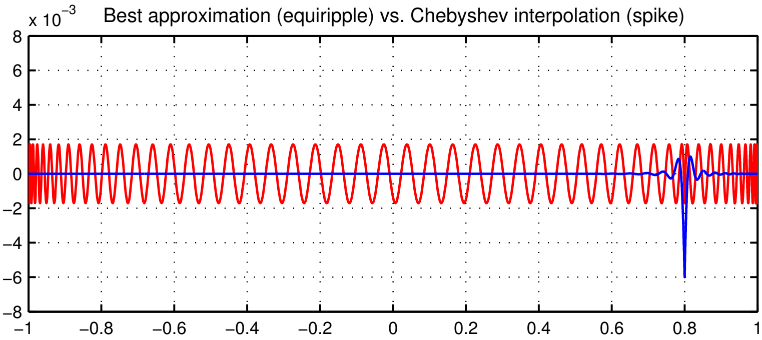
```
ans =
   98.7475
```

So we never lose more than 2 digits for degrees all the way up to 10^{66}—which might as well be ∞ for practical purposes. (For British audiences, one can give a talk on these matters with the title "10^{66} and All That".)

In fact, one might question whether best approximations are really better than near-best ones at all. Of course they are better in a literal sense, as measured in the ∞-norm. However, consider the following error curves, which are quite typical for high degree approximation of a function that is smoother in some regions than others.

```
f = abs(x-0.8);
tic, pbest = remez(f,100); toc
hold off, plot(f-pbest,'r')
tic, pcheb = chebfun(f,101); toc
hold on, plot(f-pcheb)
axis([-1 1 -.008 .008]), grid on
title('Best approximation (equiripple) vs. Chebyshev interpolation (spike)',FS,9)
```

```
Warning:  This command is deprecated. Use minimax instead.
Elapsed time is 1.603599 seconds.
Elapsed time is 0.022980 seconds.
```



We see that pbest is worse than pcheb for almost all values of x, because the damage done by the singularity at $x = 0.8$ is global. By contrast, the effect of the singularity on pcheb decays with distance. Of course, pbest is better in the ∞-norm:

```
errcheb = norm(f-pcheb,inf)
errbest = norm(f-pbest,inf)
```

```
errcheb =
    0.0060
errbest =
    0.0017
```

In the 2-norm, however, it is a good deal worse:

```
errcheb2 = norm(f-pcheb,2)
errbest2 = norm(f-pbest,2)
```

```
errcheb2 =
   4.3337e-04
errbest2 =
    0.0017
```

One might question how many applications there might be in which `pbest` was truly better than `pcheb` as an approximation to this function f. To echo a title of Corless and Watt [2004], minimax approximations are optimal, but Chebyshev interpolants may sometimes be better!

Li [2004] takes another angle on the near-optimality of Chebyshev interpolants, pointing out that for applications to elementary functions, bounds on certain derivatives usually hold that imply that the error in interpolation in Chebyshev points of the first kind exceeds that of the best approximation by less than a factor of 2, or as he calls it, "a fractional bit."

From a more theoretical point of view, we return to a notion mentioned in Theorem 12.1. Given $f \in C([-1,1])$, let ρ $(1 \le \rho \le \infty)$ be the parameter of the largest Bernstein ellipse E_ρ to which f can be analytically continued, and let $\{p_n\}$ be any sequence of approximations to f with $p_n \in \mathcal{P}_n$. Then

$$\limsup_{n\to\infty} \|f - p_n\|^{1/n} \ge \rho^{-1},$$

and if equality holds, $\{p_n\}$ is said to be *maximally convergent.* It follows from Theorem 15.1 that if $\{p_n\}$ come from a linear projection with Lebesgue constants Λ_n that grow more slowly than exponentially as $n \to \infty$, i.e., with $\limsup_{n\to\infty} \Lambda_n^{1/n} = 1$, then $\{p_n\}$ is maximally convergent for every $f \in C([-1,1])$. In particular, Chebyshev projections and interpolants are maximally convergent. This is a precise sense in which such approximations are "near-best".

Finally, we mention another kind of optimality that has received attention in the approximation theory literature [Bernstein 1931, Erdős 1961, Kilgore 1978, de Boor & Pinkus 1978]: *optimal interpolation points* (Exercise 15.5). Chebyshev points are very good, but they do not quite minimize the Lebesgue constant. Optimal points minimize the Lebesgue constant (by definition), and they level out the peaks of the Lebesgue function exactly (it has been proved)—but the improvement is negligible. The first statement of Theorem 15.2 establishes that, like Chebyshev points, they lead to Lebesgue constants that are asymptotic to $(2/\pi)\log n$ as $n \to \infty$, which means they do not even improve upon Chebyshev points by a constant factor.

Summary of Chapter 16. *The ∞-norm error in degree n Chebyshev interpolation is never greater than $2+(2/\pi)\log(n+1)$ times the ∞-norm error in degree n best approximation, and in practice, the ratio of errors rarely exceeds even a factor of 2. In the 2-norm, the interpolant is often much better than the best approximation.*

Exercise 16.1. Computing times for interpolation and best approximation. (a) Repeat the experiment of this chapter involving $|x-0.8|$ but for all the values $n = 100, 200, 300, \dots, 1000$. In each case measure the computing times for Chebyshev interpolation and best approximation as calculated by the Chebfun `remez` command, the L^2 errors of both approximants, and the L^∞ errors. Plot these results and comment on what you find. (b) In particular, produce a plot of error curves like that in the text. You may find it helpful to use a flag like `'numpts',10000` in your Chebfun plotting command.

Exercise 16.2. Approximation of a wiggly function. Define $f(x) = T_{200}(x) + T_{201}(x) + \cdots + T_{220}(x)$. Construct the Chebyshev interpolant p and best approximation p^* of degree 199. Plot the errors and measure the ∞- and 2-norms.

Exercise 16.3. Rounding errors on a grid of 10^{66} points. Suppose we had a computer with 16-digit precision capable of applying the barycentric formula (5.13) to evaluate a polynomial interpolant $p(x)$ for data on a Chebyshev grid of 10^{66} points. (For the sake of this thought experiment, imagine that the differences $x - x_j$ can be evaluated correctly to 16-digit precision rather than coming out as 0 and thereby invoking the $x = x_j$ clause of Theorem 5.2.) The evaluation would require adding up about 10^{66} numbers, entailing about 10^{66} rounding errors. Even if these errors only accumulated in the square root fashion of a random walk, it would still seem we must end up with errors on the order of 10^{33} times 10^{-16}, destroying all accuracy. Yet in fact, the computation would be highly accurate. What is the flaw in this 10^{33} reasoning?

Chapter 17

Orthogonal polynomials

`ATAPformats`

This book gives special attention to Chebyshev polynomials, since they are so useful in applications and the analogue on $[-1,1]$ of trigonometric polynomials on $[-\pi,\pi]$. However, Chebyshev polynomials are just one example of a family of orthogonal polynomials defined on the interval $[-1,1]$, and in this chapter we note some of the other possibilities, especially Legendre polynomials, which are the starting point for Gauss quadrature (Chapter 19). The study of orthogonal polynomials was initiated by Jacobi [1826] and already well developed by the end of the 19th century thanks to work by mathematicians including Chebyshev, Christoffel, Darboux, and Stieltjes. Landmark books on the subject include Szegő [1939] and Gautschi [2004].

Let $w \in C(-1,1)$ be a *weight function* with $w(x) > 0$ for all $x \in (-1,1)$ and $\int_{-1}^{1} w(x)dx < \infty$; we allow $w(x)$ to approach 0 or ∞ as $x \to \pm 1$. The function w defines an *inner product* for functions defined on $[-1,1]$:

$$(f,g) = \int_{-1}^{1} w(x)\overline{f(x)}g(x)dx. \tag{17.1}$$

(The bar over $f(x)$ indicates the complex conjugate, and can be ignored when working with real functions.) A family of *orthogonal polynomials* associated with w is a family

$$p_0, p_1, p_2, \dots$$

where p_n has degree exactly n for each n and the polynomials satisfy the orthogonality condition

$$(p_j, p_k) = 0, \quad k \neq j. \tag{17.2}$$

Notice that this condition implies that each p_n is orthogonal to all polynomials of degree $k < n$. The condition (17.2) determines the family uniquely except that each p_n can be multiplied by a constant factor. One common normalization is to require that each p_n be monic, in which case we have a family of *monic orthogonal polynomials*. Another common normalization is $p_0 > 0$ together with the condition

$$(p_j, p_k) = \delta_{jk} = \begin{cases} 1 & k = j, \\ 0 & k \neq j, \end{cases} \tag{17.3}$$

in which case we have *orthonormal polynomials*. A third choice, the standard one for Chebyshev and Legendre polynomials, is to require $p_n(1) = 1$ for each n.

As we have seen in Chapter 3, the Chebyshev polynomials $\{T_k\}$ are orthogonal with respect to the weight function

$$w(x) = \frac{2}{\pi\sqrt{1-x^2}} \tag{17.4}$$

(Exercise 3.7). If fact, if T_0 is replaced by $T_0/\sqrt{2}$, they are orthonormal. The first three Chebyshev polynomials are

$$T_0(x) = 1, \quad T_1(x) = x, \quad T_2(x) = 2x^2 - 1,$$

as we can confirm with the `chebpoly` command:

```
for j = 0:5, disp(fliplr(poly(chebpoly(j)))), end
```

```
     1
     0     1
    -1     0     2
     0    -3     0     4
     1     0    -8     0     8
     0     5     0   -20     0    16
```

The Chebyshev weight function has an inverse-square root singularity at each end of $[-1, 1]$. Allowing arbitrary power singularities at each end gives the *Jacobi weight function* $w(x) = (1-x)^{\alpha}(1+x)^{\beta}$, where $\alpha, \beta > -1$ are parameters. The associated orthogonal polynomials are known as *Jacobi polynomials* and written $\{P_n^{(\alpha,\beta)}\}$. In the special case $\alpha = \beta$ we get the *Gegenbauer* or *ultraspherical polynomials*.

The most special case of all is $\alpha = \beta = 0$, leading to *Legendre polynomials*, with the simplest of all weight functions, a constant:

$$w(x) = 1.$$

If we normalize according to (17.3), the first three Legendre polynomials are

$$p_0(x) = \sqrt{1/2}, \quad p_1(x) = \sqrt{3/2}\,x, \quad p_2(x) = \sqrt{45/8}\,x^2 - \sqrt{5/8},$$

as we can confirm by using the flag `'norm'` with the `legpoly` command:

```
format short
for j = 0:5, c = fliplr(poly(legpoly(j,'norm'))); disp(c), end
```

```
    0.7071
         0    1.2247
   -0.7906   -0.0000    2.3717
    0.0000   -2.8062   -0.0000    4.6771
    0.7955    0.0000   -7.9550   -0.0000    9.2808
    0.0000    4.3973   -0.0000  -20.5206    0.0000   18.4685
```

However, as mentioned above, it is more common to normalize Legendre polynomials by the condition $p_j(1) = 1$. Switching to an upper-case P to follow the usual notation, the first three Legendre polynomials are

$$P_0(x) = 1, \quad P_1(x) = x, \quad P_2(x) = \tfrac{3}{2}x^2 - \tfrac{1}{2}.$$

These are the polynomials returned by `legpoly` by default:

```
for j = 0:5, c = fliplr(poly(legpoly(j))); disp(c), end
```

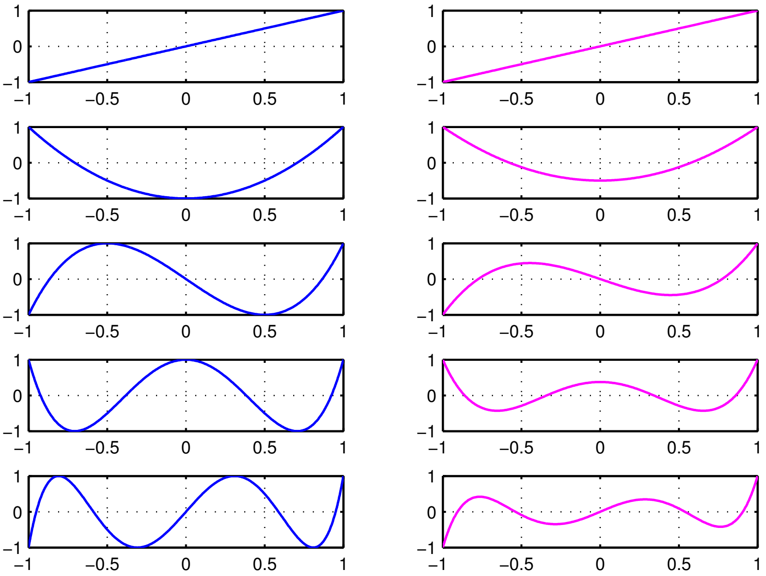
```
    1.0000
     0     1
   -0.5000   -0.0000    1.5000
    0.0000   -1.5000   -0.0000    2.5000
    0.3750    0.0000   -3.7500   -0.0000    4.3750
    0.0000    1.8750   -0.0000   -8.7500    0.0000    7.8750
```

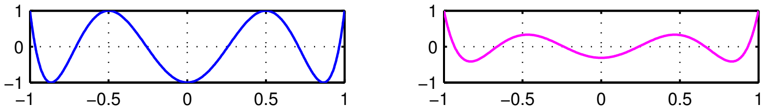
The rest of this chapter is devoted to comparing Legendre and Chebyshev polynomials. The comparison, and the consideration of orthogonal polynomials in general, will continue into the next two chapters on rootfinding (Chapter 18) and quadrature (Chapter 19). For example, Theorem 19.6 presents a fast method for calculating the barycentric weights for *Legendre points,* the zeros of Legendre polynomials. On the whole, different families of orthogonal polynomials have similar approximation properties, but Chebyshev points have the particular advantage that one can convert back and forth between interpolant and expansion by the FFT.

We begin with a visual comparison of the Chebyshev and Legendre polynomials of degrees 1–6 for $x \in [-1,1]$. The shapes are similar, with the degree n polynomial always having n roots in the interval (Exercise 17.4).

```
disp('                 Chebyshev                            Legendre')
ax = [-1 1 -1 1]; T = []; P = [];
for n = 1:6
    T{n} = chebpoly(n);
    subplot(3,2,1), plot(T{n}), axis(ax), grid on
    P{n} = legpoly(n);
    subplot(3,2,2), plot(P{n},'m'), axis(ax), grid on, snapnow
end
```

```
                 Chebyshev                            Legendre
```





For Legendre polynomials normalized by $P_j(1) = 1$, the orthogonality condition turns out to be

$$\int_{-1}^{1} P_j(x)\,P_k(x)\,dx = \begin{cases} 0 & j \neq k, \\ \dfrac{2}{2k+1} & j = k. \end{cases} \tag{17.5}$$

We can verify this formula numerically by constructing what Chebfun calls a **quasimatrix** X, that is, a "matrix" whose columns are chebfuns, and then taking inner products of each column with each other column via the quasimatrix product $X^T X$. One way to construct X is like this:

```
X = [P{1} P{2} P{3} P{4} P{5} P{6}];
```

Another equivalent method is built into `legpoly`:

```
X = legpoly(1:6);
```

Here is the quasimatrix product.

```
X'*X
```

```
ans =
    0.6667   -0.0000   -0.0000   -0.0000    0.0000   -0.0000
   -0.0000    0.4000   -0.0000    0.0000    0.0000    0.0000
   -0.0000   -0.0000    0.2857   -0.0000    0.0000   -0.0000
   -0.0000    0.0000   -0.0000    0.2222   -0.0000    0.0000
    0.0000   -0.0000    0.0000   -0.0000    0.1818   -0.0000
   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.1538
```

This matrix of inner products looks diagonal, as it should, and we can confirm the diagonal structure by checking the norm of the off-diagonal terms:

```
norm(ans-diag(diag(ans)))
```

```
ans =
   2.8600e-16
```

The entries on the diagonal are the numbers $2/3, 2/5, 2/7, \ldots$ prescribed by (17.5).

Legendre polynomials satisfy the 3-term recurrence relation

$$(k+1)P_{k+1}(x) = (2k+1)xP_k(x) - kP_{k-1}(x), \tag{17.6}$$

which may be compared with the recurrence relation (3.10) for Chebyshev polynomials. In general, orthogonal polynomials defined by (17.1)–(17.2) always satisfy a 3-term recurrence relation, and the reason is as follows. Supposing $\{p_n\}$ are monic for simplicity, one can determine p_{n+1} by the Gram–Schmidt orthogonalization procedure, subtracting off the projections of the monic degree $n+1$ polynomial xp_n onto each of the polynomials $p_0, \ldots, p_n$, with the coefficient of the projection onto p_k being given by the inner product (xp_n, p_k):

$$p_{n+1} = xp_n - (xp_n, p_n)p_n - (xp_n, p_{n-1})p_{n-1} - \cdots - (xp_n, p_0)p_0.$$

For every $k < n-1$, however, the inner product is equal to 0 because p_n is orthogonal to the lower degree polynomial xp_k:[1]

$$(xp_n, p_k) = (p_n, xp_k) = 0, \quad k < n-1. \tag{17.7}$$

Thus the series above reduces to the 3-term recurrence

$$p_{n+1} = xp_n - (xp_n, p_n)p_n - (xp_n, p_{n-1})p_{n-1}. \tag{17.8}$$

When the weight function w is even, the middle term drops out (Exercise 17.5), and the formula further simplifies to

$$p_{n+1} = xp_n - (xp_n, p_{n-1})p_{n-1} \quad \text{for } w \text{ even}. \tag{17.9}$$

We reiterate that (17.8) and (17.9) are based on the assumption that the polynomials $\{p_k\}$ are monic. For other normalizations, p_{n+1} must be multiplied by a suitable constant.

Chebyshev polynomials are not orthogonal in the standard inner product:

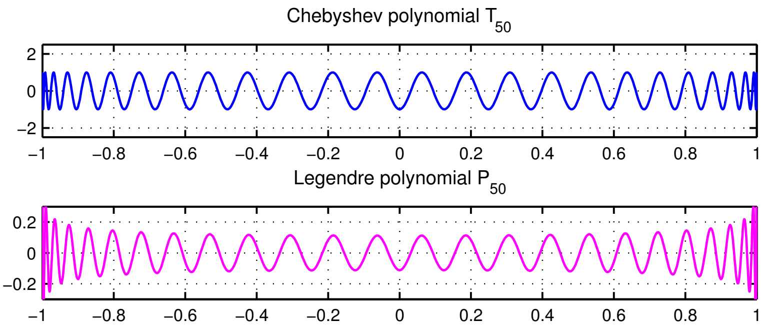
[1] What makes this calculation work, abstractly speaking, is that the operation of multiplication of a function by x is self-adjoint with respect to the inner product (17.1). It is for the same reason of self-adjointness that the Lanczos iteration in numerical linear algebra, which applies to real symmetric matrices, reduces them to tridiagonal form, whereas the Arnoldi iteration, which generalizes Lanczos to arbitrary matrices, achieves only Hessenberg form [Trefethen & Bau 1997].

```
X = chebpoly(1:6); X'*X
```

```
ans =
    0.6667    0.0000   -0.4000    0.0000   -0.0952    0.0000
   -0.0000    0.9333   -0.0000   -0.3619    0.0000   -0.0825
   -0.4000    0.0000    0.9714    0.0000   -0.3492   -0.0000
    0.0000   -0.3619   -0.0000    0.9841   -0.0000   -0.3434
   -0.0952   -0.0000   -0.3492    0.0000    0.9899    0.0000
    0.0000   -0.0825    0.0000   -0.3434    0.0000    0.9930
```

Nevertheless, Legendre and Chebyshev polynomials have much in common, as is further suggested by plots of T_{50} and P_{50}:

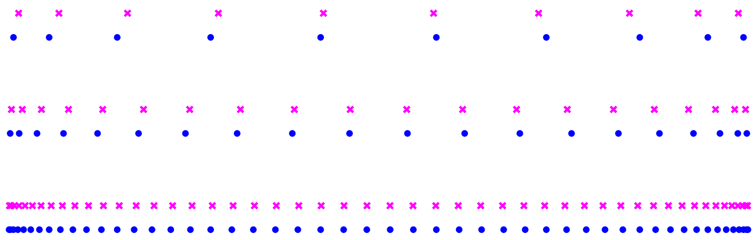
```
T50 = chebpoly(50); P50 = legpoly(50);
subplot(2,1,1), plot(T50), axis([-1 1 -2.5 2.5]), FS = 'fontsize';
grid on, title('Chebyshev polynomial T_{50}',FS,9)
subplot(2,1,2), plot(P50,'m'), axis([-1 1 -.3 .3])
grid on, title('Legendre polynomial P_{50}',FS,9)
```



The zeros of the two families of polynomials are similar, as can be confirmed by comparing Chebyshev (dots) and Legendre (crosses) zeros for degrees 10, 20, and 50. (Instead of using the `roots` command here, one could achieve the same effect with `chebpts(n,1)` and `legpts(n)`—see Chapter 19.)

```
T10 = chebpoly(10); P10 = legpoly(10);
Tr = roots(T10); Pr = roots(P10);
MS = 'markersize'; clf, plot(Tr,.8,'.b',MS,9), hold on
plot(Pr,0.9,'xm',MS,4)
T20 = chebpoly(20); P20 = legpoly(20);
Tr = roots(T20); Pr = roots(P20);
plot(Tr,0.4,'.b',MS,9), plot(Pr,0.5,'xm',MS,4)
```

```
Tr = roots(T50); Pr = roots(P50);
plot(Tr,0,'.b',MS,9), plot(Pr,0.1,'xm',MS,4)
axis([-1 1 -.1 1.1]), axis off
```

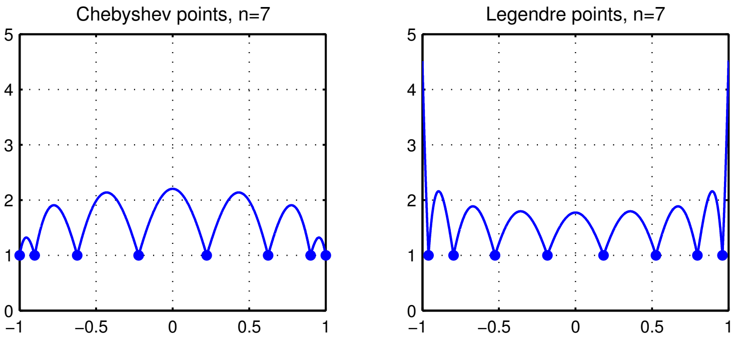


Asymptotically as $n \to \infty$, both sets of zeros cluster near ±1 with the same density distribution $n\mu(x)$, with μ given by (12.10). This behavior is made precise in Theorem 12.1.4 of [Szegő 1939] (Exercise 17.7), and exploitation of more detailed asymptotic properties of Gauss–Jacobi polynomials is the crucial idea of [Hale & Townsend 2012].

Another comparison between Chebyshev and Legendre points concerns their Lebesgue functions and Lebesgue constants. Here we repeat a computation of Lebesgue functions from Chapter 15 for 8 Chebyshev points and compare it with the analogous computation for 8 Legendre points. Chebyshev and Legendre points as we have defined them so far differ not just in which polynomials they are connected with, but in that Chebyshev points come from extrema whereas Legendre points come from zeros.

```
hold off
s = chebpts(8); [L,Lconst] = lebesgue(s);
subplot(1,2,1), plot(L), grid on, hold on, plot(s,L(s),'.'), Lconst
ylim([0,5]), title('Chebyshev points, n=7',FS,9)
s = legpts(8); [L,Lconst] = lebesgue(s);
subplot(1,2,2), plot(L), grid on, hold on, plot(s,L(s),'.'), Lconst
ylim([0,5]), title('Legendre points, n=7',FS,9)
```

```
Lconst =
    2.2022
Lconst =
    4.5135
```

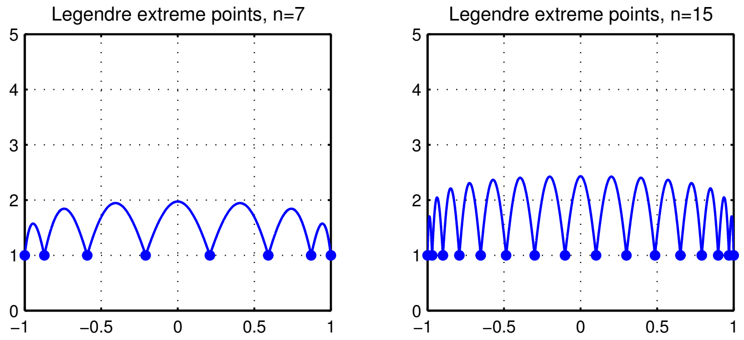


The Lebesgue functions and constants for Legendre points are a little bigger than for Chebyshev points, having size $O(n^{1/2})$ rather than $O(\log n)$ because of behavior near the endpoints [Szegő 1939, p. 338]. This small difference is of little significance for most applications: the Lebesgue constants are still quite small, and either set of points will usually deliver excellent interpolants.

Moreover, an alternative is to consider *Legendre extreme points*—the $n+1$ points in $[-1,1]$ at which $|P_n(x)|$ attains a local maximum. (The Legendre extreme points in $(-1,1)$ are also the roots of the Jacobi polynomial $P^{(1,1)}(x)$.) The Lebesgue function in this case looks even more satisfactory:

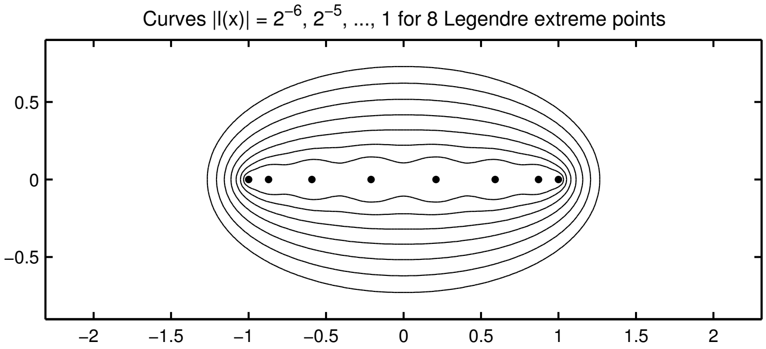
```
clf
s = [-1; roots(diff(legpoly(7))); 1]; [L,Lconst] = lebesgue(s);
subplot(1,2,1), plot(L), grid on, hold on, plot(s,L(s),'.'), Lconst
ylim([0,5]), title('Legendre extreme points, n=7',FS,9)
s15 = [-1; roots(diff(legpoly(15))); 1]; [L,Lconst] = lebesgue(s15);
subplot(1,2,2), plot(L), grid on, hold on, plot(s15,L(s15),'.'), Lconst
ylim([0,5]), title('Legendre extreme points, n=15',FS,9)
```

```
Lconst =
    1.9724
Lconst =
    2.4303
```



The Legendre extreme points have a memorable property: as shown by Stieltjes [1885], they are the Fekete or minimal-energy points in $[-1,1]$, solving the equipotential problem on that interval for a finite number of equal charges (Exercise 12.1). Here, for example, is a repetition of a figure from Chapter 11 but now for 8 Legendre extreme points instead of 8 Chebyshev points. Again the behavior is excellent.

```
ell = poly(s,domain(-1,1));
clf, plot(s,ell(s),'.k',MS,10)
hold on, ylim([-0.9,0.9]), axis equal
xgrid = -1.5:.02:1.5; ygrid = -0.9:.02:0.9;
[xx,yy] = meshgrid(xgrid,ygrid); zz = xx+1i*yy;
ellzz = ell(zz); levels = 2.^(-6:0);
contour(xx,yy,abs(ellzz),levels,'k')
title(['Curves |l(x)| = 2^{-6}, 2^{-5}, ..., 1 '...
    'for 8 Legendre extreme points'],FS,9)
```



Summary of Chapter 17. *Chebyshev polynomials are just one example of a family of polynomials orthogonal with respect to a weight function $w(x)$ on $[-1,1]$. For $w(x)$ = constant, one gets the Legendre polynomials.*

Exercise 17.1. Chebyshev and Legendre Lebesgue constants. Extend the experiments of the text to a table and a plot of Lebesgue constants of Chebyshev, Legendre, and Legendre extreme points for interpolation in $n+1$ points with $n = 1, 2, 4, \ldots, 256$. (To compute Legendre extreme points efficiently, you can use the observation about Jacobi polynomials mentioned in the text and the Chebfun command `jacpoly`.) What asymptotic behavior do you observe as $n \to \infty$?

Exercise 17.2. Chebyshev and Legendre interpolation points. Define $f(x) = x \tanh(2\sin(20x))$, and let p and p_L be the interpolants to f in $n+1$ Chebyshev or Legendre points on $[-1,1]$, respectively. The latter can be computed with `interp1` as in Chapter 13. (a) For $n+1 = 30$, plot f, p, and p_L. What are the ∞-norm errors $\|f-p\|$ and $\|f-p_L\|$? (b) For $n+1 = 300$, plot $f-p$ and $f-p_L$. What are the errors now?

Exercise 17.3. Orthogonal polynomials via QR decomposition. (a) Construct a Chebfun quasimatrix `A` with columns corresponding to $1, x, \ldots, x^5$ on $[-1,1]$. Execute `[Q,R] = qr(A)` to find an equivalent set of orthonormal functions, the columns of `Q`, and plot these with `plot(Q)`. How do the columns of `Q` compare with the Legendre polynomials normalized by (17.3)? (b) Write a **for** loop to normalize the columns of `Q` in a fashion corresponding to $P_j(1) = 1$ and to adjust R correspondingly so that the product `Q*R` continues to be equal to `A`, up to rounding errors, and plot the new quasimatrix with `plot(Q)`. How do the columns of the new `Q` compare with the Legendre polynomials normalized by $P_j(1) = 1$?

Exercise 17.4. Zeros of orthogonal polynomials. Let $\{p_n\}$ be a family of orthogonal polynomials on $[-1,1]$ defined by (17.1)–(17.2). Show by using (17.2) that the zeros of p_n are distinct and lie in $(-1,1)$.

Exercise 17.5. Even and odd orthogonal polynomials. Suppose the weight function w of (17.1) is even. Prove by induction that p_n is even when n is even and odd when n is odd.

Exercise 17.6. Legendre and Chebyshev differential equations. (a) Show from the recurrence relation (17.6) that the Legendre polynomial P_n satisfies the differential equation $(1-x^2)P'' - 2xP' + n(n+1)P = 0$. (b) Show from (3.10) that the Chebyshev polynomial T_n satisfies the differential equation $(1-x^2)T'' - xT' + n^2T = 0$. [This exercise needs more.]

Exercise 17.7. The envelope of an orthogonal polynomial. Theorem 12.1.4 of [Szegő 1939] asserts that as $n \to \infty$, the envelope of an orthonormal polynomial p_n defined by (17.1)–(17.3) approaches the curve $(w_{\text{CHEB}}(x)/w(x))^{1/2}$, where w_{CHEB} is the Chebyshev weight (17.4). Explore this prediction numerically with plots of Legendre polynomials for various n.

Exercise 17.8. Minimality of orthogonal polynomials. Let $\{p_n\}$ be the family of monic orthogonal polynomials associated with the inner product (17.1). Show that if q is any monic polynomial of degree n, then $(q,q) \ge (p_n, p_n)$.

Chapter 18

Polynomial roots and colleague matrices

`ATAPformats`

It is well known that if p is a polynomial expressed as a linear combination of monomials x^k, then the roots of p are equal to the eigenvalues of a certain *companion matrix* formed from its coefficients (Exercise 18.1). Indeed, from its beginning in the late 1970s, Matlab has included a command `roots` that calculates roots of polynomials by using this identity. This method of zerofinding is effective and numerically stable, but only in a very narrow sense. It is a numerically stable algorithm for precisely the problem just posed: given the monomial coefficients, find the roots. The trouble is, this problem is an awful one! As Wilkinson made famous beginning in the 1960s, it is a highly ill-conditioned problem in general [Wilkinson 1984]. The roots tend to be so sensitive to perturbations in the coefficients that even though the algorithm is stable in the sense that it usually produces roots that are exactly correct for a polynomial whose coefficients match the specified ones to a relative error on the order of machine precision [Goedecker 1994, Toh & Trefethen 1994], this slight perturbation is enough to cause terrible inaccuracy.

There is an exception to this dire state of affairs. Finding roots from polynomial coefficients is a well-conditioned problem in the special case of polynomials with roots on or near the unit circle (see Exercise 18.7(a) and [Sitton, Burrus, Fox & Treitel 2003]). The trouble is, most applications are not of this kind. More often, the roots of interest lie in or near a real interval, and in such cases one should avoid monomials, companion matrices, and Matlab's `roots` command completely.

Fortunately, there is a well-conditioned alternative for such problems, and that is the subject of this chapter. By now we are experts in working with functions on $[-1,1]$ by means of Chebyshev interpolants and Chebyshev series. Within this class of tools, there is a natural way of computing the roots of a polynomial by solving an eigenvalue problem. Here is the crucial result, due independently to Specht [1960, p. 222] and Good [1961].[1] The matrix C of the theorem is called a *colleague matrix.*

Theorem 18.1. Polynomial roots and colleague matrix eigenvalues. *The roots of the polynomial*

$$p(x) = \sum_{k=0}^{n} a_k T_k(x), \qquad a_n \neq 0$$

are the eigenvalues of the matrix

$$C = \begin{pmatrix} 0 & 1 & & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & & \\ & \frac{1}{2} & 0 & \frac{1}{2} & & \\ & & \ddots & \ddots & \ddots & \\ & & & & & \frac{1}{2} \\ & & & & \frac{1}{2} & 0 \end{pmatrix} - \frac{1}{2a_n} \begin{pmatrix} & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ a_0 & a_1 & a_2 & \dots & a_{n-1} \end{pmatrix}. \qquad (18.1)$$

(*Entries not displayed are zero.*) *If there are multiple roots, these correspond to eigenvalues with the same multiplicities.*

Proof. Let x be any number, and consider the nonzero n-vector

$$v = (T_0(x), T_1(x), \dots, T_{n-1}(x))^T.$$

If we multiply C by v, then in every row but the first and last the result is

$$T_k(x) \mapsto \tfrac{1}{2} T_{k-1}(x) + \tfrac{1}{2} T_{k+1}(x) = x T_k(x),$$

thanks to the three-term recurrence relation (3.9) for Chebyshev polynomials. In the first row we likewise have

$$T_0(x) \mapsto T_1(x) = x T_0(x)$$

since $T_0(x) = 1$ and $T_1(x) = x$. It remains to examine the bottom row. Here it is convenient to imagine that in the difference of matrices defining

[1] Jack Good (1916–2009) was a hero of Bayesianism who worked with Turing at Bletchley Park.

C above, the "missing" entry 1/2 is added in the $(n, n+1)$ position of the first matrix and subtracted again from the $(n, n+1)$ position of the second matrix. Then by considering the recurrence relation again we find

$$T_{n-1}(x) \mapsto xT_{n-1}(x) - \frac{1}{2a_n}(a_0T_0(x) + a_1T_1(x) + \cdots + a_nT_n(x)).$$

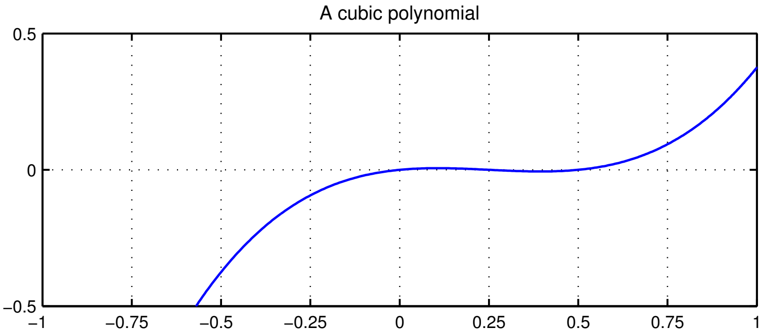
This equation holds for any x, and if x is a root of p, then the term in parentheses on the right vanishes. In other words, if x is a root of p, then Cv is equal to xv in every entry, making v is an eigenvector of C with eigenvalue x. If p has n distinct roots, this implies that they are precisely the eigenvalues of C, and this completes the proof in the case where p has distinct roots.

If p has multiple roots, we must show that each one corresponds to an eigenvalue of C with the same multiplicity. For this we can consider perturbations of the coefficients $a_0, \dots, a_{n-1}$ of p with the property that the roots become distinct. Each root must then correspond to an eigenvalue of the correspondingly perturbed matrix C, and since both roots of polynomials and eigenvalues of matrices are continuous functions of the parameters, the multiplicities must be preserved in the limit as the amplitude of the perturbations goes to zero. ▮

As mentioned above, the matrix C of (18.1) is called a colleague matrix. Theorem 18.1 has been rediscovered several times, for example by Day & Romero [2005]. Since Specht [1957] there have also been generalizations to other families of orthogonal polynomials besides Chebyshev polynomials, and the associated generalized colleague matrices are called *comrade matrices* [Barnett 1975a & 1975b]. The generalization is immediate: one need only change the entries of rows 1 to $n-1$ to correspond to the appropriate recurrence relation.

For an example to illustrate Theorem 18.1, consider the polynomial $p(x) = x(x-1/4)(x-1/2)$.

```
x = chebfun('x');
p = x.*(x-1/4).*(x-1/2);
clf, plot(p)
axis([-1 1 -.5 .5]), grid on
set(gca,'xtick',-1:.25:1)
title('A cubic polynomial','fontsize',9)
```



Obviously p has roots 0, 1/4, and 1/2. The Chebyshev coefficients are $-3/8, 7/8, -3/8, 1/4$:

```
format short
a = fliplr(chebpoly(p))
```

```
Warning: CHEBPOLY is deprecated. Please use CHEBCOEFFS instead.
a =
   -0.3750    0.8750   -0.3750    0.2500
```

As expected, the colleague matrix (18.1) for this polynomial,

```
C = [0 1 0; 1/2 0 1/2; 0 1/2 0] - ...
    (1/(2*a(4)))*[0 0 0; 0 0 0; a(1:3)]
```

```
C =
         0    1.0000         0
    0.5000         0    0.5000
    0.7500   -1.2500    0.7500
```

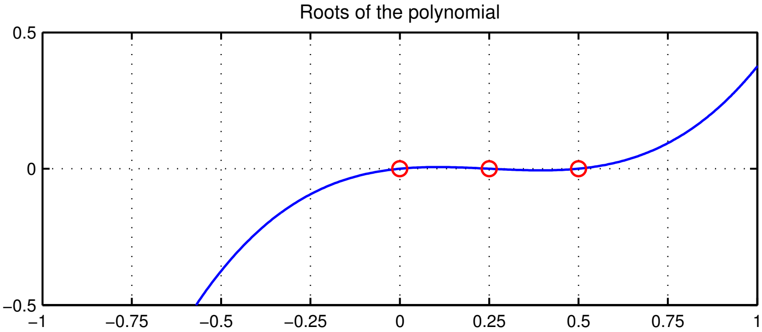
has eigenvalues that match the roots of p:

```
format long
eig(C)
```

```
ans =
  -0.000000000000000
   0.500000000000001
   0.249999999999999
```

In Chebfun, every function is represented by a polynomial or a piecewise polynomial. Theorem 18.1 provides Chebfun with its method of numerical rootfinding, implemented in the Chebfun `roots` command. For this polynomial `p`, we can call `roots` to add the roots to the plot, like this:

```
r = roots(p);
hold on, plot(r,p(r),'or','markersize',7)
title('Roots of the polynomial','fontsize',9)
```



In this example, `p` was a polynomial from the start. The real power of Theorem 18.1, however, comes when it is applied to the problem of finding the roots on $[-1,1]$ of a general function f. To do this, we first approximate f by a polynomial, then find the roots of the polynomial. This idea was proposed in Good's original 1961 paper [Good 1961]. In a more numerical era, it has been advocated in a number of papers by John Boyd, including [Boyd 2002], and it is exploited virtually every time Chebfun is used.

For example, here is the chebfun corresponding to $\cos(50\pi x)$ on $[-1,1]$:

```
f = cos(50*pi*x); length(f)
```

```
ans =
   215
```

It doesn't take long to compute its roots,

```
tic, r = roots(f); toc
```

```
Elapsed time is 0.350356 seconds.
```

The exact roots of this function on $[-1, 1]$ are $-0.99, -0.97, \ldots, 0.97, 0.99$. Inspecting a few of the computed results shows they are accurate to close to machine precision:

```
r(1:5)
```

```
ans =
  -0.990000000000000
  -0.970000000000000
  -0.950000000000000
  -0.930000000000000
  -0.910000000000000
```

Changing the function to $\cos(500\pi x)$ makes the chebfun ten times longer,

```
f = cos(500*pi*x); length(f)
```

```
ans =
        1685
```

One might think this would increase the rootfinding time greatly, since the number of operations for an eigenvalue computation grows with the cube of the matrix dimension. (The colleague matrix has special structure that can be used to bring the operation count down to $O(n^2)$, but this is not done in a straightforward Matlab call to `eig`.) However, an experiment shows that the timing is still quite good,

```
tic, r = roots(f); toc
```

```
Elapsed time is 1.844424 seconds.
```

and the accuracy is still outstanding:

```
r(1:5)
```

```
ans =
  -0.999000000000000
  -0.997000000000000
  -0.995000000000000
  -0.993000000000000
  -0.991000000000000
```

We can make sure all 1000 roots are equally accurate by computing a norm:

```
exact = [-0.999:0.002:0.999]'; norm(r-exact,inf)
```

```
ans =
     3.330669073875470e-16
```

The explanation of this great speed in finding the roots of a polynomial of degree in the thousands is that the complexity of the algorithm has been improved from $O(n^3)$ to $O(n^2)$ by recursion. If a chebfun has length greater than 100, the interval is divided recursively into subintervals, with a chebfun constructed on each subinterval of appropriately lower degree. Thus no eigenvalue problem is ever solved of dimension greater than 100. This idea of rootfinding based on recursive subdivision of intervals and Chebyshev eigenvalue problems was developed by John Boyd in the 1980s and 1990s and published by him in 2002 [Boyd 2002]. Details of the original Chebfun implementation of `roots` were presented in [Battles 2005], and in 2012 the Chebfun algorithm was speeded up substantially by Pedro Gonnet (unpublished).

These techniques are remarkably powerful for practical computations. For example, how many zeros does the Bessel function J_0 have in the interval $[0, 5000]$? Chebfun finds the answer in a fraction of a second:

```
tic, f = chebfun(@(x) besselj(0,x),[0,5000]);
r = roots(f); toc
length(r)
```

```
Elapsed time is 1.850265 seconds.
ans =
        1591
```

What is the the 1000th zero?

```
r(1000)
```

```
ans =
     3.140807295225079e+03
```

We readily verify that this zero is an accurate one:

```
besselj(0,ans)
```

```
ans =
     5.756205180307391e-17
```

This example, like a few others scattered around the book, makes use of a chebfun defined on an interval other than the default $[-1, 1]$. The mathematics is straightforward; $[0, 5000]$ is reduced to $[-1, 1]$ by a linear transformation.

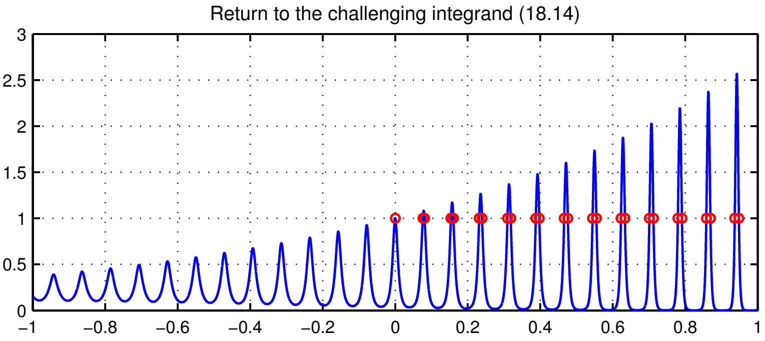
Here is another illustration of recursive colleague matrix rootfinding for a high-order polynomial. The function

$$f(x) = e^x[\operatorname{sech}(4\sin(40x))]^{\exp(x)} \tag{18.1}$$

features a row of narrower and narrower spikes. Where in $[-1, 1]$ does it take the value 1? We can find the answer by using `roots` to find the zeros of the equation $f(x) - 1 = 0$:

```
ff = @(x) exp(x).*sech(4*sin(40*x)).^exp(x);
tic, f = ff(x); r = roots(f-1); toc
clf, plot(f), grid on, FS = 'fontsize';
title('Return to the challenging integrand (18.14)',FS,9)
hold on, plot(r,f(r),'or','markersize',4)
```

```
Elapsed time is 2.674122 seconds.
```



Notice that we have found the roots here of a polynomial of quite high degree:

```
length(f)
```

```
ans =
        3679
```

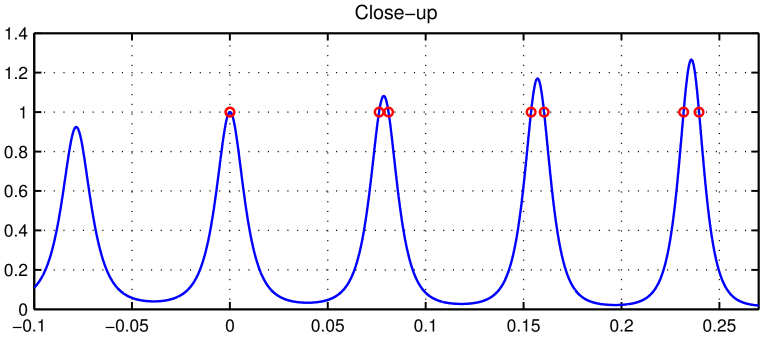
A numerical check confirms that the roots are accurate,

```
max(abs(ff(r)-1))
```

```
ans =
     3.308464613382966e-14
```

and zooming in gives perhaps a more convincing plot:

```
xlim([-.1 .27])
title('Close-up',FS,9)
```



Computations like this are examples of *global rootfinding*, a special case of *global optimization*. They are made possible by the combination of fast methods of polynomial approximation with the extraordinarily fast and accurate methods for matrix eigenvalue problems that have been developed in the years since Francis invented the QR algorithm in the very same year as Good proposed his colleague matrices [Francis 1961]. (A crucial algorithmic feature that makes these eigenvalue calculations so accurate is known as "balancing", introduced in [Parlett & Reinsch 1969]—see [Toh & Trefethen 1994] and Exercise 18.3.)

Global rootfinding is a step in many other practical computations. It is used by Chebfun, for example, in computing minima, maxima, 1-norms, and absolute values.

It is worth mentioning that as an alternative to eigenvalue problems based on Chebyshev expansion coefficients, it is possible to relate roots of polynomials to eigenvalue problems (or generalized eigenvalue problems) constructed from function values themselves at Chebyshev or other points. Mathematical processes along these lines are described in [Fortune 2001], [Amiraslani, et al. 2004], and [Amiraslani 2006]. So far there has not been much numerical exploitation of these ideas, but preliminary experiments suggest that in the long run they may be competitive.

We close this chapter by clarifying a point that may have puzzled the reader, and which has fascinating theoretical connections. In plots like the last two, we see only real roots of a function. Yet if the function is a chebfun based on a polynomial representation, won't there be complex roots too? This is indeed the case, but the Chebfun `roots` command by default returns only those roots in the interval where the function is defined. This default behavior can be overridden by the use of the flags `'all'` or `'complex'` (see Exercise 14.2). For example, suppose we make a chebfun corresponding to the function $f(x) = (x - 0.5)/(1 + 10x^2)$, which has just one root in the complex plane, at $x = 0.5$:

```
f = (x-0.5)./(1+10*x.^2); length(f)
```

```
ans =
   119
```

Typing `roots` alone gives just the root at $x = 0.5$:

```
roots(f)
```

```
ans =
   0.499999999999999
```

With `roots(f,'all')`, however, we get 106 roots:

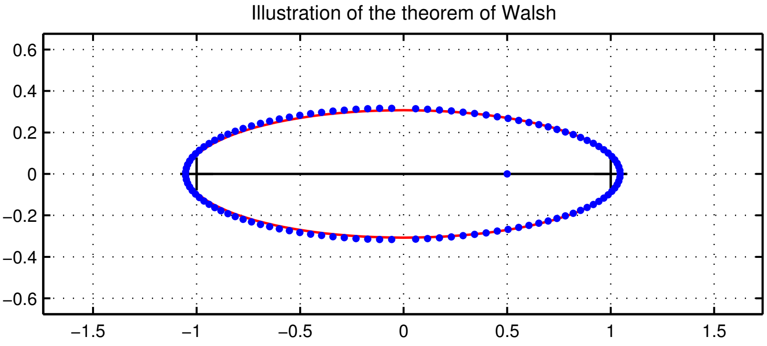
```
r = roots(f,'all'); length(r)
```

```
ans =
   112
```

The complex roots are meaningless from the point of view of the underlying function f; they are an epiphenomenon that arises in the process of approximating f on $[-1, 1]$. A plot reveals that they have a familiar distribution, lying almost exactly on the Chebfun ellipse for this function:

```
hold off, chebellipseplot(f,'r')
hold on, plot(r,'.','markersize',10)
xlim(1.2*[-1 1]), grid on, axis equal
FS = 'fontsize';
title('Illustration of the theorem of Walsh',FS,9)
```

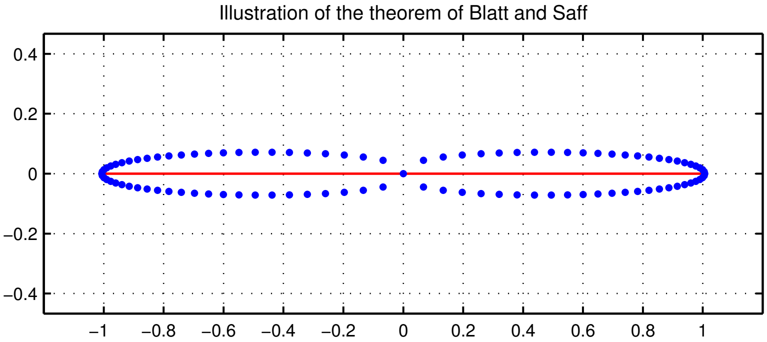
```
Warning: CHEBELLIPSEPLOT is deprecated.  Please use PLOTREGION instead.
```



The fact that roots of best and near-best approximations cluster along the maximum Bernstein ellipse of analyticity is a special case of a theorem due to Walsh [1959]. Blatt and Saff [1986] extended Walsh's result to the case in which the function being approximated has no ellipse of analyticity, but is merely continuous on $[-1, 1]$. They showed that in this case, the zeros of the best approximants always cluster at every point of the interval as $n \to \infty$. This phenomenon applies not only to the best approximations, but to all near-best best approximations that are maximally convergent as defined in Chapter 12, hence in particular to Chebyshev interpolants. Here for example are the roots of the degree 100 Chebyshev interpolant to $|x|$:

```
f = chebfun('abs(x)',101); length(f)
r = roots(f,'all');
hold off, plot([-1,1],[0,0],'r')
hold on, plot(r,'.','markersize',10)
xlim(1.2*[-1 1]), grid on, axis equal
title('Illustration of the theorem of Blatt and Saff',FS,9)
```

```
ans =
   101
```



The Walsh and Blatt–Saff theorems are extensions of *Jentzsch's theorem*, which asserts that the partial sums of Taylor series have roots clustering along every point of the circle of convergence [Jentzsch 1914].

> SUMMARY OF CHAPTER 18. *The roots of a polynomial are equal to the eigenvalues of a colleague matrix formed from its coefficients in a Chebyshev series, tridiagonal except in the final row. This identity, combined with recursive subdivision, leads to a stable and efficient numerical method for computing roots of a polynomial in an interval. For orthogonal polynomials other than Chebyshev, the colleague matrix generalizes to a comrade matrix with the same almost-tridiagonal structure.*

Exercise 18.1. Companion matrix. Prove that the roots of the polynomial $p(x) = a_0 + a_1 x + \cdots + a_n x^n$, $a_n \neq 0$, are the eigenvalues of the $n \times n$ matrix with zero entries everywhere except for the value 1 in the first superdiagonal and the values $-a_0/a_n, \ldots, -a_{n-1}/a_n$ in the last row.

Exercise 18.2. Four forms of colleague matrix. A matrix C has the same eigenvalues and eigenvalue multiplicities as C^T and also as SCS^{-1}, where S is any nonsingular matrix. Use these properties to derive three alternative forms of the colleague matrix in which the Chebyshev coefficients appear in (a) the first row, (b) the first column, (c) the last column.

Exercise 18.3. Some forms more stable than others. Mathematically, all the matrices described in the last exercise have the same eigenvalues. Numerically, however, some may suffer more than others from rounding errors, and in fact Chebfun works with the first-column option for just this reason. (a) Determine the 11×11 colleague matrix corresponding to roots $-1, -0.8, -0.6, \ldots, 1$. Get the entries of the matrix exactly, either analytically or by intelligent guesswork based on Matlab's `rat` command. (b) How does the accuracy of the eigenvalues of the four matrix variants compare? Which one is best? Is the difference significant? (c) What happens if you solve the four eigenvalue problems again using Matlab's `'nobalance'` option in the `eig` command?

Exercise 18.4. Legendre polynomials. The Legendre polynomials satisfy $P_0(x) = 1$, $P_1(x) = x$, and for $k \geq 1$, the recurrence relation (17.6). (a) Derive a "comrade matrix" analogue of Theorem 18.1 for the roots of a polynomial expanded as a linear combination of Legendre polynomials. (b) Verify numerically that the roots of the particular polynomial $P_0 + P_1 + \cdots + P_5$ match the prediction of your theorem. (Try `sum(legpoly(0:5),2)` to construct this polynomial elegantly in Chebfun and don't forget `roots(...,'all')`.)

Exercise 18.5. Complex roots. For each of the following functions defined on $[-1,1]$, construct corresponding chebfuns and plot all their roots in the complex plane with `plot(roots(f,'all'))`. Comment on the patterns you observe. (Your comments are not expected to go very deep.) (a) $x^{20} - 1$, (b) $\exp(x)(x^{20} - 1)$, (c) $1/(1 + 25x^2)$, (d) $x\exp(30ix)$, (e) $\sin(10\pi x)$, (f) $\sqrt{1.1 - x}$, (g) An example of your own choosing.

Exercise 18.6. The Szegő curve. If f is entire, then it has no maximal Bernstein ellipse of analyticity. Plot the roots in the complex x-plane of the Chebfun polynomial approximation to e^x on $[-1,1]$, and for comparison, the "Szegő curve" defined by $|xe^{1-x}| = 1$ and $|x| \leq 1$ [Szegő 1924, Saff & Varga 1978b, Pritsker & Varga 1997].

Exercise 18.7. Roots of random polynomials. (a) Use Matlab's `roots` command to plot the roots of a polynomial $p(z) = a_0 + a_1 z + \cdots + a_{200} z^{200}$ with coefficients selected from the standard normal distribution. (b) Use `chebfun('randn(201,1)','coeffs')` and `plot(roots(p,'all'))` to plot the roots of a polynomial $p(x) = a_0 T_0 + a_1 T_1(x) + \cdots + a_{200} T_{200}(x)$ with the same kind of random coefficients. (Effects like these are analyzed rigorously in [Shiffman & Zelditch 2003].)

Chapter 19

Clenshaw–Curtis and Gauss quadrature

```
ATAPformats
```

One thing that is famous about Legendre points and polynomials is their connection with *Gauss quadrature*, invented by Gauss [1814]. Chebyshev points, similarly, are the basis of *Clenshaw–Curtis quadrature* [Clenshaw & Curtis 1960], and equispaced points are the basis of *Newton–Cotes quadrature.* Quadrature is the standard term for the numerical calculation of integrals. It is one of the areas where approximation theory has an immediate link to applications, as we shall see in Theorems 19.3–19.5.

In the basic quadrature problem, we are given a function $f \in C([-1,1])$ and wish to calculate

$$I = \int_{-1}^{1} f(x)\,dx. \tag{19.1}$$

(More generally the integral may include a weight function $w(x)$ as in (17.1).) There is a standard idea for doing this that is the basis of the Gauss, Clenshaw–Curtis, and Newton–Cotes formulas and many others besides. Given $n \ge 0$, we sample f at a certain set of $n+1$ distinct *nodes* $x_0, \dots, x_n$ in $[-1,1]$. We then approximate I by I_n, the exact integral of the degree n polynomial interpolant p_n of f in these nodes:

$$I_n = \int_{-1}^{1} p_n(x)\,dx. \tag{19.2}$$

One might wonder, why use a polynomial rather than some other interpolant? This is a very good question, and in Chapter 22 we shall see that other interpolants may in fact be up to $\pi/2$ times more efficient. Nevertheless, polynomial interpolants have been the standard idea in numerical quadrature since the 18th century.

To integrate p_n, we do not construct it explicitly. Instead, I_n is computed from the formula

$$I_n = \sum_{k=0}^{n} w_k f(x_k), \tag{19.3}$$

where the numbers $w_0, \dots, w_n$ are a set of $n+1$ *weights* that have been predetermined so that the value of I_n will come out right. From (5.1) it is clear that the weights must be the integrals of the Lagrange polynomials,

$$w_k = \int_{-1}^{1} \ell_k(x)\,dx. \tag{19.4}$$

Another way to write (19.3) is to say that I_n is given by an inner product,

$$I_n = w^T v, \tag{19.5}$$

where w and v are column vectors of the weights w_k and function values $f(x_k)$. Any linear process of computing an approximate integral from $n+1$ sample points must be representable in this inner product form, and the integration of polynomial interpolants is a linear process. The mapping from $\{f(x_k)\}$ to I_n is a *linear functional* (Exercise 19.1).

When the weights $\{w_k\}$ of a quadrature formula (19.3) are determined by the principle of integrating the polynomial interpolant, i.e. by (19.4), then the formula is said to be *interpolatory.* (Logically, the term should really be *polynomial interpolatory.*) For the following theorem, we say that a formula is *exact* when applied to f if the result it gives is the exactly correct integral of f.

Theorem 19.1. Polynomial degree of quadrature formulas. *For any $n \ge 0$, an $(n+1)$-point interpolatory quadrature formula such as Clenshaw–Curtis, Gauss, or Newton–Cotes is exact for $f \in \mathcal{P}_n$. The $(n+1)$-point Gauss formula is exact for $f \in \mathcal{P}_{2n+1}$.*

Proof. Since an interpolatory formula is constructed by integration of a polynomial interpolant of degree n, it is immediate that it is exact for $f \in \mathcal{P}_n$. The nontrivial property to be established is that Gauss quadrature achieves more than this, being exact for polynomials all the way up to degree $2n+1$. The following standard argument, based on orthogonal polynomials, comes from [Jacobi 1826]. Gauss's original work twelve years earlier was based on continued fractions rather than orthogonal polynomials.

Suppose that $f \in \mathcal{P}_{2n+1}$. Such a function can be written in the form $f(x) = P_{n+1}(x)\, q_n(x) + r_n(x)$, where P_{n+1} is the $(n+1)$st Legendre polynomial and $q_n, r_n \in \mathcal{P}_n$. This implies

$$I = \int_{-1}^{1} f(x)\, dx = \int_{-1}^{1} P_{n+1}(x)\, q_n(x)\, dx \, + \int_{-1}^{1} r_n(x)\, dx.$$

The first of the integrals on the right is zero because of the orthogonality property of Legendre polynomials, leaving us with

$$I = \int_{-1}^{1} r_n(x)\, dx.$$

Now consider I_n, the $(n+1)$-point Gauss quadrature approximation to I. The nodes of this formula are the zeros of $P_{n+1}(x)$. Accordingly, at each node x_k we have $f(x_k) = r_n(x_k)$. Thus the value I_n the Gauss formula gives for f will be the same as the value it gives for r_n. But $r_n \in \mathcal{P}_n$, so this value is exactly the integral of r_n, that is, $I_n = I$. ▮

Theorem 19.1 is famous, but we shall see that it is misleading. It suggests that there is a significant gap between Clenshaw–Curtis and Newton–Cotes quadrature, with one rate of convergence, and Gauss quadrature, with a rate twice as high. In fact, the great gap is between Newton–Cotes, which does not converge at all in general, and both Clenshaw–Curtis and Gauss, which converge for every continuous f and do so typically at similar rates.

First, let us give some more details of the Clenshaw–Curtis and Gauss formulas. For Clenshaw–Curtis quadrature, one way to compute I_n is by constructing the weight vector w explicitly. It can be shown that the weights are all positive and sum to 2 (the same properties also hold for Gauss quadrature weights, whose computation we discuss later in the chapter). From a practical point of view, this approach may be advantageous for integrating a collection of functions on a single Chebyshev grid. There is a classical formula for calculation of the weights with $O(n^2)$ operations [Davis & Rabinowitz 1984, Trefethen 2000], and it is also possible to compute the weights faster, in $O(n \log n)$ operations, using the FFT [Waldvogel 2006]. This fast algorithm is invoked by Chebfun when the command `chebpts` is called with two arguments, as we illustrate with $n+1=3$:

```
[nodes,weights] = chebpts(3)
```

```
nodes =
    -1
     0
     1
weights =
   0.333333333333333   1.333333333333333   0.333333333333333
```

By increasing 3 to one million we see the speed of Waldvogel's algorithm:

```
tic, [nodes,weights] = chebpts(1000000); toc
```

```
Elapsed time is 1.151043 seconds.
```

The other way to carry out Clenshaw–Curtis quadrature, simplest when just one or a small number of integrands are involved, is to use the FFT to transform the problem to coefficient space (see Chapter 3) at a cost of $O(n \log n)$ operations per integrand. (This idea was not proposed by Clenshaw and Curtis, who wrote before the rediscovery of the FFT in 1965, but by Morven Gentleman a few years later [Gentleman 1972a, 1972b].) To see how this works, we observe that the integral of the Chebyshev polynomial T_k from -1 to 1 is zero if k is odd and

$$\int_{-1}^{1} T_k(x)\,dx = \frac{2}{1-k^2} \tag{19.6}$$

if k is even (Exercise 19.6). This gives us the following theorem, the basis of the FFT realization of Clenshaw–Curtis quadrature:

Theorem 19.2. Integral of a Chebyshev series. *The integral of a degree n polynomial expressed as a Chebyshev series is*

$$\int_{-1}^{1} \sum_{k=0}^{n} c_k T_k(x)\,dx \;=\; \sum_{k=0,\,k\,\mathrm{even}}^{n} \frac{2c_k}{1-k^2}.$$

Proof. Follows from (19.6). ∎

Chebfun applies Theorem 19.2 every time one types `sum(f)`, and this theorem is also the basis of Waldvogel's algorithm mentioned above.

By combining (19.6) with Theorems 8.1 and 19.1, we can now write down a theorem about the geometric convergence of Clenshaw–Curtis and Gauss quadrature for analytic integrands. For Gauss quadrature, this estimate is due to Rabinowitz [1969], and the extension to Clenshaw–Curtis can be found in [Trefethen 2008]. This result is fundamental and very important. *For analytic integrands, the Gauss and Clenshaw–Curtis formulas converge geometrically.* Every numerical analysis textbook should state this fact.

Theorem 19.3. Quadrature formulas for analytic integrands. *Let a function f be analytic in $[-1,1]$ and analytically continuable to the open Bernstein ellipse E_ρ, where it satisfies $|f(z)| \le M$ for some M. Then $(n+1)$ -point Clenshaw–Curtis quadrature with $n \ge 2$ applied to f satisfies*

$$|I - I_n| \le \frac{64}{15}\frac{M\rho^{1-n}}{\rho^2-1} \tag{19.7}$$

and $(n+1)$ -point Gauss quadrature with $n \ge 1$ satisfies

$$|I - I_n| \le \frac{64}{15}\frac{M\rho^{-2n}}{\rho^2-1}. \tag{19.8}$$

The factor ρ^{1-n} in (19.7) can be improved to ρ^{-n} if n is even, and the factor $64/15$ can be improved to $144/35$ if $n \ge 4$ in (19.7) or $n \ge 2$ in (19.8).

Proof. If the constants $64/15$ are increased to 8 and ρ^2-1 is reduced to $\rho-1$, these conclusions can be obtained as corollaries of Theorem 8.2. The key is to note that that the error in integrating f will be the same as the error in integrating $f - f_n$. Applying the triangle inequality, this gives us

$$|I - I_n| \le |I(f - f_n)| + |I_n(f - f_n)|.$$

By Theorem 8.2, $|(f - f_n)(x)| \le 2M\rho^{-n}/(\rho-1)$ for each $|x|$. Since the interval $[-1,1]$ has length 2, this implies

$$|I(f - f_n)| \le \frac{4M\rho^{-n}}{\rho-1}.$$

In addition to this, there also holds the analogous property

$$|I_n(f - f_n)| \le \frac{4M\rho^{-n}}{\rho-1}.$$

This follows from the fact that the weights are positive.

To get the sharper results stated, we use an additional fact: both Gauss and Clenshaw–Curtis formulas get the right answer when integrating an odd function, namely zero. In particular the error is zero in integration of $T_k(x)$

for any odd k. Now by Theorem 19.1, Gauss quadrature is exact through the term of degree $2n+1$ in the Chebyshev expansion of f. Since odd terms do not contribute, we see that the error in integrating f by $(n+1)$-point Gauss quadrature will thus be the error in integrating

$$a_{2n+2}T_{2n+2}(x) + a_{2n+4}T_{2n+4}(x) + \dots,$$

a series in which the smallest index that appears is at least 4. Now by (19.6), the true integral of T_k for $k \ge 4$ is at most $2/15$. When T_k is integrated over $[-1,1]$ by the Gauss quadrature formula, the result will be at most 2 since the weights are positive and add up to 2. Thus the error in integrating each T_k is at most $2 + 2/15 = 32/15$. Combining this estimate with the bound $|a_k| \le 2M\rho^{-k}$ of Theorem 8.1 gives (19.8). The argument for (19.7) is analogous. For the improvement from 64/15 to 144/35, see Exercise 19.5. ∎

Just as Theorem 19.3 follows from the results of Chapter 8 for analytic integrands, there is an analogous result for differentiable integrands based on the results of Chapter 7.

Theorem 19.4. Quadrature formulas for differentiable integrands. *For any $f \in C([-1,1])$, both the Clenshaw–Curtis and Gauss approximations I_n converge to the integral I as $n \to \infty$. For an integer $\nu \ge 1$, let f and its derivatives through $f^{(\nu-1)}$ be absolutely continuous on $[-1,1]$ and suppose the νth derivative $f^{(\nu)}$ is of bounded variation V. Then $(n+1)$-point Clenshaw–Curtis quadrature applied to f satisfies*

$$|I - I_n| \le \frac{32}{15}\frac{V}{\pi\nu(n-\nu)^{\nu}} \tag{19.9}$$

for $n > \nu$ and $(n+1)$-point Gauss quadrature satisfies

$$|I - I_n| \le \frac{32}{15}\frac{V}{\pi\nu(n-2\nu-1)^{2\nu+1}} \tag{19.10}$$

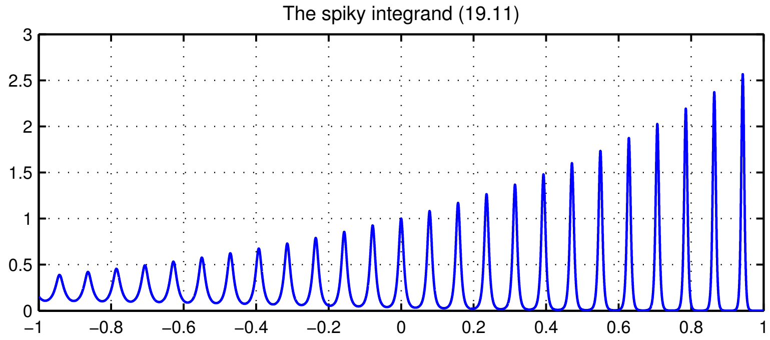
for $n > 2\nu + 1$.

Proof. The first assertion, for arbitrary continuous f, is due to Stieltjes [1884]. As for (19.9) and (19.10), these can be derived as in the previous proof, but now using Theorem 7.2. ∎

Here is a numerical example, the integration of the function (18.1) with a sequence of spikes:

$$I = \int_{-1}^{1} e^x\,[\operatorname{sech}(4\sin(40x))]^{\exp(x)}\,dx \tag{19.11}$$

```
ff = @(x) exp(x).*sech(4*sin(40*x)).^exp(x);
x = chebfun('x'); f = ff(x);
FS = 'fontsize';
clf, plot(f), grid on, title('The spiky integrand (19.11)',FS,9)
```



The corresponding chebfun is not exactly short:

```
length(f)
```

```
ans =
        3679
```

Nevertheless, Chebfun computes its integral to 15 digits of accuracy in a fraction of a second:

```
sum(f)
```

```
ans =
   0.543384000907901
```

Now let us look at Gauss quadrature. The nodes for the $n+1$-point Gauss formula are the roots of the Legendre polynomial $P_{n+1}(x)$. A good method for computing these numbers is implicit in Theorem 18.1 and the comment after it. According to that theorem, the roots of a polynomial expressed as a Chebyshev series are equal to the eigenvalues of a colleague matrix whose structure is tridiagonal apart from a nonzero final row. If the Chebyshev series reduces to the single polynomial T_{n+1}, the matrix reduces to tridiagonal without the extra row. Similarly the roots of a polynomial expressed as a series in Legendre polynomials are the eigenvalues of a comrade matrix, which is again tridiagonal except for a final row, and for the roots of P_{n+1}

itself, the matrix reduces to tridiagonal. When symmetrized, this matrix is called a *Jacobi matrix* (Exercise 19.7). The classic numerical algorithm for implementing Gauss quadrature formulas comes from Golub and Welsch in 1969, who showed that the weights as well as the nodes can be obtained by solving the eigenvalue problem for this Jacobi matrix [Golub & Welsch 1969]. The Golub–Welsch algorithm can be coded in six lines of Matlab (see `gauss.m` in [Trefethen 2000]), and the operation count is in principle $O(n^2)$, although it is $O(n^3)$ in the simple implementation since Matlab does not offer a command to exploit the tridiagonal structure of the eigenvalue problem.

For larger values of n, a much faster alternative algorithm was introduced by Glaser, Liu, and Rokhlin [2007], based on numerical solution of certain linear ordinary differential equations by high-order Taylor series approximations combined with Newton iteration. This GLR algorithm shrank the operation count dramatically to $O(n)$ and became the default algorithm invoked by Chebfun during 2009–2012 when the `legpts` command is called with two output arguments. Most recently an even faster algorithm has been introduced by Hale and Townsend [2012], which is Chebfun's default at the time of this writing. The key idea of the Hale–Townsend algorithm is to start from high accuracy asymptotic approximations for nodes and then take one or two Newton steps, with P_n and P_n' evaluated to machine precision by known asymptotic formulas. When n is large enough, one may not even need any Newton steps at all. A crucial feature is that the method treats the nodes independently, so that it vectorizes readily, and this is a primary reason why it is approximately 20 times faster than the GLR algorithm in a Matlab implementation.

Following the illustration of Clenshaw–Curtis quadrature earlier, here are nodes and weights for Gauss quadrature with $n + 1 = 3$:

```
[nodes,weights] = legpts(3)
```

```
nodes =
  -0.774596669241483
                   0
   0.774596669241483
weights =
   0.555555555555556   0.888888888888889   0.555555555555556
```

And here is the time it takes to compute Gauss quadrature nodes and weights for one million points, not much slower than Clenshaw–Curtis:

```
tic, [nodes,weights] = legpts(1000000); toc
```

```
Elapsed time is 0.611738 seconds.
```

For example, here is the integral (19.11) computed by n-point Gauss quadrature for various values of n. We write `w*gg(s)` rather than `w’*gg(s)` since w as returned by `legpts` is a row vector, not a column vector.

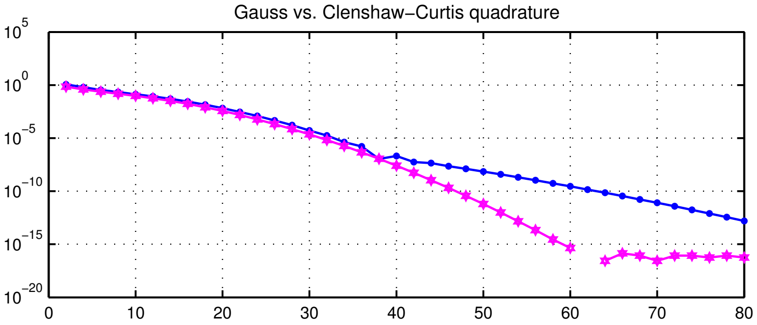
```
for n = 500:500:2000
  tic
  [s,w] = legpts(n+1);
  I = w*ff(s); t = toc;
  fprintf(’n = %4d,  I = %16.14f,  time = %6.4f\n’,n,I,t)
end
```

```
n =  500,  I = 0.54339275810622,  time = 0.0627
n = 1000,  I = 0.54338400182558,  time = 0.0161
n = 1500,  I = 0.54338400090784,  time = 0.0109
n = 2000,  I = 0.54338400090790,  time = 0.0085
```

Gauss quadrature has not often been employed for numbers of nodes in the thousands, because with traditional algorithms the computations are too expensive. It is clear from this experiment that the GLR and Hale–Townsend algorithms make such computations feasible after all.

So is Gauss quadrature the formula of choice? In particular, how does it compare with Clenshaw–Curtis quadrature as $n \to \infty$? As mentioned above, the traditional expectation, based on Theorem 19.1 and seemingly supported by Theorems 19.3 and 19.4, is that Gauss should converge twice as fast as Clenshaw–Curtis. However, numerical experiments show that the truth is not so simple. We begin with the easy integrand $f(x) = \exp(-100x^2)$.

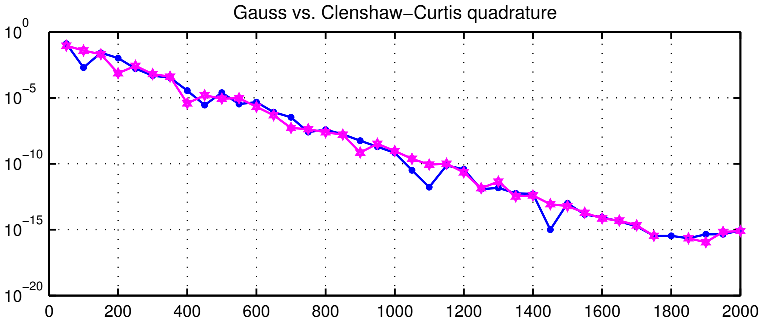
```
gg = @(x) exp(-100*x.^2);
I = sum(chebfun(gg));
errcc = []; errgauss = [];
nn = 2:2:80;
for n = nn
   Icc = sum(chebfun(gg,n+1));
   errcc = [errcc abs(I-Icc)];
   [s,w] = legpts(n+1);
   Igauss = w*gg(s);
   errgauss = [errgauss abs(I-Igauss)];
end
hold off, semilogy(nn,errcc,'.-','markersize',10), grid on
hold on, semilogy(nn,errgauss,'h-m','markersize',4), grid on
title('Gauss vs. Clenshaw-Curtis quadrature',FS,9)
```



This behavior is typical: for smaller values of n, Clenshaw–Curtis (dots) and Gauss quadrature (stars) have similar accuracy, not a difference of a factor of 2. This effect was pointed out by Clenshaw and Curtis in their original paper [1960]. Only at a sufficiently large value of n, if the integrand is analytic, does a kink appear in the Clenshaw–Curtis convergence curve, whose further convergence is then about half as fast as before. An explanation of this effect based on ideas of rational approximation is given in Figures 4–6 of [Trefethen 2008], and another explanation based on aliasing can be derived from Theorems 4.2 and 19.2 and goes back to O'Hara and Smith [1968] (Exercise 19.4). For a full analysis, see [Weideman & Trefethen 2007].

Here is a similar comparison for the harder integral (19.11):

```
I = sum(f);
errcc = []; errgauss = []; tcc = []; tgauss = [];
nn = 50:50:2000;
for n = nn
   tic, Icc = sum(chebfun(ff,n+1)); t = toc;
   tcc = [tcc t]; errcc = [errcc abs(I-Icc)];
   tic, [s,w] = legpts(n+1); t = toc;
   Igauss = w*ff(s);
   tgauss = [tgauss t]; errgauss = [errgauss abs(I-Igauss)];
end
hold off, semilogy(nn,errcc,'.-','markersize',10), grid on
hold on, semilogy(nn,errgauss,'h-m','markersize',4)
title('Gauss vs. Clenshaw-Curtis quadrature',FS,9)
```



This time, for the values of n under study, the kink does not appear at all. Clenshaw–Curtis has approximately the same accuracy as Gauss throughout, and in particular, it obtains the correct integral to machine precision by around $n = 1800$, which is about half the length of the chebfun, `length(f)`, reported earlier! This is typical of Clenshaw–Curtis quadrature: just as with Gauss quadrature, the quadrature value often converges about twice as fast as the underlying polynomial approximation, even though Theorems 19.1, 19.3, and 19.4 give no hint of such behavior.

There is a theorem that substantiates this effect. The following result, whose proof we shall not give, comes from [Trefethen 2008].

Theorem 19.5. Clenshaw–Curtis quadrature for differentiable integrands. *Under the hypotheses of Theorem* 19.4*, the same conclusion* (19.10) *also holds for* $(n+1)$*-point Clenshaw–Curtis quadrature:*

$$|I - I_n| \le \frac{32}{15}\frac{V}{\pi\nu(n-2\nu-1)^{2\nu+1}}. \tag{19.12}$$

The only difference is that this bound applies for all sufficiently large n (depending on ν but not f) rather than for $n > 2\nu + 1$.

Proof. See [Trefethen 2008]. Here, the definition of V is somewhat different from the one in [Trefethen 2008], but this does not affect the argument leading to (19.12). ∎

All in all, though Gauss quadrature is more celebrated than Clenshaw–Curtis, and certainly has some beautiful properties, its behavior in practice is often not very much different.

For an extensive survey of many aspects of Gauss quadrature, see [Gautschi 1981], and for general information about numerical integration, see [Davis & Rabinowitz 1984]. In practical applications and software implementations it is common to use adaptive formulas of low or moderate order rather than letting n increase toward ∞ with a global grid, though Chebfun is an exception to this pattern.

As mentioned earlier, both Gauss and Clenshaw–Curtis quadrature grids can be improved by a factor approaching $\pi/2$ by the introduction of a change of variables, taking us beyond the realm of polynomial approximations. These ideas are discussed in Chapter 22.

We have not said much about Newton–Cotes quadrature formulas, based on equispaced points. For smaller orders these are of practical interest: $n = 4$ gives *Simpson's rule*, and Espelid has used Newton–Cotes rules of order up to 33 as the basis of excellent codes `coted2a` and `da2glob` for adaptive quadrature [Espelid 2004]. The weights $\{w_j\}$ of Newton–Cotes formula, however, oscillate in sign between magnitudes on the order of 2^n, a reflection of the Runge phenomenon, causing terrible numerical instability for large n. Even in the absence of rounding errors, the results of Newton–Cotes formulas do not converge in general as $n \to \infty$, even for analytic functions. It was clear upon publication of Runge's paper in 1901 that such divergence was likely, and a theorem to this effect was proved by Pólya [1933].

We close this chapter by mentioning an elegant application of Gauss quadrature nodes and weights pointed out by Wang and Xiang [2012].

Theorem 19.6. Barycentric weights for Legendre points. *Let the numbers $\lambda_0, \ldots, \lambda_k$ be defined by*

$$\lambda_k = (-1)^k \sqrt{(1 - x_k^2) w_k}, \tag{19.13}$$

where $\{x_k\}$ and $\{w_k\}$ are the nodes and weights for $(n+1)$-point Gauss quadrature. If these numbers are taken as weights in the barycentric formula (5.11)*, they yield the polynomial interpolant through Legendre points.*

Proof. See Theorem 3.1 of [Wang & Xiang 2012]. ∎

In view of the Glaser–Liu–Rokhlin algorithm for Gauss quadrature, this theorem implies that polynomial interpolants in Legendre points, like Chebyshev points, can be evaluated in $O(n)$ operations. The formulas are implemented in Chebfun and accessed when one calls `legpts`, `jacpts`, `hermpts` or `lagpts` with three output arguments [Hale & Trefethen 2012].

Summary of Chapter 19. *Clenshaw–Curtis quadrature is derived by interpolating a polynomial interpolant in Chebyshev points, and Gauss quadrature from Legendre points. The nodes and weights for both families can be computed quickly and accurately, even for millions of points. Though Gauss has twice the polynomial order of accuracy of Clenshaw–Curtis, their rates of convergence are approximately the same for non-analytic integrands.*

Exercise 19.1. Riesz Representation Theorem. (a) Look up the Riesz Representation Theorem and write down a careful mathematical statement of it . (b) Show that the computation of an approximate integral I_n from $n+1$ samples of a function $f \in C([-1,1])$ by integrating the degree n polynomial interpolant through a fixed set of $n+1$ nodes in $[-1,1]$ is an example of the kind of linear functional to which this theorem applies, provided we work in a finite-dimensional space rather than all of $C([-1,1])$. (c) In what sense is the Riesz Representation Theorem significantly more general than is needed for this particular application to quadrature?

Exercise 19.2. quad, quadl, quadgk. Evaluate (19.11) with Matlab's `quad`, `quadl`, and `quadgk` commands. As a function of the specified precision, what is the actual accuracy obtained and how long does the computation take? How do these results compare with Chebfun `sum`?

Exercise 19.3. Quadrature weights. (a) Use Chebfun to illustrate the identity (19.4) for Clenshaw–Curtis quadrature in the case $n = 20$, $k = 7$. (b) Do the same for Gauss quadrature.

Exercise 19.4. Accuracy of Clenshaw–Curtis quadrature. (a) Using theorems of Chapters 4 and 19, derive an exact expression for the error $I - I_n$ in Clenshaw–Curtis quadrature applied to the function $f(x) = T_k(x)$ for $k > n$. (b) [to be continued. See eqs (9) and (9') of Gentleman [1972a].]

Exercise 19.5. Sharpening Theorem 19.3. Suppose we assume $n \geq 2$ instead of $n \geq 1$ in the Gauss quadrature bound of Theorem 19.3. Show why the constant 64/15 improves to 144/35. What is this actual "constant" as a function of n?

Exercise 19.6. Integral of a Chebyshev polynomial. Derive the formula (19.6) for the integral of $T_k(x)$ with k even. (Hint: Following the proof of Theorem 3.1, replace $T_k(x)dx$ by $(z^k + z^{-k})(dx/dz)dz$.)

Exercise 19.7. Symmetrization in the Golub–Welsch algorithm. The nodes $\{x_j\}$ of the $(n+1)$-point Gauss quadrature rule are the zeros of the Legendre polynomial P_{n+1}. From the recurrence relation (17.6), it follows as in Theorem 18.1 that they are the eigenvalues of the $(n+1) \times (n+1)$ tridiagonal matrix with zeros on the main diagonal, [xxx] on the first superdiagonal, and [xxx] on the first subdiagonal. Find the unique diagonal matrix $D = \mathrm{diag}(d_0, \dots, d_n)$ with $d_0 = 1$ and $d_j > 0$ for $j \geq 1$ such that $B = DAD^{-1}$, which has the same eigenvalues as A, is real symmetric. What are the entries of B? (This symmetrized matrix is the Jacobi matrix that is the basis of the Golub–Welsch algorithm.)

Exercise 19.8. Integrating the Bernstein polynomial. Given $f \in C([-1,1])$, let $B_n(x)$ be the Bernstein polynomial defined by (6.1) and let I_n be the approximation to $\int_{-1}^{1} f(x)dx$ defined by $I_n = \int_{-1}^{1} B_n(x)dx$. (a) Show that $I_n = (n+1)^{-1}\sum_{k=1}^{n} f(k/n)$. (b) Is this an interpolatory quadrature formula? (c) What is its order of accuracy α as defined by the condition $I - I_n = O(n^{-\alpha})$?

Chapter 20

Caratheodory–Fejer approximation

```
ATAPformats
```

We have seen that Chebyshev interpolants are near-best approximations in the sense that they come within a factor of at most $O(\log n)$ of best approximations, usually even closer. For most applications, this is all one could ask for. But there is another kind of near-best approximations that are so close to best that for smooth functions, they are often indistinguishable from best approximations to machine precision on a computer. These are *CF (Carathéodory–Fejér) approximations*, introduced by Gutknecht and Trefethen [1982]. Earlier related ideas were proposed in [Darlington 1970, Elliott 1973, Lam 1972, Talbot 1976], and the theoretical basis goes back to the early 20th century [Carathéodory & Fejér 1911, Schur 1918].[1]

Before explaining the mathematics of CF approximants, let us illustrate the remarkable degree of near-optimality they sometimes achieve. Here is the optimal ∞-norm error in approximation of $f(x) = e^x$ on $[-1, 1]$ by a polynomial of degree 2:

```
x = chebfun('x'); format long
f = exp(x); n = 2;
pbest = remez(f,n);
errbest = norm(f-pbest,inf)
```

[1] Logically, this chapter could have appeared earlier, perhaps just after Chapter 10. We have deferred it to this point of the book, however, since the material is relatively difficult and none of the other chapters depend on it.

```
Warning:  This command is deprecated. Use minimax instead.
errbest =
   0.045017388402824
```

Here is the corresponding error for CF approximation computed by the Chebfun `cf` command:

```
pcf = cf(f,n);
errcf = norm(f-pcf,inf)


errcf =
   0.045017388414604
```

These two numbers agree to an extraordinary 9 significant digits. Comparing the best and CF polynomials directly to one another, we confirm that they are almost the same:

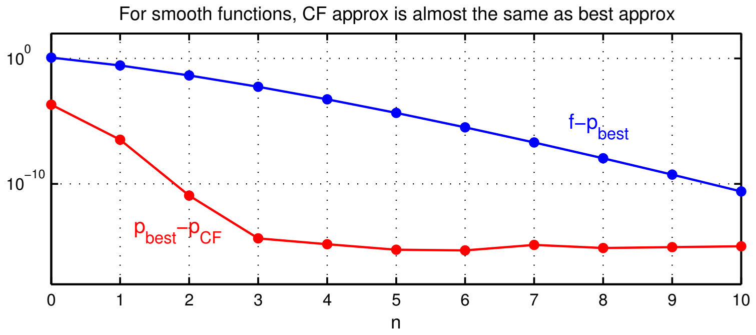
```
norm(pbest-pcf,inf)


ans =
     1.179117914418271e-11
```

That was for degree $n = 2$, and the near-optimality of the CF approximants grows stronger as n increases. Let us explore the dependence on n. On a semilog plot, the upper curve in the next figure shows the accuracy of the best polynomial as an approximation to $f(x)$, while the lower curve shows the accuracy of the CF polynomial as an approximation to the best polynomial. The two errors are of entirely different orders, and for $n > 3$, the CF and best polynomials are indistinguishable in floating point arithmetic.

```
nn = 0:10; err1 = []; err2 = [];
for n = nn
    pbest = remez(f,n); err1 = [err1 norm(f-pbest,inf)];
    pcf = cf(f,n); err2 = [err2 norm(pbest-pcf,inf)];
end
hold off, semilogy(nn,err1,'.-'), grid on
hold on, semilogy(nn,err2,'.-r')
FS = 'fontsize';
text(7.5,2e-6,'f-p_{best}','color','b',FS,10)
text(1.2,1e-14,'p_{best}-p_{CF}','color','r',FS,10)
ylim([1e-18,1e2]), xlabel('n',FS,9)
title(['For smooth functions, ' ...
     'CF approx is almost the same as best approx'],FS,9)
```

```
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
```

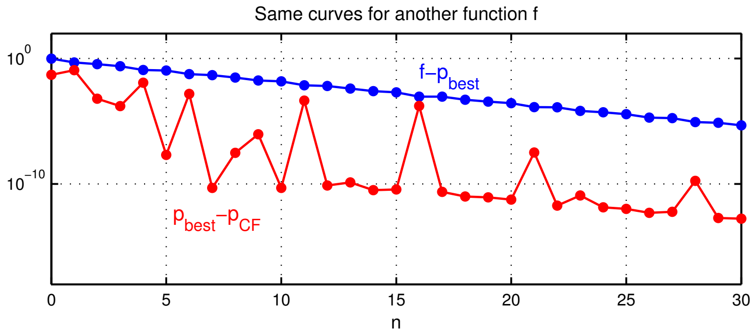


Here is the same experiment repeated for $f(x) = \tanh(4(x-0.3))$.

```
f = tanh(4*(x-.3));
nn = 0:30; err1 = []; err2 = [];
for n = nn
    pbest = remez(f,n); err1 = [err1 norm(f-pbest,inf)];
    pcf = cf(f,n); err2 = [err2 norm(pbest-pcf,inf)];
end
hold off, semilogy(nn,err1,'.-'), grid on
hold on, semilogy(nn,err2,'.-r')
text(16,2e-2,'f-p_{best}','color','b',FS,10)
text(5.3,1e-13,'p_{best}-p_{CF}','color','r',FS,10)
ylim([1e-18,1e2]), xlabel('n',FS,9)
title('Same curves for another function f',FS,9)
```

```
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
```

```
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
```



Again we see that `pbest−pcf` is much smaller than `f−pbest`, implying that the CF approximant is for practical purposes essentially optimal. (Concerning the erratic oscillations, see Exercise 20.4.) Yet it is far easier to compute:

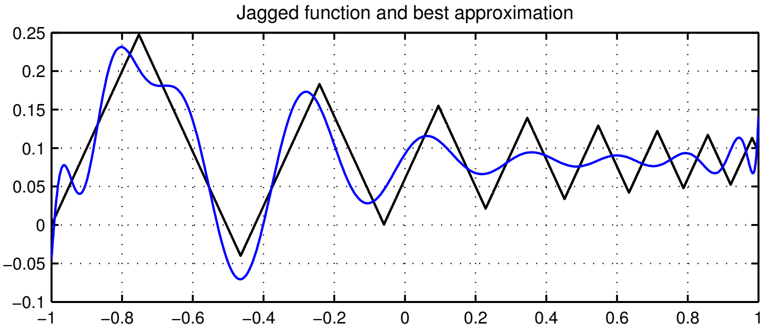
```
tic, remez(f,20); tbest = toc
tic, cf(f,20); tcf = toc
```

```
Warning:  This command is deprecated. Use minimax instead.
tbest =
   0.755624000000000
tcf =
   0.074389000000000
```

Turning to a non-smooth function, here again is the jagged example from Chapter 10 with its best approximation of degree 20:

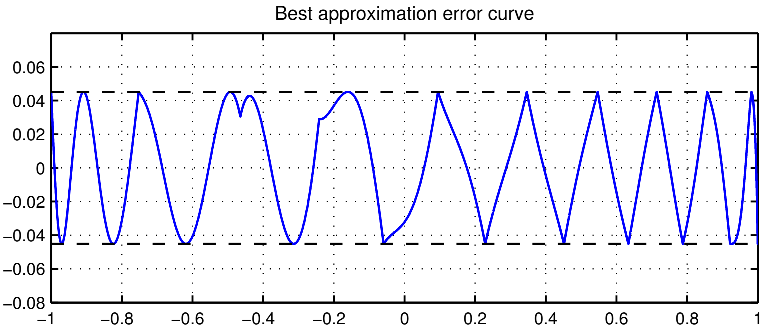
```
f = cumsum(sign(sin(20*exp(x))));
hold off, plot(f,'k'), grid on
tic, [pbest,err] = remez(f,20); tbest = toc;
hold on, plot(pbest)
title('Jagged function and best approximation',FS,9)
```

```
Warning:  This command is deprecated. Use minimax instead.
```



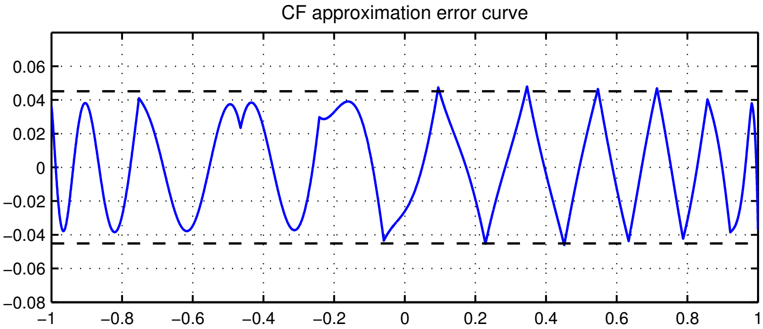
We saw the error curve before:

```
hold off, plot(f-pbest), grid on, hold on, axis([-1 1 -.08 .08])
plot([-1 1],err*[1 1],'--k'), plot([-1,1],-err*[1 1],'--k')
title('Best approximation error curve',FS,9)
```



In CF approximation, we must start from a polynomial, not a jagged function. As a rule of thumb, truncating the Chebyshev series at 5 times the degree of the desired approximation is usually pretty safe. Here is what we get:

```
f100 = chebfun(f,100);
tic, pcf = cf(f100,20); tcf = toc;
hold off, plot(f-pcf), grid on, hold on, axis([-1 1 -.08 .08])
plot([-1 1],err*[1 1],'--k'), plot([-1,1],-err*[1 1],'--k')
title('CF approximation error curve',FS,9)
```



Evidently the error falls short of optimality by just a few percent. Yet again the computation is much faster:

```
tbest
```

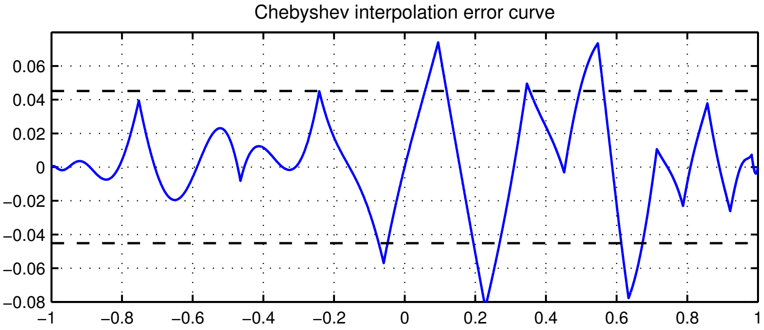
```
tbest =
   2.452415000000000
```

```
tcf
```

```
tcf =
   0.120896000000000
```

Here for comparison is the error in Chebyshev interpolation.

```
pinterp = chebfun(f,21);
hold off, plot(f-pinterp), grid on, hold on, axis([-1 1 -.08 .08])
plot([-1 1],err*[1 1],'--k'), plot([-1,1],-err*[1 1],'--k')
title('Chebyshev interpolation error curve',FS,9)
```



The time has come to describe what CF approximation is all about. We shall see that the hallmark of this method is the use of eigenvalues and eigenvectors (or singular values and singular vectors) of a Hankel matrix of Chebyshev coefficients.

We start with a real function f on $[-1,1]$, which we want to approximate by a polynomial of degree $n \geq 0$. Following Theorem 3.1, we assume that f is Lipschitz continuous, so it has an absolutely convergent Chebyshev series

$$f(x) = \sum_{k=0}^{\infty} a_k T_k(x).$$

Since our aim is polynomial approximation, there is no loss of generality if we suppose that $a_0 = a_1 = \cdots = a_n = 0$, so that the Chebyshev series of f begins at the term T_{n+1}. For technical simplicity, let us further suppose that the series is a finite one, ending at the term T_N for some $N \geq n+1$. Then f has the Chebyshev series

$$f(x) = \sum_{k=n+1}^{N} a_k T_k(x).$$

We now transplant f to a function F on the unit circle in the complex z-plane by defining $F(z) = F(z^{-1}) = f(x)$ for $|z| = 1$, where $x = \operatorname{Re} z = (z+z^{-1})/2$.

As in the proof of Theorem 3.1, this gives us a formula for F as a Laurent polynomial,

$$F(z) = \frac{1}{2}\sum_{k=n+1}^{N} a_k(z^k + z^{-k}).$$

We can divide F into two parts, $F(z) = G(z) + G(z^{-1})$, with

$$G(z) = \frac{1}{2}\sum_{k=n+1}^{N} a_k z^k.$$

The function G is called the *analytic part* of F, since it can be analytically continued to an analytic function in $|z| \le 1$. Similarly $G(z^{-1})$ is the *coanalytic part* of F, analytic for $1 \le |z| \le \infty$.

Now we ask the following question: what is the best approximation $\tilde{P}$ to G on the unit circle of the form

$$\tilde{P}(z) = \frac{1}{2}\sum_{k=-\infty}^{n} b_k z^k, \tag{20.1}$$

where the series converges for all z with $1 \le |z| < \infty$? In other words, $\tilde{P}$ must be analytic in the exterior of the unit disk apart from a pole of order at most n at $z = \infty$. This is the problem that Carathéodory and Fejér solved, and the solution is elegant. First of all, $\tilde{P}$ exists, and it is unique. Secondly, $G - \tilde{P}$ maps the unit circle onto a perfect circle that winds counterclockwise around the origin a number of times: the winding number is at least $n + 1$. Third, as shown by Schur a few years after Carathéodory and Fejér [Schur 1918], $\tilde{P}$ can be constructed explicitly by solving a certain matrix singular value problem. Let H denote the $(N - n) \times (N - n)$ real symmetric matrix of Chebyshev coefficients arranged like this,

$$H = \begin{pmatrix} a_{n+1} & a_{n+2} & a_{n+3} & \dots & a_N \\ a_{n+2} & a_{n+3} & & & \\ a_{n+3} & & & & \\ \vdots & & & & \\ a_N & & & & \end{pmatrix}, \tag{20.2}$$

where the entries in the lower-right triangle are zero. A matrix with this structure, constant along diagonals so that a_{ij} depends only on $i + j$, is called a *Hankel matrix*. Let λ be the largest eigenvalue of H in absolute value, let $u = (u_0, u_1, \dots, u_{N-n-1})^T$ be a corresponding real eigenvector, and define

$$u(z) = u_0 + u_1 z + \cdots + u_{N-n-1} z^{N-n-1}.$$

Here is the theorem due to Carathéodory and Fejér and Schur.

Theorem 20.1. Carathéodory–Fejér–Schur theorem. *The approximation problem described above has a unique solution $\tilde{P}$, and it is given by the error formula*

$$(G-\tilde{P})(z) = \lambda z^{n+1}\, \frac{u(z)}{\overline{u(z)}}. \tag{20.3}$$

The function $G-\tilde{P}$ maps the unit circle to a circle of radius $|\lambda|$ and winding number $\ge n+1$, and if $|\lambda| > |\mu|$ for all other eigenvalues μ, the winding number is exactly $n+1$.

Proof. The result is due to Carathéodory and Fejér [1911] and Schur [1918]. See Theorem 1.1 of [Gutknecht & Trefethen 1982] and Theorem 4 of [Hayashi, Trefethen & Gutknecht 1990]. ∎

Theorem 20.1 is a mathematical assertion about the approximation of a function G on the unit circle by an infinite series. We use this result to construct the polynomial CF approximant as follows. Since $G-\tilde{P}$ maps the unit circle to a circle of winding number $\ge n+1$, its real part (times 2)

$$(G-\tilde{P})(z) + (G-\tilde{P})(z^{-1})$$

maps $[-1,1]$ to an equioscillating curve with at least $n+2$ extreme points. Thus the function

$$\tilde{p}(x) = \tilde{P}(z) + \tilde{P}(z^{-1})$$

yields the equioscillatory behavior that characterizes a best approximation polynomial of degree n to $f(x)$ on $[-1,1]$ (Theorem 10.1). Unfortunately, $\tilde{p}(x)$ is not a polynomial of degree n. However, it will generally be very close to one. The function $\tilde{P}$ will normally have Laurent series coefficients b_k that decay as $k\to-\infty$. We truncate these at degree $-n$ to define

$$\mathcal{P}_{\rm CF}(z) = \frac{1}{2}\sum_{k=-n}^{n} b_k z^k,$$

with real part (times 2)

$$p_{\rm CF}(x) = \mathcal{P}_{\rm CF}(z) + \mathcal{P}_{\rm CF}(z^{-1}) = \frac{1}{2}\sum_{k=-n}^{n} (b_k + b_{-k}) z^k.$$

If the truncated terms are small, $f - p_{\rm CF}$ maps $[-1,1]$ to a curve that comes very close to equioscillation with $\ge n+2$ extrema, and thus $p_{\rm CF}$ is close to optimal.

For more details on real polynomial CF approximation, with numerical examples, see [Gutknecht & Trefethen 1982], [Trefethen 1983], and [Hayashi, Trefethen & Gutknecht 1990].

Our experiments in the opening pages of this chapter showed that CF approximants can be exceedingly close to best. The truncation described above gives an idea of how this happens. In the simplest case, suppose f is an analytic function on $[-1,1]$. Then by Theorem 8.1, its Chebyshev coefficients decrease geometrically, and let us suppose that this happens smoothly at a rate $a_k = O(\rho^k)$. Then, roughly speaking, the dominant degree $n+1$ term of f is of order ρ^{-n-1}, and the terms $b_n, b_{n-1}, \dots, b_{-n}$ are of orders $\rho^{-n-2}, \rho^{-n-3}, \dots, \rho^{-3n-2}$. This suggests that the truncation in going from $\tilde{p}$ to p_{CF} will introduce an error of order ρ^{-3n-3}. This is usually a very small number, and in particular, much smaller than the error $\|f - p^*\|$ of order ρ^{-n-1}.

In fact, the actual order of accuracy for polynomial CF approximation is one order higher, ρ^{-3n-4} rather than ρ^{-3n-3}. (The reason is that the first truncated term is a multiple of T_{3n+3}, the same Chebyshev polynomial that dominates the error $f - p^*$ itself, and so it is not until the second truncated term, T_{3n+4}, that the equioscillation is broken.) On the other hand, to go from this rough argument to a precise theorem is not so easy, because in fact, Chebyshev series need not decay smoothly (Exericse 20.3). Here we quote without proof a theorem from [Gutknecht & Trefethen 1982].

Theorem 20.2. Accuracy of polynomial CF approximation. *For any fixed $m \ge 0$, let f have a Lipschitz continuous $(3m+3)$rd derivative on $[-1,1]$ with a nonzero $(m+1)$st derivative at $x=0$, and for each $s \in (0,1]$, let p^* and p_{CF} be the best and the CF approximations of degree m to $f(sx)$ on $[-1,1]$, respectively. Then as $s \to 0$,*

$$\|f - p^*\| = O(s^{m+1}) \tag{20.4}$$

and

$$\|f - p^*\| \ne O(s^{m+2}) \tag{20.5}$$

and

$$\|p_{CF} - p^*\| = O(s^{3m+4}). \tag{20.6}$$

Proof. See Theorem 3.4 of [Gutknecht & Trefethen 1982]. ▮

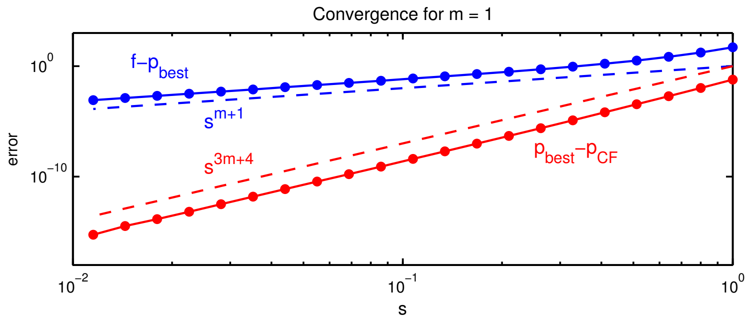
We can verify this result numerically. The two plots below display norms for $m = 1$ and $m = 2$ for the function $f(x) = e^{5x}$.

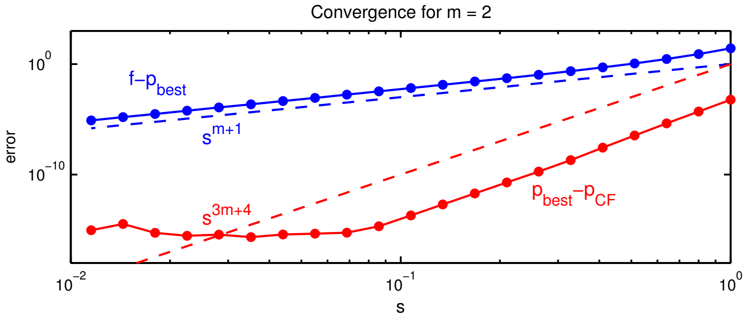
```
ff = @(x) exp(5*x);
for m = 1:2
  ss = .8.^(0:20); errfp = []; errpp = [];
  for s = ss
    f = chebfun(@(x) ff(s*x));
    pbest = remez(f,m); pcf = cf(f,m);
    errfp = [errfp norm(f-pbest,inf)];
    errpp = [errpp norm(pcf-pbest,inf)];
  end
  hold off, loglog(ss,errfp,'.-')
  hold on, loglog(ss,errpp,'.-r'), loglog(ss,ss.^(m+1),'--');
  s = 0.025; text(s,.1*s^(m+1)/4,'s^{m+1}','color','b',FS,10)
  loglog(ss,ss.^(3*m+4),'--r')
  text(s,.02*s^(3*m+4)*1e4,'s^{3m+4}','color','r',FS,10)
  text(.015,.01+(2-m)*.5,'f-p_{best}','color','b',FS,10)
  text(.25,1e-12+(2-m)*1e-8,'p_{best}-p_{CF}','color','r',FS,10)
  axis([1e-2 1 1e-18 1e3]), xlabel('s',FS,9), ylabel error
  title(['Convergence for m = ' int2str(m)],FS,9), snapnow
end
```

```
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
```

```
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
```

```
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
```





In this chapter we have considered CF approximation in its simplest context of approximation of one polynomial f of degree N by another polynomial $p_{\rm CF}$ of degree n. In fact, the method is much more general. So long as f has an absolutely convergent Chebyshev series, which is implied for example if it is Lipschitz continuous, then Theorem 20.1 still applies [Hayashi, Trefethen & Gutknecht 1990]. Now H is an infinite matrix which can be shown to represent a compact operator on ℓ^2 or ℓ^1, its dominant eigenvector is an infinite vector, and $u(z)$ is defined by an infinite series. The error curve is still a continuous function of winding number at least $n+1$.

Another generalization is to approximation by rational functions rather than polynomials. Everything goes through in close analogy to what has been written here, and now the other eigenvalues of the Hankel matrix come into play. The theoretical underpinnings of rational CF approximation can be found in papers of Takagi [1924], Adamyan, Arov and Krein [1971], and Trefethen and Gutknecht [1983b], as well as the article by Hayashi, Trefethen and Gutknecht cited above. Quite apart from theory, one can compute these approximations readily by the Chebfun `cf` command using capabilities introduced by Joris Van Deun. For details and examples see [Van Deun & Trefethen 2011].

Further generalizations of CF approximation concern approximation of vector or matrix functions rather than just scalars, and here, such techniques are associated with the name H^∞ *approximation*. An important early paper was Glover [1984], and there have been many extensions and generalizations since then [Antoulas 2005, Zhou, Doyle & Glover 1996].

We have emphasized the practical power of CF approximants as providing near-best approximations at low computational cost. The conceptual and theoretical significance of the technique, however, goes beyond this. Indeed, the eigenvalue/singular value analysis of Carathéodory–Fejér approximation seems to be the principal known algebraic window into the detailed analysis of best approximations, and in most cases where best approximations of a function happen to be known exactly, these best approximations are CF approximations in which an approximant like $\widetilde{P}$ or $\tilde{p}$ already has the required finite form, so that nothing must be truncated to get to P or p [Gutknecht 1983].

Summary of Chapter 20. *Carathéodory–Fejér approximation constructs near-best approximations of a function $f \in C([-1,1])$ from the singular values and vectors of a Hankel matrix of Chebyshev coefficients. If f is smooth, CF approximants are often indistinguishable in machine precision from true best approximants.*

Exercise 20.1. Approximating $\cos(nx)$. (a) For $n = 2, 4, 8, 16, \ldots$, compute the degree n CF approximant to $f(x) = \cos(nx)$ and plot the error curve. How high can you go in this sequence? (b) What happens if $\cos(nx)$ is changed to $\cos(0.9nx)$?

Exercise 20.2. Approximating the jagged function. Four of the figures of this chapter concerned approximations of degree 20 to a jagged function. (a) How do the L^2 norms of the best and CF approximations compare? (b) The CF approximation was based on truncation of the Chebyshev series at term $N = 100$. How does the ∞-norm of the error vary with N? (c) Draw a conclusion from this exploration: is the imperfect equioscillation of the error curve in the figure given in the text for this function mostly to the fact that CF approximation is not best approximation, or to the fact that $N < \infty$?

Exercise 20.3. Complex approximation on the unit disk. (a) Suppose f is an analytic function on the closed unit disk and p is a polynomial of degree n. Prove that p is a best approximation to f in the ∞-norm on the disk $|z| \le 1$ if and only if it is a best approximation on the circle $|z| = 1$. (b) Look up Rouché's theorem and write down a careful statement, citing your source. (c) Suppose f is an analytic function in the closed unit disk and p is a polynomial of degree n such that $f - p$ maps the unit circle to a circle of winding number at least $n+1$. Prove that p is a best approximation to f on the unit disk. (In fact it is unique, though this is not obvious.)

Exercise 20.4. Irregularity of CF approximation. The second figure of this chapter showed quite irregular dependence of $\|p_{\mathrm{CF}} - p^*\|$ on the degree n for the function $f(x) = \tanh(4(x - 0.3))$. In particular, $n = 15$ and $n = 16$ give very different results. Following the derivation of p_{CF} in the text, investigate this difference numerically. (a) For $n = 15$, how do the coefficients $|b_k|$ depend on k, and how big are the truncated terms in going from $\tilde{p}$ to p_{CF}? (b) Answer the same questions for $n = 16$.

Chapter 21

Spectral methods

`ATAPformats`

Theorem 8.2 described the geometric convergence of Chebyshev projections and interpolants for an analytic function f defined on $[-1,1]$. For such a function, it is not just the polynomials that converge geometrically, but also their derivatives. The following theorem makes this precise. An early publication with a result along these lines is [Tadmor 1986].

Theorem 21.1. Geometric convergence of derivatives. *Let a function f be analytic in $[-1,1]$ and analytically continuable to the closed Bernstein ellipse $\overline{E}_\rho$ for some $\rho > 1$. Then for each integer $\nu \ge 0$, the νth derivatives of the Chebyshev projections f_n and interpolants p_n satisfy as $n \to \infty$*

$$\|f^{(\nu)} - f_n^{(\nu)}\| = O(\rho^{-n}), \quad \|f^{(\nu)} - p_n^{(\nu)}\| = O(\rho^{-n}). \tag{21.1}$$

Proof. Here is an outline, to be filled in in Exercise 21.1. If f is analytic in the closed region $\overline{E}_\rho$, it is also analytic and bounded in the open region $E_{\tilde\rho}$ for some $\tilde\rho > \rho$. By Theorem 8.1 it follows that the Chebyshev coefficients satisfy $a_k = O(\tilde\rho^{-k})$. The bounds (21.1) follow by differentiating the Chebyshev series for $f^{(\nu)} - f_n^{(\nu)}$ and $f^{(\nu)} - p_n^{(\nu)}$ term by term. The differentiations introduce powers of n, since T_n' is of size $O(n^2)$ on $[-1,1]$, for example, but since $n^\alpha \tilde\rho^{-n} = O(\rho^{-n})$ as $n \to \infty$ for any fixed α, we still get $O(\rho^{-n})$ convergence for any fixed ν. ▮

The phenomenon captured in Theorems 8.2 and 21.1 is a general one in complex analysis. When a property holds for an analytic function, there is a good chance that a similar property holds for its derivatives too. The ultimate reason is that both function and derivative can be related to Cauchy integrals, and indeed, an alternative proof of Theorem 21.1 can be based on the Hermite integral formula.

The present chapter is a practical one, devoted to outlining some of the wide-ranging consequences of Theorem 21.1 for scientific computing: the whole field of *spectral methods* for the numerical solution of differential equations. Spectral methods are noted for achieving *spectral accuracy,* which means accuracy that is limited not by the order of the numerical discretization, but only by the smoothness of the function being approximated. This is in contrast to a traditional finite difference or finite element method, which might achieve just $O((\Delta x)^2)$ or $O((\Delta x)^4)$ accuracy as $\Delta x \to 0$, say, where Δx is a grid spacing, even when the function being approximated is C^∞ or analytic. For a leisurely introduction to spectral methods on Chebyshev grids, see [Trefethen 2000].

We now put aside $\{f_n\}$ and focus on *spectral collocation methods,* based on point values and polynomial interpolants, as opposed to *spectral Galerkin methods,* based on integrals.

The starting point of spectral collocation methods is the notion of a *differentiation matrix.* If p is a polynomial of degree n, it is determined by its values on an $(n+1)$-point grid in $[-1,1]$. The derivative p', a polynomial of degree $n-1$, is determined by its values on the same grid. The classical spectral differentiation matrix is the $(n+1)\times(n+1)$ matrix that represents the linear map from the vector of values of p on the grid to the vector of values of p'. (Later we shall mention rectangular alternatives to this classical square matrix formulation.) An explicit formula for this matrix follows from equation (5.8) and was first published by Bellman, Kashev and Casti [1972] (Exercise 21.9):

$$D_{ij} = \ell_j'(x_i) = \begin{cases} \dfrac{\lambda_j}{\lambda_i(x_i - x_j)} & i \neq j, \\ \dfrac{x_j}{1 - x_j^2} & i = j. \end{cases} \qquad (21.2)$$

The particularly important special case is that of a Chebyshev grid. For example, the function $\sin(x)$ can be represented to machine precision by a Chebyshev interpolant p on a grid of 14 points:

```
x = chebfun('x'); p = sin(x); length(p)
```

```
ans =
    14
```

Suppose we wish to calculate the values of p' on the same grid. In Chebfun we can write

```
pp = diff(p); x14 = chebpts(14); pp14 = pp(x14)
```

```
pp14 =
   0.540302305868151
   0.564522388819888
   0.632936510563864
   0.732703188872979
   0.842943722651218
   0.937783753082982
   0.992744245701782
   0.992744245701782
   0.937783753082982
   0.842943722651218
   0.732703188872979
   0.632936510563864
   0.564522388819888
   0.540302305868151
```

But we can also get our hands on the differentiation matrix explicitly with these commands involving a chebfun object known as a “chebop”:

```
D = chebop(@(u) diff(u)); D14 = D(14);
```

```
Warning: FEVAL(N, DIM) or N(DIM) exists only to provide backwards
compatibility with ATAP. The preferred method for visualizing a
discretization of a linear CHEBOP is MATRIX(N, DIM). Note, however,
that these may not give the same result due to changes in how
CHEBOP discretizes differential operators.
```

If the matrix `D14` is multiplied by the vector `p(x14)`, the result is the same vector `pp14` of sampled derivatives, up to rounding errors:

```
norm(pp14-D14*p(x14))
```

```
ans =
     4.711415555570459e-14
```

Above, we put a semicolon after `D(14)` to avoid printing a 14 × 14 matrix. To give the idea while using up a little less space, here are the 3×3 and 5×5 Chebyshev differentiation matrices on $[-1,1]$:

```
format short, D(3)


ans =
   -1.5000    2.0000   -0.5000
   -0.5000         0    0.5000
    0.5000   -2.0000    1.5000
```

```
D(5)


ans =
   -5.5000    6.8284   -2.0000    1.1716   -0.5000
   -1.7071    0.7071    1.4142   -0.7071    0.2929
    0.5000   -1.4142    0.0000    1.4142   -0.5000
   -0.2929    0.7071   -1.4142   -0.7071    1.7071
    0.5000   -1.1716    2.0000   -6.8284    5.5000
```

Formulas for the entries of Chebyshev differentiation matrices were first published by Gottlieb, Hussaini & Orszag [1984], and recurrence relations for computing them fast and stably were given by Welfert [1997], based on earlier work by Fornberg [1988]. Welfert's paper in turn led to the influential Matlab Differentiation Matrix Suite by Weideman and Reddy [2000], and another Matlab code `cheb` for generating these matrices can be found in [Trefethen 2000].

There is no need to stop at the first derivative. Here is the 5 × 5 Chebyshev matrix corresponding to the second derivative on $[-1,1]$:

```
D2 = chebop(@(u) diff(u,2)); D2(5)


ans =
   17.0000  -28.4853   18.0000  -11.5147    5.0000
    9.2426  -14.0000    6.0000   -2.0000    0.7574
   -1.0000    4.0000   -6.0000    4.0000   -1.0000
    0.7574   -2.0000    6.0000  -14.0000    9.2426
    5.0000  -11.5147   18.0000  -28.4853   17.0000
```

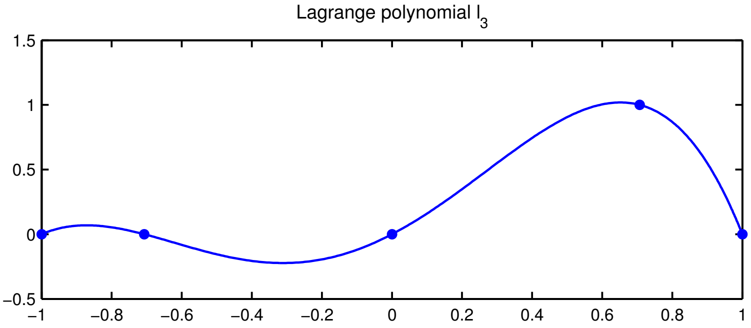
Yes, `D2(5)` is the square of `D(5)`:

```
norm(D2(5)-(D(5))^2)
```

```
ans =
   9.5799e-14
```

The entries of this matrix can be interpreted as follows. The jth column ($0 \leq j \leq n$) contains the second derivatives of the Lagrange polynomial $\ell_j(x)$ evaluated at grid points $x_0, \dots, x_n$. That is, its (i,j) entry (with indexing from 0 to n) is $\ell_j''(x_i)$. (We have seen Lagrange polynomials in Chapters 5, 9, 11, and 15.) For example, here is the Lagrange polynomial supported at x_3:

```
p3 = chebfun([0 0 0 1 0]'); FS = 'fontsize';
clf, plot(p3,'.-'), title('Lagrange polynomial l_3',FS,9)
```



Its second derivatives at the grid points are the values in the fourth column of the matrix $D(5)$ just shown:

```
p3pp = diff(p3,2); x5 = chebpts(5); p3pp(x5)
```

```
ans =
  -11.5147
   -2.0000
    4.0000
  -14.0000
  -28.4853
```

In Chebfun, an object like D or D2 is called a *linear chebop* (and internally within the Chebfun system, a *linop*). A linear chebop is not a matrix, but rather a prescription for how to construct matrices of arbitrary order. (A computer science term for the process of filling such prescriptions is *lazy evaluation*.) If D is applied to an integer argument, the matrix of that dimension is produced:

```
size(D(33))
```

```
ans =
    33    33
```

If D is applied to a chebfun, it has the effect appropriate to the length of that chebfun:

```
f = sin(7*x).*exp(x).*tan(x); norm(diff(f)-D*f)
```

```
ans =
     0
```

More generally, a chebop can be defined for any differential (or integral) operator. For example, here is the chebop corresponding to the map $L : u \mapsto u'' + u' + 100u$ on $[-1, 1]$:

```
L = chebop(@(u) diff(u,2) + diff(u) + 100*u);
```

Here is the 5×5 realization of this operator:

```
L(5)
```

```
ans =
  111.5000  -21.6569   16.0000  -10.3431    4.5000
    7.5355   86.7071    7.4142   -2.7071    1.0503
   -0.5000    2.5858   94.0000    5.4142   -1.5000
    0.4645   -1.2929    4.5858   85.2929   10.9497
    5.5000  -12.6863   20.0000  -35.3137  122.5000
```

We can illustrate its use by applying it to the chebfun for e^x:

```
f = exp(x); Lf = L*f;
Lfexact = 102.*exp(x); norm(Lf-Lfexact)
```

```
ans =
   7.5694e-14
```

Now we come at last to spectral methods proper. If we just wanted to apply differential operators to functions, we would not need matrices. To solve a differential equation, however, we need to invert the process of applying a differential operator. We want to find a function u satisfying certain boundary conditions such that Lu is equal to a prescribed function f. This is where the matrices come in, for matrices can be inverted.

Suppose, for example, we seek a function u that satisfies the equation

$$u'' + u' + 100u = x, \quad u(-1) = u(1) = 0 \tag{21.3}$$

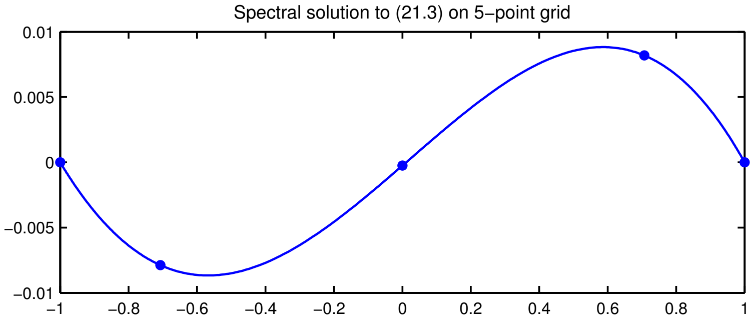
with $x \in [-1, 1]$. The matrix realization above had no boundary conditions. Now we need to impose them, and a standard way of doing this is to modify one or more initial or final rows of the matrix, one row for each boundary condition (see Chapters 7 and 13 of [Trefethen 2000]). For Dirichlet boundary conditions as in (21.3), we change the first and last rows to correspond to rows of the identity:

```
L.bc = 'dirichlet'; feval(L,5,'oldschool')
```

```
ans =
    1.0000         0         0         0         0
    7.5355   86.7071    7.4142   -2.7071    1.0503
   -0.5000    2.5858   94.0000    5.4142   -1.5000
    0.4645   -1.2929    4.5858   85.2929   10.9497
         0         0         0         0    1.0000
```

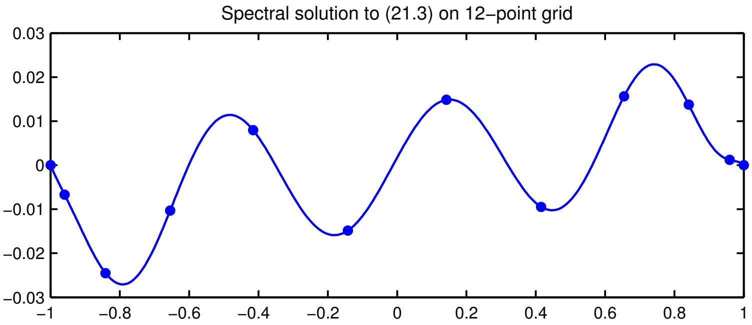
(We shall explain the clumsy command `feval(L,5,'oldschool')` in a moment.) Thus, instead of imposing the differential equation at the boundary points x_0 and x_n, we are imposing boundary conditions at those points. We can now use exactly this matrix to solve the ODE approximately with a 5×5 spectral discretization. The right-hand side of the matrix problem will be the vector of x sampled at the Chebyshev points—except that the first and last components of the vector will be changed to the appropriate Dirichlet values at x_0 and x_n, namely zero.

```
x5 = chebpts(5); x5([1 end]) = 0;
u5 = feval(L,5,'oldschool')\x5;
plot(chebfun(u5),'.-')
title('Spectral solution to (21.3) on 5-point grid',FS,9)
```



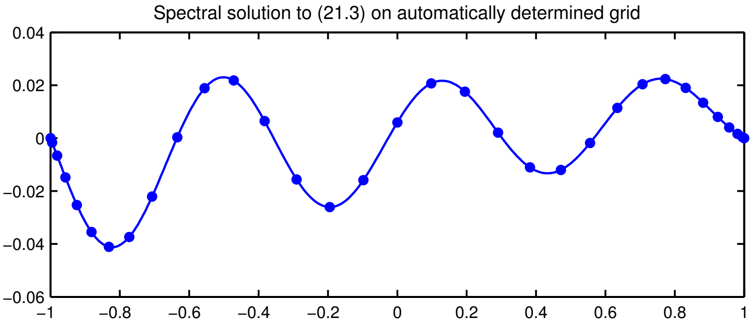
We have just computed our first solution of a boundary value problem with a spectral method. From the picture it is not evident whether the result is close to correct or not. In fact it is not, as increasing the resolution reveals:

```
x12 = chebpts(12); x12([1 end]) = 0;
u12 = feval(L,12,'oldschool')\x12;
plot(chebfun(u12),'.-')
title('Spectral solution to (21.3) on 12-point grid',FS,9)
```



This curve is beginning to get close to the true solution. How fine a grid do we need to reach approximately machine precision? In Chebfun, the appropriate grid is determined automatically when one solves the problem without specifying dimensions, still with the backslash command:

```
u = L\x; plot(u,'.-')
title(['Spectral solution to (21.3) on ' ...
    'automatically determined grid'],FS,9)
```



To get this result, Chebfun has solved matrix problems of sizes 9, 17, 33, and 65, at which point it found that its convergence criteria were satisfied. The final length is

```
length(u)
```

```
ans =
    33
```

and we can verify that the accuracy is good:

```
norm(L*u-x)
```

```
ans =
   2.3002e-12
```

This brings us to the clumsy expression `feval(L,5,'oldschool')` in the demonstration above. This notation instructs Chebfun to display a spectral differentiation matrix corresponding to boundary conditions imposed in the classical way that we have just described, in which certain rows of a square differentiation matrix are replaced by rows corresponding to boundary conditions [Trefethen 2000]. This method of applying boundary conditions relies on the assumption that for each boundary condition, there is a clear choice of which row of the ODE discretization matrix it should replace. In fact, this ceases to be clear in various situations involving systems of equations or more complicated boundary conditions, as well as more general side conditions such as $\int u(x)dx = 0$. Around 2010, Driscoll and Hale realized that more robust and flexible discretizations could be obtained by switching to a different approach based on rectangular differentiation matrices. For an order d differential operator to be applied on an $(n+1)$-point grid, the Driscoll–Hale discretization begins with a matrix of dimension $(n+1-d)\times(n+1)$

corresponding to a map from data on an $(n+1)$-grid to data on an $(n+1-d)$-grid, and then appends an additional d rows for boundary conditions. No collocation equation gets replaced in this process. This is now the discretization strategy used routinely by Chebfun, and it is what Chebfun actually did in solving the problem `u = L\x` above. To see the matrices, one can type the more natural expression `L(5)` instead of `feval(L,5,'oldschool')`. We shall not go into details here; see [Driscoll & Hale 2012].

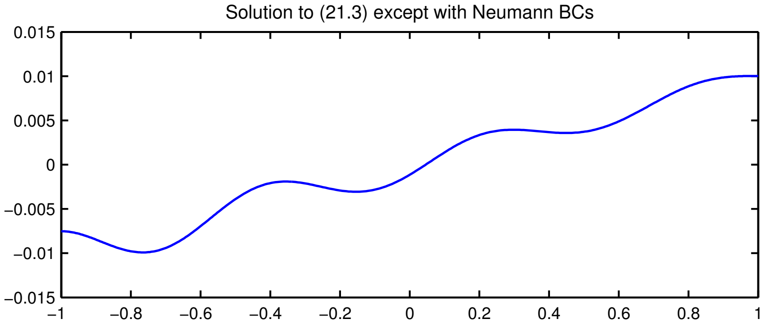
Homogeneous Dirichlet conditions at both ends are only the simplest of many possible boundary conditions for a boundary value problem. To solve (21.3) again except with Neumann conditions $u'(-1) = u'(1) = 0$, the first and last rows of the discretization matrix would classically get replaced by the corresponding rows of the first derivative matrix:

```
L.bc = 'neumann'; format short, feval(L,5,'oldschool')
```

```
ans =
   -5.5000    6.8284   -2.0000    1.1716   -0.5000
    7.5355   86.7071    7.4142   -2.7071    1.0503
   -0.5000    2.5858   94.0000    5.4142   -1.5000
    0.4645   -1.2929    4.5858   85.2929   10.9497
    0.5000   -1.1716    2.0000   -6.8284    5.5000
```

Here is the Chebfun solution, again based on the Driscoll–Hale discretization, now plotted without dots:

```
u = L\x; plot(u), ylim([-0.015 0.015])
title('Solution to (21.3) except with Neumann BCs',FS,9)
```

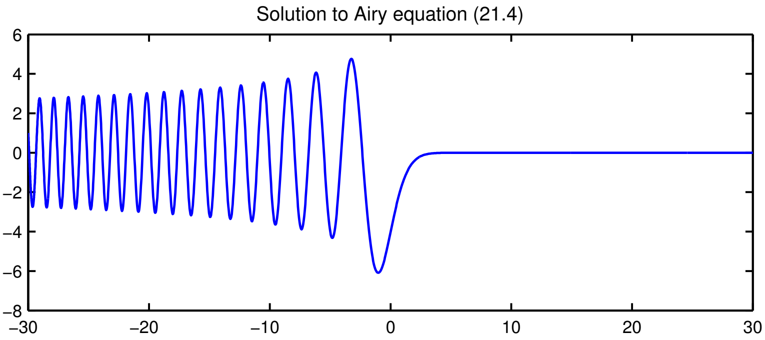


Spectral methods can also solve problems with variable coefficients. For example, suppose we wish to solve the Airy equation boundary value problem

$$u'' - xu = 0, \quad u(-30) = 1, \ u(30) = 0 \tag{21.4}$$

for $x \in [-30, 30]$. Here is the solution:

```
L = chebop(@(x,u) diff(u,2)-x.*u,[-30,30]);
L.lbc = 1; L.rbc = 0; u = L\0;
plot(u), title('Solution to Airy equation (21.4)',FS,9)
```



For nonlinear problems, one would normally use a Newton iteration or some variant. Chebfun handles these cases too. For example, the equation

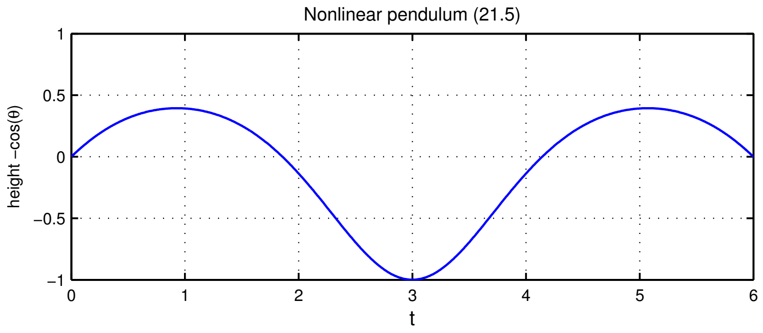
$$\theta'' + \sin(\theta) = 0, \quad \theta \in [0,6] \tag{21.5a}$$

describes the motion in time of a nonlinear pendulum situated at height $h(t) = -\cos(\theta(t)) \in [-1,1]$. If we prescribe boundary conditions

$$u(0) = -\pi/2, \quad u(6) = \pi/2, \tag{21.5b}$$

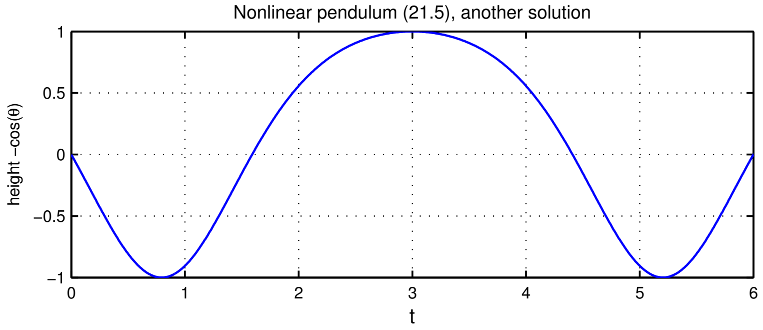
we can solve the system numerically with Chebfun as follows. Notice that the solution is still invoked by the backslash command, though we are very far now from the original Matlab notion of backslash for solving a square system of linear equations.

```
N = chebop(0,6);
N.op = @(theta) diff(theta,2) + sin(theta);
N.lbc = -pi/2; N.rbc = pi/2; theta = N\0;
plot(-cos(theta)), grid on, ylim([-1 1])
title('Nonlinear pendulum (21.5)',FS,9)
xlabel('t',FS,10), ylabel('height -cos(\theta)')
```



This solution corresponds to the pendulum first going up above height 0 for a time, then swinging over to the other side, where it again goes above height 0 before falling back down again. On the other hand, suppose we change the right-hand boundary condition to $5\pi/2$. Then another solution appears, corresponding to the pendulum swinging once around the top:

```
N.lbc = -pi/2; N.rbc = 5*pi/2; theta = N\0;
plot(-cos(theta)), grid on, ylim([-1 1])
title('Nonlinear pendulum (21.5), another solution',FS,9)
xlabel('t',FS,10), ylabel('height -cos(\theta)')
```



These two solutions do not exhaust the full set of possibilities for this nonlinear problem; see Exercise 21.7.

To compute solutions of nonlinear differential equations, Chebfun uses variants of Newton's method implemented for continuous functions rather than discrete vectors. Where one might expect to encounter Jacobian matrices in the solution process, Chebfun actually utilizes their continuous analogues known as Fréchet derivative operators, which are constructed by a process of automatic differentiation, again exploiting lazy evaluation. These capabilities are due to Birkisson and Driscoll [2012]. Chebfun can also solve systems of equations, eigenvalue problems, and problems specified by coefficients that are just piecewise smooth.

This is a book about approximation theory, not differential equations, and we began this chapter with an approximation result, a theorem about the $O(\rho^{-n})$ accuracy of derivatives of approximations of analytic functions. It would be excellent if this theorem implied that spectral methods converge to analytic solutions at the rate $O(\rho^{-n})$, but it does not. Theorem 21.1 ensures that if u is an analytic solution to a boundary value problem $Lu = f$, then the Chebyshev interpolants to Lu would converge geometrically to f as $n \to \infty$. In spectral computations, however, we do not have the exact solution available to discretize, but must approximate it by solving matrix problems. One can hope that the approximations will converge at the expected rate, and indeed they do so under many circumstances, but proving this requires further arguments, which we shall not attempt to discuss here. As a rule, in this business, the practice is ahead of the theory.

Some of the ideas behind spectral methods are as old as Fourier and Chebyshev expansions, and many people contributed in the early years of computers, including Lanczos, Elliott, Fox, and Clenshaw. But it was their application to the partial differential equations of fluid mechanics by Orszag and Gottlieb and others beginning around 1970 that made these methods famous, and it was Orszag who coined the term “spectral methods” [Orszag 1971a & 1971b]. Spectral methods divide into Fourier methods, for periodic problems, and Chebyshev and related methods, for nonperiodic problems. As always in this book, we have emphasized the nonperiodic case, which is less obvious, even though at bottom it is essentially the same. In applications, Fourier and Chebyshev discretizations are often found mixed together. For example, a 3D cylindrical geometry may be discretized by a Chebyshev grid for the radial variable, a periodic Fourier grid for the circumferential variable, and another periodic grid serving as an approximation to an ideal infinite Fourier grid for the longitudinal variable. When the grids are fine, implementations are often based on the Fast Fourier Transform rather than matrices.

For details of the spectral methods incorporated in Chebfun, see [Driscoll, Bornemann & Trefethen 2008] and [Driscoll & Hale 2012] for the linear case and [Birkisson & Driscoll 2012] for nonlinear aspects. For information about spectral methods in general, see texts such as [Fornberg 1996], [Trefethen 2000], [Boyd 2001], [Canuto, Hussaini, Quarteroni & Zang 2006], [Hesthaven, Gottlieb & Gottlieb 2007], and [Shen, Tang & Wang 2011].

This chapter began by noting that if a function is smooth, the derivatives of its interpolants converge rapidly. A contrapositive of this observation is the phenomenon that if the discrete approximations to derivatives of a function blow up as the mesh is refined, it is not smooth. Chebfun exploits this principle as the basis of its edge detection algorithm for breaking piecewise smooth functions into subintervals, which was illustrated at the end of Chapter 9. This algorithm was developed by Rodrigo Platte and is described in [Pachón, Platte & Trefethen 2010].

Summary of Chapter 21. *Spectral collocation methods are numerical algorithms for solving differential equations based on polynomial or trigonometric interpolants. For problems whose solutions are analytic, they typically converge geometrically as the grid is refined.*

Exercise 21.1. Proof of Theorem 21.1. Write down a careful proof of Theorem 21.1 as a corollary of Theorems 3.1 and 8.1. Be sure to state precisely what properties of the Chebyshev polynomials $\{T_k\}$ your proof depends on.

Exercise 21.2. Extension of Theorem 21.1. Theorem 21.1 quantifies the accuracy of the derivatives of Chebyshev interpolants based on an assumption of analyticity in a Bernstein ellipse. State and prove a different theorem about the convergence of the derivatives for *any* sequence of polynomials $p_n \in \mathcal{P}_n$ for which the errors satisfy $\|f - p_n\| = O(\rho^{-n})$ for some $\rho > 1$.

Exercise 21.3. Differentiation matrices. (a) The text displayed the 3×3 matrix $D(3)$. Derive the entries of this matrix analytically. (b) Also displayed was the 5×5 matrix $D2(5)$. Derive the entries of the middle column of this matrix analytically.

Exercise 21.4. Linear boundary value problems. Solve the following linear ODE boundary value problems numerically with Chebfun. In each case plot the solution and report the value of u at the midpoint of the interval and the length of the chebfun representing u.
(a) $0.001u'' + xu' - u = \exp(-10x^2)$, $x \in [-1, 1]$, $u(-1) = 2$, $u(1) = 1$.
(b) $0.001u'' + (1 - x^2)u = 1$, $x \in [-5, 5]$, $u(-5) = 0$, $u(5) = 0$.
(c) $0.001u'' + \sin(x)u = 1$, $x \in [-10, 10]$, $u(-10) = 0$, $u'(10) = 0$.

Exercise 21.5. Nonlinear boundary value problems. Find a solution numerically to each of the following nonlinear ODE boundary value problems. In each case plot the solution and report the value of u at the midpoint of the interval.
(a) $0.05u'' + (u')^2 - u = 1$, $x \in [0, 1]$, $u(0) = 2$, $u(1) = 1$.
(b) $0.01u'' - uu' - u = 0$, $x \in [-1, 1]$, $u(-1) = 1$, $u(1) = 2$.

Exercise 21.6. Convergence with *n*. The text solved the boundary value problem $u'' + u' + 100u = x$ on $[-1, 1]$ with boundary conditions $u(-1) = u(1) = 0$

for grid parameters $n + 1 = 5$, 12, and 35. Perform a numerical study of the ∞-norm error of the solution as a function of n, and comment on the results.

Exercise 21.7. Nonunique solutions. (a) For each of the two nonlinear pendulum problems solved at the end of the chapter, determine exactly how many solutions there must be. (You can use physical reasoning, or phase plane analysis.) (b) Find them all numerically with Chebfun by using sufficiently close initial guesses specified by a command of the form `N.init = f(theta)` to start the iteration. Report the maximum heights $-\cos(\theta)$ of the pendulum in all cases, and the time(s) at which these heights are reached.

Exercise 21.8. Painlevé equation. Solutions to the second Painlevé equation, $u'' = 2u^3 + xu$, typically blow up at various locations on the x-axis. There exist special solutions, however, that are smooth for all real x. Characterized by the asymptotic boundary conditions $u \sim \pm\sqrt{-x/2}$ as $x \to -\infty$ and $u \to 0$ as $x \to +\infty$, these are the so-called Hastings–McLeod solutions. Truncate the problem to the interval $[-L, L]$ with boundary conditions $u(-L) = \sqrt{L/2}$, $u(L) = 0$ and compute and plot solutions for $L = 1, 2, 4, 8, 16$. Produce a table of $u(0)$ and $u'(0)$ for each value of L. To ten digits, what do you think are the values of $u(0)$ and $u'(0)$ in the limit $L \to \infty$?

Exercise 21.9. Formula for square differentiation matrix. Derive (21.2) from (5.8).

Chapter 22

Linear approximation: beyond polynomials

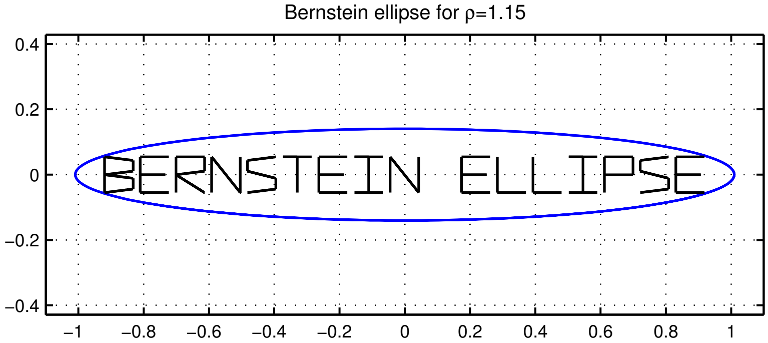
```
ATAPformats
```

Several times in the previous chapters, we have hinted that polynomials are not optimal functions for linear approximation on $[-1,1]$. (Nonlinear approximations are another matter and will make their appearance in the next chapter.) It is now time to explain these hints and introduce alternative approximations that may be up to $\pi/2$ times more efficient. One reason the alternatives are valuable is that they have practical advantages in some applications, especially for spectral methods in more than one space dimension. An equally important reason is that they push us to think more deeply about what it means to approximate a function and what may or may not be special about polynomials. The ideas of this chapter originate in [Bakhvalov 1967] and [Hale & Trefethen 2008]. Related ideas are the basis of work on sinc function numerical methods [Stenger 1993 & 2010, Richardson & Trefethen 2011], tanh and double exponential or tanh-sinh quadrature [Sag & Szekeres 1964, Takahasi & Mori 1974, Mori & Sugihara 2001], and the transformed-grid spectral methods introduced by Kosloff and Tal-Ezer [1993].

Recall from Chapter 8 that if f is analytic on $[-1,1]$, then to investigate its polynomial approximations, we ask how large a Bernstein ellipse E_ρ it can be analytically continued to. Here for example is the ellipse E_ρ with $\rho = 1.15$. The words "Bernstein ellipse" written inside will help in a moment to visualize a conformal map. (Mathematically, these words are a piecewise linear complex function of a real variable constructed by the Chebfun `scribble` command.)

```
x = chebfun('x'); w = exp(2i*pi*x);
```

```
z = @(rho) (rho*w+(rho*w).^(-1))/2;
clf, plot(z(1.15)), xlim([-1.1,1.1]), axis equal, grid on
FS = 'fontsize';
title('Bernstein ellipse for \rho=1.15',FS,9)
f = .01-.055i+.93*scribble('Bernstein ellipse');
hold on, plot(f,'k','linewidth',1.2)
```



Bernstein ellipses are unavoidable if one works with polynomial interpolants, but from the user's point of view, they have an unfortunate property: they are thicker in the middle than near the ends! For a function f to be analytic in the region just shown, its Taylor series about a point $x \approx 0$ must have radius of convergence 0.14 or more. For $x \approx \pm 1$, on the other hand, a radius of convergence of 0.01 or less is sufficient. Thus the smoothness requirement on f is highly nonuniform, and this is not an artifact of the analysis. Polynomials of a given degree really can resolve rougher behavior of a function f near the endpoints than in the middle. This phenomenon turns up in one form of another whenever approximation theorists seek sharp results about polynomial approximations, whether f is analytic or not. See for example [Timan 1951], [Lorentz 1986], [Ditzian & Totik 1987], and Chapter 8 of [DeVore and Lorentz 1993].

Of course, there are some functions that have most of their complexity near ± 1, and for these, the nonuniform approximation power of polynomials may be an advantage. For example, functions of this kind arise in fluid mechanics problems with boundary layers. More often, however, the nonuniform approximation power of polynomials is a disadvantage from a practical point of view, as well as being a conceptual complication. If only those ellipses had constant width for all $x \in [-1, 1]$!

As soon as one frames the difficulty in this way, a possibility for a solution suggests itself. The idea is to change variables by means of a function that conformally maps ellipses, approximately at least, to straight-sided ε-neighborhoods of $[-1,1]$, while mapping $[-1,1]$ to itself. To explore this idea we shall use the variable x for the domain where f is defined and introduce a new variable s for the parameter domain, where the Chebyshev points and ellipses live. Our conformal map will be $x = g(s)$, and we shall approximate a function $f(x)$ on $[-1,1]$ by $p(g^{-1}(x)) = p(s)$, where p is a polynomial. Equivalently, we shall approximate $f(g(s))$ on $[-1,1]$ by a polynomial. In the remainder of this chapter we explore the consequences of this idea, considering just one fixed example of a map g,

$$g(s) = \frac{1}{53089}(40320s + 6720s^3 + 3024s^5 + 1800s^7 + 1225s^9), \qquad (22.1)$$

or as a Chebfun command,

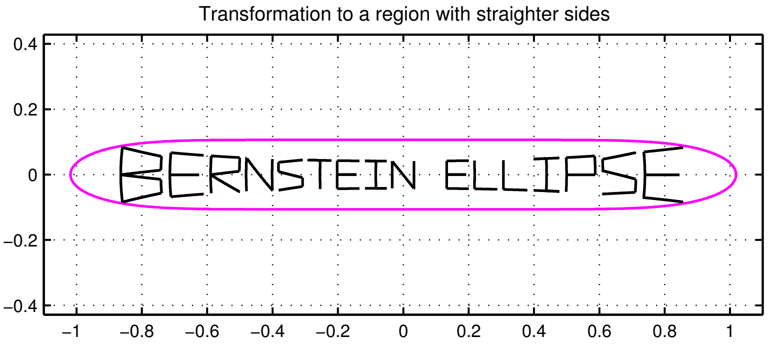
```
g = chebfun(@(s) (40320*s + 6720*s.^3 + 3024*s.^5 + ...
                                1800*s.^7 + 1225*s.^9)/53089);
```

This function g is derived by truncating the Taylor series of $(2/\pi)\sin^{-1}(x)$ and then rescaling the result so that $g(\pm 1) = \pm 1$. See [Hale & Trefethen 2008] for a discussion of this and other possible choices of g, some of which (notably a conformal map onto an infinite strip) come closer to realizing the maximum possible improvement by a factor of $\pi/2$. See also Exercises 22.2 and 22.3.

To begin the discussion, let us look at how g transforms ellipses about $[-1,1]$. Here is a plot of $g(E_{1.15})$, the transformed version of the ellipse shown earlier. Notice the much straighter sides.

```
hold off, plot(g(z(1.15)),'m')
xlim([-1.1,1.1]), axis equal, grid on
title('Transformation to a region with straighter sides',FS,9)
hold on, plot(g(f),'k','linewidth',1.2)
```

```
Warning: F should be real valued to construct G(F).
Results may be inaccurate if G is not a polynomial.
```



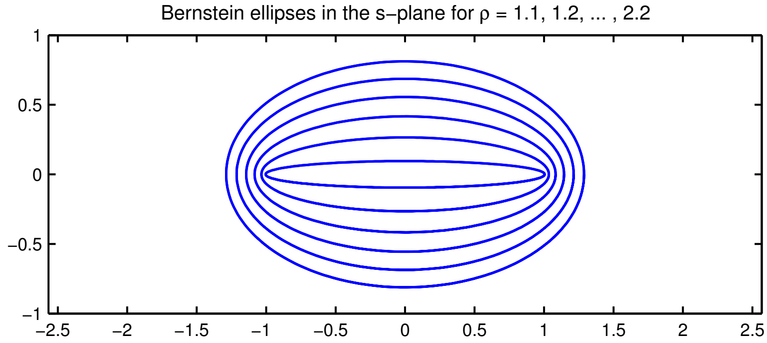
Following [Hale & Trefethen 2008], we call g a *sausage map* and $g(E_{1.15})$ a *sausage region.* The crucial property is that for most of its length, the sausage is narrower than the ellipse, as the distorted "Bernstein ellipse" label makes clear. The ellipse has half-width approximately $\rho - 1$, which is about 32% more than the half-width $0.76(\rho - 1)$ of the sausage:

```
format short
ellipse_width = max(imag(z(1.15)))
sausage_width = max(imag(g(z(1.15))))
ratio = ellipse_width/sausage_width
```

```
ellipse_width =
    0.1402
sausage_width =
    0.1061
ratio =
    1.3210
```

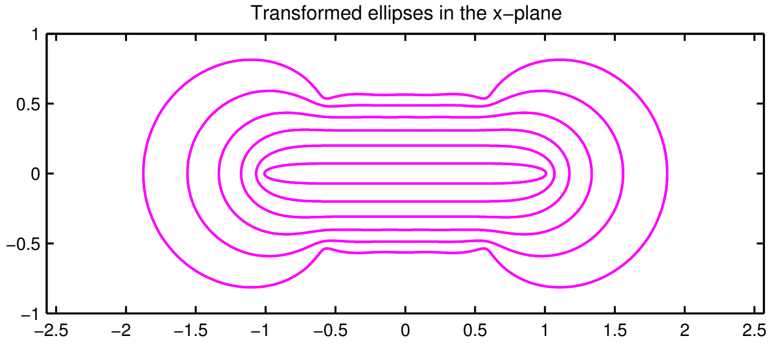
We can learn more by looking at a family of ellipses. Following Chapter 8, here is a plot of E_ρ for $\rho = 1, 1.2, \ldots, 2.2$:

```
w = exp(2i*pi*x); hold off
for rho = 1.1:0.2:2.2
    plot((rho*w+(rho*w).^(-1))/2), hold on
end
ylim([-1 1]), axis equal
title(['Bernstein ellipses in the s-plane'...
       ' for \rho = 1.1, 1.2, ... , 2.2'],FS,9)
```



Here is the corresponding figure for the images $g(E_\rho)$:

```
hold off
for rho = 1.1:0.2:2.2
    plot(g((rho*w+(rho*w).^(-1))/2),'m'), hold on
end
ylim([-1 1]), axis equal
title('Transformed ellipses in the x-plane',FS,9)
```

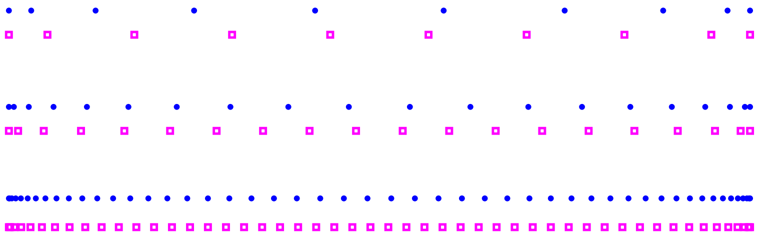


It is clear that near $[-1,1]$, the transformed ellipses are narrower and more uniform in shape than the ellipses, but further away, their behavior is more irregular. We shall see some of the implications of these shapes as we explore the uses of this map.

Chapter 2 considered polynomial interpolants in Chebyshev points $\{s_k\}$. With the transformation g, f is interpolated by transformed polynomials $p(g^{-1}(x))$ in the points $\{g(s_k)\}$. We illustrate the difference between Chebyshev and transformed Chebyshev points by adapting a code segment from Chapter 17. The squares show the transformed points.

```
ss = chebpts(10);
```

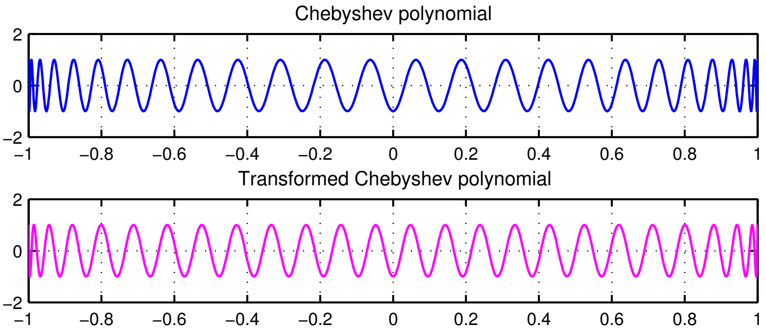
```
clf, plot(ss,.9,'.b','markersize',8), hold on
plot(g(ss),.8,'sm','markersize',3)
ss = chebpts(20);
plot(ss,.5,'.b','markersize',8), plot(g(ss),.4,'sm','markersize',3)
ss = chebpts(50);
plot(ss,.12,'.b','markersize',8), plot(g(ss),0,'sm','markersize',3)
axis([-1 1 -.1 1.1]), axis off
```



Note that the squares are more evenly distributed than the dots, and in particular, they are denser in the middle, providing finer resolution.

Chapter 3 considered Chebyshev polynomials and series. We adapt another code segment from Chapter 17 to illustrate how a Chebyshev polynomial $T_n(x)$ compares to the corresponding transformed polynomial $T_n(g^{-1}(x))$. For this we need the inverse map g^{-1}.

```
gi = inv(g);
T50 = chebpoly(50); subplot(2,1,1), plot(T50), axis([-1 1 -2 2])
title('Chebyshev polynomial',FS,9), grid on, subplot(2,1,2)
plot(T50(gi),'m'), axis([-1 1 -2 2])
grid on, title('Transformed Chebyshev polynomial',FS,9)
```



Notice that the lower curves are more like uniform sine waves than the upper ones.

Theorem 3.1 summarized some basic facts about Chebyshev series, and these carry over immediately to a theorem for transformed Chebyshev series. The theorem as stated assumes g is analytic, though in fact, continuous differentiability would be enough.

Theorem 22.1. Transformed Chebyshev series. *Let g be an analytic function on $[-1,1]$ mapping $[-1,1]$ to itself with $g'(s)>0$. Then if f is Lipschitz continuous on $[-1,1]$, it has a unique representation as an absolutely convergent series*

$$f(x)=\sum_{k=0}^{\infty} a_k T_k(g^{-1}(x)), \tag{22.2}$$

and the coefficients are given for $k\ge 1$ by the formula

$$a_k=\frac{2}{\pi}\int_{-1}^{1}\frac{f(g(s))T_k(s)}{\sqrt{1-s^2}}ds, \tag{22.3}$$

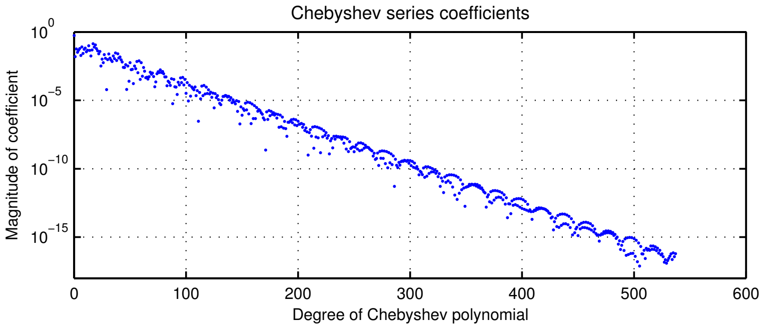
and for $k=0$ by the same formula with the factor $2/\pi$ changed to $1/\pi$.

Proof. This is a consequence of Theorem 3.1. ∎

For many functions f, the transformed series are 20–30% more efficient than the originals. For example, Chebyshev interpolation of $(2+\cos(20x+1))^{-1}$ requires about 520 terms for 15-digit accuracy:

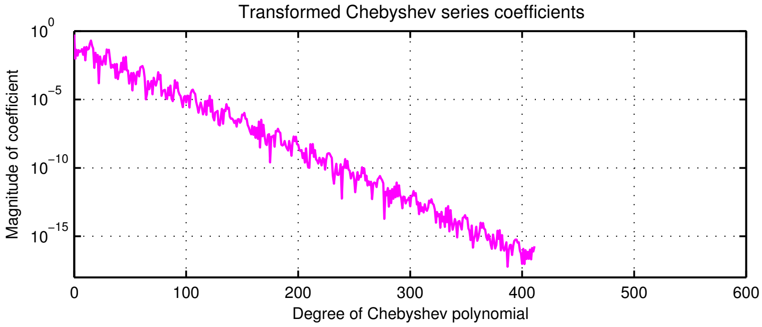
```
f = 1./(2+cos(20*x+1));
clf, chebpolyplot(f), grid on, axis([0 600 1e-18 1])
title('Chebyshev series coefficients',FS,9)
```

```
Warning: CHEBPOLYPLOT is deprecated. Please use PLOTCOEFFS instead.
```



For the transformed interpolants the figure is closer to 400:

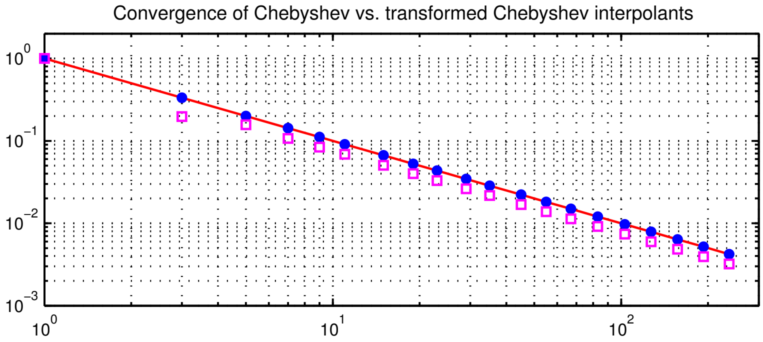
```
chebpolyplot(f(g),'m'), grid on, axis([0 600 1e-18 1])
title('Transformed Chebyshev series coefficients',FS,9)
```



Chapter 7 considered convergence for differentiable functions. Theorem 7.2 can readily be restated for the transformed context—see Exercise 22.1. For a numerical illustration, here is a repetition of the experiment from Chapter 7 involving $f(x) = |x|$. On the loglog scale, the transformed approximants run parallel to the same line as the Chebyshev interpolants, but lower.

```
f = abs(x); fg = f(g);
nn = 2*round(2.^(0:.3:7))-1; ee = 0*nn; ee2 = 0*nn;
for j = 1:length(nn)
    n = nn(j);
    fn = chebfun(f,n+1); ee(j) = norm(f-fn,inf);
    fn2 = chebfun(fg,n+1); ee2(j) = norm(fg-fn2,inf);
end
hold off, loglog(nn,1./nn,'r'), grid on, axis([1 300 1e-3 2])
hold on, loglog(nn,ee,'.'), loglog(nn,ee2,'sm','markersize',5)
ratio = ee(end-4:end)./ee2(end-4:end)
title(['Convergence of Chebyshev vs. '...
       'transformed Chebyshev interpolants'],FS,9)
```

```
ratio =
    1.3167     1.3167     1.3167     1.3167     1.3167
```



Chapter 8 considered convergence for analytic functions. Here is the transformed equivalent of Theorems 8.1 and 8.2.

Theorem 22.2. Transformed coefficients of analytic functions. *For given $\rho > 1$, let g and f be analytic functions on $[-1, 1]$ that can be analytically continued to E_ρ and $g(E_\rho)$, respectively, with $|f(z)| \le M$ for $z \in g(E_\rho)$. Then the transformed Chebyshev coefficients of Theorem* 22.1 *satisfy*

$$|a_k| \le 2M\rho^{-n}, \tag{22.4}$$

the truncated transformed series satisfy

$$\|f - f_n(g^{-1}(x))\| \le \frac{2M\rho^{-n}}{\rho - 1}, \tag{22.5}$$

and the transformed Chebyshev interpolants satisfy

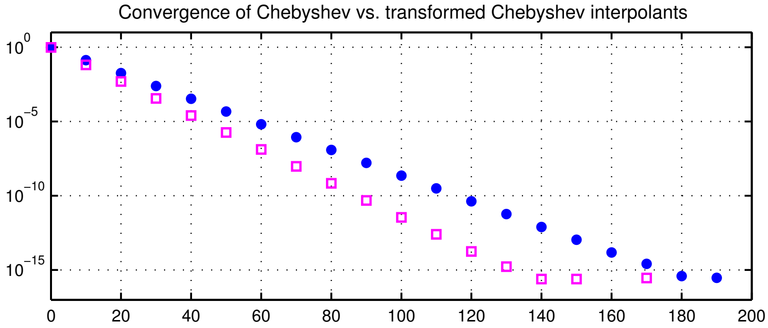
$$\|f - p_n(g^{-1}(x))\| \le \frac{4M\rho^{-n}}{\rho - 1}. \tag{22.6}$$

Proof. These results follow from Theorems 8.2 and 22.1. ∎

Here is a repetition of the Chapter 8 experiment for the Runge function, now with squares to show the transformed approximants.

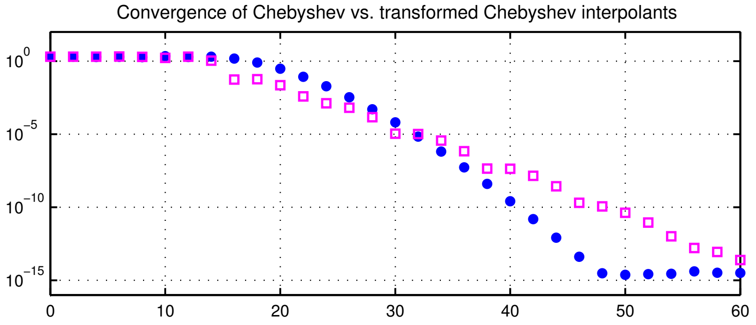
```
f = 1./(1+25*x.^2); fg = f(g);
nn = 0:10:200; ee = 0*nn; ee2 = 0:nn;
for j = 1:length(nn)
    n = nn(j);
    fn = chebfun(f,n+1); ee(j) = norm(f-fn,inf);
    fn2 = chebfun(fg,n+1); ee2(j) = norm(fg-fn2,inf);
end
```

```
hold off, semilogy(nn,ee,'.')
hold on, semilogy(nn,ee2,'sm','markersize',5)
grid on, axis([0 200 1e-17 10])
title(['Convergence of Chebyshev vs. '...
       'transformed Chebyshev interpolants'],FS,9)
```



The speedup is clear. On the other hand, here is a repetition of the experiment with $\cos(20x)$.

```
f = cos(20*x); fg = f(g);
nn = 0:2:60; ee = 0*nn; ee2 = 0:nn;
for j = 1:length(nn)
    n = nn(j);
    fn = chebfun(f,n+1); ee(j) = norm(f-fn,inf);
    fn2 = chebfun(fg,n+1); ee2(j) = norm(fg-fn2,inf);
end
hold off, semilogy(nn,ee,'.')
hold on, semilogy(nn,ee2,'sm','markersize',5)
grid on, axis([0 60 1e-16 100])
title(['Convergence of Chebyshev vs. '...
       'transformed Chebyshev interpolants'],FS,9)
```

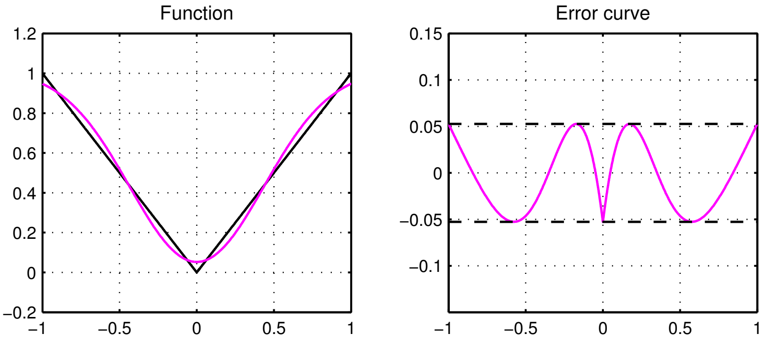


Now the result is ambiguous: the transformed method starts out ahead, but the standard Chebyshev method wins eventually. The explanation can be found in the nested ellipses E_ρ and their images plotted earlier. The function $\cos(20x)$ is entire, and for larger n, the Chebyshev points take good advantage of its analyticity well away from $[-1, 1]$. The transformed points do not do as well. (The advantage of the transformation becomes decisive again if we change $\cos(20x)$ to $\cos(100x)$, at least down to 16-digit precision.)

We can see similar effects if we look at best approximations. For a non-smooth function like $|x|$, transformed polynomials typically approximate better than true ones. The following figures should be compared with those of Chapter 10, and the variable `ratio` quantifies the degree of improvement.

```
f = abs(x);
subplot(1,2,1), hold off, plot(f,'k'), grid on
fg = f(g);
[p,err] = remez(fg,4);
hold on, plot(p(gi),'m'), axis([-1 1 -.2 1.2])
title('Function',FS,9)
subplot(1,2,2), hold off
plot(g,f-p(gi),'m'), grid on, hold on, axis([-1 1 -.15 .15])
plot([-1 1],err*[1 1],'--k'), plot([-1 1],-err*[1 1],'--k')
[p2,err2] = remez(f,4); ratio = err2/err, title('Error curve',FS,9)
```

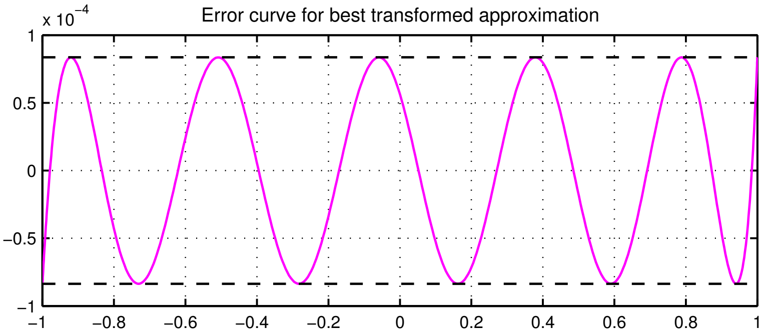
```
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
ratio =
    1.2847
```



On the other hand for a gentle entire function like $\exp(x)$, pure polynomials converge very fast and transformed polynomials cannot compete. The following error curve is seven orders of magnitude larger than that of Chapter 10.

```
f = exp(x);
fg = f(g);
[p,err] = remez(fg,10);
clf, plot(g,fg-p,'m'), grid on, hold on
plot([-1 1],err*[1 1],'--k'), plot([-1 1],-err*[1 1],'--k')
[p2,err2] = remez(f,10); ratio = err2/err
xlim([-1 1])
title('Error curve for best transformed approximation',FS,9)
```

```
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
ratio =
   2.9939e-07
```



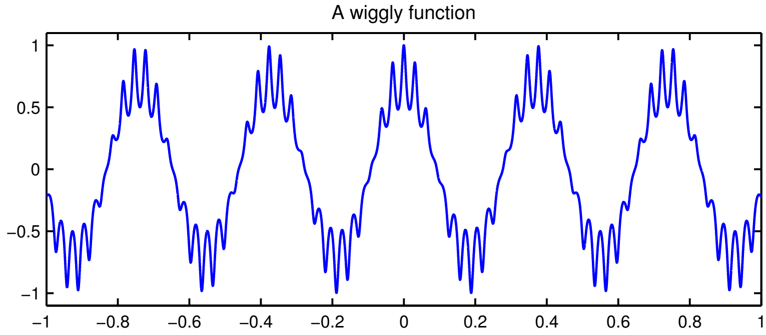
Our final application of transformed polynomial approximants is the one that was the subject of [Hale & Trefethen 2008]: quadrature. As described in Chapter 19, standard quadrature formulas are based on the idea of integrating a function numerically by interpolating it by a polynomial, then integrating the interpolant. This is the basis of all the well-known quadrature formulas, including Gauss, Newton–Cotes, Simpson, and Clenshaw–Curtis. But why should quadrature formulas be based on polynomials? This is a question not often raised in the quadrature literature. Some of the explanation surely has to do with custom going back centuries, before the appearance of computers, when the algebraic simplicity of polynomials would have been a telling advantage. If one had to give a mathematical answer with still some validity today, it would probably be that a polynomial formula is optimal if the order is fixed while the grid size is decreased to zero. If the order increases to ∞ on a fixed interval of integration, however, polynomial formulas are in no sense optimal.

In particular, a "transformed Gauss" quadrature formula can be obtained by applying Gauss quadrature to the integral on the right in the formula

$$\int_{-1}^{1} f(x) = \int_{-1}^{1} f(g(s))g'(s)ds. \tag{22.7}$$

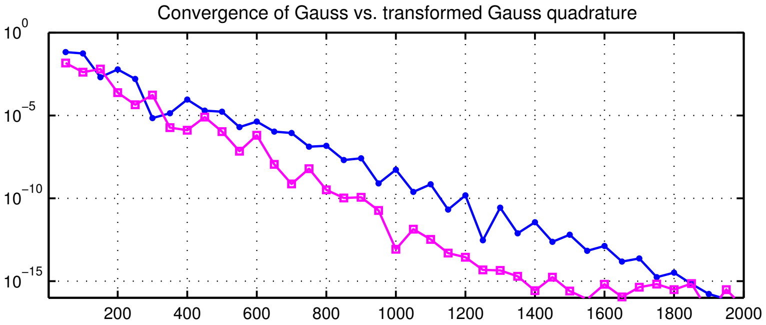
To illustrate this transplanted quadrature idea we pick a wiggly function,

```
f = cos(17*x)./(1+sin(100*x).^2); clf, plot(f), ylim([-1.1 1.1])
title('A wiggly function','fontsize',9)
```



Here is a code in which I represents Gauss quadrature and I2 is transformed Gauss quadrature—and we see that the dots decrease about 30% more slowly than the squares.

```
gp = diff(g); Iexact = sum(f);
err = []; err2 = []; nn = 50:50:2000;
for n = nn
   [s,w] = legpts(n);
   I = w*f(s); err = [err abs(I-Iexact)];
   I2 = w*(f(g(s)).*gp(s)); err2 = [err2 abs(I2-Iexact)];
end
hold off, semilogy(nn,err,'.-','markersize',9), grid on
hold on, semilogy(nn,err2,'s-m','markersize',4), axis([1 2000 1e-16 1])
title('Convergence of Gauss vs. transformed Gauss quadrature',FS,9)
```



We emphasize: in the end a quadrature formula is just a quadrature formula, as specified in (19.3):

$$I_n = \sum_{k=0}^{n} w_k f(x_k). \qquad (22.8)$$

Gauss leads to one choice of nodes and weights, Clenshaw–Curtis leads to another, transplanted Gauss leads to a third, transplanted Clenshaw–Curtis to a fourth. Regardless of what concepts may have been employed in the derivation, in the end the quadrature formula just takes a linear combination of function values, and the transformed formulas usually outperform the classical ones. For example, in [Hale & Trefethen 2008] it is proved that the transformed Gauss formulas based on mapping $E_{1.1}$ to an infinite strip converges 50% faster than Gauss quadrature for the class of functions analytic in the ε-neighborhood of $[-1,1]$, for any $\varepsilon < 0.05$.

This chapter has shown that polynomials are not the only effective general linear class of approximants for general functions f on an interval and indeed are often suboptimal. There is much more that can be said on this subject. For example, there is the matter of how the mapping g was derived and what other maps might be useful; an influential family of maps was introduced by Kosloff and Tal-Ezer [1993]. Another topic we have not discussed is the application to spectral methods, Kosloff and Tal-Ezer's motivation, and it is here that transformations of variables are perhaps most important in practice. Finally, there is the idea of using the map g for rational functions rather than polynomials. The last two ideas have been combined powerfully in Tee's adaptive rational spectral collocation method based on adaptively determined conformal maps [Tee & Trefethen 2006, Hale & Tee 2009].

Summary of Chapter 22. *Although many numerical methods are based on polynomial approximations of a function $f \in C([-1,1])$, such approximations are not optimal in any natural sense, for polynomials have higher resolution near the endpoints of the interval than near the middle. By a conformal transplantation one can derive approximations that are up to $\pi/2$ times more efficient.*

Exercise 22.1. A challenging integrand. Repeat the Gauss vs. transformed Gauss quadrature experiment for the "challenging integrand" (18.14). By approximately what percentage is Gauss slower than transformed Gauss for this function? How do you account for this behavior?

Exercise 22.2. Chebfun 'map'. Chebfun contains a `'map'` parameter that enables one to explore some of the ideas of this chapter in an automatic fashion (try `help maps` for information). To illustrate this, construct `f = 1./(1+25*x.^2)` with both `x = chebfun('x')` as usual and also `x = chebfun('x','map',{'sausage',9})`. How do the `chebpolyplot` results compare? (b) What if the parameter 9 is varied to $1, 3, 5, \dots 15$? (This is the degree of the expansion in (22.1).)

Exercise 22.3. Transformed Clenshaw–Curtis quadrature. Generate the final plot of this chapter again, but now with two further curves added corresponding to Clenshaw–Curtis and transformed Clenshaw–Curtis quadrature. How do the results compare with those for Gauss and transformed Gauss?

Exercise 22.4. Gauss quadrature transformed by an infinite strip. Better than a sausage map for some applications is a map onto an infinite strip. Following the last two exercises, use `x = chebfun('x','map',{'strip',1.4})` to reproduce the final plot of this chapter again, now with one other curve added corresponding to Gauss quadrature transformed by the strip map of the Bernstein ellipse of parameter $\rho = 1.4$. How do the results compare with those from the sausage transformation?

Exercise 22.5. Interpolation of equispaced data. Here is a scheme for interpolation of data at equispaced points on $[-1,1]$: use a conformal map g^{-1} to transform the equispaced grid to an approximate Chebyshev grid, and then compute a polynomial interpolant by means of the barycentric formulas (5.11)–(5.12). Explore this method in Chebfun for interpolation of the Runge function $f(x) = 1/(1+25x^2)$ where g is the map (22.1), using `interp1` to compute the interpolant. Do these approximants weaken the Runge phenomenon? (A theorem of [Platte, Trefethen & Kuijlaars 2011] asserts that no approximation scheme can eliminate the Runge phenomenon entirely.)

Chapter 23

Nonlinear approximation: why rational functions?

`ATAPformats`

Up to now, this book has been about polynomials, or in the last chapter, their transplants. The final six chapters of the book are about rational functions, which have been a mainstay of approximation theory from the beginning. Why do rational approximations occupy such a large place in the literature? Polynomials are familiar and comfortable, but rational functions seem complex and specialized. Is their position in approximation theory justified, or is it an artifact of history, perhaps a holdover from the pre-computer era? In this chapter we attempt to answer these questions, and in doing so we shall find ourselves considering the broader question of what the uses are of the whole subject of approximation theory.

I think the answer is this. Although rational functions indeed became an established part of approximation theory long before computers and many of today's practical applications, their place in the subject is deserved. Their importance stems from a conjunction of two facts. On the one hand, rational functions are more powerful than polynomials at approximating functions *near singularities* and on *unbounded domains.* On the other hand, for various reasons related for example to partial fraction decompositions, they are easier to work with than their nonlinearity might suggest—indeed, sometimes no more complicated than polynomials.

A rational function is the ratio of two polynomials, and in particular, given $m \ge 0$ and $n \ge 0$, we say that r is a rational function of *type* (m,n) if it can be written as a quotient p_m/q_n with $p_m \in \mathcal{P}_m$ and $q_n \in \mathcal{P}_n$. The set of all rational functions of type (m, n) is denoted by $\mathcal{R}_{mn}$, and any $r \in \mathcal{R}_{mn}$ can be written in the form

$$r(x) = \sum_{k=0}^{m} a_k x^k \Big/ \sum_{k=0}^{n} b_k x^k \tag{23.1}$$

for some real or complex coefficients $\{a_k\}$ and $\{b_k\}$. The degrees need not be exact, i.e., there is no requirement that a_m or b_n must be nonzero. Nor do we require that the numerator and denominator are relatively prime, that is, that they have no common zeros.

Suppose, however, that for some nonzero $r \in \mathcal{R}_{mn}$, we choose a representation with relatively prime numerator and denominator. Define $\mu \le m$ to be the index of the highest degree nonzero numerator coefficient and similarly $\nu \le n$ for the denominator, and further normalize the coefficients by requiring $b_\nu = 1$. Then we can write

$$r(x) = \sum_{k=0}^{\mu} a_k x^k \Big/ \sum_{k=0}^{\nu} b_k x^k \ , \quad a_\mu \ne 0, \ b_\nu = 1. \tag{23.2}$$

In this case r has exactly μ finite zeros and ν finite poles, counted with multiplicity: we say that r is of *exact type* (μ, ν). (The case in which r is identically zero is a special one, with no nonzero coefficients in the numerator, and we say it has exact type $(-\infty, 0)$.) If $\mu > \nu$, then r has a pole at $x = \infty$ of order $\mu - \nu$, and if $\nu > \mu$ it has a zero at $x = \infty$ of order $\nu - \mu$. Basic properties of rational functions are described in books of complex analysis such as [Ahlfors 1953, Henrici 1974, Markushevich 1985].

These representations highlight the nonlinearity of rational functions, but a different perspective is suggested when we represent them by *partial fractions.* (Partial fractions were the subject of Jacobi's PhD thesis [1825], and an excellent general reference is Chapter 7 of [Henrici 1974].) In the simplest situation, consider

$$r(x) = \sum_{k=1}^{n} \frac{c_k}{x - \xi_k}, \tag{23.3}$$

where $\{\xi_k\}$ are distinct real or complex numbers. For any coefficients $\{c_k\}$, this is a rational function of type $(n-1, n)$. The number c_k is the *residue* of r at ξ_k. This representation highlights the linear aspects of rational functions. For example, whereas computing the integral of r written in the form p/q looks daunting, in the representation (23.3) we have simply

$$\int^x r(s)ds = C + \sum_{k=1}^{n} c_k \log(x - \xi_k). \tag{23.4}$$

In applications, it is interesting how often a formula like this turns out to be instrumental in making a rational function useful.

The partial fraction form (23.3) does not apply to all rational functions. One limitation is that it always represents a rational function of exact type (μ,ν) with $\mu < \nu$. Another is that it does not represent all functions of this kind, since it cannot account for poles of multiplicity greater than 1. The following theorem gives a partial fraction representation for the general case.

Theorem 23.1. Partial fraction representation. *Given $m, n \geq 0$, let $r \in \mathcal{R}_{mn}$ be arbitrary. Then r has a unique representation in the form*

$$r(x) = p_0(x) + \sum_{k=1}^{\mu} p_k((x-\xi_k)^{-1}), \tag{23.5}$$

where p_0 is a polynomial of exact degree ν_0 for some $\nu_0 \leq m$ (unless $p = 0$) and $\{p_k\}$, $1 \leq k \leq \mu$, are polynomials of exact degrees $\nu_k \geq 1$ with $p_k(0) = 0$ and $\sum_{k=1}^{\mu} \nu_k \leq n$.

Proof. See Theorem 4.4h of [Henrici 1974]. ▮

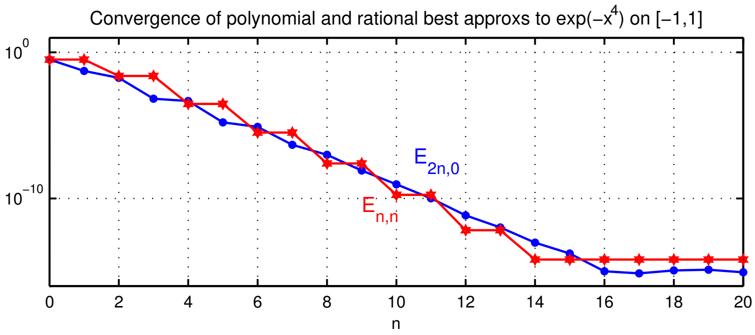
The function p_0 is the *polynomial part* of r, and $p_k((x-\xi_k)^{-1})$ is its *principal part* at ξ_k.

This is all we shall say for the moment about the mathematics of rational functions. Let us now turn to the main subject of this chapter, the discussion of why these functions are useful in approximation theory and approximation practice.

The right place to start is with a cautionary observation. Rational functions are not always better than polynomials. Indeed, consider the most basic of all situations, in which a fuction f is analytic in a ρ-ellipse E_ρ for some $\rho > 1$. For such a function, by Theorem 8.2, polynomial approximations will converge at the rate $O(\rho^{-n})$. It turns out that a typical convergence rate for type (n,n) rational functions is $O(\rho^{-2n})$. So, doubling the number of parameters to be determined sometimes just approximately doubles the convergence rate. (In fact, sometimes it does not increase the convergence rate at all [Szabados 1970].) For applications of this kind, rational functions may outperform polynomials, but often it is by a rather modest factor.

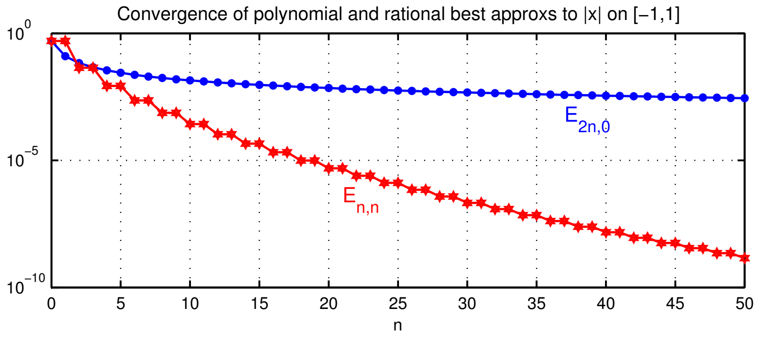
For example, here are a pair of curves showing $\|f - p_{2n}^*\|$ (dots) and $\|f - r_{nn}^*\|$ (stars) as functions of n for $f(x) = \exp(-x^4)$, where p_{2n}^* and r_{nn}^* are the best approximations to f in $\mathcal{P}_{2n}$ and $\mathcal{R}_{nn}$, respectively. (We shall discuss rational best approximation in the next chapter.) Both curves decrease geometrically, and there is not much difference between them. (The rational approximations here should in principle be computed with `remez`, but Chebfun's rational remez algorithm is currently not robust enough, so `cf` is used instead.)

```
x = chebfun('x'); f = exp(-x.^4); warning off
nn = 0:20; errp = []; errr = [];
for n = nn
  p2n = remez(f,2*n); errp = [errp norm(f-p2n,inf)];
  [p,q,foo] = cf(f,n,n); rnn = p./q; errr = [errr norm(f-rnn,inf)];
end
clf, semilogy(nn,errp,'.-','markersize',12), grid on, ylim([1e-16 10])
hold on, semilogy(nn,errr,'h-r','markersize',4), FS = 'fontsize';
text(10.5,2e-8,'E_{2n,0}',FS,10,'color','b')
text(9,1e-11,'E_{n,n}',FS,10,'color','r'), xlabel n
title(['Convergence of polynomial and rational '...
       'best approxs to exp(-x^4) on [-1,1]'],FS,9)
```



What makes rational functions important is that, in contrast to this example, there are many problems where one wants to operate near singularities, or on unbounded domains. For these problems, rational approximations may converge much faster than polynomials. For example, here is an experiment like the last one, but with $f(x) = |x|$. For this function, a type (n, n) rational approximant with $n = 150$ gives 16-digit accuracy, whereas polynomial approximants would need $n = 10^{15}$ to do so well. (Again this code should in principle use `remez` but cannot, so known best approximation errors are hardwired into the code.)

```
f = abs(x); xx = linspace(-1,1,1000); nn = 0:50; errp = [];
errr = [.5 4.37e-2 8.50e-3 2.28e-3 7.37e-4 2.69e-4 1.07e-4 ...
        4.60e-5 2.09e-5 9.89e-6 4.88e-6 2.49e-6 1.30e-6 ...
        6.3*exp(-pi*sqrt(26:2:max(nn)))];
errr = kron(errr,[1 1]); errr(end) = [];
for n = nn
  p2n = remez(f,2*n); errp = [errp norm(f(xx)-p2n(xx),inf)];
end
hold off, semilogy(nn,errp,'.-','markersize',12), grid on
hold on, semilogy(nn,errr,'h-r','markersize',4)
text(37,3e-4,'E_{2n,0}',FS,10,'color','b')
text(21,2e-7,'E_{n,n}',FS,10,'color','r'), xlabel n
title(['Convergence of polynomial and rational '...
       'best approxs to |x| on [-1,1]'],FS,9)
```



The approximation of $|x|$ by rational functions is one of the "two famous problems" to be considered in Chapter 25. Half a century ago Donald Newman proved that whereas polynomial approximants to $|x|$ converge just at the rate $O(n^{-1})$, for rational approximants the rate is $\exp(-C\sqrt{n})$ with $C > 1$ [Newman 1964]. This result rigorously established the possibility of an exponential difference in effectiveness of the two types of approximations.

The rest of this chapter is devoted to an outline of twelve applications in which rational approximations are useful. In most of these examples, there is a singularity or unbounded domain in the picture. The exceptions are applications #1 and #8, where rational functions outperform polynomials less decisively.

1. Elementary and special functions. Classically, approximation theory brings to mind the problem of designing subroutines for computers to evaluate elementary functions, like $\sin(x)$, and special functions, like Airy or Bessel functions. For some of these applications, especially when the number of digits of accuracy required is known in advance, rational approximations prove to be the best choice. A classic project in this line is the SPECFUN software package [Cody 1993], descendant of the earlier FUNPACK [Cody 1975], which uses rational best approximations to evaluate Bessel functions, error functions, gamma functions and exponential integrals to 18 digits of accuracy. For many years a driving force behind these software products and an expert on the matter of practical rational approximations was W. J. Cody at the Argonne National Laboratory; Cody's version of the rational Remez algorithm is described in [Cody, Fraser & Hart 1968]. For a presentation of some of the state of the art early in the 21st century, see [Muller 2006].

2. Digital filters. In electrical engineering, the construction of low-pass, high-pass, and other digital filters often involves approximation of functions with jumps. (For these problems the approximation domain is usually the unit circle in the complex plane.) Jumps amount to singularities on or near the domain of approximation, and Theorem 8.3 implies that polynomials have no chance of rapid convergence for such functions. As Newman's theorem would lead us to expect, rational approximations sometimes do much better. Engineers use the term FIR (Finite Impulse Response) for polynomial filters and IIR (Infinite Impulse Response) in the rational case [Oppenheim, Schafer & Buck 1999].

3. Convergence acceleration for sequences and series. The mathematical sciences are full of problems of extrapolation. For example, one might be interested in $\lim_{h\to 0} f(h)$, where $f(h)$ is a quantity computed numerically on a grid of spacing h. For such a problem, f is often analytic at $h = 0$, in which case *Richardson extrapolation,* based on interpolating the data by a polynomial, may be beautifully effective. On the other hand, suppose we want to evaluate $\lim_{n\to\infty} a_n$ for a sequence $\{a_n\}$. We can regard this problem too as $\lim_{h\to 0} f(h)$ with the definition $f(1/n) = a_n$, but now, in many applications, $f(h)$ will not be analytic at $h = 0$ and Richardson extrapolation will be ineffective. The more powerful extrapolation methods that have been developed for such problems, such as Aitken extrapolation and the epsilon algorithm, are mostly based on rational approximations. See Chapter 28.

4. Determination of poles. Suppose a function f is analytic on $[a, b]$ and has some real or complex poles nearby whose positions and residues are of interest. Classic examples of such problems arise in the study of phase transitions in condensed matter physics. If we approximate f by polynomials on $[a, b]$, then by Theorem 8.3, the convergence fails outside a ρ-ellipse of analyticity, so not much information about poles can be obtained. If we approximate by rational functions, exponential convergence to some of the poles can often be achieved. Specifically, a good strategy is to consider the poles of r_{mn} for moderate values of n, where r_{mn} is a rational approximant to f obtained by Padé or Chebyshev–Padé approximation or rational interpolation or least-squares. See Chapters 26–28.

5. Analytic continuation. If f is analytic on $[a, b]$, then in many applications it can be analytically continued, in theory, to the rest of the complex plane, apart from exceptional points and curves in the form of poles, other singularities, and branch cuts. Computing such continuations numerically, however, is a difficult problem. One could try approximating f by a polynomial, but this approach will be useless outside the largest Bernstein ellipse in which f is analytic. Rational functions, by contrast, may be effective for continuation much further out. Again see Chapter 28.

6. Eigenvalues and eigenvectors of matrices. Suppose we want to compute an eigenvector of a matrix A. One approach, the *power method,* is to pick a starting vector x and compute $\lim_{n\to\infty} A^n x$, but the convergence of this polynomial-based idea is very slow in general. A much faster method, *inverse iteration*, is based on rational approximations: find an approximation μ to some eigenvalue λ and compute $\lim_{n\to\infty} (A - \mu I)^{-n} x$. The convergence gets faster the closer μ is to the singularity λ, and exploitation of this effect leads to the spectacularly effective QR algorithm for matrix eigenvalues and eigenvectors [Francis 1961]. Experts in numerical linear algebra do not usually think about rational approximations when discussing inverse iteration or the QR algorithm, but such approximations come explicitly to the fore in the analysis of extensions such as shift-and-invert Arnoldi or rational Krylov iteration [Güttel 2010].

7. Model reduction and optimal control. A major topic in numerical linear algebra and control theory is the approximation of complex input-output systems by simpler ones for more efficient computation. Via the Laplace transform, problems of this kind (in the case of continuous as opposed to discrete time) can in many cases be reduced to problems of approximation on the imaginary axis in the complex plane. The unbounded domain makes rational approximations a natural choice, and in fortunate cases, a system with hundreds of thousands of degrees of freedom may be reduced to a model with just dozens or hundreds. One set of methods for such problems goes by the name of H_∞ approximation, based on results by Adamyan, Arov and Krein [1971] and Glover [1984] that are related to CF approximation (Chapter 20). For more information see [Antoulas 2005, Zhou, Doyle & Glover 1996, Embree & Sorensen 2012].

8. Exponential of a matrix. A famous paper in numerical analysis is "Nineteen dubious ways to compute the exponential of a matrix", by Moler and Van Loan in 1978, reprinted in expanded form 25 years later [Moler & Van Loan 2003]. These authors compared many algorithms for computing e^A and reached the conclusion that the most effective was a scaling-and-squaring method based on Padé approximation [Ward 1977]. Here, first A is scaled so that its norm is on the order of 1. Then e^A is approximated by $r(A)$, where r is a type (n, n) Padé approximant to e^x (Chapter 27). This is an example where rational approximations outperform polynomials not decisively but by a more or less constant factor. This approach is used by the matrix exponential program `expm` in Matlab, which for many years was based on type $(6, 6)$ Padé approximation. A more careful analysis of the scaling-and-squaring algorithm was later provided by Higham [2009], who concluded that a better choice was type $(13, 13)$, and the `expm` code was adjusted accordingly in Matlab Version 8. In [Higham & Al-Mohy 2010, Appendix] the authors conclude that Padé approximants are up to 23% more efficient than Taylor polynomials in this application.

9. Numerical solution of stiff PDEs. A particularly important set of problems related to matrix exponentials are derived from partial differential equations. The starting point of such applications is the Laplace operator Δ on a spatial domain Ω with Dirichlet boundary conditions, which has an infinite set of negative real eigenvalues diverging to $-\infty$. To solve the heat equation $\partial u/\partial t = \Delta u$ numerically on Ω with initial data $u(x, 0) = u_0$, one would like to be able to compute the matrix exponential product $e^{tA}v_0$, where A is a matrix discretization of Δ and v_0 is a discretization of u_0. The wide range of eigenvalues makes such a problem "stiff", posing challenges for numerical methods. One method for coping with stiffness is to find a rational function $r(x)$ that approximates e^x accurately on $(-\infty, 0]$, hence in particular at all of the eigenvalues of A, and then to compute $r(tA)v_0$. Polynomials cannot approximate a bounded function on an infinite interval, but rational functions can. This problem of rational approximation of e^x on $(-\infty, 0]$ goes back to Cody, Meinardus and Varga [1969], whose "1/9 conjecture", eventually settled by Gonchar and Rakhmanov [1986], is the other famous problem considered in Chapter 25. Generalizations have become important in scientific computing in recent years in the design of *exponential integrators* for the fast numerical solution of stiff nonlinear ordinary and partial differential equations [Hochbruck & Ostermann 2010, Kassam & Trefethen 2005, Schmelzer & Trefethen 2007].

10. Quadrature formulas. As we have seen in Chapter 19, a quadrature formula approximates an integral $I = \int_a^b f(x)dx$ by a finite linear combination $I_n = \sum_{k=0}^n w_k f(x_k)$. If the weights w_k are interpreted as residues of a rational function $r(x)$ with poles at the nodes x_k, then by estimation of a Cauchy integral over a contour Γ enclosing $[a, b]$ in the complex plane, one can show that the error $I - I_n$ is bounded in terms of the size of f in the region enclosed by Γ times the error in approximation of the analytic function $\log((x+1)/(x-1))$ by r over the same region [Takahasi & Mori 1971]. So every quadrature formula is connected with a rational approximation problem. In fact, Gauss's original derivation of the $(n+1)$-point Gauss quadrature formula on $[-1, 1]$ was based on exactly this connection: he used type $(n, n+1)$ Padé approximation of $\log((x+1)/(x-1))$ at $x = \infty$ [Gauss 1814].

11. Adaptive spectral methods for PDEs. The barycentric interpolation formula has the form of a rational function that reduces to a polynomial for a special choice of weights (Chapter 5). Regardless of the choice of weights, however, one still gets an interpolant, and in some applications there is no compelling reason to force the interpolant to be a polynomial. This opens up the possibility of much more flexible rational interpolants, which have the particular advantage of not being so sensitive to the distribution of the interpolation points. These ideas originated with Salzer [1981] and Schneider and Werner [1986], building on earlier work going as far back as Jacobi [1846], and were later developed by Berrut [1988], and Floater and Hormann [2007]. For ordinary and partial differential equations, they form the basis of adaptive spectral methods for solving problems whose solutions have singularities close to the region of approximation [Berrut, Baltensperger & Mittelmann 2005, Tee & Trefethen 2006, Hale & Tee 2009].

12. One-way wave equations. Our final application became well known in the 1970s and 1980s [Halpern & Trefethen 1988]. The usual wave equation permits energy propagation in all directions, but there are applications where one would like to restrict to half the permitted angles, a 180° range. For example, this idea is useful in underwater acoustics [Tappert 1977], in geophysical migration [Claerbout 1985], and in the design of absorbing boundary conditions for numerical simulations [Lindman 1975, Engquist & Majda 1977]. How can we define a system that behaves like $u_{tt} = u_{xx} + u_{yy}$ for leftgoing waves, say, with negative x-component of velocity, while not propagating rightgoing waves? (The subscripts represent partial derivatives.) A Fourier transform shows that the dispersion relation of such a system should be $\xi = \omega\sqrt{1 - s^2}$, where $s = \eta/\omega$ and ω, ξ, η are the dual variables to t, x, y. Only the positive branch of the square root should be present, making this system a *pseudodifferential operator.* However, a rational approximation

$\sqrt{1-s^2} \approx r(s)$ simplifies this to a differential equation. For example, the type $(2,2)$ Padé approximation $r(s) = (1-\frac{3}{4}s^2)/(1-\frac{1}{4}s^2)$ leads to the PDE $u_{xtt} - \frac{1}{4}u_{xyy} = u_{ttt} - \frac{3}{4}u_{tyy}$, sometimes known as the "45° equation" because it has high accuracy approximately for angles up to 45°. In this application, rational functions are superior to polynomials both because of higher accuracy in view of the singularities at $s = \pm 1$, and because polynomial approximations lead to PDEs that are ill-posed [Trefethen & Halpern 1986].

We have just seen a list of twelve applications. In concluding this chapter I would like to consider what light these may shed on the biggest question of all, namely, what is the use of approximation theory?

To see some possible views, let us go back to 1901. That was the year of Runge's landmark paper (Chapter 13), whose title was[1]

"On empirical functions and interpolation between equidistant ordinates."

In reading this today, one is struck by the word "empirical". The empirical theme is echoed in the opening sentence:

The relationship between two measurable quantities can, strictly speaking, not be found by observation.

Runge goes on to mention "observations" six times more in the opening paragraph. It would seem that his motivation is the processing of scientific data: interpolation in the traditional sense of evaluating a function at points lying between those at which it is listed in a table.

The next year, 1902, brought another landmark of approximation theory: Kirchberger's PhD thesis under Hilbert in Göttingen, which included the first systematic statement and proof of the equioscillation theorem for polynomial approximation (Theorem 10.1). Here is the first paragraph of Kirchberger's thesis, as reprinted in the first paragraph of his published paper a year later [1903], setting forth a clear motivation for approximation theory. We may imagine that this was probably also Hilbert's view of the subject.[2]

[1] Title: "Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten." First sentence: "Die Abhängigkeit zwischen zwei messbaren Grössen kann, strenge genommen, durch Beobachtung überhaupt nicht gefunden werden."

[2] "Mit dem Begriff der Funktion ist das Postulat der numerischen Berechnung der Funktionswerte für irgendwelche Werte der unabhängigen Variabeln gegeben. Da aber die vier elementaren Spezies der Addition, Subtraktion, Multiplikation und Division,

The notion of a function entails the assumption that a numerical value of the function can be calculated for any value of the independent variable. But since the only operations that can really be carried out numerically are the four elementary operations of addition, subtraction, multiplication and division, or strictly speaking only the first three of these, it follows that we are really only masters of more general functions insofar as we can replace them by rational functions, that is, represent them approximately. This highlights the great significance of approximation problems for the whole of mathematics and the special role of approximation by polynomials and rational functions. Indeed, for numerical calculation at least, any use of other approximations such as trigonometric functions presupposes that these can in turn be approximated by rational functions.

Updated to 2012, we may say that Kirchberger's justification of approximation theory is all about *machine arithmetic.* Approximation by polynomials and rational functions is important, he is saying, because ultimately computers can only carry out polynomial and rational operations.

Both Runge's emphasis on data and Kirchberger's emphasis on arithmetic capture aspects of approximation theory that remain valid today. In particular, Kirchberger's paragraph seems a remarkably clear statement of a justification of approximation theory that in a certain philosophical sense seems almost unarguable (although the line between "primitive" operations like + and "derived" ones like $\sin(\cdot)$ is not always so clear on actual computers, with their multiple levels of hardware, software and microcode). The same argument is often seen nowadays.

Nevertheless, I do not think data analysis or machine arithmetic get at the heart of why approximation theory is important and interesting. In fact I don't think Runge's words even capture the truth of why *he* was interested in the subject! (He becomes more of a mathematician in the second half of his paper.) What these observations miss is the importance of *algorithms.*

oder streng genommen nur die erste drei derselben, die einzigen numerisch ausführbaren Rechnungsarten, alle andern aber nur insoweit durchführbar sind, als sie sich auf diese zurückführen lassen, so folgt hieraus, dass wir sämtliche Funktionen nur insoweit numerisch beherrschen, als sie sich durch rationale Funktionen ersetzen, d. h. angenähert darstellen lassen. Hieraus erhellt die große Bedeutung der Annäherungsprobleme für die gesamte Mathematik und die ausgezeichnete Stellung, die die Probleme der Annäherung durch rationale oder ganze rationale Funktionen einnehmen. In der Tat setzt, wenigstens für die numerische Berechnung, jede Annäherung durch andere, z. B. trigonometrische Funktionen, die annäherungsweise Ersetzbarkeit dieser Funktionen durch rationale voraus."

Let us look again at the list of applications. Kirchberger's motivation could be said to be on target for #1 and #2 (evaluation of functions, digital filters), and Runge's for #3, #4, and #5 (extrapolation, determination of poles, analytic continuation). But the remaining seven items need to be accounted for in other ways. It is noteworthy that applications #6 to #9 all involve matrices, sometimes of very large dimension (eigenvalues and eigenvectors, model reduction, exponentials of matrices, stiff PDEs). Applications #9 to #12 all involve integrals and differential equations (stiff PDEs, quadrature, adaptive spectral methods, one-way wave equations). In most of these problems we seem a long way from scalars x and $r(x)$: the polynomial and rational operations are applied to matrices and operators, not just numbers.

Chebfun provides another interesting data point (for polynomials rather than rational functions). Chebfun is built on a century of developments in polynomial interpolation and approximation, and it makes it possible to work with univariate functions numerically in almost unlimited ways. A particularly important Chebfun capability is finding roots of a function $f(x)$, which enables many further operations like computing extrema, absolute values, and 1-norms. Chebfun finds the roots by the algorithm proposed by Good [1961] and Boyd [2002] and described in Chapter 18: approximate f by polynomial interpolants, then find roots of the polynomials by computing eigenvalues of colleague matrices. This is as powerful an application of approximation theory as one could ask for, but it has little to do with data analysis or machine arithmetic.

Why are polynomial and rational approximations useful? Not because $r(x)$ is easier to evaluate than $\exp(x)$, but because $r(A)$ is easier to evaluate than $\exp(A)$, and $r(\partial/\partial x)$ is easier to evaluate than $\exp(\partial/\partial x)$! Not because we can *evaluate* $p(x)$, but because we can *find its roots*!

Summary of Chapter 23. *Rational functions are more powerful than polynomials for approximating functions near singularities or on unbounded domains. This is the reason for their importance in approximation theory and approximation practice.*

Exercise 23.1. Examples of partial fractions. Express the following functions in partial fraction form: (a) $x^3/(1-x)$ (b) $x/(x^2-4)$, (c) $x^2/(x^2-4)^2$, (d) $(1-x^3)/(1+x^2)$.

Exercise 23.2. Uses of partial fractions. (a) Express the function $r(x) = (x(x+1)(x+2))^{-1}$ in partial fraction form. (b) What is its integral from 1 to t? (c) What is the sum of the infinite series $r(1) + r(2) + r(3) + \cdots$?

Exercise 23.3. Another infinite series. (a) Based on numerical experiments, conjecture a value of the infinite sum $1/(1 \cdot 3 \cdot 5) + 2/(3 \cdot 5 \cdot 7) + 3/(5 \cdot 7 \cdot 9) + \cdots$. (b) Verify your conjecture with partial fractions.

Exercise 23.4. A trigonometric identity. Verify the identity $1/(1 \cdot 3 \cdot 5) - 1/(7 \cdot 9 \cdot 11) + 1/(13 \cdot 15 \cdot 17) - \cdots = \pi/48$.

Exercise 23.5. Polynomial vs. rational experiments. Produce plots comparing $E_{2n,0}(f)$ and $E_{n,n}(f)$ for the following functions f defined on $[-1,1]$: (a) $\log(1+x^2)$, (b) $\tanh(5x)$, (c) $\exp(x)/(2-x)$.

Exercise 23.6. Approximation of a gamma function. Consider the function $f(x) = \Gamma(x+2)$ on $[-1,1]$, which has simple poles at $x = -2, -3, \ldots$. Determine analytically the geometric convergence rates to be expected as $m \to \infty$ for rational approximants to f of types (a) $(m, 0)$, (b) $(m, 1)$, (c) $(m, 2)$.

Chapter 24

Rational best approximation

```
ATAPformats
```

Chapter 10 considered best or “minimax” approximation by polynomials, that is, approximation in the ∞-norm, where optimality is characterized by an equioscillating error curve. This chapter presents analogous results for approximation by rational functions. Much remains the same, but a crucial new feature is the appearance in the equioscillation condition of a number known as the *defect*, which leads to the phenomenon of square blocks of degenerate entries in the “Walsh table” of best approximations. This complication adds a fascinating new ingredient to the theory, but it is a complication with destructive consequences in terms of the fragility of rational approximations and the difficulty of computing them numerically. An antidote to some such difficulties may be the use of algorithms based on linearized least-squares and the singular value decomposition, a theme we shall take up in Chapters 26 and 27.

Another new feature in rational approximation is that we must now be careful to distinguish real and complex situations, because of a curious phenomenon: best rational approximations to real functions are in general complex. This effect is intriguing, but it has little relevance to practical problems, so for the most part we shall restrict our attention to approximations in the space $\mathcal{R}_{mn}^{\text{real}}$ consisting of functions in $\mathcal{R}_{mn}$ with real coefficients.

We will first state the main theorem, then give some examples, and then present a proof. To begin the discussion, we must define the defect. Suppose $r \in \mathcal{R}_{mn}$, that is, r is a rational function of type (m, n). As discussed in the last chapter, this means that r can be written as a fraction p/q in lowest terms with p and q having exact degrees $\mu \le m$ and $\nu \le n$. The *defect* d of

r in $\mathcal{R}_{mn}$ is the number between 0 and n defined by

$$d = \min\{m - \mu, n - \nu\} \geq 0. \tag{24.1}$$

Note that d is a measure of how far *both* the numerator and the denominator degrees fall short of their maximum allowed values. Thus $(1 - x^2)/(1 + x^2)$, for example, has defect 0 in $\mathcal{R}_{22}$ or $\mathcal{R}_{23}$ and defect 1 in $\mathcal{R}_{33}$.

A special case to be noted is the situation in which $r = 0$, that is, r is identically zero. Recall that in this case we defined $\mu = -\infty$ and $\nu = 0$, so that r is said to have exact type $(-\infty, 0)$. The definition (24.1) remains in force in this case, so if $r = 0$, we say that r has defect $d = n$ in $\mathcal{R}_{mn}$, regardless of m.

The reason why defects matter has to do with the counting of zeros. Suppose $r = p/q \in \mathcal{R}_{mn}$ has exact type (μ, ν) and $\tilde{r} = \tilde{p}/\tilde{q}$ is another function in $\mathcal{R}_{mn}$. Then we have

$$r - \tilde{r} = \frac{p}{q} - \frac{\tilde{p}}{\tilde{q}} = \frac{p\tilde{q} - \tilde{p}q}{q\tilde{q}},$$

a rational function of type $(\max\{\mu+n, m+\nu\}, n+\nu)$. By (24.1), this implies that $r - \tilde{r}$ is of type $(m+n-d, 2n-d)$. Thus $r - \tilde{r}$ can have at most $m+n-d$ zeros, and this zero count is a key to equioscillation and uniqueness results.

Here is our main theorem. The study of rational best approximations goes back to Chebyshev [1859], including the idea of equioscillation, though without a precise statement of what form an alternant must take. Existence was first proved by de la Vallée Poussin [1911] and Walsh [1931], and equioscillation is due to Achieser [1930].

Theorem 24.1. Equioscillation characterization of best approximants. *A real function $f \in C([-1, 1])$ has a unique best approximation $r^* \in \mathcal{R}_{mn}^{\text{real}}$, and a function $r \in \mathcal{R}_{mn}^{\text{real}}$ is equal to r^* if and only if $f - r$ equioscillates between at least $m + n + 2 - d$ extreme points, where d is the defect of r in $\mathcal{R}_{mn}$.*

"Equioscillation" here is defined just as in Chapter 10. For $f - r$ to equioscillate between k extreme points means that there exists a set of numbers $-1 \leq x_1 < \cdots < x_k \leq 1$ such that

$$f(x_j) - r(x_j) = (-1)^{j+i}\|f - r\|, \qquad 1 \leq j \leq k$$

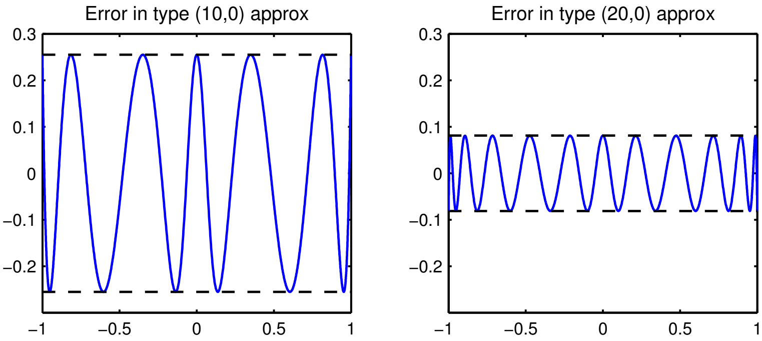
with $i = 0$ or 1. Here and throughout this chapter, $\|\cdot\|$ is the supremum norm.

We now give some examples. To begin with, here is a function with a spike at $x = 0$:

```
x = chebfun('x'); f = exp(-100*x.^2);
```

Polynomial approximations of this function converge rather slowly. For example, it takes $n = 20$ to achieve one digit of accuracy:

```
[p,err] = remez(f,10);
subplot(1,2,1), hold off, plot(f-p), hold on
plot([-1 1],err*[1 1],'--k'), plot([-1 1],-err*[1 1],'--k')
FS = 'fontsize';
title('Error in type (10,0) approx',FS,9), ylim(.3*[-1 1])
[p,err] = remez(f,20);
subplot(1,2,2), hold off, plot(f-p), hold on
plot([-1 1],err*[1 1],'--k'), plot([-1 1],-err*[1 1],'--k')
title('Error in type (20,0) approx',FS,9), ylim(.3*[-1 1])
```



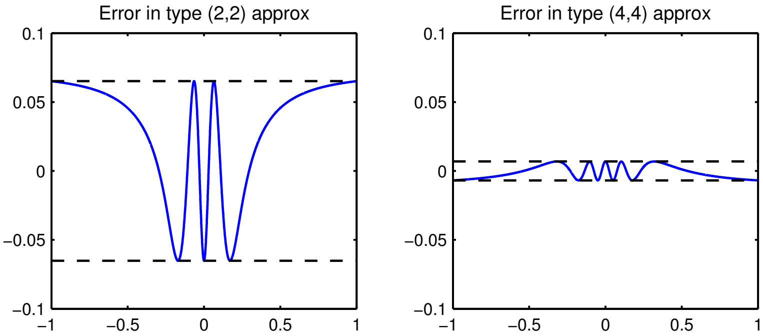
Notice that the extreme points of these error curves are distributed all across $[-1, 1]$, even though the challenging part of the function would appear to be in the middle. As discussed in Chapter 16, this is typical of polynomial best approximations.

If we switch to rational approximations, which can also be computed by Chebfun's `remez` command [Pachón & Trefethen 2009, Pachón 2010], the accuracy improves. Here we see error curves for approximations of types $(2, 2)$ and $(4, 4)$, with much smaller errors although the degrees are low. Note that most of the extreme points are now localized in the middle.

```
[p,q,rh,err] = remez(f,2,2);
subplot(1,2,1), hold off, plot(f-p./q), hold on
plot([-1 1],err*[1 1],'--k'), plot([-1 1],-err*[1 1],'--k')
```

```
title('Error in type (2,2) approx',FS,9), ylim(.1*[-1 1])
[p,q,rh,err] = remez(f,4,4);
subplot(1,2,2), hold off, plot(f-p./q), hold on
plot([-1 1],err*[1 1],'--k'), plot([-1 1],-err*[1 1],'--k')
title('Error in type (4,4) approx',FS,9), ylim(.1*[-1 1])
```

```
Warning:  This command is deprecated. Use minimax instead.
```

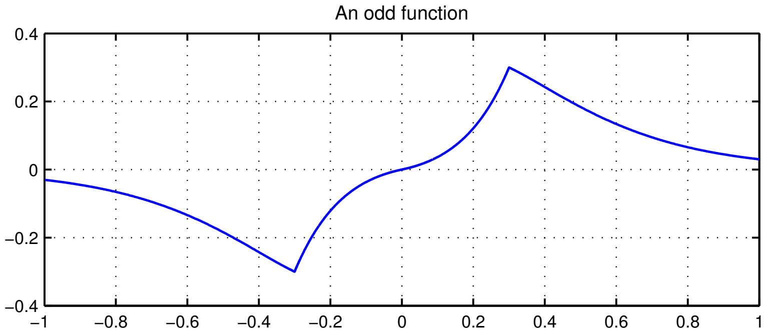


The error curves just plotted provide good examples of the role of the defect in the characterization of best approximants. The function f is even, and so are its best approximations (Exercise 24.1). Thus we expect that the type $(2, 2)$, $(3, 2)$, $(2, 3)$ and $(3, 3)$ best approximations will all be the same function, a rational function of exact type $(2, 2)$ whose error curve has 7 points of equioscillation. For $(m, n) = (2, 2)$, the defect is 0 and there is one more equioscillation point than the minimum $m + n + 2 - d = 6$. For $(m, n) = (3, 2)$ or $(2, 3)$, the defect is 0 and the number of equiscillation points is exactly the minimum $m+n+2-d$. For $(m, n) = (3, 3)$, the defect is 1 and the number of equiscillation points is again exactly the minimum $m + n + 2 - d$.

Similarly, the error curve in the plot on the right, with 11 extrema, indicates that this rational function is a best approximation not only of type $(4, 4)$ but also of types $(5, 4)$, $(4, 5)$, and $(5, 5)$.

Here is another example, an odd function:

```
f = x.*exp(-5*abs(abs(x)-.3));
clf, plot(f), grid on, ylim(.4*[-1 1])
title('An odd function','fontsize',9)
```



If we look for a best approximation of type $(4,5)$, we find that the numerator has exact degree 3:

```
[p,q,rh,err] = remez(f,4,4); format short, chebpoly(p)
```

```
Warning:  This command is deprecated. Use minimax instead.
Warning: CHEBPOLY is deprecated. Please use CHEBCOEFFS instead.
ans =
    0.0045    0.0000    0.0154    0.0000
```

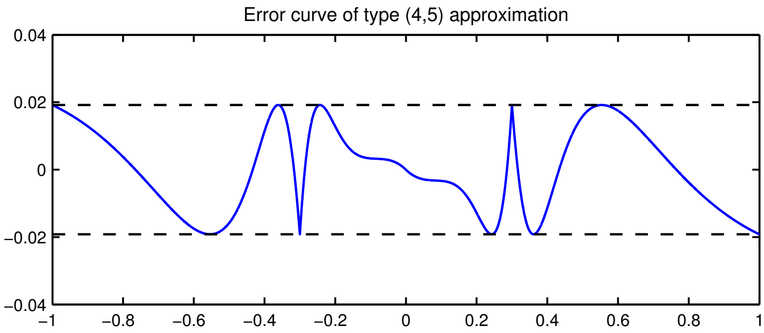
and the denominator has exact degree 4:

```
chebpoly(q)
```

```
ans =
    0.0574    0.0000    0.1987    0.0000    0.1468
```

The defect is 1, so there must be at least $4+5+2-1=10$ extreme points in the error curve. In fact, there are exactly 10:

```
plot(f-p./q), hold on, ylim(.04*[-1 1])
plot([-1 1],err*[1 1],'--k'), plot([-1 1],-err*[1 1],'--k')
title('Error curve of type (4,5) approximation','fontsize',9)
```



We conclude that r is the best approximation of types $(4,4),(4,5),(3,4)$ and $(3,5)$.

Let us now turn to the proof of Theorem 24.1. For polynomial approximations, our analogous theorem was Theorem 10.1, whose proof proceeded in four steps:

1. Existence proof via compactness,

2. Equioscillation ⇒ optimality,

3. Optimality ⇒ equisoscillation,

4. Uniqueness proof via equioscillation.

For rational functions, we shall follow the same sequence. The main novelty is in step 1, where compactness must be applied in a subtler way. We shall see an echo of this argument one more time in Chapter 27, in the proof of Theorem 27.1 for Padé approximants.

Part 1 of proof: Existence via compactness. For polynomial approximation, in Chapter 10, we noted that $\|f-p\|$ is a continuous function on $\mathcal{P}_n$, and since one candidate approximation was the zero polynomial, it was enough to look in the bounded subset $\{p \in \mathcal{P}_n : \|f-p\| \le \|f\|\}$. Since this set was compact, the minimum was attained.

For rational functions, $\|f-r\|$ is again a continuous function on $\mathcal{R}_{mn}^{\rm real}$, and again it is enough to look in the bounded subset $\{r \in \mathcal{R}_{mn}^{\rm real} : \|f-r\| \le \|f\|\}$, or more simply, the larger bounded set $\{r \in \mathcal{R}_{mn}^{\rm real} : \|r\| \le 2\|f\|\}$. The difficulty is that bounded sets of rational functions are not in general compact. To illustrate this fact, consider the family of functions

$$r_\varepsilon(x) = \frac{x^3+\varepsilon}{x^2+\varepsilon}, \tag{24.2}$$

where $\varepsilon > 0$ is a parameter. For each ε, $r_\varepsilon(x)$ is a continuous function on $[-1,1]$ with $\|r_\varepsilon\| = 1$. As $\varepsilon \to 0$, however, r_ε behaves discontinuously:

$$\lim_{\varepsilon\to 0} r_\varepsilon(x) = \begin{cases} 1 & x = 0, \\ x & x \ne 0. \end{cases}$$

So we cannot find a limit function r_0 by taking a limit as $\varepsilon \to 0$. What saves us, however, is that the spaces of numerators and denominators are both compact, so we can argue that the numerators and denominators separately approach limits p_0 and q_0, which in this example would be x^3 and x^2. We then define a limiting rational function by $r_0 = p_0/q_0$ and argue by continuity that it has the necessary properties. This kind of reasoning is spelled out in greater generality in [Walsh 1931].

Suppose then that $\{r_k\}$ is a sequence of functions in $\mathcal{R}_{mn}^{\text{real}}$ with $\|r_k\| \le 2\|f\|$ and

$$\lim_{k\to\infty} \|f - r_k\| = E = \inf_{r\in\mathcal{R}_{mn}^{\text{real}}} \|f - r\|.$$

Write each r_k in the form p_k/q_k with $p_k \in \mathcal{P}_m$, $q_k \in \mathcal{P}_n$, $q_k(x) \ne 0$ for all $x \in [-1,1]$, and $\|q_k\| = 1$, hence $\|p_k\| \le \|q_k\|\|r_k\| \le 2\|f\|$. Since $\{p_k\}$ and $\{q_k\}$ lie in compact sets, we may assume by passing to a subsequence if necessary that $p_k \to p^*$ and $q_k \to q^*$ for some $p^* \in \mathcal{P}_m$ and $q^* \in \mathcal{P}_n$. Since $\|q_k\| = 1$ for each k, $\|q^*\| = 1$ too, and thus q^* is not identically zero but has at most a finite set of zeros on $[-1,1]$. Now define $r^* = p^*/q^* \in \mathcal{R}_{mn}^{\text{real}}$. For all $x \in [-1,1]$ except perhaps the zeros of q^*, $|f(x) - r^*(x)| = \lim_{k\to\infty} |f(x) - r_k(x)| \le E$. By continuity, the same must hold for all $x \in [-1,1]$, with p^* having zeros in $[-1,1]$ wherever q^* does. Thus r^* is a best approximation to f. ▮

Part 2 of proof: Equioscillation ⇒ optimality. Suppose $f - r$ takes equal extreme values with alternating signs at $m+n+2-d$ points $x_0 < x_1 < \cdots < x_{m+n+1-d}$, and suppose $\|f - \tilde r\| < \|f - r\|$ for some $\tilde r \in \mathcal{R}_{mn}^{\text{real}}$. Then $r - \tilde r$ must take nonzero values with alternating signs at the equioscillation points, implying that it must take the value zero in at least $m + n + 1 - d$ points in-between. However, as observed above, $r - \tilde r$ is of type $(m+n-d, 2n-d)$. Thus it cannot have $m + n + 1 - d$ zeros unless it is identically zero, a contradiction. ▮

Part 3 of proof: Optimality ⇒ equisoscillation. Suppose $f - r$ equioscillates at fewer than $m + n + 2 - d$ points, and set $E = \|f - r\|$. Without loss of generality suppose the leftmost extremum is one where $f - r$ takes the value $-E$. Then by a compactness argument, for all sufficiently small $\varepsilon > 0$, there are numbers $-1 < x_1 < \cdots < x_k < 1$ with $k \le m + n - d$ such that $(f - r)(x) < E - \varepsilon$ for $x \in [-1, x_1 + \varepsilon] \cup [x_2 - \varepsilon, x_3 + \varepsilon] \cup [x_4 - \varepsilon, x_5 + \varepsilon] \cup \cdots$ and $(f - r)(x) > -E + \varepsilon$ for $x \in [x_1 - \varepsilon, x_2 + \varepsilon] \cup [x_3 - \varepsilon, x_4 + \varepsilon] \cup \cdots$. Let r be written in the form p/q, where p has degree $\mu \le m - d$ and q has degree $\nu \le n - d$, with p and q having no roots in common. The proof now consists of showing that r can be perturbed to a function $\tilde r = (p + \delta p)/(q + \delta q) \in \mathcal{R}_{mn}$ with the properties that $\|\tilde r - r\| < \varepsilon$ and $\tilde r - r$ is strictly negative for $x \in [-1, x_1 - \varepsilon] \cup [x_2 + \varepsilon, x_3 - \varepsilon] \cup [x_4 + \varepsilon, x_5 - \varepsilon] \cup \cdots$ and strictly positive for $x \in [x_1 + \varepsilon, x_2 - \varepsilon] \cup [x_3 + \varepsilon, x_4 - \varepsilon] \cup \cdots$. Such a function $\tilde r$ will have error less than E throughout the whole interval $[-1,1]$. We calculate

$$\tilde{r} = \frac{p+\delta p}{q+\delta q} = \frac{(p+\delta p)(q-\delta q)}{q^2} + O(\|\delta q\|^2)$$

and therefore

$$\tilde{r} - r = \frac{q\delta p - p\delta q}{q^2} + O(\|\delta p\|\|\delta q\| + \|\delta q\|^2).$$

We are done if we can show that δp and δq can be chosen so that $q\delta p - p\delta q$ is a nonzero polynomial of degree exactly k with roots $x_1, \dots, x_k$; by scaling this δp and δq sufficiently small, the quadratic terms above can be made arbitrarily small relative to the others, so that the required ε conditions are satisfied. This can be shown by the Fredholm alternative of linear algebra. The map from the $(m+n+2)$-dimensional set of choices of δp and δq to the $(m+n+1-d)$-dimensional space of polynomials $q\delta p - p\delta q$ is linear. To show the map is surjective, it is enough to show that its kernel has dimension $d+1$ but no more. Suppose then that $q\delta p - p\delta q$ is zero, that is, $q\delta p = p\delta q$. Then since p and q have no roots in common, all the roots of p must be roots of δp and all the roots of q must be roots of δq. In other words we must have $\delta p = gp$ and $\delta q = gq$ for some polynomial g. Since δp has degree no greater than m and δq has degree no greater than n, g can have degree no greater than d. The set of polynomials of degree d has dimension $d+1$, so we are done. ▮

Part 4 of proof: Uniqueness via equioscillation. Finally, to prove uniqueness, suppose r is a best approximation whose error curve equioscillates between extreme points at $x_0 < x_1 < \cdots < x_{m+n+1-d}$, and suppose $\|f-\tilde{r}\| \le \|f-r\|$ for some $\tilde{r} \in \mathcal{R}_{mn}^{\text{real}}$. Then (without loss of generality) $(r-\tilde{r})(x)$ must be ≤ 0 at $x_0, x_2, x_4, \dots$ and ≥ 0 at $x_1, x_3, x_5, \dots$. This implies that $r - \tilde{r}$ has roots in each of the $m+n+1-d$ closed intervals $[x_0, x_1], \dots, [x_{m+n-d}, x_{m+n+1-d}]$, and since $r-\tilde{r}$ is a rational function of type $(m+n-d, 2n-d)$, the same must hold for its numerator polynomial. We wish to conclude that its numerator polynomial has at least $m+n+1-d$ roots in total, counted with multiplicity, implying that $r = \tilde{r}$. The argument for this is the same as given in the proof of Theorem 10.1. ▮

We have now finished the substantial mathematics. It is time to look at some of the consequences.

One of the recurring themes in the subject of rational approximation is the phenomenon of *square blocks in the Walsh table.* Suppose that a real function

$f \in C([-1,1])$ is given, and consider the set of all of its real rational best approximations of type (m,n) for various $m,n \geq 0$. We can imagine these laid out in an array, with m along the horizontal and n along the vertical. This array is called the *Walsh table* for f [Walsh 1934].

Generically, all the entries in the Walsh table for a given f will be distinct, and in this case we say that f is *normal.* Sometimes, however, certain entries in the table may be repeated, and in fact this is a frequent occurrence because it happens whenever f is even or odd. If f is even, then for any nonnegative integers j and k, all of its rational approximations of types $(2j,2k)$, $(2j+1,2k)$, $(2j,2k+1)$ and $(2j+1,2k+1)$ must be the same. Similarly, if f is odd, then all of its approximations of types $(2j+1,2k)$, $(2j+2,2k)$, $(2j+1,2k+1)$ and $(2j+2,2k+1)$ must be the same. We have already seen a number of examples.

More generally, repeated entries or "degeneracies" in the Walsh table may take complicated forms. Nevertheless the equioscillation condition imposes quite a bit of structure on the chaos. Degeneracies always appear precisely in a pattern of square blocks. The following statement of this result is taken from [Trefethen 1984], where a discussion of various aspects of this and related problems can be found. We shall return to the subject of square blocks in Chapter 27, on Padé approximation.

Theorem 24.2. Square blocks in the Walsh table. *The Walsh table of best real rational approximants to a real rational function $f \in C([-1,1])$ breaks into precisely square blocks containing identical entries. (If f is rational, one of these will be infinite in extent.) The only exception is that if an entry $r = 0$ appears in the table, then it fills all of the columns to the left of some fixed index $m = m_0$.*

Proof. Given a nonrational function f, let $r \neq 0$ be a best approximation in $\mathcal{R}_{\mu\nu}^{\text{real}}$ of exact type (μ,ν). (The cases of rational f or $r = 0$ can be handled separately.) By Theorem 24.1, the number of equioscillation points of $f - r$ is $\mu + \nu + 2 + k$ for some integer $k \geq 0$. We note that r is an approximation to f in $\mathcal{R}_{mn}^{\text{real}}$ for any $m \geq \mu$ and $n \geq \nu$, and the defect is $\min\{m-\mu, n-\nu\}$. Thus by Theorem 24.1, r is the best approximation to f precisely for those values of (m,n) satisfying $m \geq \mu$, $n \geq \nu$, and $\mu+\nu+2+k \geq m+n+2-\min\{m-\mu,n-\nu\}$. The latter condition simplifies to $n \leq \nu + k$ and $m \leq \mu + k$, showing that r is the best approximation to f precisely in the square block $\mu \leq m \leq \mu + k$, $\nu \leq n \leq \nu + k$. ∎

Within a square block in the Walsh table, the defect d is equal to zero precisely in the first column and the first row. An approximation with $d = 0$ is sometimes said to be *nondegenerate*. It can have more points of equioscillation than the generic number $m + n + 2$, but never fewer.

As mentioned above, the theory of equioscillation and degeneracies is very appealing mathematically. As an example we note a result due to Werner [1964], in completion of earlier work of Maehly and Witzgall [1960]: the type (m, n) *best approximation operator*, which maps functions f to their best approximations r^*_{mn}, is continuous at f with respect to the supremum norm if and only if $f \in \mathcal{R}_{mn}$ or the corresponding function r^*_{mn} is nondegenerate. The essential reason for this effect is that if a function r^* is the best approximation to f in a nontrivial square block, then a small perturbation $f \to \tilde{f}$ might fracture that block into pieces of size 1×1 [Trefethen 1984]. If (m, n) corresponds to a degenerate position in the block, with $d > 0$, then the best approximation $\tilde{r}^*$ for such an $\tilde{f}$ would need to have a higher equioscillation number than that of r^* for f, requiring $\tilde{r}^*$ to be far from r^* if $\|f - r^*\|$ is positive.

These complications hint at some of the practical difficulties of rational approximation. For example, the Remez algorithm is based on explicit manipulation of alternant sets. If the number of extremal points is not known a priori, it is plausible that one may expect numerical difficulties in certain circumstances. Indeed, this is the case, and so far as I am aware, no implementation of the Remez algorithm for rational approximation, including Chebfun's, can be called fully robust. Other kinds of algorithms may have better prospects.

We finish by returning to the matter of best *complex* approximations to real functions. Nonuniqueness of certain complex rational approximations was pointed out by Walsh in the 1930s. Later Lungu [1971] noticed, following a suggestion of Gonchar, that the nonuniqueness arises even for approximation of a real function f on $[-1, 1]$, with examples as simple as type $(1, 1)$ approximation of $|x|$. (Exercise 24.3 gives another proof that there must exist such examples.) These observations were rediscovered independently by Saff and Varga [1978a]. Ruttan [1981] showed that complex best approximations are always better than real ones in the strict lower-right triangle of a square block, that is, when a type (m, n) best approximation equioscillates in no more than $m + n + 1$ points. Trefethen and Gutknecht [1983a] showed that for every (m, n) with $n \geq m + 3$, examples exist where the ratio of the optimal complex and real errors is arbitrarily small. Levin, Ruttan and Varga showed that the minimal ratio is exactly 1/3 for $n = m + 2$ and exactly 1/2 for $1 \le n \le m+1$ [Ruttan & Varga 1989]. None of this has much to do with practical approximation, but it is fascinating.

> Summary of Chapter 24. *Any real function $f \in C([-1,1])$ has a unique best approximation $r^* \in \mathcal{R}_{mn}^{\text{real}}$ with respect to the ∞-norm, and r^* is characterized by having an error curve that equioscillates between at least $m + n + 2 - d$ extreme points, where d is the defect of r in $\mathcal{R}_{mn}$. In the Walsh table of all best approximations to f indexed by m and n, repeated entries, if any, lie in exactly square blocks.*

Exercise 24.1. Approximating even functions. Prove that if a real function $f \in C([-1,1])$ is even, then its real best approximations of all types (m, n) are even.

Exercise 24.2. Approximating the Gaussian. The first figures of this chapter considered lower degree polynomial and rational approximations of $\exp(-100x^2)$ on $[-1,1]$. Make a plot of the errors in approximations of types $(n, 0)$ and (n, n), now taking n as high as you can. (You may find that the `cf` command takes you farther than `remez`.) How do the polynomial and rational approximations compare?

Exercise 24.3. Complex approximations and nonuniqueness. (a) Suppose a real function $f \in C([-1,1])$ takes both the values 1 and -1. Prove that no real rational function $r \in \mathcal{R}_{0n}^{\text{real}}$, for any n, can have $\|f - r\| < 1$. (b) On the other hand, show that for any $\varepsilon > 0$, there is a complex rational function $r \in \mathcal{R}_{0n}$ for some n with $\|f - r\| < \varepsilon$. (Hint: perturb f by an imaginary constant and consider its reciprocal.) (c) Conclude that type $(0, n)$ complex rational best approximations in $C([-1,1])$ are nonunique in general for large enough n.

Exercise 24.4. A function with a spike. Plot chebfuns of the function (24.2) for $\varepsilon = 1, 0.1, \dots, 10^{-6}$ and determine the polynomial degree $n(\varepsilon)$ of the chebfun in each case. What is the observed asymptotic behavior of $n(\varepsilon)$ as $\varepsilon \to 0$? How accurately can you explain this observation based on the theory of Chapter 8?

Exercise 24.5. de la Vallée Poussin lower bound. Suppose an approximation $r \in \mathcal{R}_{mn}^{\text{real}}$ to $f \in C([-1,1])$ approximately equioscillates in the sense that there are points $-1 \le s_0 < s_1 < \cdots < s_{m+n+1-d} \le 1$ at which $f - r$ alternates in sign with $|f(s_j) - r(s_j)| \ge \varepsilon$ for some $\varepsilon > 0$, where d is the defect of r in $\mathcal{R}_{mn}$. Show that the best approximation $r^* \in \mathcal{R}_{mn}^{\text{real}}$ satisfies $\|f - r^*\| \ge \varepsilon$. (Compare Exercise 10.3.)

Exercise 24.6. A rational lethargy theorem. Let $\{\varepsilon_n\}$ be a sequence decreasing monotonically to 0. Adapt the proof of Exercise 10.7 to show that that there is a function $f \in C([-1,1])$ such that $\|f - r_{nn}^*\| \ge \varepsilon_n$ for all n.

Chapter 25

Two famous problems

`ATAPformats`

In this chapter we discuss two problems of rational approximation that have been the focus of special attention over the years: approximation of $|x|$ on $[-1,1]$, a prototype of approximation of non-smooth functions, and approximation of e^x on $(-\infty,0]$, a prototype of approximation on unbounded domains. Both stories go back many decades and feature initial theorems, later conjectures based on numerical experiments, and eventual proofs of the conjectures based on mathematical methods related to potential theory. We shall not present the proofs of the sharpest results, but we shall show that the essential rates of approximation can be achieved by using the trick that appears several times in this book: if a function $f(x)$ can be written as an integral with respect to a variable s, then an approximation $r(x)$ in partial fractions form is obtained by applying a quadrature formula (19.3) to the integral.

The problem of approximation of $|x|$ on $[-1,1]$ originates at the beginning of the 20th century, when polynomial approximations of this function were of interest to Lebesgue, de la Vallée Poussin, Jackson, and Bernstein. This was an era when the fundamental results of approximability were being developed, and $|x|$ served as a function from which many other results could be derived. Bernstein's prize-winning article on the subject ran for 104 pages [1912B] and was followed by another of 57 pages [1914B]. Among other things, Bernstein proved that in best polynomial approximation of $|x|$ as $n\to\infty$, the errors decrease linearly but no faster, that is, at the rate $O(n^{-1})$ but not $o(n^{-1})$.

Why linearly? This is an example of the fundamental fact of approximation theory which we mentioned first in Chapter 7: the close connection between the smoothness of a function and its rate of approximation. The function

$f(x) = |x|$ has a derivative of bounded variation $V = 2$ on $[-1,1]$, so by Theorem 7.2, its Chebyshev projections $\{f_n\}$ satisfy

$$\|f - f_n\| \le \frac{4}{\pi(n-1)}$$

for $n \ge 2$, and its Chebyshev interpolants $\{p_n\}$ satisfy the same bound with 4 replaced by 8. Thus approximations to $|x|$ converge at least at the rate $O(n^{-1})$. What Bernstein showed is that the rate is in fact no better than this: no approximations to $|x|$ can beat Chebyshev projection or interpolation by more than a constant factor. Or to put it another way, convergence of polynomial approximants to a function f at a rate faster than $O(n^{-1})$ implies that f is in some sense smoother than $|x|$. Such results in the direction *approximability* $\Longrightarrow$ *smoothness* go by the general name of *Bernstein theorems.* In this book we have presented one result of this kind: Theorem 8.3, asserting that geometric convergence implies analyticity.

It is hard not to be curious about the constants. Bernstein in fact proved in [Bernstein 1914B] that there exists a number β such that the best approximation errors satisfy

$$E_n(|x|) \sim \frac{\beta}{n} \tag{25.1}$$

as $n \to \infty$, and he obtained the bound

$$0.278 < \beta < 0.286.$$

(Theorem 7.2 gives $\beta \le 4/\pi \approx 1.27$.) He noted as a "curious coincidence" that $1/2\sqrt{\pi} \approx 0.28209\ldots$ falls in this range, and the idea that β might take exactly this value became known as *Bernstein's conjecture.* Seventy years later, Varga and Carpenter [1985] investigated the problem numerically to great accuracy and found that in fact

$$\beta \approx 0.2801694990238691330364364 9\ldots.$$

(Of course the difference between 0.282 and 0.280 would not have the slightest practical importance.) Along with this numerical result, which was based on Richardson extrapolation, Varga and Carpenter established the rigorous bounds

$$0.2801685 < \beta < 0.2801734. \tag{25.2}$$

For example, here are the values of $nE_n(|x|)$ for $n = 1, 2, 4, \ldots, 64$, showing quadratic convergence to the limit value. A comparison with the much more accurate Table 2.1 of [Varga & Carpenter 1985] indicates that the Chebfun results are accurate in all but the last digit or two.

```
x = chebfun('x'); f = abs(x); limit = 0.280169499023869133;
disp('            n           n*err        n*err - limit')
for n = 2.^(0:6)
  [p,err] = remez(f,n);
  fprintf('%14d %16.8f %16.2e\n',n,n*err,n*err-limit)
end
```

```
            n           n*err        n*err - limit
Warning:  This command is deprecated. Use minimax instead.
             1       0.50000000         2.20e-01
Warning:  This command is deprecated. Use minimax instead.
             2       0.25000000        -3.02e-02
Warning:  This command is deprecated. Use minimax instead.
             4       0.27048360        -9.69e-03
Warning:  This command is deprecated. Use minimax instead.
             8       0.27751782        -2.65e-03
Warning:  This command is deprecated. Use minimax instead.
            16       0.27948884        -6.81e-04
Warning:  This command is deprecated. Use minimax instead.
            32       0.27999815        -1.71e-04
Warning:  This command is deprecated. Use minimax instead.
            64       0.28012659        -4.29e-05
```

Now all this is for polynomial approximation. What about rational functions? As mentioned in Chapter 23, the dramatic discovery here came from Donald Newman, fifty years after Bernstein: best rational approximants to $|x|$ converge “root-exponentially”. Newman’s bounds were these:

$$\frac{1}{2}e^{-9\sqrt{n}} \le E_{nn}(|x|) \le 3e^{-\sqrt{n}}. \tag{25.3}$$

We have already seen in the second plot of Chapter 23 what an improvement in convergent rate this is as compared with (25.1). For approximating non-smooth functions, rational functions can be far more powerful than polynomials.

Again mathematicians could not resist trying to sharpen the constants. First, Vyacheslavov [1975] found that the exact exponent is midway between Newman’s bounds of 1 and 9: it is π. Then Varga, Ruttan and Carpenter [1993] performed computations with a version of the Remez algorithm to 200 decimal places, leading to numerical evidence for the conjecture

$$E_{nn} \sim 8e^{-\pi\sqrt{n}}$$

as $n \to \infty$. Soon afterwards this result was proved by Stahl [1993]. Later Stahl generalized the result to approximation of x^α on $[0, 1]$ for any $\alpha > 0$ [Stahl 2003].

The following theorem summarizes the results we have mentioned.

Theorem 25.1. Approximation of $|x|$ on $[-1, 1]$. *The errors in best polynomial and rational approximation of $|x|$ on $[-1, 1]$ satisfy as $n \to \infty$*

$$E_{n0}(|x|) \sim \frac{\beta}{n}, \quad \beta = 0.2801\ldots \tag{25.4}$$

and

$$E_{nn}(|x|) \sim 8e^{-\pi\sqrt{n}}. \tag{25.5}$$

Proof. Equation (25.4) is due to Varga and Carpenter [1985] and (25.5) is due to Stahl [1993]. ▮

Why can rational approximations of $|x|$ achieve $O(C^{-\sqrt{n}})$ accuracy? The crucial fact is that the poles of r can be chosen to cluster near the singular point $x = 0$. In particular, a good choice is to make the poles approach 0 geometrically, for each fixed n, with a geometric factor depending on $\sqrt{n}$.

Here is a derivation of a rational approximation that achieves the right root-exponential convergence. (Arguments like this have been made by Stenger in various publications; see for example [Stenger 1986].) We start from the identity

$$\frac{1}{|x|} = \frac{2}{\pi} \int_0^\infty \frac{dt}{t^2 + x^2},$$

which is derived in calculus courses. Multiplying by x^2 gives

$$|x| = \frac{2x^2}{\pi} \int_0^\infty \frac{dt}{t^2 + x^2}. \tag{25.6}$$

(This formula is perhaps due to Roberts [1980], though the essence of the matter dates to Zolotarev in the 1870s.) The change of variables $t = e^s$, $dt = e^s ds$ converts this to

$$|x| = \frac{2x^2}{\pi} \int_{-\infty}^\infty \frac{e^s ds}{e^{2s} + x^2}, \tag{25.7}$$

which is an attractive integral to work with because the integrand decays exponentially as $|s| \to \infty$. We now get a rational approximation of $|x|$ by approximating this integral by the trapezoid rule with node spacing $h > 0$:

$$r(x) = \frac{2hx^2}{\pi} \sum_{k=-(n-2)/4}^{(n-2)/4} \frac{e^{kh}}{e^{2kh} + x^2}. \tag{25.8}$$

Here n is a positive even number, and there are $n/2$ terms in the sum, so $r(x)$ is a rational function of x of type (n, n). There are two sources of error that make $r(x)$ differ from $|x|$. The fact that the sum has been terminated at a limit $n < \infty$ introduces an error on the order of $e^{-nh/4}$, and the finite step size $h > 0$ introduces an error on the order of $e^{-\pi^2/h}$. (The integrand is analytic in the strip around the real s-axis of half-width $a = \pi/2$, corresponding to a convergence rate $e^{-2\pi a/h}$.) Balancing these sources of error suggests the condition $e^{-nh/4} \approx e^{-\pi^2/h}$, that is,

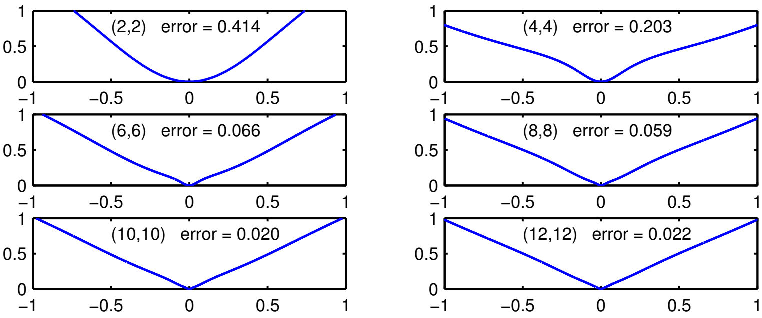
$$h \approx 2\pi/\sqrt{n}, \tag{25.9}$$

with error of order

$$e^{-(\pi/2)\sqrt{n}}. \tag{25.10}$$

We can see these approximations with an experiment.

```
for n = 2:2:12
  r = 0*x; h = 2*pi/sqrt(n);
  for k = -(n-2)/4:(n-2)/4
    r = r + exp(k*h)./(exp(2*k*h)+x.^2);
  end
  r = (2*h/pi)*x.^2.*r; err = norm(f-r,inf);
  subplot(3,2,n/2), plot(r), ylim([0 1])
  ss = sprintf('(%1d,%1d)   error = %5.3f',n,n,err);
  FS = 'fontsize'; text(-.5,.78,ss,FS,8)
end
```



The poles of (25.8)–(25.9) in the x-plane lie at

$$\pm i e^{2\pi k/\sqrt{n}}. \tag{25.11}$$

Here are these numbers (those in the upper half-plane) for the six approximations plotted above, showing the wide range of amplitudes associated with the exponential spacing.

```
disp('Poles of rational approximants to |x|:')
for n = 2:2:12
  h = 2*pi/sqrt(n); k = -(n-2)/4:(n-2)/4; y = exp(k*h);
  fprintf('%8.2ei  ',y), disp(' ')
end
```

```
Poles of rational approximants to |x|:
1.00e+00i
2.08e-01i  4.81e+00i
7.69e-02i  1.00e+00i  1.30e+01i
3.57e-02i  3.29e-01i  3.04e+00i  2.80e+01i
1.88e-02i  1.37e-01i  1.00e+00i  7.29e+00i  5.32e+01i
1.07e-02i  6.58e-02i  4.04e-01i  2.48e+00i  1.52e+01i  9.32e+01i
```

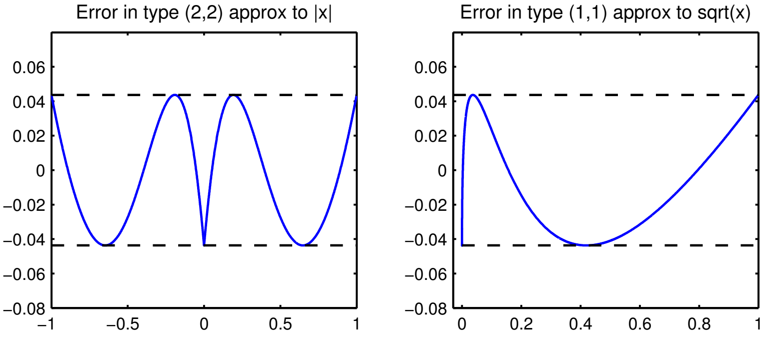
The approximations aren't optimal, but they are close. The convergence rate (25.10) as $n \to \infty$ is one-quarter of the optimal rate (25.5) in the sense that we need 4 times as large a value of n to achieve a certain accuracy in (25.10) as in (25.5).

Above, we computed errors for best polynomial approximations to $|x|$ with the Chebfun command `remez`. In the rational case, `remez` does not succeed in computing best approximations beyond a certain low order. This difficulty is related to the exponential spacing of the oscillations of $f - r^*$ near $x = 0$.

It is worth noting that the problem of approximating $|x|$ on $[-1, 1]$ is equivalent to certain other approximation problems. If $r(x)$ is a type (m, n) approximation to $|x|$ on $[-1, 1]$, then normally r will be an even function of x and m and n can be taken to be even too. Thus $r(x) = \tilde{r}(x^2)$, where $\tilde{r}$ is a rational function of type $(m/2, n/2)$. Since $\tilde{r}(x^2)$ approximates $|x|$ for $x \in [-1, 1]$, $\tilde{r}(x)$ approximates $\sqrt{x}$ for $x \in [0, 1]$. This reasoning holds for any approximations, and in particular, by counting equioscillations one finds that best type (m, n) approximation of $|x|$ on $[-1, 1]$ is equivalent to best type $(m/2, n/2)$ approximation of $\sqrt{x}$ on $[0, 1]$. The following pair of plots illustrates this equivalence. Notice that the error curves are the same apart from the scaling of the x-axis.

```
f = abs(x); [p,q,rh,err] = remez(f,2,2); clf
subplot(1,2,1), plot(f-p./q), hold on, ylim(.08*[-1 1])
plot([-1 1],err*[1 1],'--k'), plot([-1 1],-err*[1 1],'--k')
title('Error in type (2,2) approx to |x|',FS,9)
f = chebfun('sqrt(x)',[0,1],'splitting','on');
[p,q,rh,err] = remez(f,1,1);
subplot(1,2,2), plot(f-p./q), hold on, axis([-.03 1 .08*[-1 1]])
plot([-.03 1],err*[1 1],'--k'), plot([0 1],-err*[1 1],'--k')
title('Error in type (1,1) approx to sqrt(x)',FS,9)
```

```
Warning:  This command is deprecated. Use minimax instead.
Warning:  This command is deprecated. Use minimax instead.
```


Error in type (2,2) approx to |x|

Error in type (1,1) approx to sqrt(x)

For applications in scientific computing, the approximation of $\sqrt{x}$ on an interval $[a, b]$ is particularly interesting because of the case in which x is a matrix A with eigenvalues in $[a, b]$, which might come from discretizing a differential operator. Rational approximations of the square root lead to powerful algorithms for evaluating $A^{1/2}v$ for vectors v, as described in [Hale, Higham & Trefethen 2008] and [Higham 2008]. At the other end of the historical spectrum, approximation of square roots was the problem addressed by Poncelet in the very first paper on minimax approximation [Poncelet 1835].

We now turn to the second of the famous problems of this chapter: approximation of e^x on $(-\infty, 0]$. This problem was introduced in a paper of Cody, Meinardus and Varga [1969], which drew attention to the connection of such approximations with the numerical solution of partial differential equations, since a rational approximation can be used to compute the exponential of a matrix arising from a numerical discretization [Moler & Van Loan 2003].[1] Curiously, despite that good motivation from applied mathematics, the influence of this paper was mainly in theoretical approximation theory for quite a few decades, until computers and numerical linear algebra had advanced to the point where it became more practical to take advantage of algorithms based on rational functions.

The first thing we may note about approximation of e^x on $(-\infty, 0]$ is that polynomials cannot do the job at all. Since any non-constant polynomial $p(x)$ diverges to $\pm\infty$ as $x \to -\infty$, the only polynomials that can approximate e^x with finite error on $(-\infty, 0]$ are constants, so the minimax error can never be less than $1/2$.

Inverse-polynomials of the form $1/p_n(x)$, however, can be chosen to converge geometrically. This makes sense when you consider that e^x on $(-\infty, 0]$ is the same as $1/e^x$ for $x \in [0, \infty)$. Cody, Meinardus and Varga noted that to achieve geometric convergence, it is enough to consider $1/p_n(x)$, where p_n is the degree-n truncation of the Taylor series for e^x. They showed that these approximations converge at a rate $O(2^{-n})$, and then they improved this rate to $O(2.298^{-n})$ by a shift of origin. It was later proved by Schönhage [1973] that the optimal rate for inverse-polynomials is $O(3^{-n})$.

Since $1/p_n(x)$ is a rational function of type (n, n), these observations tell us that best rational type (n, n) approximations to e^x on $(-\infty, 0]$ converge at least geometrically. Newman [1974] proved that the convergence is no faster than geometric. What is the optimal rate? With twice as many parameters to work with as with inverse-polynomials, one might guess that it should be $O(9^{-n})$, and this idea became known in the 1970s as the 1/9 *conjecture.* In fact, the optimal convergence rate turned out to be $O(H^n)$ with $H \approx 1/9.28903$, a number now known as *Halphen's constant*, equal to the unique positive root of the equation

$$h(s) = \sum_{k=1}^{\infty} \frac{ks^n}{1-(-s)^n} = \frac{1}{8}. \tag{25.12}$$

This number was conjectured numerically based on Carathéodory–Fejér singular values by Trefethen and Gutknecht [1983B], verified to many digits

[1] The Cody–Meinardus–Varga paper was important in my life. As a graduate student in the Numerical Analysis Group at Stanford, I happened to come across it one evening around 1980 in a pile of Gene Golub's discarded reprints—"help yourself". Its mix of theory and numerical calculations appealed to me greatly and led to my computation of the constant 9.28903... a few years later [Trefethen & Gutknecht 1983b].

by high-precision Remes algorithms by Carpenter, Ruttan and Varga [1984], conjectured to have the exact value associated with a certain problem of elliptic functions treated by Halphen [1886] by Magnus via the Carathéodory–Fejér method [Magnus 1985, Magnus & Meinguet 2000], and then proved using quite different methods of potential theory by Gonchar and Rakhmanov [1989]. This work represents a fascinating and important line of investigation in approximation theory, and for a summary of many of the ideas with wide generalizations to related problems, a good place to start is [Stahl & Schmelzer 2009]. Presentations of some of the potential theory underlying results in this area can be found in [Saff & Totik 1997].

Following the idea presented earlier for $|x|$ on $[-1,1]$, it is interesting to see what can be achieved for this problem by the trapezoid rule approximation of a contour integral. Here is a derivation of a rational approximation that achieves the rate $O((2.849\ldots)^{-n})$, adapted from [Weideman & Trefethen 2007]; such approximations are discussed more generally in [Trefethen, Weideman & Schmelzer 2006]. We begin with a Laplace transform identity that is easily proved by residue calculus,

$$e^x = \frac{1}{2\pi i}\int \frac{e^t dt}{t-x}$$

for $x \in (-\infty, 0]$, where the integral is over any contour in the complex plane that starts at $-\infty$ below the t-axis, circles around $t=0$, and finishes at $-\infty$ above the t-axis. Choosing the contour to be a parabola, we convert this to an integral over the real s-axis by the change of variables

$$t = (is+a)^2, \quad dt = 2i(is+a)ds$$

for some constant $a > 0$, which gives

$$e^x = \frac{1}{\pi}\int \frac{e^{(is+a)^2}(is+a)ds}{(is+a)^2 - x}. \tag{25.13}$$

As in (25.8), we now approximate this integral by the trapezoid rule with node spacing $h > 0$:

$$r(x) = \frac{h}{\pi}\sum_{k=-(n-1)/2}^{(n-1)/2} \frac{e^{(ikh+a)^2}(ikh+a)}{(ikh+a)^2 - x}. \tag{25.14}$$

Here n is a positive even number, and since x rather than x^2 appears in each term we now take n terms in the sum rather than $n/2$ as in (25.8) to make $r(x)$ a rational function of x of type (n,n).

This time, the integral has square-exponential rather than just exponential decay as $s \to \infty$, so choosing $h = O(1/\sqrt{n})$ is enough to make the errors from endpoint truncation exponentially small. We also have the parameter a to play with. By taking $a = O(\sqrt{n})$, we can make the errors due to grid spacing exponentially small too, and in this fashion we can achieve geometric convergence. More precisely, the choices

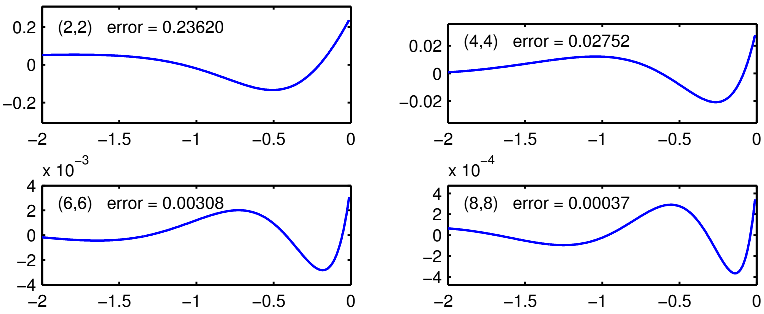
$$a = \sqrt{\frac{\pi n}{24}}, \qquad h = \sqrt{\frac{3\pi}{2n}} \tag{25.15}$$

lead to the convergence rate

$$\|f - r_{nn}\| = O(e^{-\pi n/3}) \approx O((2.849\ldots)^{-n}). \tag{25.16}$$

As before, we can see these approximations with an experiment, this time plotting $f - r$ rather than r itself.

```
x = chebfun('x',[-2,-.01]); f = exp(x);
for n = 2:2:8
  r = 0*x; h = sqrt(3*pi/(2*n)); a = sqrt(pi*n/24);
  for k = -(n-1)/2:(n-1)/2
    r = r + exp((1i*k*h+a)^2)*(1i*k*h+a)./((1i*k*h+a)^2-x);
  end
  r = (h/pi)*real(r); subplot(2,2,n/2), plot(f-r)
  err = norm(f-r,inf); ss = sprintf('(%1d,%1d)   error = %7.5f',n,n,err);
  axis([-2,0,1.3*err*[-1 1]]), text(-1.9,.85*err,ss,FS,8)
end
```



Let us summarize these results with a theorem, which goes further to include the precise leading-order asymptotic behavior of the best approximation errors as conjectured by Magnus [1994] and proved by Aptekarev [2002].

Theorem 25.2. Approximation of e^x on $(-\infty, 0]$. *The errors in best type* $(0, n)$ *and* (n, n) *rational approximation of* $\exp(x)$ *on* $(-\infty, 0]$ *satisfy as*

$n \to \infty$

$$\lim_{n\to\infty} E_{0n}^{1/n} = \frac{1}{3} \tag{25.17}$$

and

$$E_{nn} \sim 2H^{n+1/2}, \quad H = 1/9.2890254919208\ldots \tag{25.18}$$

Proof. Equation (25.17) is due to Schönhage [1973] and (25.18) to Aptekarev [2002], extending the earlier result on nth root asymptotics and the constant H by Gonchar and Rakhmanov [1989]. ▮

We finish this chapter by showing that the numerical computation of these best approximants is surprisingly easy. The crucial matter is to note that the change of variables

$$x = a\frac{s-1}{s+1}, \qquad s = \frac{a+x}{a-x} \tag{25.19}$$

where a is a positive parameter, maps the negative real axis $(-\infty, 0]$ in x to the interval $(-1, 1]$ in s. Since the mapping is a rational function of type $(1, 1)$, it transplants a rational function of type (n, n) in s or x to a rational function of type (n, n) in the other variable. In particular, for the approximation of $f(x) = e^x$ on $(-\infty, 0]$, let us define

$$F(s) = e^{a(s-1)/(s+1)}, \quad s \in (-1, 1]. \tag{25.20}$$

A good choice of the parameter is $a = 9$, which has a big effect for numerical computation in improving the conditioning of the approximation problem. We now find we have a function that can be approximated to machine precision by a Chebyshev interpolating polynomial $p(s)$ of degree less than 50:

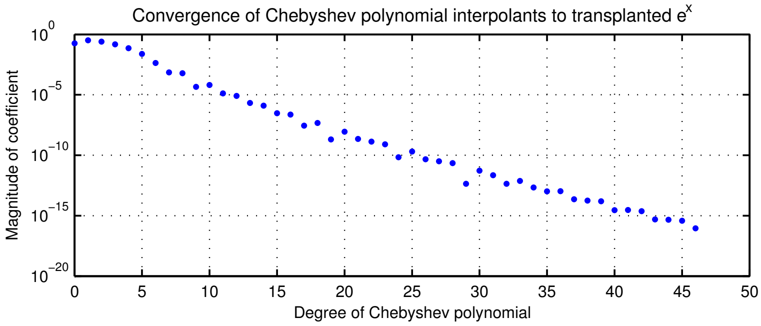
```
s = chebfun('s',[-1,1]);
F = exp(9*(s-1)./(s+1));
length(F)
```

```
ans =
    47
```

The Chebyshev series of F decreases at a good exponential rate:

```
clf, chebpolyplot(F), grid on
title(['Convergence of Chebyshev polynomial ' ...
       'interpolants to transplanted e^x'],FS,9)
```

```
Warning: CHEBPOLYPLOT is deprecated. Please use PLOTCOEFFS instead.
```

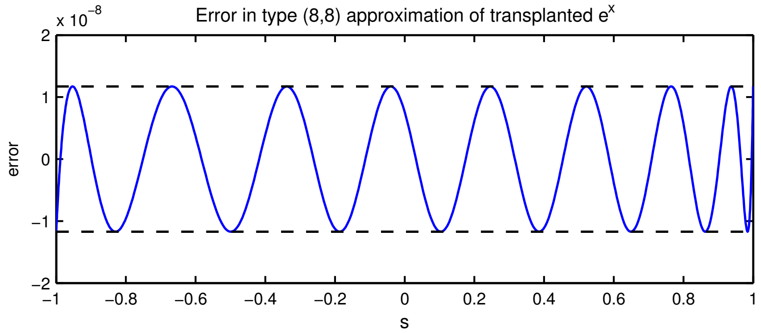


This gives us yet another way to compute rational approximations to e^x on $(-\infty,0]$: truncate this Chebyshev series in s, then transplant by (25.19) to get rational functions in x.

Alternatively, we can get true best approximations from (25.19) by applying the Chebfun `remez` command. Here for example is the error for the best approximation of type $(8,8)$ plotted in the s variable, showing 18 points of equioscillation.

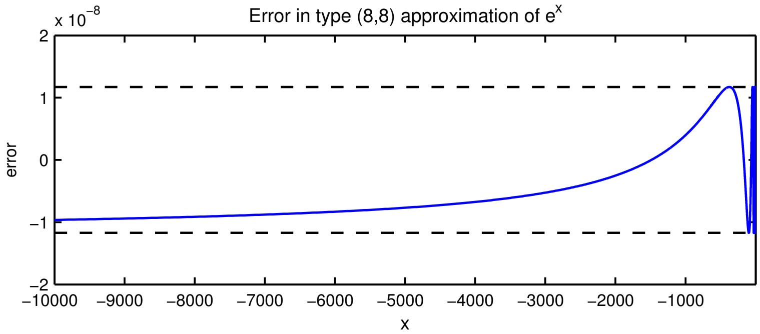
```
[P,Q,RH,err] = remez(F,8,8); R = P./Q;
hold off, plot(F-R), hold on
plot([-1 1],err*[1 1],'--k'), plot([-1 1],-err*[1 1],'--k')
xlabel('s',FS,9), ylabel error, ylim(2e-8*[-1,1])
title(['Error in type (8,8) approximation'...
       ' of transplanted e^x'],FS,9)
```

```
Warning:  This command is deprecated. Use minimax instead.
```



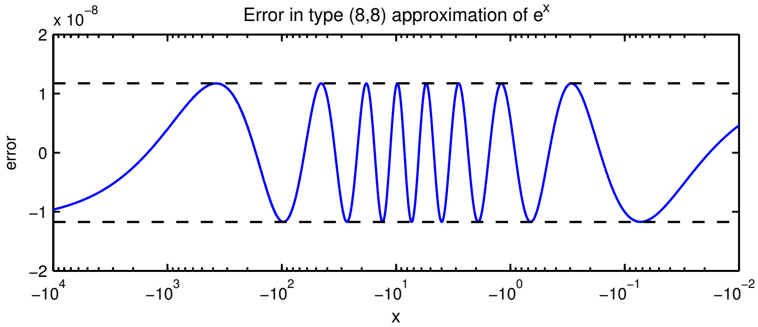
If we plot the same curve in the x variable, it's hard to see much because of the varying scale:

```
s1 = -.999; s2 = .999; s = chebfun('s',[s1 s2]); x = 9*(s-1)./(s+1);
hold off, plot(x,F{s1,s2}-R{s1,s2}), hold on
xx = [-1e4 -1e-2];
plot(xx,err*[1,1],'--k'), plot(xx,-err*[1,1],'--k'), xlim(xx)
xlabel('x',FS,9), ylabel error, ylim(2e-8*[-1,1])
title('Error in type (8,8) approximation of e^x',FS,9)
```



Putting the x axis on a log scale, however, makes the plot informative again:

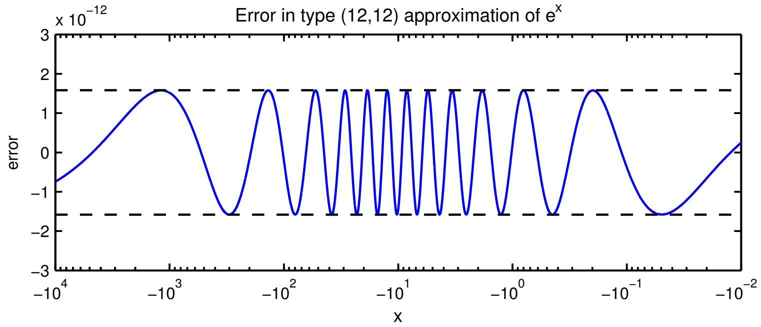
```
hold off, semilogx(x,F{s1,s2}-R{s1,s2}), hold on
semilogx(xx,err*[1,1],'--k'), plot(xx,-err*[1,1],'--k'), xlim(xx)
xlabel('x',FS,9), ylabel error, ylim(2e-8*[-1,1])
title('Error in type (8,8) approximation of e^x',FS,9)
```



Here is the analogous plot for type $(12, 12)$ approximation:

```
[P,Q,RH,err] = remez(F,12,12); R = P./Q;
hold off, semilogx(x,F{s1,s2}-R{s1,s2}), hold on
plot(xx,err*[1,1],'--k'), plot(xx,-err*[1,1],'--k'), xlim(xx)
xlabel('x',FS,9), ylabel error, ylim(3e-12*[-1,1])
title('Error in type (12,12) approximation of e^x',FS,9)
```

```
Warning:  This command is deprecated. Use minimax instead.
```



These plots are modeled after [Trefethen, Weideman & Schmelzer 2006], where it is shown that Carathéodory–Fejér approximation is equally effective and even faster than the Remes algorithm at computing these approximations.

Summary of Chapter 25. *Two problems involving rational functions have attracted special attention, highlighting the power of rational approximations near singularities and on unbounded domains. For approximating $|x|$ on $[-1,1]$, best rational functions converge root-exponentially whereas polynomials converge linearly. For approximating e^x on $(-\infty,0]$, best rational functions converge geometrically whereas polynomials do not converge at all. Both rates of approximation can be achieved by constructing partial fractions from trapezoid rule approximations to certain integrals.*

Exercise 25.1. Newton iteration for $|x|$. (This problem has roots in [Roberts 1980].) (a) Let x be a number, and suppose we want to solve the equation $r^2 = x^2$ for the unknown r using Newton iteration. Show that the iteration formula is $r^{(k+1)} = ((r^{(k)})^2 + x^2)/2r^{(k)}$. (b) If the initial guess is $r^{(0)} = 1$, then for $k \ge 1$, what is the smallest n for which the rational function $r^{(k)}(x)$ is of type (n,n)? (c) Use Chebfun to compute and plot the approximations $r^{(0)}(x), ..., r^{(5)}(x)$ on the interval $[-1,1]$. What is the sup-norm error $\||x| - r^{(k)}(x)\|$, and where is it attained? (d) What rate of convergence does this correspond to for $\||x| - r^{(k)}(x)\|$ as a function of n? How does this compare with the optimal rate given by Theorem 25.1? (e) Make a semilog plot of $|\,|x| - r^{(5)}(x)|$ as a function of $x \in [-1,1]$ and comment further on the nature of these rational approximations.

Exercise 25.2. An elementary approximant for e^x on $(-\infty,0]$. A degree n polynomial $p(s)$ on $[-1,1]$ can be transplanted to a type (n,n) rational function

$r(x)$ on $(-\infty, 0]$ by the map (25.19). Combine this observation with Theorem 8.2 to show that type (n, n) approximants to e^x on $(-\infty, 0]$ exist with accuracy $O(\exp(-Cn^{-2/3}))$ for some $C > 0$ as $n \to \infty$.

Exercise 25.3. Computing Halphen's constant. Write a short Chebfun program that computes Halphen's constant to 10 or more digits based on the condition (25.12).

Exercise 25.4. Best approximation errors for e^x. (a) Using `remez` and the change of variables (25.20), compute best approximation errors in type (n, n) approximation of e^x on $(-\infty, 0]$ for $n = 0, 1, \ldots, 13$. Plot the results on a log scale and compare them with estimates from the asymptotic formula (25.18). Also on a log scale, plot the difference between the estimates and the true errors, and comment on the results. (b) Repeat the computation with CF instead of `remez`. This time, plot the different between the CF and true errors on a log scale, and comment on the results.

Exercise 25.5. Behavior of approximants of $|x|$ in the complex plane. It is shown in [Blatt, Iserles & Saff 1988] that the type (n, n) best approximants to $|x|$ on $[-1, 1]$ have all their zeros and poles on the imaginary axis and converge to x for $\mathrm{Re}(x) > 0$ and to $-x$ for $\mathrm{Re}(x) < 0$ as $n \to \infty$. Verify this result numerically by plotting $|x - r^*_{nn}(x)|$ against $\mathrm{Re}(x)$ for $x \in [-1 + 0.5i, 1 + 0.5i]$ for $n = 1, 2, 3, 4$.

Exercise 25.6. Behavior of approximants of e^x in the complex plane. It is stated in [Stahl & Schmelzer 2009] that the poles of best type (n, n) approximations to e^x on $(-\infty, 0]$ move off to ∞ as $n \to \infty$, and the convergence at nth-root rate governed by $h \approx 1/9.28903$ applies on any compact set in the complex plane. With this result in mind, produce contour plots in the complex z-plane for the errors $|e^z - r_{nn}(z)|$ for the approximations (25.14)–(25.15) with $n = 2, 4, 6, 8, 10$. Does it appear likely that these approximations too converge on all compact sets in the plane?

Chapter 26

Rational interpolation and linearized least-squares

`ATAPformats`

For polynomials, we have emphasized that although best approximations with their equioscillating error curves are fascinating, Chebyshev interpolants or projections are just as good for most applications and simpler to compute since the problem is linear. To some degree at least, the same is true of rational functions. Best rational approximations are fascinating, but for practical purposes, it is often a better idea to use rational interpolants, and again an important part of the problem is linear since one can multiply through by the denominator.

But there is a big difference. Rational interpolation problems are not entirely linear, and unlike polynomial interpolation problems, they suffer from both nonexistence and discontinuous dependence on data in some settings. To use rational interpolants effectively, one must formulate the problem in a way that minimizes such effects. The method we shall recommend for this, here and in the next two chapters, makes use of the singular value decomposition (SVD) and the generalization of the linearized interpolation problem to one of least-squares fitting. This approach originates in [Pachón, Gonnet & Van Deun 2012] and [Gonnet, Pachón & Trefethen 2011]. The literature of rational interpolation goes back to Cauchy [1821] and Jacobi [1846], but much of it is rather far from computational practice.

Here is an example to illustrate the difficulties. Suppose we seek a rational function $r \in \mathcal{R}_{11}$ satisfying the conditions

$$r(-1) = 2, \quad r(0) = 1, \quad r(1) = 2. \tag{26.1}$$

Since a function in $\mathcal{R}_{11}$ is determined by three parameters, the count appears right for this problem to be solvable. In fact, however, there is no solution, and one can prove this by showing that if a function in $\mathcal{R}_{11}$ takes equal values at two points, it must be a constant (Exercise 26.1). We conclude: solutions to seemingly sensible rational interpolation problems do not always exist.

Let us modify the problem and seek a function $r \in \mathcal{R}_{11}$ satisfying the conditions

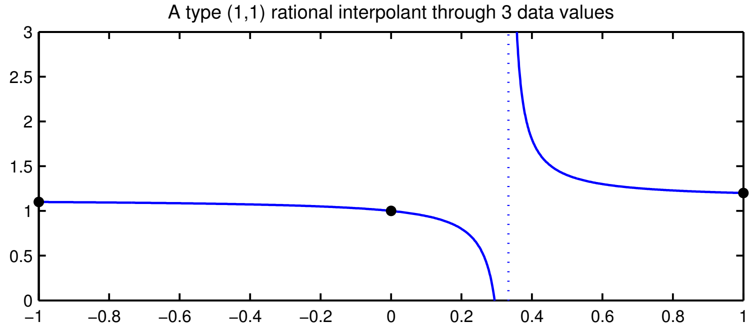
$$r(-1) = 1 + \varepsilon, \quad r(0) = 1, \quad r(1) = 1 + 2\varepsilon, \tag{26.2}$$

where ε is a parameter. Now there is a solution for any ε, namely

$$r(z) = 1 + \frac{\frac{4}{3}\varepsilon x}{x - \frac{1}{3}}. \tag{26.3}$$

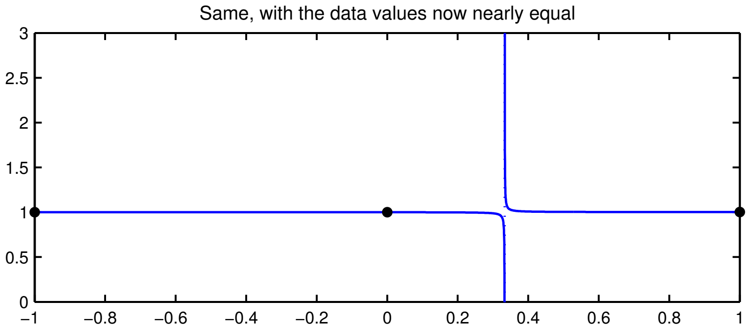
However, this is not quite the smooth interpolant one might have hoped for. Here is the picture for $\varepsilon = 0.1$:

```
x = chebfun('x'); r = @(ep) 1 + (4/3)*ep*x./(x-(1/3));
ep = 0.1; hold off, plot(r(ep)), ylim([0 3])
hold on, plot([-1 0 1],[1+ep 1 1+2*ep],'.k')
FS = 'fontsize';
title('A type (1,1) rational interpolant through 3 data values',FS,9)
```



And here it is for $\varepsilon = 0.001$:

```
ep = 0.001; hold off, plot(r(ep)), ylim([0 3])
hold on, plot([-1 0 1],[1+ep 1 1+2*ep],'.k')
title('Same, with the data values now nearly equal',FS,9)
```



Looking back at the formula (26.3), we see that for any nonzero value of ε, this function has a pole at $x = 1/3$. When ε is small, the effect of the pole is quite localized, and we may confirm this by calculating that the residue is $(4/3)\varepsilon$. Another way to interpret the local effect of the pole is to note that r has a zero at a distance just $O(\varepsilon)$ from the pole:

$$\text{pole: } x = \tfrac{1}{3}, \qquad \text{zero: } x = \tfrac{1}{3}/(1 - \tfrac{4}{3}\varepsilon).$$

For $|x - \frac{1}{3}| \gg \varepsilon$, the pole and the zero will effectively cancel. This example shows that even when a rational interpolation problem has a unique solution, the problem may be ill-posed in the sense that the solution depends discontinuously on the data. For $\varepsilon = 0$, (26.3) reduces to the constant $r = 1$, whereas for any nonzero ε there is a pole, though it seems to have little to do with approximating the data. Such poles are often called *spurious poles*. Since a spurious pole is typically associated with a nearby zero that approximately cancels its effect further away, another term is *Froissart doublet*, named after the physicist Marcel Froissart [Froissart 1969]. We may also say that the function has a *spurious pole-zero pair*.

Here is an example somewhat closer to practical approximation. Consider the function $f(x) = \cos(e^x)$,

```
f = cos(exp(x));
```

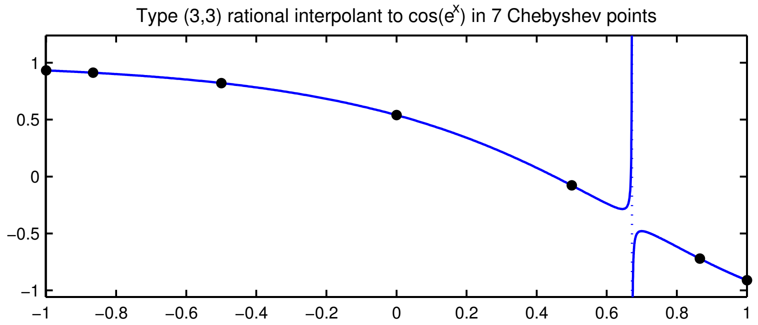
and suppose we want to construct rational interpolants of type (n, n) to f based on samples at $2n + 1$ Chebyshev points in $[-1, 1]$. Chebfun has a command `ratinterp` that will do this, and here is a table of the maximum errors obtained by `ratinterp` for $n = 1, 2, \ldots, 6$:

```
disp('    (n,n)        Error ')
for n = 1:6
  [p,q] = ratinterp(f,n,n);
  err = norm(f-p./q,inf);
  fprintf('    (%1d,%1d)     %7.2e\n',n,n,err)
end
```

```
    (n,n)       Error
    (1,1)     2.46e-01
    (2,2)     7.32e-03
    (3,3)         Inf
    (4,4)     6.11e-06
    (5,5)     4.16e-07
    (6,6)     6.19e-09
```

We seem to have very fast convergence, but what has gone wrong with the type $(3,3)$ approximant? A plot reveals that the problem is a spurious pole:

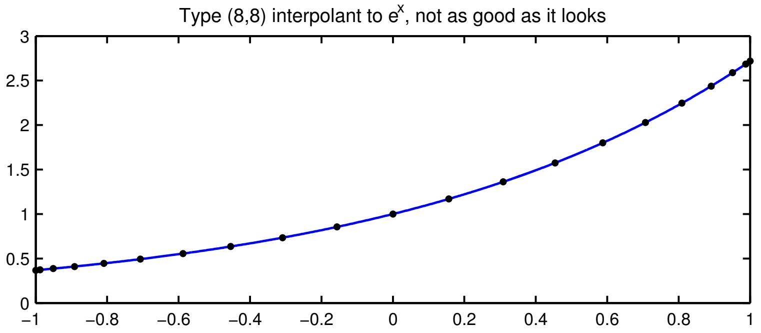
```
[p,q] = ratinterp(f,3,3);
hold off, plot(p./q), hold on
xx = chebpts(7); plot(xx,f(xx),'.k')
title(['Type (3,3) rational interpolant ' ...
       'to cos(e^x) in 7 Chebyshev points'],FS,9)
xlim([-1.001,1])
```



One might suspect that this artifact has something to do with rounding errors on the computer, but it is not so. The spurious pole is in the mathematics, with residue equal to about -0.0013.

In other examples, on the other hand, spurious poles do indeed arise from rounding errors. In fact, they appear very commonly when one aims for approximations with accuracy close to machine precision. Here, for example, is what happens when `ratinterp` is called upon to compute the interpolant of type $(8,8)$ of e^x in 17 Chebyshev points:

```
[p,q] = ratinterp(exp(x),8,8,[],[],0);
hold off, plot(p./q), hold on
xx = chebpts(21); plot(xx,exp(xx),'.k','markersize',10)
title(['Type (8,8) interpolant to e^x, ' ...
       'not as good as it looks'],FS,9)
```



The picture looks fine, but that is only because Chebfun has failed to detect that p/q has a spurious pole-zero pair:

```
spurious_zero = roots(p)
spurious_pole = roots(q)
```

```
spurious_zero =
   0.354105165744785
spurious_pole =
   0.354105165744784
```

One could attempt to get around this particular pathology by computing in higher precision arithmetic. However, quite apart from the practical difficulties of high-precision arithmetic, that approach would not really address the challenges of rational interpolation at a deeper level. The better response is to adjust the formulation of the rational interpolation problem so as to make it more robust. In this last example, it seems clear that a good algorithm should be sensible enough to reduce the number of computed poles. We now show how this can be done systematically with the SVD.

At this point, we shall change settings. Logically, we would now proceed to develop a robust rational interpolation strategy on $[-1, 1]$. However, that route would require us to combine new ideas related to robustness with the complexities of Chebyshev points, Chebyshev polynomials, and rational barycentric interpolation formulas. Instead, now and for most of the rest of the book, we shall move from the real interval $[-1,1]$ to the unit disk and switch variable names from x to z. This will make the presentation simpler, and it fits with the fact that many applications of rational interpolants and approximants involve complex variables.

Specifically, here is the problem addressed in the remainder of this chapter, following [Pachón, Gonnet & Van Deun 2012] and [Gonnet, Pachón & Trefethen 2011] but with roots as far back as Jacobi [1846]. Suppose f is a function defined on the unit circle in the complex plane and we consider its values $f(z_j)$ at the $(N+1)$st roots of unity for some $N \geq 0$,

$$z_j = e^{2\pi ij/(N+1)}, \quad 0 \leq j \leq N.$$

Using this information, how can we construct a good approximation $r \in \mathcal{R}_{mn}$? We assume for the moment that m, n and N are related by $N = m+n$. The parameter count is then right for an interpolant $r = p/q$ satisfying

$$\frac{p(z_j)}{q(z_j)} = f(z_j), \quad 0 \leq j \leq N. \tag{26.4}$$

The problem of finding such a function r is known as the *Cauchy interpolation problem.* As we have seen, however, a solution does not always exist.

Our first step towards greater robustness will be to linearize the problem and seek polynomials $p \in \mathcal{P}_m$ and $q \in \mathcal{P}_n$ such that

$$p(z_j) = f(z_j)q(z_j), \quad 0 \leq j \leq N. \tag{26.5}$$

By itself, this set of equations isn't very useful, because it has the trivial solution $p = q = 0$. Some kind of normalization is needed, and for this we introduce the representations

$$p(z) = \sum_{k=0}^{m} a_k z^k, \qquad q(z) = \sum_{k=0}^{n} b_k z^k$$

with

$$\mathbf{a} = (a_0, \dots, a_m)^T, \qquad \mathbf{b} = (b_0, \dots, b_n)^T.$$

Our normalization will be the condition

$$\|\mathbf{b}\| = 1, \tag{26.6}$$

where $\|\cdot\|$ is the standard 2-norm on vectors,

$$\|\mathbf{b}\| = \left(\sum_{k=0}^{n} |b_k|^2\right)^{1/2},$$

and similarly for vectors of dimensions other than $n+1$. Our linearized rational interpolation problem consists of solving the two equations (26.5)–(26.6). A solution with $q(z_j) \neq 0$ for all j will also satisfy (26.4), but if $q(z_j) = 0$ for some j, then (26.4) may or may not be satisfied. A point where it is not attained is called an *unattainable point.*

We turn (25.5)–(25.6) into a matrix problem as follows. Given an arbitrary $(n+1)$-vector $\mathbf{b}$, there is a corresponding polynomial $q \in \mathcal{P}_n$, which we may evaluate at the $(N+1)$st roots of unity $\{z_j\}$. Multiplying by the values $f(z_j)$ gives a set of $N+1$ numbers $f(z_j)q(z_j)$. There is a unique polynomial $\hat{p} \in \mathcal{P}_N$ that interpolates these data,

$$\hat{p}(z_j) = f(z_j)q(z_j), \quad 0 \leq j \leq N.$$

Let $\hat{p}$ be written as

$$\hat{p}(z) = \sum_{k=0}^{N} \hat{a}_k z^k, \quad \hat{\mathbf{a}} = (\hat{a}_0, \ldots, \hat{a}_N)^T.$$

Then $\hat{\mathbf{a}}$ is a linear function of $\mathbf{b}$, and we may accordingly express it as the product

$$\hat{\mathbf{a}} = \hat{C}\mathbf{b},$$

where $\hat{C}$ is a rectangular matrix of dimensions $(N+1) \times (n+1)$ depending on f. It can be shown that $\hat{C}$ is a Toeplitz matrix with entries given by the discrete Laurent or Fourier coefficients

$$c_{jk} = \frac{1}{N+1} \sum_{\ell=0}^{N} z_\ell^{k-j} f(z_\ell). \tag{26.7}$$

And now we can solve (26.5)–(26.6). Let $\tilde{C}$ be the $n \times (n+1)$ matrix consisting of the last n rows of $\hat{C}$. Since $\tilde{C}$ has more columns than rows, it has a nontrivial null vector, and for $\mathbf{b}$ we take any such null vector normalized to length 1:

$$\tilde{C}\mathbf{b} = 0, \quad \|\mathbf{b}\| = 1. \tag{26.8}$$

The corresponding vector $\hat{\mathbf{a}} = \hat{C}\mathbf{b}$ is equal to zero in positions $m+1$ through N, and we take $\mathbf{a}$ to be the remaining, initial portion of $\hat{\mathbf{a}}$: $a_j = \hat{a}_j$, $0 \leq j \leq m$. In matrix form we can write this as

$$\mathbf{a} = C\mathbf{b}, \tag{26.9}$$

where C is the $(m+1) \times (n+1)$ matrix consisting of the first $m+1$ rows of $\hat{C}$. Equations (26.8)–(26.9) constitute a solution to (26.5)–(26.6).

In a numerical implementation of the algorithm just described, the operations should properly be combined into a Matlab function, and a function like this called `ratdisk` is presented in [Gonnet, Pachón & Trefethen 2011]. Here, however, for the sake of in-line presentation, we shall achieve the necessary effect with a string of anonymous functions.

The first step is to construct the Toeplitz matrix $\hat{C}$ using the Matlab `fft` command. The `real` command below eliminates imaginary parts at the level of rounding errors, and would need to be removed for a function f that was not real on the real axis.

```
fj = @(f,N) f(exp(2i*pi*(0:N)'/(N+1)));
extract = @(A,I,J) A(I,J);
column = @(f,N) real(fft(fj(f,N)))/(N+1);
row = @(f,n,N) extract(column(f,N),[1 N+1:-1:N+2-n],1);
Chat = @(f,n,N) toeplitz(column(f,N),row(f,n,N));
```

Next we extract the submatrices $\tilde{C}$ and C:

```
Ctilde = @(f,m,n,N) extract(Chat(f,n,N),m+2:N+1,:);
C = @(f,m,n,N) extract(Chat(f,n,N),1:m+1,:);
```

Finally we compute the vector $\mathbf{b}$ using the Matlab `null` command, which makes use of the SVD, and multiply by C to get $\mathbf{a}$:

```
q = @(f,m,n,N) null(Ctilde(f,m,n,N));
p = @(f,m,n,N) C(f,m,n,N)*q(f,m,n,N);
```

For example, here are the coefficients of the type $(2,2)$ interpolant to e^z in the 5th roots of unity:

```
f = @(z) exp(z); m = 2; n = 2; N = m+n;
pp = p(f,m,n,N), qq = q(f,m,n,N)
```

```
pp =
  -0.893131422200046
  -0.446418130422149
  -0.074390723603151
qq =
  -0.891891822763679
   0.446093473426966
  -0.074361209330862
```

The zeros lie in the left half-plane and the poles in the right half-plane:

```
rzeros = roots(flipud(pp))
rpoles = roots(flipud(qq))
```

```
rzeros =
 -3.000495954331878 + 1.732909565613550i
 -3.000495954331878 - 1.732909565613550i
rpoles =
  2.999503890813022 + 1.731191260767684i
  2.999503890813022 - 1.731191260767684i
```

Here are the values of the interpolant at $z = 0$ and $z = 2$, which one can see are not too far from e^0 and e^2:

```
r = @(z) polyval(flipud(pp),z)./polyval(flipud(qq),z);
approximation = r([0 2])
exact = exp([0 2])
```

```
approximation =
   1.001389854021227   7.011719966971134
exact =
   1.000000000000000   7.389056098930650
```

Now let us take stock. We have derived an algorithm for computing rational interpolants based on the linearized formula (26.5), but we have not yet dealt with spurious poles. Indeed, the solution developed so far has neither uniqueness nor continuous dependence on data. It is time to take our second step toward greater robustness, again relying on the SVD.

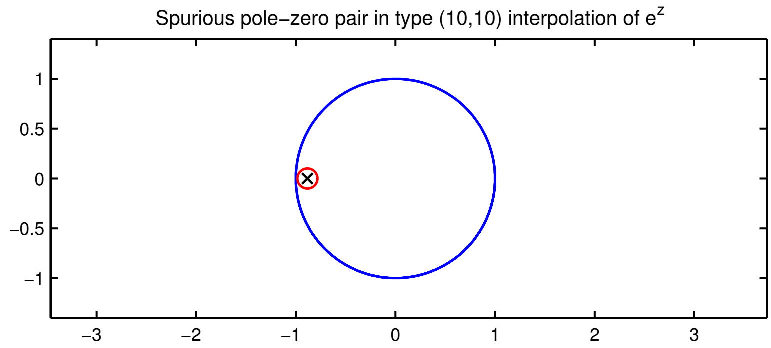
An example will illustrate what needs to be done. Suppose that instead of a type $(2,2)$ interpolant to e^z in 5 points, we want a type $(8,8)$ interpolant in 17 points. (This is like the type $(8,8)$ interpolant computed earlier, but now in roots of unity rather than Chebyshev points.) Here is what we find:

```
m = 8; n = 8; N = m+n;
format short
pp = p(f,m,n,N)
qq = q(f,m,n,N)
```

```
pp =
   -0.7863   -0.6055
   -1.2853    0.1338
   -0.5391    0.1482
   -0.1167    0.0406
   -0.0157    0.0061
   -0.0014    0.0006
   -0.0001    0.0000
   -0.0000    0.0000
   -0.0000    0.0000
qq =
   -0.7863   -0.6055
   -0.4989    0.7393
    0.3530   -0.2884
   -0.0892    0.0602
    0.0129   -0.0079
   -0.0012    0.0007
    0.0001   -0.0000
   -0.0000    0.0000
    0.0000   -0.0000
```

Instead of the expected vectors **a** and **b**, we have matrices of dimension 9×2, and the reason is, $\tilde{C}$ has a nullspace of dimension 2. This would not be true in exact arithmetic, but it is true in 16-digit floating-point arithmetic. If we construct an interpolant from one of these vectors, it will have a spurious pole-zero pair. Here is an illustration, showing that the spurious pole (cross) and zero (circle) are near the unit circle, which is typical. The other seven non-spurious poles and zeros have moduli about ten times larger.

```
rpoles = roots(flipud(pp(:,1)));
rzeros = roots(flipud(qq(:,1)));
hold off, plot(exp(2i*pi*x))
ylim([-1.4 1.4]), axis equal, hold on
plot(rpoles,'xk','markersize',7)
plot(rzeros,'or','markersize',9)
title(['Spurious pole-zero pair in type ' ...
       '(10,10) interpolation of e^z'],FS,9)
```



Having identified the problem, we can fix it as follows. If $\tilde{C}$ has rank $n-d$ for some $d \geq 1$, then it has a nullspace of dimension $d+1$. (We intentionally use the same letter d as was used to denote the defect in Chapter 24.) There must exist a vector $\mathbf{b}$ in this nullspace whose final d entries are zero. We could do some linear algebra to construct this vector, but a simpler approach is to reduce m and n by d and N by $2d$ and compute the interpolant again. Here is a function for computing d with the help of the Matlab `rank` command, which is based on the SVD. The tolerance 10^{-12} ensures that contributions close to machine precision are discarded.

```
d = @(f,m,n,N) n-rank(Ctilde(f,m,n,N),1e-12);
```

We redefine `q` and `p` to use this information:

```
q = @(f,m,n,N,d) null(Ctilde(f,m-d,n-d,N-2*d));
p = @(f,m,n,N,d) C(f,m-d,n-d,N-2*d)*q(f,m,n,N,d);
```

Our example now gives vectors instead of matrices, with no spurious poles.

```
pp = p(f,m,n,N,d(f,m,n,N)); qq = q(f,m,n,N,d(f,m,n,N));
format long
disp('           pp                   qq'), disp([pp qq])
```

```
            pp                   qq
  -0.889761508241581  -0.889761508241427
  -0.444881276255395   0.444880231986110
  -0.101109523960421  -0.101109001825797
  -0.013481293295988   0.013481177243662
  -0.001123443568774  -0.001123429053829
  -0.000056172338559   0.000056171300329
  -0.000001337441819  -0.000001337407096
```

This type $(7,7)$ rational function approximates e^z to approximately machine precision in the unit disk. To verify this loosely, we write a function `error` that measures the maximum of $|f(z) - r(z)|$ over 1000 random points in the disk:

```
r = @(z) polyval(flipud(pp),z)./polyval(flipud(qq),z);
z = sqrt(rand(1000,1)).*exp(2i*pi*rand(1000,1));
error = @(f,r) norm(f(z)-r(z),inf);
error(f,r)
```

```
ans =
     8.204290648309170e-13
```

Mathematically, in exact arithmetic, the trick of reducing m and n by d restores uniqueness and continuous dependence on data, making the rational interpolation problem well-posed. On a computer, we do the same but rely on finite tolerances to remove contributions from singular values close to machine epsilon. A much more careful version of this algorithm can be found in the Matlab code `ratdisk` from [Gonnet, Pachón & Trefethen 2011], mentioned earlier.

We conclude this chapter by taking a third step towards robustness. So far, we have spoken only of interpolation, where the number of data values exactly matches the number of parameters in the fit. In some approximation problems, however, it may be better to have more data than parameters and perform a least-squares fit. This is one of those situations, and in particular, a least-squares formulation will reduce the likelihood of obtaining poles in the region near the unit circle where one is hoping for good approximation. This is why we have included the parameter N throughout the derivation of the last six pages. We will now consider the situation $N > m + n$. Typical choices for practical applications might be $N = 2(m + n)$ or $N = 4(m + n)$.

Given an $(n + 1)$-vector $\mathbf{b}$ and corresponding function q, we have already defined $\|\mathbf{b}\|$ as the usual 2-norm. For the function q, let us now define

$$\|q\|_N = (N+1)^{-1/2} \sum_{k=0}^{N} |q(z_j)|^2,$$

a weighted 2-norm of the values of $q(z)$ over the unit circle. So long as $N \geq n$, the two norms are equal:

$$\|q\|_N = \|\mathbf{b}\|.$$

The norm $\|\cdot\|_N$, however, applies to any function, not just a polynomial. In particular, our linearized least-squares rational approximation problem is this generalization of (26.5)–(26.6):

$$\|p - fq\|_N = \text{minimum}, \quad \|q\|_N = 1. \tag{26.10}$$

The algorithm we have derived for interpolation solves this problem too. What changes is that the matrix $\tilde{C}$, of dimension $(N-m)\times(n+1)$, may no longer have a null vector. If its singular values are $\sigma_1 \ge \cdots \ge \sigma_{n+1} \ge 0$, then the minimum error will be

$$\|p - fq\|_N = \sigma_{n+1},$$

which may be positive or zero. If $\sigma_n > \sigma_{n+1}$, $\mathbf{b}$ is obtained from the corresponding singular vector and that is all there is to it. If

$$\sigma_{n-d} > \sigma_{n-d+1} = \cdots = \sigma_{n+1}$$

for some $d \ge 1$, then the minimum singular space is of dimension $d+1$, and as before, we reduce m and n by d. The parameter N can be left unchanged, so f does not need to be evaluated at any new points.

For example, let f be the function $f(z) = \log(1.44 - z^2)$,

```
f = @(z) log(1.44-z.^2);
```

with branch points at ± 1.2, and suppose we want a type $(40,40)$ least-squares approximant with $N = 400$. The approximation delivered by the SVD algorithm comes out with exact type $(18,18)$:

```
m = 40; n = 40; N = 400;
pp = p(f,m,n,N,d(f,m,n,N)); qq = q(f,m,n,N,d(f,m,n,N));
mu = length(pp)-1; nu = length(qq)-1;
fprintf('    mu = %2d    nu = %2d\n',mu,nu)
```

```
    mu = 18    nu = 18
```

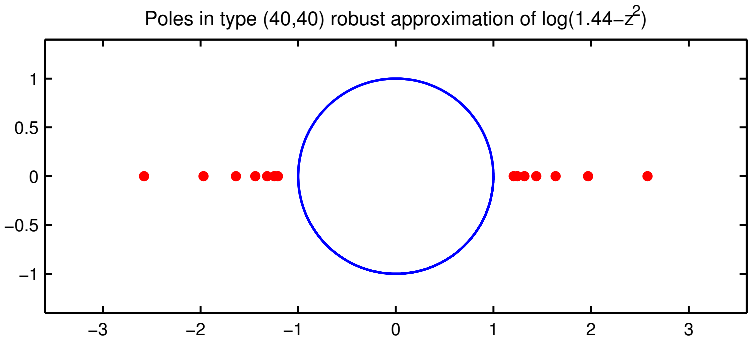
The accuracy in the unit disk is good (Exercise 26.4):

```
r = @(z) polyval(flipud(pp),z)./polyval(flipud(qq),z);
error(f,r)
```

```
ans =
     1.082379778434609e-11
```

Here are the poles:

```
rpoles = roots(flipud(qq));
hold off, plot(exp(2i*pi*x))
ylim([-1.4 1.4]), axis equal, hold on
plot(rpoles+1e-10i,'.r','markersize',14)
title(['Poles in type (40,40) robust ' ...
       'approximation of log(1.44-z^2)'],FS,9)
```

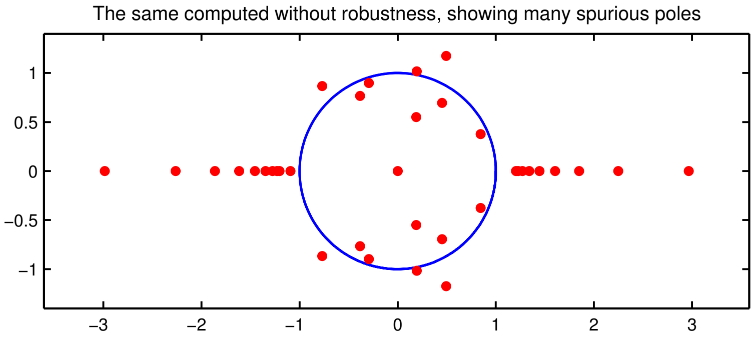


For comparison, suppose we revert to the original definitions of the anonymous functions p and q, with no removal of negligible singular values:

```
q = @(f,m,n,N) null(Ctilde(f,m,n,N));
p = @(f,m,n,N) C(f,m,n,N)*q(f,m,n,N);
```

Now the computation comes out with exact type $(40, 40)$, and half the poles are spurious:

```
m = 40; n = 40; N = 400;
pp = p(f,m,n,N); pp = pp(:,end);
qq = q(f,m,n,N); qq = qq(:,end);
rpoles = roots(flipud(qq));
hold off, plot(exp(2i*pi*x))
ylim([-1.4 1.4]), axis equal, hold on
plot(rpoles+1e-10i,'.r','markersize',14)
title(['The same computed without robustness, ' ...
       'showing many spurious poles'],FS,9)
```



The error looks excellent,

```
r = @(z) polyval(flipud(pp),z)./polyval(flipud(qq),z);
error(f,r)
```

```
ans =
     3.116648361300168e-14
```

but in fact it is not so good. Because of the spurious poles, the maximum error in the unit disk is actually infinite, but this has gone undetected at the 1000 random sample points used by the `error` command.

In closing this chapter we return for a moment to the variable x on the interval $[-1,1]$. Earlier we used the Chebfun command `ratinterp` to compute a type $(8,8)$ interpolant to e^x in Chebyshev points and found that it had a spurious pole-zero pair introduced by rounding errors. This computation was one of pure interpolation, with no SVD-related safeguards of the kind described in the last few pages. However, `ratinterp` is actually designed to incorporate SVD robustness by default. The earlier computation called `ratinterp(exp(x),8,8,[],[],0)` in order to force a certain SVD tolerance to be 0 instead of the default 10^{-14}. If we repeat the computation with the default robustness turned on, we find that an approximation of exact type $(8,4)$ is returned and it has no spurious pole and zero:

```
[p,q,rh,mu,nu] = ratinterp(exp(x),8,8);
mu, nu
spurious_zero = roots(p)
spurious_pole = roots(q)
```

```
mu =
     8
nu =
     4
spurious_zero =
   Empty matrix: 0-by-1
spurious_pole =
   Empty matrix: 0-by-1
```

Summary of Chapter 26. *Generically, there exists a unique type (m, n) rational interpolant through $m+n+1$ data points, but such interpolants do not always exist, depend discontinuously on the data, and exhibit spurious pole-zero pairs both in exact arithmetic and even more commonly in floating point. They can be computed by solving a matrix problem involving a Toeplitz matrix of discrete Fourier coefficients. Uniqueness, continuous dependence, and avoidance of spurious poles can be achieved by reducing m and n when the minimal singular value of this matrix is multiple. It may also be helpful to oversample and solve a least-squares problem.*

Exercise 26.1. Nonexistence of certain interpolants. Show that if a function in $\mathcal{R}_{11}$ takes equal values at two points, it must be a constant.

Exercise 26.2. An invalid argument. We saw that the type $(3,3)$ interpolant to $\cos(e^x)$ in 7 Chebyshev points has a pole near $x = 0.6$. What is the flaw in the following argument? (Spell it out carefully, don't just give a word or two.) The interpolant through these 7 data values can be regarded as a combination of cardinal functions, i.e., type $(3,3)$ rational interpolants through Kronecker delta functions supported at each of the data points. If the sum has a pole at x_0, then one of the cardinal interpolants must have a pole at x_0. So type $(3,3)$ rational interpolants to almost every set of data at these 7 points will have a pole at exactly the same place.

Exercise 26.3. Explicit example of degeneracy. Following the example (26.2)–(26.3), but now on the unit circle, let r be the type $(1,1)$ rational function satisfying $r(1) = 1$, $r(\omega) = 1 + i\varepsilon$, $r(\overline{\omega}) = 1 - i\varepsilon$, where ω is the cube root of 1 in the upper half-plane and $\varepsilon > 0$ is a parameter. (a) What is r? (b) What is the 2×3 matrix $\hat{C}$ of (26.7)? (c) How do the singular values of $\hat{C}$ behave as $\varepsilon \to 0$?

Exercise 26.4. Rational vs. polynomial approximation. The final computational example of this chapter considered type (n, n) rational approximation of $f(z) = \log(1.44 - z^2)$ with $n = 40$, which was reduced to $n = 18$ by the robust algorithm. For degree $2n$ polynomial approximation, one would expect accuracy of order $O(\rho^{-2n})$ where ρ is the radius of convergence of the Taylor series of f at $z = 0$. How large would n need to be for this figure to be comparable to the observed accuracy of 10^{-11}?

Exercise 26.5. Rational Gibbs phenomenon (from [Pachón 2010, Sec. 5.1]). We saw in Chapter 9 that if $f(x) = \operatorname{sign}(x)$ is interpolated by polynomials in Chebyshev points in $[-1, 1]$, the errors decay inverse-linearly with distance from the discontinuity. Use `ratinterp` to explore the analogous rates of decay for type $(m, 2)$ and $(m, 4)$ linearized rational interpolants to the same function, keeping m odd for simplicity. What do the decay rates appear to be?

Exercise 26.6. A function with two rows of poles. After Theorem 22.1 we considered as an example the function $f(x) = (2 + \cos(20x + 1))^{-1}$. (a) Call `ratinterp` with $(m, n) = (100, 20)$ to determine a rational approximation r to f on $[-1, 1]$ with up to 20 poles. How many poles does r in fact have, and what are they? (b) Determine analytically the full set of poles of f and produce a plot of the approximations from (a) together with the nearby poles of f. How accurate are these approximations?

Chapter 27

Padé approximation

`ATAPformats`

Suppose f is a function with a Taylor series

$$f(z) = c_0 + c_1 z + c_2 z^2 + \cdots \tag{27.1}$$

at $z = 0$.[1] Whether or not the series converges doesn't matter in this chapter (it is enough for f to be a *formal power series*). For any integer $m \geq 0$, the *degree* $\boldsymbol{m}$ *Taylor approximant* to f is the unique polynomial $p_m \in \mathcal{P}_m$ that matches the series as far as possible, which will be at least through degree m,

$$(f - p_m)(z) = O(z^{m+1}). \tag{27.2}$$

Padé approximation is the generalization of this idea to rational approximation. For any integers $m, n \geq 0$, $r \in \mathcal{R}_{mn}$ is the *type* (m,n) *Padé approximant* to f if their Taylor series at $z = 0$ agree as far as possible:

$$(f - r_{mn})(z) = O(z^{\text{maximum}}). \tag{27.3}$$

In these conditions the "big O" notation has its usual precise meaning. Equation (27.2) asserts, for example, that the first nonzero term in the Taylor series for $f - p_m$ is of order z^k for some $k \geq m+1$, but not necessarily $k = m+1$.

Padé approximation can be viewed as the special case of rational interpolation in which the interpolation points coalesce at a single point. Thus there is a close analogy between the mathematics of the last chapter and this one, though some significant differences too that spring from the fact that

[1] This chapter is adapted from Gonnet, Güttel and Trefethen [2012].

the powers $z^0, z^1, \dots$ are ordered whereas the roots of unity are all equal in status. We shall see that a key property is that r_{mn} exists and is unique. Generically, it matches f through term $m+n$,

$$(f - r_{mn})(z) = O(z^{m+n+1}), \tag{27.4}$$

but in some cases, the matching will be to lower or higher order.

For example, the type $(1,1)$ Padé approximant to e^z is $(1+\frac{1}{2}z)/(1-\frac{1}{2}z)$, as we can verify numerically with the Chebfun command `padeapprox`:

```
[r,a,b] = padeapprox(@exp,1,1);
fprintf('    Numerator coeffs: %19.15f %19.15f\n',a)
fprintf('  Denominator coeffs: %19.15f %19.15f\n',b)
```

```
    Numerator coeffs:   1.000000000000000   0.500000000000000
  Denominator coeffs:   1.000000000000000  -0.500000000000000
```

The algorithm used by `padeapprox` will be discussed in the second half of this chapter.

The early history of Padé approximation is hard to disentangle because every continued fraction can be regarded as a Padé approximant (Exercise 27.7), and continued fractions got a lot of attention in past centuries. For example, Gauss derived the idea of Gauss quadrature from a continued fraction that amounts to a Padé approximant to the function $\log((z+1)/(z-1))$ at the point $z = \infty$ [Gauss 1814, Takahasi & Mori 1971, Trefethen 2008]. Ideas related to Padé approximation have been credited to Anderson (1740), Lambert (1758) and Lagrange (1776), and contributions were certainly made by Cauchy [1826] and Jacobi [1846]. The study of Padé approximants began to come closer to the current form with the papers of Frobenius [1881] and Padé himself [1892], who was a student of Hermite and published many articles after his initial thesis on the subject. Throughout the early literature, and also in the more recent era, much of the discussion of Padé approximation is connected with continued fractions, determinants, and recurrence relations, but here we shall follow a more robust matrix formulation.

We begin with a theorem about existence, uniqueness, and characterization, analogous to Theorem 24.1 for rational best approximation on an interval. There, the key idea was to count points of equioscillation of the error function $f - r$. Here, we count how many initial terms of the Taylor series of $f - r$

are zero. The arguments are similar, and again, everything depends on the integer known as the defect. Recall that if $r \in \mathcal{R}_{mn}$ is of exact type (μ, ν) for some $\mu \le m$, $\nu \le n$, then the defect of r with respect to $\mathcal{R}_{mn}$ is $d = \min\{m-\mu, n-\nu\} \ge 0$, with $\mu = -\infty$ and $d = n$ in the special case $r = 0$.

Theorem 27.1: Characterization of Padé approximants. *For each $m, n \ge 0$, a function f has a unique Padé approximant $r_{mn} \in \mathcal{R}_{mn}$ as defined by the condition* (27.3)*, and a function $r \in \mathcal{R}_{mn}$ is equal to r_{mn} if and only if $(f-r)(z) = O(z^{m+n+1-d})$, where d is the defect of r in $\mathcal{R}_{mn}$.*

Proof. The first part of the argument is analogous to parts 2 and 4 of the proof of Theorem 24.1: we show that if r satisfies $(f-r)(z) = O(z^{m+n+1-d})$, then r is the unique type (m, n) Padé approximant to f as defined by the condition (27.3). Suppose then that $(f-r)(z) = O(z^{m+n+1-d})$ and that $(f-\tilde{r})(z) = O(z^{m+n+1-d})$ also for some possibly different function $\tilde{r} \in \mathcal{R}_{mn}$. Then $(r-\tilde{r})(z) = O(z^{m+n+1-d})$. However, $r-\tilde{r}$ is of type $(m+n-d, 2n-d)$, so it can only have $m+n-d$ zeros at $z = 0$ unless it is identically zero. This implies $\tilde{r} = r$.

The other half of the proof is to show that there exists a function r with $(f-r)(z) = O(z^{m+n+1-d})$. This part of the argument makes use of linear algebra and is given in the two paragraphs following (27.8). ▮

Let us consider some examples to illustrate the characterization of Theorem 27.1. First, a generic case, we noted above that the type (1,1) Padé approximant to e^z is $r_{11}(z) = (1+\frac{1}{2}z)/(1-\frac{1}{2}z)$. The defect of r_{11} in $\mathcal{R}_{11}$ is $d = 0$, and we have

$$r_{11}(z) - e^z = \tfrac{1}{12}z^3 + \tfrac{1}{12}z^4 + \cdots = O(z^3).$$

Since $m+n+1-d = 3$, this confirms that r_{11} is the Padé approximant.

On the other hand, if f is even or odd, we soon find ourselves in the non-generic case. Suppose we consider

$$f(z) = \cos(z) = 1 - \tfrac{1}{2}z^2 + \tfrac{1}{24}z^4 - \cdots$$

and the rational approximation

$$r(z) = 1 - \tfrac{1}{2}z^2$$

of exact type $(2, 0)$. This gives

$$(f-r)(z) = O(z^4), \quad \ne O(z^5).$$

By Theorem 27.1, this implies that r is the Padé approximation to f for four different choices of (m,n): $(2,0)$, $(3,0)$, $(2,1)$, and $(3,1)$. With $(m,n)=(2,0)$, for example, the defect is $d=0$ and we need $(f-r)(z)=O(z^{2+0+1-0})=O(z^3)$, and with $(m,n)=(3,1)$, the defect is $d=1$ and we need $(f-r)(z)=O(z^{3+1+1-1})=O(z^4)$. Both matching conditions are satisfied, so r is the Padé approximant of both types $(2,0)$ and $(3,1)$. Similarly it is also the Padé approximant of types $(3,0)$ and $(2,1)$, but for no other values of (m,n).

This example involving an even function suggests the general situation. In analogy to the Walsh table of Chapter 24, the *Padé table* of a function f consists of the set of its Padé approximants for various $m,n\ge 0$ laid out in an array, with m along the horizontal and n along the vertical:

$$\begin{pmatrix} r_{00} & r_{10} & r_{20} & \cdots \\ r_{01} & r_{11} & r_{21} & \cdots \\ r_{02} & r_{12} & r_{22} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

The idea of the Padé table was proposed by Padé [1892], who called it "a table of approximate rational fractions... analogous to the multiplication table, unbounded to the right and below." Like the Walsh table for real rational approximation on an interval, the Padé table breaks into square blocks of degenerate entries, again as a consequence of the equioscillation-type characterization [Trefethen 1987]:

Theorem 27.2. Square blocks in the Padé table. *The Padé table for any function f breaks into precisely square blocks containing identical entries. (If f is rational, one of these will be infinite in extent.) The only exception is that if an entry $r=0$ appears in the table, then it fills all of the columns to the left of some fixed index $m=m_0$.*

Proof. Essentially the same as the proof of Theorem 24.2. ▮

As in the case of best real approximation on an interval discussed in Chapter 24, square blocks and defects have a variety of consequences for Padé approximants. In particular, the *Padé approximation operator*, which maps Taylor series f to their Padé approximants r_{mn}, is continuous at f with respect a norm based on Taylor coefficients if and only if r_{mn} has defect $d=0$. Another related result is that best supremum-norm approximations on intervals $[-\varepsilon,\varepsilon]$ converge to the Padé approximant as $\varepsilon\to 0$ if $d=0$, but not, in general, if $d>0$. These results come from [Trefethen & Gutknecht 1985], with earlier partial results due to Walsh; Werner and Wuytak; and Chui, Shisha and Smith.

At this point we have come a good way into the theory of Padé approximation without doing any algebra. To finish the job, and to lead into algorithms, it is time to introduce vectors and matrices, closely analogous to those of the last chapter.

Given a function f with Taylor coefficients $\{c_j\}$, suppose we look for a representation of the Padé approximant r_{mn} as a quotient $r = p/q$ with $p \in \mathcal{P}_m$ and $q \in \mathcal{P}_n$. Equation (27.4) is nonlinear, but multiplying through by the denominator suggests the linear condition

$$p(z) = f(z)q(z) + O(z^{m+n+1}), \tag{27.5}$$

just as (26.4) led to (26.5). To express this equation in matrix form, suppose that p and q are represented by coefficient vectors $\mathbf{a}$ and $\mathbf{b}$:

$$\mathbf{a} = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{pmatrix}, \qquad \mathbf{b} = \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{pmatrix},$$

$$p(z) = \sum_{k=0}^{m} a_k z^k, \qquad q(z) = \sum_{k=0}^{n} b_k z^k.$$

Then (27.5) can be written as an equation involving a Toeplitz matrix of Taylor coefficients of f, that is, a matrix with constant entries along each diagonal. For $m \ge n$, the equation looks like this:

$$\left(\begin{array}{c} a_0 \\ a_1 \\ \vdots \\ a_n \\ \vdots \\ a_m \\ \hline a_{m+1} \\ \vdots \\ a_{m+n} \end{array}\right) = \left(\begin{array}{cccc} c_0 & & & \\ c_1 & c_0 & & \\ \vdots & \vdots & \ddots & \\ c_n & c_{n-1} & \cdots & c_0 \\ \vdots & \vdots & & \vdots \\ c_m & c_{m-1} & \cdots & c_{m-n} \\ \hline c_{m+1} & c_m & \cdots & c_{m+1-n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m+n} & c_{m+n-1} & \cdots & c_m \end{array}\right) \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{pmatrix} \tag{27.6}$$

coupled with the condition

$$a_{m+1} = \cdots = a_{m+n} = 0. \tag{27.7}$$

In other words, $\mathbf{b}$ must be a null vector of the $n \times (n+1)$ matrix displayed below the horizontal line. If $m < n$, the equation looks like this:

$$\left(\begin{array}{c} a_0 \\ a_1 \\ \vdots \\ a_m \\ \hline a_{m+1} \\ \vdots \\ a_n \\ \vdots \\ a_{m+n} \end{array}\right) = \left(\begin{array}{ccccccc} c_0 & & & & & & \\ c_1 & c_0 & & & & & \\ \vdots & \vdots & \ddots & & & & \\ c_m & c_{m-1} & \cdots & c_0 & & & \\ \hline c_{m+1} & c_m & \cdots & c_1 & c_0 & & \\ \vdots & \vdots & & & \ddots & \ddots & \\ c_n & c_{n-1} & & \ddots & & c_1 & c_0 \\ \vdots & \vdots & & & & & \vdots \\ c_{m+n} & c_{m+n-1} & & \cdots & & & c_m \end{array}\right) \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{pmatrix}.$$

For simplicity we shall use the label (27.6) to refer to both cases, writing the $n \times (n+1)$ matrix always as

$$C = \begin{pmatrix} c_{m+1} & c_m & \cdots & c_{m+1-n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m+n} & c_{m+n-1} & \cdots & c_m \end{pmatrix} \qquad (27.8)$$

with the convention that $c_k = 0$ for $k < 0$.

One solution to (27.6)–(27.7) would be $\mathbf{a} = \mathbf{0}$ and $\mathbf{b} = \mathbf{0}$, corresponding to the useless candidate $r = 0/0$. However, an $n \times (n+1)$ matrix always has a nonzero null vector,

$$C\mathbf{b} = \mathbf{0}, \quad \mathbf{b} \neq \mathbf{0},$$

and once $\mathbf{b}$ is chosen, the coefficients $a_0, \ldots, a_m$ of p can be obtained by multiplying out the matrix-vector product above the line. Thus there is always a solution to (27.5) with $q \neq 0$.

If $b_0 \neq 0$, then dividing (27.5) by q shows that p/q is a solution to (27.4). Some nonzero null vectors $\mathbf{b}$, however, may begin with one or more zero components. Suppose that $\mathbf{b}$ is a nonzero null vector with $b_0 = b_1 = \cdots = b_{\sigma-1} = 0$ and $b_\sigma \neq 0$ for some $\sigma \geq 1$. Then the corresponding vector $\mathbf{a}$ will also have $a_0 = a_1 = \cdots = a_{\sigma-1} = 0$ (and a_σ might be zero or nonzero). It follows from the Toeplitz structure of (27.6) that we can shift both $\mathbf{a}$ and $\mathbf{b}$ upward by σ positions to obtain new vectors $\tilde{\mathbf{a}} = (a_\sigma, \ldots, a_m, 0, \ldots, 0)^T$ and $\tilde{\mathbf{b}} = (b_\sigma, \ldots, b_n, 0, \ldots, 0)^T$ while preserving the quotient $r = \tilde{p}/\tilde{q} = p/q$. Thus r has defect $d \geq \sigma$, and equations (27.6)–(26.7) are still satisfied except that $\tilde{a}_{m+n-\sigma+1}, \ldots, \tilde{a}_{m+n}$ may no longer be zero, implying $(f-r)(z) =$

$O(z^{m+n+1-\sigma})$. Thus $(f-r)(z) = O(z^{m+n+1-d})$, and this completes the proof of Theorem 27.1.

We have just shown that any nonzero null vector of the matrix C of (27.8) gives a function r that satisfies the condition for a Padé approximation, hence must be the unique approximant provided by Theorem 27.1. So we have proved the following theorem.

Theorem 27.3. Linear algebra solution of Padé problem. *Given a function f with Taylor coefficients $\{c_j\}$, let $\mathbf{b}$ be any nonzero null vector of the matrix C of* (27.8)*, let $\mathbf{a}$ be the corresponding vector obtained from* (27.6)*, and let $p \in \mathcal{P}_m$ and $q \in \mathcal{P}_n$ be the corresponding polynomials. Then $r_{mn} = p/q$ is the unique type (m,n) Padé approximant to f.*

We emphasize that the vectors $\mathbf{a}$ and $\mathbf{b}$ are not unique in general: a function in $\mathcal{R}_{mn}$ may have many representations p/q. Nevertheless, all choices of $\mathbf{a}$ and $\mathbf{b}$ lead to the same r_{mn}.

From Theorems 27.1–27.3 one can derive a precise characterization of the algebraic structure of the Padé approximants to a function f, as follows. Let $\hat{r}$ be a rational function of exact type (μ,ν) that is the Padé approximant to f in a $(k+1)\times(k+1)$ square block for some $k \ge 0$:

$$\begin{pmatrix} r_{\mu\nu} & \cdots & r_{\mu+k,\nu} \\ \vdots & & \vdots \\ r_{\mu,\nu+k} & \cdots & r_{\mu+k,\nu+k} \end{pmatrix}.$$

Write $\hat{r} = \hat{p}/\hat{q}$ with $\hat{p}$ and $\hat{q}$ of exact degrees μ and ν. From Theorem 27.1 we know that the defect d must be distributed within the square block according to this pattern illustrated for $k=5$:

$$\text{defect } d\text{:} \quad \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 2 & 2 & 2 \\ 0 & 1 & 2 & 3 & 3 & 3 \\ 0 & 1 & 2 & 3 & 4 & 4 \\ 0 & 1 & 2 & 3 & 4 & 5 \end{pmatrix}. \qquad (27.9)$$

According to Theorem 27.3, the polynomials p and q that result from solving the matrix problem (27.6)–(27.7) must be related to $\hat{p}$ and $\hat{q}$ by

$$p(z) = \pi(z)\hat{p}(z), \quad q(z) = \pi(z)\hat{q}(z)$$

for some polynomial π of degree at most d. Now let us define the **deficiency** λ of r as the distance below the cross-diagonal in the square block, with the following pattern:

$$\text{deficiency } \lambda\text{:} \quad \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 2 & 3 & 4 \\ 0 & 1 & 2 & 3 & 4 & 5 \end{pmatrix}. \tag{27.10}$$

From Theorem 27.1, we know that in the positions of the block with $\lambda > 0$, $(f - r)(z) = O(z^{m+n+1-\lambda})$, $\neq O(z^{m+n+2-\lambda})$, for otherwise, the block would be bigger. For this to happen, $\pi(z)$ must be divisible by z^λ, so that at least λ powers of z are lost when solutions p and q from (27.6) are normalized to $\hat{p}$ and $\hat{q}$. Since π may have degree up to d, the number of degrees of freedom remaining in p and q is $d - \lambda$, an integer we denote by χ, distributed within the block according to this pattern:

$$\text{rank deficiency } \chi\text{:} \quad \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 2 & 2 & 1 & 0 \\ 0 & 1 & 2 & 2 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{27.11}$$

Thus the dimensionality of the space of vectors q is $\chi + 1$, and the same for p. We call χ the *rank deficiency* of r because of a fact of linear algebra: the rank of the $n \times (n+1)$ matrix C of (27.8) must be equal to $n - \chi$, so that its space of null vectors will have the required dimension $\chi + 1$. Some ideas related to these developments can be found in [Heinig & Rost 1984].

We have just outlined a proof of the following theorem, which can be found in Section 3 of [Gragg 1972].

Theorem 27.4. Structure of a Padé approximant. *Let f and $m, n \geq 0$ be given, let the type (m, n) Padé approximant r_{mn} of f have exact type (μ, ν), and let $\hat{p}$ and $\hat{q} \neq 0$ be polynomials of exact degrees μ and ν with $r_{mn} = \hat{p}/\hat{q}$. Let the defect d, deficiency λ, and rank deficiency $\chi = d - \lambda$ be defined as above. Then the matrix C of* (27.8) *has rank $n - \chi$, and two polynomials $p \in P_m$ and $q \in P_n$ satisfy* (27.5) *if and only if*

$$p(z) = \pi(z)\hat{p}(z), \quad q(z) = \pi(z)\hat{q}(z) \tag{27.12}$$

for some $\pi \in \mathcal{P}_d$ divisible by z^λ.

Although we did not state it in the last chapter, there is an analogue of this theorem for rational interpolation in distinct points, proved by Maehly and Witzgall [1960] and discussed also in [Gutknecht 1990] and [Pachón, Gonnet & Van Deun 2011].

With the results of the past few pages to guide us, we are now prepared to talk about algorithms.

At one level, the computation of Padé approximants is trivial, just a matter of implementing the linear algebra of (27.6)–(27.7). In particular, in the generic case, the matrix C of (27.8) will have full rank, and so will its $n \times n$ submatrix obtained by deleting the first column. One computational approach to Padé approximation is accordingly to normalize $\mathbf{b}$ by setting $b_0 = 1$ and then determine the rest of the entries of $\mathbf{b}$ by solving a system of equations involving this square matrix.

This approach will fail, however, when the square matrix is singular, and it is nonrobust with respect to rounding errors even when the matrix is nonsingular. To work with (27.8) robustly, it is a better idea to normalize by the condition

$$\|\mathbf{b}\| = 1,$$

where $\|\cdot\|$ is the vector 2-norm, as in equation (26.6) of the last chapter. We then again consider the SVD (singular value decomposition) of C, a factorization

$$C = U\Sigma V^*, \tag{27.13}$$

where U is $n\times n$ and unitary, V is $(n+1)\times(n+1)$ and unitary, and Σ is an $n\times(n+1)$ real diagonal matrix with diagonal entries $\sigma_1 \ge \sigma_2 \ge \cdots \ge \sigma_n \ge 0$.

Suppose $\sigma_n > 0$. Then C has rank n, and the final column of V provides a unique nonzero null vector $\mathbf{b}$ of C up to a scale factor. This null vector defines a polynomial $q \in \mathcal{P}_n$. Moreover, from (27.11), we know that (m,n) must lie on the outer boundary of its square block in the Padé table. If q is divisible by z^λ for some $\lambda \ge 1$, then (m,n) must lie in the bottom row or right column, and dividing p and q by z^λ brings it to the left column or top row, respectively. A final trimming of any trailing zeros in p or q brings them to the minimal forms $\hat{p}$ and $\hat{q}$ with exact degrees μ and ν.

On the other hand, suppose $\sigma_n = 0$, so that the number of zero singular values of C is $\chi \ge 1$. In this case (27.11) tells us that (m,n) must lie in the interior of its square block at a distance χ from the boundary. Both m and n can accordingly be reduced by χ and the process repeated with a new matrix and a new SVD, χ steps closer to the upper-left (μ, ν) corner. After a small number of such steps (never more than $2 + \log_2(d + 1)$, where d is the defect), convergence is guaranteed.

These observations suggest the following SVD-based algorithm, introduced in [Gonnet, Güttel & Trefethen 2012].

Algorithm 27.1. Pure Padé approximation in exact arithmetic

Input: $m \geq 0$, $n \geq 0$, *and a vector* $\mathbf{c}$ *of Taylor coefficients* $c_0, \dots, c_{m+n}$ *of a function* f.

Output: Polynomials $p(z) = a_0 + \cdots + a_\mu z^\mu$ *and* $q(z) = b_0 + \cdots + b_\nu z^\nu$, $b_0 = 1$, *of the minimal degree type* (m, n) *Padé approximation of* f.

1. *If* $c_0 = \cdots = c_m = 0$, *set* $p = 0$ *and* $q = 1$ *and stop.*
2. *If* $n = 0$, *set* $p(z) = c_0 + \cdots + c_m z^m$ *and* $q = 1$ *and go to Step* 8.
3. *Compute the SVD* (27.13) *of the* $n \times (n + 1)$ *matrix* C. *Let* $\rho \leq n$ *be the number of nonzero singular values.*
4. *If* $\rho < n$, *reduce* n *to* ρ *and* m *to* $m - (n - \rho)$ *and return to Step* 2.
5. *Get* q *from the null right singular vector* $\mathbf{b}$ *of* C *and then* p *from the upper part of* (27.6).
6. *If* $b_0 = \cdots = b_{\lambda-1} = 0$ *for some* $\lambda \geq 1$, *which implies also* $a_0 = \cdots = a_{\lambda-1} = 0$, *cancel the common factor of* z^λ *in* p *and* q.
7. *Divide* p *and* q *by* b_0 *to obtain a representation with* $b_0 = 1$.
8. *Remove trailing zero coefficients, if any, from* $p(z)$ *or* $q(z)$.

In exact arithmetic, this algorithm produces the unique Padé approximant r_{mn} in a minimal-degree representation of type (μ, ν) with $b_0 = 1$. The greatest importance of Algorithm 27.1, however, is that it generalizes readily to numerical computation with rounding errors, or with noisy Taylor coefficients $\{c_j\}$. All one needs to do is modify the tests for zero singular values or zero coefficients so as to incorporate a suitable tolerance, such as 10^{-14} for computations in standard 16-digit arithmetic. The following modified algorithm also comes from [Gonnet, Güttel & Trefethen 2012].

Algorithm 27.2. Robust Padé approximation for noisy data or floating point arithmetic

Input: $m \geq 0$, $n \geq 0$, *a vector* $\mathbf{c}$ *of Taylor coefficients* $c_0, \dots, c_{m+n}$ *of a function* f, *and a relative tolerance* `tol` ≥ 0.

Output: Polynomials $p(z) = a_0 + \cdots + a_\mu z^\mu$ *and* $q(z) = b_0 + \cdots + b_\nu z^\nu$, $b_0 = 1$, *of the minimal degree type* (m, n) *Padé approximation of a function close to*

f.

1. *Rescale $f(z)$ to $f(z/\gamma)$ for some $\gamma > 0$ if desired to get a function whose Taylor coefficients $c_0, \dots, c_{m+n}$ do not vary too widely.*
2. *Define $\tau = \texttt{tol} \cdot \|\mathbf{c}\|_2$. If $|c_0| = \cdots = |c_m| \le \tau$, set $p = 0$ and $q = 1$ and stop.*
3. *If $n = 0$, set $p(z) = c_0 + \cdots + c_m z^m$ and $q = 1$ and go to Step 7.*
4. *Compute the SVD (27.13) of the $n \times (n+1)$ matrix C. Let $\rho \le n$ be the number of singular values of C that are greater than τ.*
5. *If $\rho < n$, reduce n to ρ and m to $m - (n - \rho)$ and return to Step 3.*
6. *Get q from the null right singular vector $\mathbf{b}$ of C and then p from the upper part of (27.6).*
7. *If $|b_0|, \dots, |b_{\lambda-1}| \le \texttt{tol}$ for some $\lambda \ge 1$, zero the first λ coefficients of p and q and cancel the common factor z^λ.*
8. *If $|b_{n+1-\lambda}|, \dots, |b_n| \le \texttt{tol}$ for some $\lambda \ge 1$, remove the last λ coefficients of q. If $|a_{m+1-\lambda}|, \dots, |a_m| \le \tau$ for some $\lambda \ge 1$, remove the last λ coefficients of p.*
9. *Divide p and q by b_0 to obtain a representation with $b_0 = 1$.*
10. *Undo the scaling of Step 1 by redefining $\gamma^j a_j$ as a_j and $\gamma^j b_j$ as b_j for each j.*

Algorithm 27.2 has been implemented in a Matlab code called `padeapprox` that is included in the Chebfun distribution, though it does not involve chebfuns. In its basic usage, `padeapprox` takes as input a vector $\mathbf{c}$ of Taylor coefficients together with a specification of m and n, with $\texttt{tol} = 10^{-14}$ by default. For example, following [Gragg 1972], suppose

$$f(z) = \frac{1 - z + z^3}{1 - 2z + z^2} = 1 + z + z^2 + 2z^3 + 3z^4 + 4z^5 + \cdots.$$

Then the type $(2,5)$ Padé approximation of f comes out with the theoretically correct exact type $(0,3)$:

```
c = [1 1 (1:50)];
[r,a,b] = padeapprox(c,2,5);
format short
disp('Coefficients of numerator:'), disp(a.')
disp('Coefficients of denominator:'), disp(b.')
```

```
Coefficients of numerator:
    1.0000
Coefficients of denominator:
    1.0000   -1.0000    0.0000   -1.0000
```

To illustrate the vital role of the SVD in such a calculation, here is what happens if robustness is turned off by setting `tol` $= 0$:

```
[r,a,b] = padeapprox(c,2,5,0);
disp(’Coefficients of numerator:’), disp(a.’)
disp(’Coefficients of denominator:’), disp(b.’)
```

```
Coefficients of numerator:
   1.0e+15 *
    0.0000    0.0000    6.1337
Coefficients of denominator:
   1.0e+15 *
    0.0000   -0.0000    6.1337   -6.1337    0.0000   -6.1337
```

We now see longer vectors with enormous entries, on the order of the inverse of machine precision. The type appears to be $(2,5)$, but the zeros and poles reveal that this is spurious:

```
format long g
disp(’Zeros:’), disp(roots(a(end:-1:1)))
disp(’Poles:’), disp(roots(b(end:-1:1)))
```

```
Zeros:
      -7.27051934844095e-17 +   1.2768444696564e-08i
      -7.27051934844095e-17 -   1.2768444696564e-08i
Poles:
         -0.341163901914009 +      1.16154139999725i
         -0.341163901914009 -      1.16154139999725i
          0.682327803828019
      -7.27051934954803e-17 +   1.2768444696564e-08i
      -7.27051934954803e-17 -   1.2768444696564e-08i
```

We see that the two zeros are virtually cancelled by two poles that differ from them by only about 10^{-24}. Thus this approximant has two spurious pole-zero pairs, or Froissart doublets, introduced by rounding errors. Many

Padé computations over the years have been contaminated by such effects, and in an attempt to combat them, many authors have asserted that it is necessary to compute Padé approximations in high precision arithmetic.

If `padeapprox` is called with a Matlab function handle f rather than a vector as its first argument, then it assumes f is a function analytic in a neighborhood of the closed unit disk and computes Taylor coefficients by the Fast Fourier Transform. For example, here is the type $(2,2)$ Padé approximant of $f(z) = \cos(z)$:

```
format long
[r,a,b] = padeapprox(@cos,2,2);
disp('Coefficients of numerator:'), disp(a.')
disp('Coefficients of denominator:'), disp(b.')
```

```
Coefficients of numerator:
   1.000000000000000                   0  -0.416666666666667
Coefficients of denominator:
   1.000000000000000                   0   0.083333333333333
```

One appealing application of `padeapprox` is the numerical computation of block structure in the Padé table for a given function f. For example, here is a table of the computed pair (μ,ν) for each (m,n) in the upper-left portion of the Padé table of $\cos(z)$ with $0 \le m,n \le 8$. One sees the 2×2 block structure resulting from the evenness of $\cos(z)$.

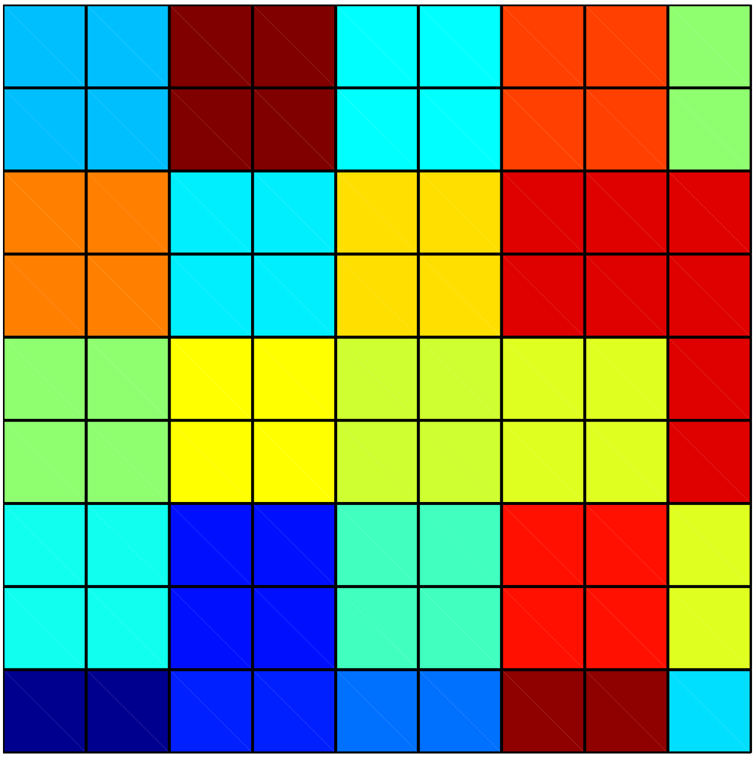
```
nmax = 8;
for n = 0:nmax
  for m = 0:nmax
    [r,a,b,mu,nu] = padeapprox(@cos,m,n); fprintf(' (%1d,%1d)',mu,nu)
  end
  fprintf('\n')
end
```

```
 (0,0) (0,0) (2,0) (2,0) (4,0) (4,0) (6,0) (6,0) (8,0)
 (0,0) (0,0) (2,0) (2,0) (4,0) (4,0) (6,0) (6,0) (8,0)
 (0,2) (0,2) (2,2) (2,2) (4,2) (4,2) (6,2) (6,2) (8,2)
 (0,2) (0,2) (2,2) (2,2) (4,2) (4,2) (6,2) (6,2) (8,2)
 (0,4) (0,4) (2,4) (2,4) (4,4) (4,4) (6,4) (6,4) (8,4)
 (0,4) (0,4) (2,4) (2,4) (4,4) (4,4) (6,4) (6,4) (8,4)
 (0,6) (0,6) (2,6) (2,6) (4,6) (4,6) (6,6) (6,6) (8,6)
 (0,6) (0,6) (2,6) (2,6) (4,6) (4,6) (6,6) (6,6) (8,6)
 (0,8) (0,8) (2,8) (2,8) (4,8) (4,8) (6,8) (6,8) (8,8)
```

We can also show the block structure with a color plot, like this:

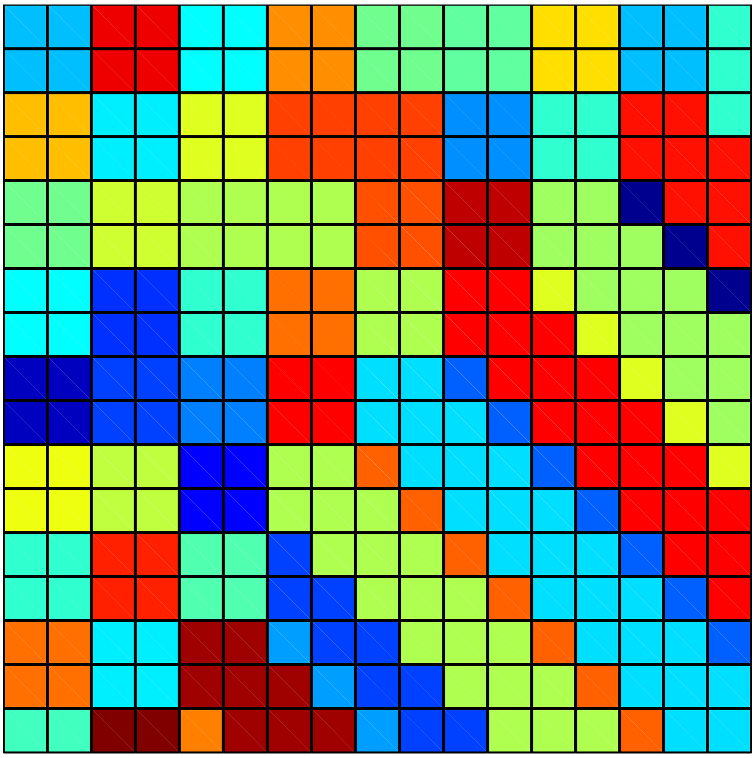
```
d = zeros(nmax+2);
rand('state',7); h = tan(2*rand(50)-1); h(8,1) = 1;
for n = 0:nmax, for m = 0:nmax
  [r,a,b,mu,nu] = padeapprox(@cos,m,n); d(n+1,m+1) = h(mu+1,nu+1);
end, end
pcolor(d), axis ij square off
```

```
Warning: Using 'state' to set RAND's internal state causes RAND, RANDI, and R
to use legacy random number generators.
```



The pattern of 2 × 2 blocks is broken if we compute a larger segment of the table, such as $0 \leq m, n \leq 16$:

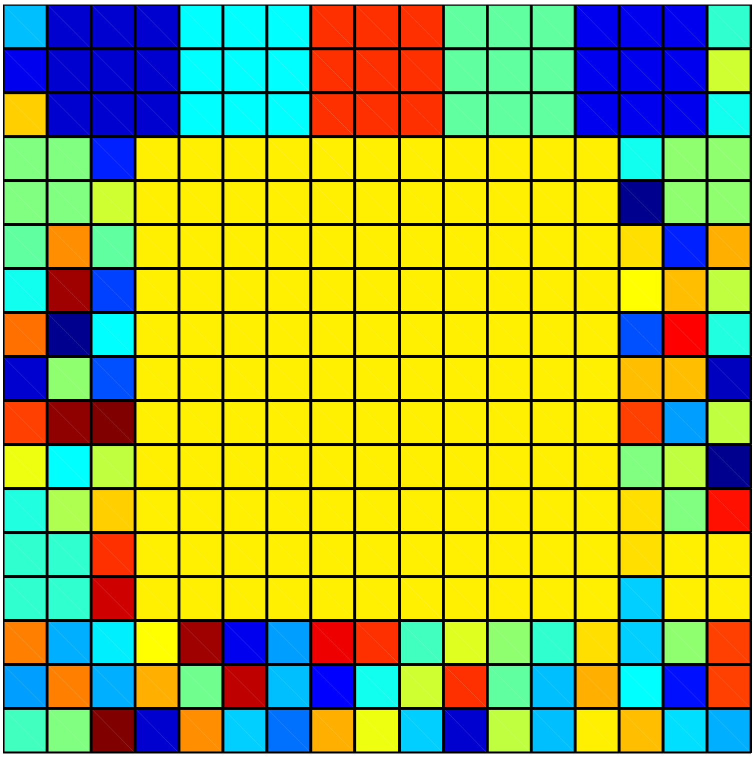
```
nmax = 16; d = zeros(nmax+2);
for n = 0:nmax, for m = 0:nmax
  [r,a,b,mu,nu] = padeapprox(@cos,m,n); d(n+1,m+1) = h(mu+1,nu+1);
end, end
pcolor(d), axis ij square off
```



What is going on here is that for $m+n$ greater than about 16, $\cos(z)$ is resolved to machine precision, and the diagonal stripes of the plot show that `padeapprox` has automatically cut m and n down to this level.

For an "arbitrary" function f with gaps in its Taylor series, the block structure can be quite intriguing, as illustrated by this example with $f(z) = 1+z+z^4+z^7+z^{10}+z^{13}+z^{16}+z^{17}$:

```
nmax = 16; d = zeros(nmax+2);
f = @(z) 1+z+z.^4+z.^7+z.^10+z.^13.+z.^16+z.^17;
for n = 0:nmax, for m = 0:nmax
  [r,a,b,mu,nu] = padeapprox(f,m,n); d(n+1,m+1) = h(mu+1,nu+1);
end, end
pcolor(d), axis ij square off
```



Apart from z^{17}, these are the initial terms of the Taylor series of

$$f(z) = \frac{1 + z - z^3}{1 - z^3}, \tag{27.14}$$

an example for which Padé worked out the block structure for $0 \le m \le 7$, $0 \le n \le 5$ [Padé 1892], showing vividly a 2×2 block, two 3×3 blocks, and the beginning of the infinite block at position $(3, 3)$.

In this chapter we have discussed how to compute Padé approximants, but not what they are useful for. As outlined in chapter 23, applications of these approximations typically involve situations where we know a function in one region of the z-plane and wish to evaluate it in another region that lies near or beyond certain singularities. The next chapter is devoted to a practical exploration of such problems.

From a theoretical perspective, a central question for more than a century has been, what sort of convergence of Padé approximants of a function f can we expect as m and/or n increase to ∞? In the simplest case, suppose that f is an entire function, that is, analytic for all z. Then for any compact set K in the complex plane, we know that the type $(m, 0)$ Padé approximants converge uniformly on K as $m \to \infty$, since these are just the Taylor approximants. One might hope that the same would be true of type (m, n_0) approximants for fixed $n_0 \geq 1$ as $m \to \infty$, or of type (n, n) approximants as $n \to \infty$, but in fact, pointwise convergence need not occur in either of these situations. The problem is that spurious pole-zero pairs, Froissart doublets, may appear at seemingly arbitrary locations in the plane. As m and/or n increase, the doublets get weaker and their effects more localized, but they can never be guaranteed to go away. (In fact, there exist functions f whose Padé approximants have so many spurious poles that the sequence of (n, n) approximants is unbounded for every $z \neq 0$ [Perron 1929, Wallin 1972].) The same applies if f is meromorphic, i.e., analytic apart from poles, or if it has more complicated singularities such as branch points. All this is in true in exact mathematics, and when there are rounding errors on a computer, the doublets become ubiquitous.

Despite these complexities, important theorems have been proved. The theorem of de Montessus de Ballore [1902] concerns the case of $m \to \infty$ with fixed n, guaranteeing convergence in a disk about $z = 0$ if f has exactly n poles there. The Nuttall–Pommerenke theorem [Nuttall 1970, Pommerenke 1973] concerns $m = n \to \infty$ and ensures convergence for meromorphic f not pointwise but *in measure* or *in capacity*, these being precise notions that require accuracy over most of a region as $m, n \to \infty$ while allowing for localized anomalies. This result was powerfully generalized for functions with branch points by Stahl [1987], who showed that as $n \to \infty$, almost all the poles of type (n, n) Padé approximants line up along branch cuts that have a property of minimal capacity in the z^{-1}-plane. For discussion of these results see [Baker & Graves-Morris 1996]. There are also analogous results for multipoint Padé approximation and other forms of rational interpolation. For example, an analogue of the de Montessus de Ballore theorem for interpolation as in the last chapter was proved by Saff [1972].

As a practical matter, these complexities of convergence are well combatted by the SVD approach we have described, which can be regarded as a method of regularization of the Padé problem.

For reasons explained in the last chapter, the whole discussion of this chapter has been based on the behavior of a function $f(z)$ at $z = 0$ rather than this book's usual context of a function $f(x)$ on an interval such as $[-1, 1]$. There is an analogue of Padé approximation for $[-1, 1]$ called *Chebyshev–Padé approximation*, developed by Hornecker [1959], Maehly [1963], Frankel and Gragg [1973], Clenshaw and Lord [1974], and Geddes

[1981]. The idea is to consider the analogue of (27.3) for Chebyshev series rather than Taylor series:

$$(f - r_{mn})(x) = O(T_{\text{maximum}}(x)). \tag{27.14}$$

(The Maehly version starts from the analogue of the linearized form (27.5).) In analogy to Theorem 27.1, it turns out that any $r \in \mathcal{R}_{mn}$ satisfying $(f - r)(x) = O(T_{m+n+1-d}(x))$ is the unique Chebyshev–Padé approximant according to this definition, but now, there is no guarantee that such a function r exists. For theoretical details, see [Trefethen & Gutknecht 1987], and for computations in Chebfun, there is a code called `chebpade`. As of today, there has not yet been a study of Chebyshev–Padé approximation employing the SVD-based robustness ideas described in this chapter for Padé approximation.

For extensive information about Padé approximation, see the book by Baker and Graves-Morris [1996]. However, that monograph uses an alternative definition according to which a Padé approximant only exists if equation (27.4) can be satisfied, and in fact the present treatment is mathematically closer to the landmark review of Gragg [1972], which uses the definition (27.3).

Summary of Chapter 27. *Padé approximation is the generalization of Taylor polynomials to rational approximation, that is, rational interpolation at a single point. Padé approximants are characterized by a kind of equioscillation condition and can be computed robustly by an algorithm based on the SVD. The analogue on the interval $[-1, 1]$ is known as Chebyshev–Padé approximation.*

Exercise 27.1. Padé approximation of a logarithm. Show from Theorem 27.1 that the function $f(z) = \log(1+z)$ has Padé approximants $r_{00} = 0$, $r_{1,0}(z) = z$, $r_{01}(z) = 0$, and $r_{11} = z/(1 + \frac{1}{2}z)$.

Exercise 27.2. Reciprocals and exponentials. (a) Suppose r_{mn} is the type (m, n) Padé approximant to a function f with $f(0) \neq 0$. Show that $1/r_{mn}$ is the type (n, m) Padé approximant to $1/f$. (b) As a corollary, state a theorem relating the (m, n) and (n, m) Padé approximants of e^z.

Exercise 27.3. Prescribed block structures. Devise functions f with the following structures in their Padé tables, and verify your claims numerically by color plots for $0 \leq m, n \leq 20$. (a) 3×3 blocks everywhere. (b) 1×1 blocks everywhere, except that $r_{11} = r_{21} = r_{12} = r_{22}$. (c) 1×1 blocks everywhere, except that all r_{mn} with $n \le 2$ are the same.

Exercise 27.4. Order stars. The *order star* of a function f and its approximation r is the set of points z in the complex plane for which $|f(z)| = |r(z)|$. Use the Matlab `contour` command to plot the order stars of the Padé approximations r_{11}, r_{22}, r_{32} and r_{23} to e^z. Comment on the behavior near the origin.

Exercise 27.5. Nonsingularity and normality. Show that for a given f and (m, n), the Padé approximation r_{mn} has defect $d = 0$ if and only if the square matrix obtained by deleting the first column of (27.8) is nonsingular. (If all such matrices are nonsingular, the Padé table of f is accordingly **normal**, with all its entries distinct.)

Exercise 27.6. Arbitrary patterns of square blocks? Knowing that degeneracies in the Padé table always occupy square blocks, one might conjecture that, given any tiling of the quarter-plane $m \ge 0$, $n \ge 0$ by square blocks, there exists a function f with this pattern in its Padé table. Prove that this conjecture is false. (Hint: consider the case where the first two rows of the table are filled with 2×2 blocks [Trefethen 1984].)

Exercise 27.7. Continued fractions and the Padé table. If $d_0, d_1, \dots$ is a sequence of numbers, the *continued fraction*

$$d_0 + \cfrac{d_1 z}{1 + \cfrac{d_2 z}{1 + \cdots}} \qquad (27.15)$$

is a shorthand for the sequence of rational functions

$$d_0,\ d_0 + d_1 z,\ d_0 + \frac{d_1 z}{1 + d_2 z},\ \dots, \qquad (27.16)$$

known as *convergents* of the continued fraction. (a) Show that if $d_0, \dots, d_{k-1} \ne 0$ and $d_k = 0$, then (27.15) defines a rational function $r(z)$, and determine its exact type. (b) Assuming $d_k \ne 0$ for all k, show that the convergents are the Padé approximants of types $(0, 0), (1, 0), (1, 1), (2, 1), (2, 2), \dots$ of a certain formal power series.

Chapter 28

Analytic continuation and convergence acceleration

```
ATAPformats
```

We have considered techniques for rational approximation by best approximation on an interval (Chapter 24, `remez`), interpolation or linearized least-squares fitting on an interval or disk (Chapter 26, `ratinterp` and `ratdisk`), and Padé approximation at a point or Chebyshev–Pade approximation on an interval (Chapter 27, `padeapprox` and `chebpade`). In this final chapter, we turn to the application of such approximations for extrapolating a function to real or complex values z outside the region where it is initially known. Three of the applications listed in Chapter 23 fall into this category: those numbered 3 (convergence acceleration for sequences and series), 4 (determination of poles), and 5 (analytic continuation).

It will be a chapter more of examples than theory. For an example to begin the discussion, suppose we pretend that we can evaluate

$$f(z) = \tanh(z)$$

for real values of z but know nothing about complex values, and we wish to estimate where f has poles. How might we proceed? (Of course we really know the answer: there are poles at all the odd multiples of $\pm\pi i/2$.)

The first thing to try might be polynomials. For example, we could use Chebfun to construct a polynomial that approximates f to 16 digits on $[-1,1]$,

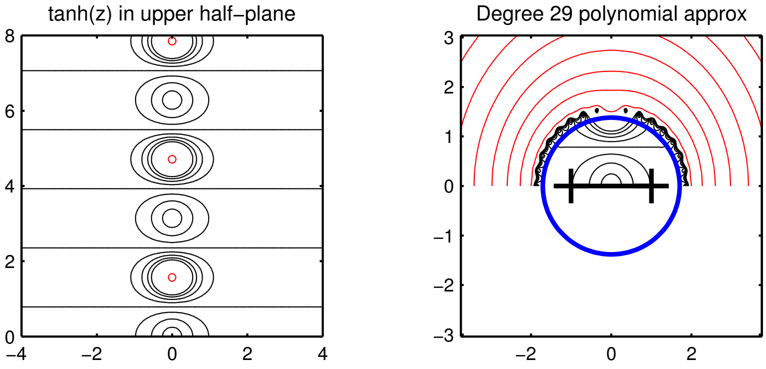
```
f = @(z) tanh(z); p = chebfun(f); length(p)
```

```
ans =
    32
```

From here, however, it is hard to make much progress. As we know from Chapter 8, p will be a good approximation to f within a certain Bernstein ellipse, the Chebfun ellipse, which can be plotted by the command `chebellipseplot`. We can expect this ellipse to reach approximately out to the first singularities at $\pm\pi i/2$. Once we hit the ellipse, however, everything will change. According to the theory of Walsh [1959] and Blatt and Saff [1986] mentioned in Chapter 18, zeros of p will cluster all along the boundary, and a further result of Blatt and Saff states that outside the ellipse, there will be no convergence at all. The polynomial p will simply grow rapidly, its behavior having nothing to do with that of f. We can confirm this prediction with contour plots. Here are plots of $|f(z)|$ and $|p(z)|$ in the upper half-plane, with black contours at levels $0.25, 0.5, \ldots, 3$ and red contours at $10^1, 10^3, 10^5, \ldots, 10^{19}$. We see immediately that p matches f very well inside the Chebfun ellipse, which is marked in blue, but not at all outside.

```
x = -4:.05:4; y = 0:.05:8;
[xx yy] = meshgrid(x,y); zz = xx + 1i*yy;
ff = f(zz); pp = p(zz);
lev1 = .25:.25:2; lev2 = 10.^(1:2:19);
subplot(1,2,1), hold off, contour(x,y,abs(ff),lev1,'k'), hold on
contour(x,y,abs(ff),lev2,'r'), FS = 'fontsize';
axis([-4 4 0 8]), axis square, title('tanh(z) in upper half-plane',FS,9)
subplot(1,2,2), hold off, contour(x,y,abs(pp),lev1,'k'), hold on
contour(x,y,abs(pp),lev2,'r')
axis([-4 4 0 8]), axis square, title('Degree 29 polynomial approx',FS,9)
chebellipseplot(p,'b','linewidth',2)
```

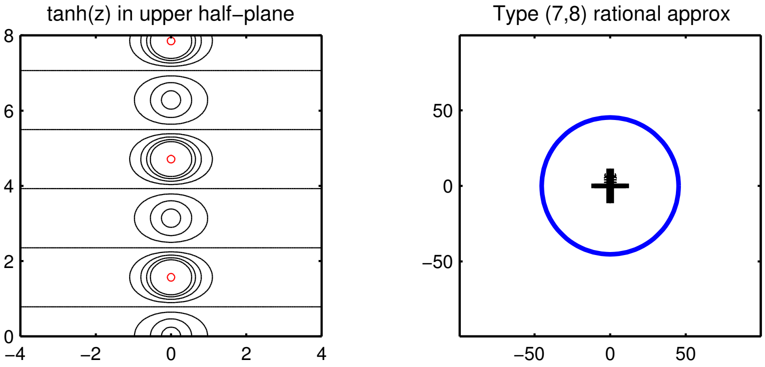
```
Warning: CHEBELLIPSEPLOT is deprecated.  Please use PLOTREGION instead.
```



To get better information, we turn to rational approximation. A practical approach is to use `ratinterp` to compute rational linearized least-squares approximations of f in $[-1,1]$. Specifically, suppose we take r to be the type $(7,8)$ approximation to f in 1000 Chebyshev points and draw the same contour plots as before. The picture changes completely, showing very impressive agreement over most of the range plotted. This is the power and the promise of rational approximation.

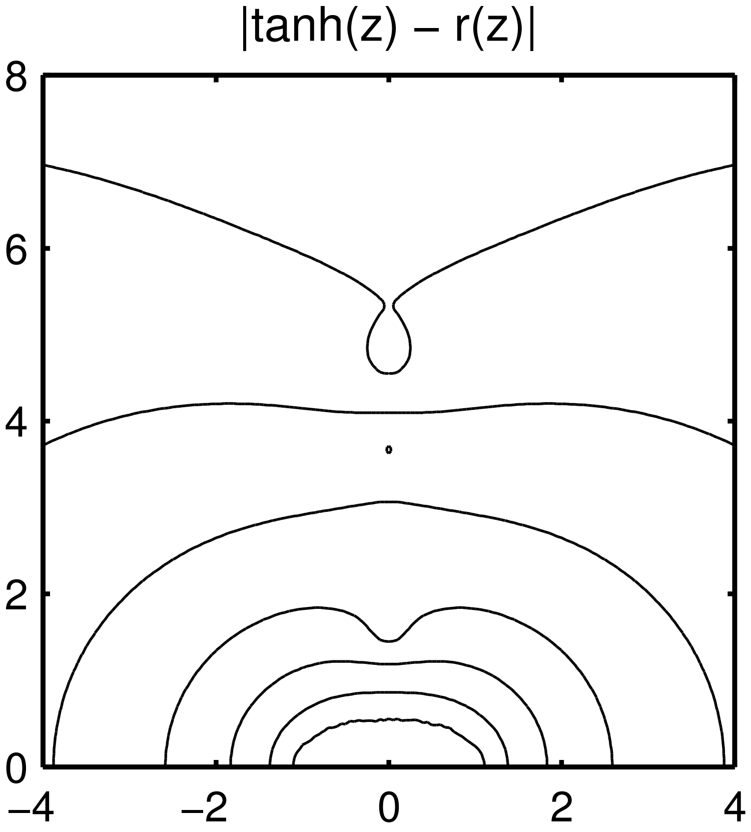
```
d = domain(-1,1);
[p,q,r,mu,nu,poles] = ratinterp(d,f,7,8,1000); rr = r(zz);
subplot(1,2,1), hold off, contour(x,y,abs(ff),lev1,'k'), hold on
contour(x,y,abs(ff),lev2,'r')
axis([-4 4 0 8]), axis square, title('tanh(z) in upper half-plane',FS,9)
subplot(1,2,2), hold off, contour(x,y,abs(rr),lev1,'k'), hold on
contour(x,y,abs(rr),lev2,'r')
axis([-4 4 0 8]), axis square, title('Type (7,8) rational approx',FS,9)
chebellipseplot(p,'b','linewidth',2)
```

```
Warning: Using a DOMAIN object as an input to RATINTERP is deprecated.
Specify domains using a two-element row vector instead.
Warning: CHEBELLIPSEPLOT is deprecated.  Please use PLOTREGION instead.
```



For a direct measure of the accuracy of r as an approximation to f, we can look at $|f(z)-r(z)|$. In the following plot the contours, from bottom to top, lie at $10^{-14}, 10^{-12}, \dots, 10^{-2}$. Evidently the approximation is excellent over a wide region.

```
levels = 10.^(-14:2:-2);
clf, subplot(1,2,1), contour(x,y,abs(ff-rr),levels,'k')
axis([-4 4 0 8]), axis square, title('|tanh(z) - r(z)|',FS,9)
```



Results like these become all the more remarkable when one recalls that the problem of analytic continuation is ill-posed: analytic continuations are unique, but they do not depend continuously on the data. For example, the following observation shows the ill-posedness of the problem of continuing a function analytically from the interval $(-1, 1)$ to the unit disk. If f is analytic in the disk, then for any $\varepsilon > 0$, there is another function g analytic in the disk such that $\|f - g\| \geq 1$ on the disk and yet $\|f - g\| \leq \varepsilon$ on the interval. (Proof: perturb f by $\varepsilon \sin(Mz)$ for a suitable value of M.) Because of this ill-posedness, every successful example of numerical analytic continuation must entail some smoothness assumptions about f, whether implicit or explicit. That is to say, numerical analytic continuation always involves some kind of regularization. (A standard reference on this subject is [Hansen 1998].) In the computations just shown, the regularization is introduced by the use of the SVD in `ratinterp`.

The question with which we opened the discussion was, where are the poles of $\tanh(z)$? To experiment with this, let us now apply `ratinterp` to compute approximants of types $(2, 2), (3, 3), \ldots, (8, 8)$, and examine the poles of these approximations. In the next output, following the convention of the past few chapters, (m, n) represents the permitted type of each approximant and (μ, ν) the exact type, with $\mu \le m$ and $\nu \le n$. Note that (μ, ν) always comes out in the form $(\mathrm{odd}, \mathrm{even})$, because f is an odd function. Thus there are always an even number of poles, which come in complex conjugate pairs and are pure imaginary, and we print just their positive imaginary parts.

```
for n = 2:8
  [p,q,r,mu,nu,poles] = ratinterp(d,f,n,n,1000);
  fprintf('\n(m,n)=(%d,%d), (mu,nu)=(%d,%d):\n',n,n,mu,nu)
  yi = sort(imag(poles)); fprintf('%15.10fi',yi(yi>0))
end
```

```
(m,n)=(2,2), (mu,nu)=(1,2):
   1.8048291471i
(m,n)=(3,3), (mu,nu)=(3,2):
   1.5884736641i
(m,n)=(4,4), (mu,nu)=(3,4):
   1.5716968677i   6.6346803797i
(m,n)=(5,5), (mu,nu)=(5,4):
   1.5708250772i   5.0809800415i
(m,n)=(6,6), (mu,nu)=(5,6):
   1.5707969475i   4.7823012576i  13.7250512960i
(m,n)=(7,7), (mu,nu)=(7,6):
   1.5707963366i   4.7228430583i   9.4128754604i
(m,n)=(8,8), (mu,nu)=(7,8):
   1.5707963264i   4.7128732394i   8.2544969405i  22.7745274849i
```

The table shows that for larger values of (m, n), two of the poles lie near $1.5707963i$ and $4.71i$. We compare these with the actual first three poles of $\tanh(z)$ in the upper half-plane:

```
disp('Exact poles:'), fprintf('%15.10fi',(pi/2)*[1 3 5])
```

```
Exact poles:
   1.5707963268i   4.7123889804i   7.8539816340i
```

Evidently the type $(7, 8)$ approximation has captured the first two poles to 9 and 3 digits of accuracy, respectively, numbers that are consistent with the contour levels near $z = 1.57i$ and $4.71i$ in the last contour plot.

To understand computations like this, it is important to recognize that the "goal" of r is not to find the poles of f, but simply to approximate f over $[-1, 1]$. If r turns out to have poles near those of f, this is a by-product, a side effect that happened because placing poles there is an effective strategy for approximation.[1] To illustrate this, suppose we compare the type $(7, 8)$ approximation above to one of type $(15, 8)$. One might expect that with more degrees of freedom, the new approximation would capture the first pole more accurately. In fact, the approximation returned has exact type $(15, 2)$, and the accuracy of the pole has deteriorated, because the denominator is less important to the quality of the least-squares approximation:

```
[p,q,r,mu,nu,poles] = ratinterp(d,f,15,8,1000);
fprintf('\n(m,n)=(15,8), (mu,nu)=(%d,%d):\n',mu,nu)
yi = sort(imag(poles)); fprintf('%15.10fi',yi(yi>0))
```

```
(m,n)=(15,8), (mu,nu)=(15,2):
   1.5707963890i
```

If we go further and ask for a type $(35, 8)$ approximant, `ratinterp` returns an approximation with no poles at all. The numerator now provides so much flexibility for the least-squares problem that the degrees of freedom in the denominator are not needed in 16-digit arithmetic, putting us back in the situation of the Chebfun ellipse of the first plot of this chapter.

```
[p,q,r,mu,nu,poles] = ratinterp(d,f,35,8,1000);
fprintf('\n(m,n)=(35,8), (mu,nu)=(%d,%d):\n',mu,nu)
```

```
(m,n)=(35,8), (mu,nu)=(25,0):
```

One must always bear this in mind when using rational approximations for extrapolation: increasing m and/or n does not always improve the accuracy of the quantities one cares about.

[1] Still, side effects can be the basis of powerful algorithms. An example is the Lanczos iteration in numerical linear algebra, which is the standard method of computing extreme eigenvalues of large symmetric matrices [Trefethen & Bau 1997]. Using this method, it is often possible to find a few dozen eigenvalues of a matrix even if the dimension is in the millions. Yet at bottom, the Lanczos iteration does nothing but construct a polynomial to minimize a certain norm. The accurate eigenvalues are a by-product of the minimization, since the optimal polynomial has roots close to some of the eigenvalues of the matrix [Trefethen & Greenbaum 1994, Kuijlaars 2006].

One way to get an idea of the dependence of an approximation on m and n is to print a table of digits of accuracy. The following table, for example, indicates the number of digits of accuracy in the computed first pole of $\tanh(z)$ for $m = 1, 3, 5, \dots, 19$ and $n = 2, 4, 6, \dots, 20$, all based on robust least-squares fits in 200 Chebyshev points in 16-digit arithmetic. The table shows again the effect that increasing m beyond a certain small value—moving right in the table—diminishes the accuracy of the pole.

```
err = zeros(1,10); disp('DIGITS OF ACCURACY: LEAST-SQUARES')
for n = 2:2:20
  for m = 1:2:19
    [p,q,r,mu,nu,poles] = ratinterp(d,f,m,n,200);
    p1 = imag(poles(abs(poles-1.6i)==min(abs(poles-1.6i))));
    err((m+1)/2) = -round(log10(abs(p1-pi/2)));
  end
  fprintf('%3d',err), disp(' ')
end
```

```
DIGITS OF ACCURACY: LEAST-SQUARES
  1  2  3  4  4  5  6  7  7  6
  1  3  5  6  7  8  9  7  7  6
  2  4  6  8  9  8  9  7  7  6
  2  5  8  9  9  8  9  7  7  6
  3  7  9  9  9  8  9  7  7  6
  4  8  9  9  9  8  9  7  7  6
  4  7  9  9  9  8  9  7  7  6
  5  7  9  9  9  8  9  7  7  6
  5  7  9  9  9  8  9  7  7  6
  5  7  9  9  9  8  9  7  7  6
```

The use of rational approximations for locating poles or other singularities has an honorable history. Many applications are mentioned in the monograph by Baker and Graves-Morris [1996], which is a standard reference on Padé approximation. One interesting kind of application is to locating singularities of solutions of ODEs or PDEs computed numerically, an idea explored among others by Weideman [2003]. For Chebfun-based explorations, including the application of `ratinterp` to find complex singularities of solutions to the Lorenz and Lottka–Volterra equations, see [Pachón 2010] and [Webb 2012].

Having just mentioned Padé approximation, which was the subject of the last chapter, let us now turn to this alternative method of constructing rational approximations. Here is a repetition of the last experiment, the table of digits of accuracy in the first pole of $\tanh(z)$, but now based on Padé approximation instead of rational least-squares. The results are similar, but better. This is not a general conclusion: it depends on the problem.

```
disp('    DIGITS OF ACCURACY: PADE')
for n = 2:2:20
  for m = 1:2:19
    [r,a,b,mu,nu,poles] = padeapprox(f,m,n);
    p1 = imag(poles(abs(poles-1.57i)==min(abs(poles-1.57i))));
    err((m+1)/2) = -round(log10(abs(p1-pi/2)));
  end
  fprintf('%3d',err), disp(' ')
end
```

```
    DIGITS OF ACCURACY: PADE
  1  2  3  4  5  6  7  8  9 10
  2  3  5  6  8  9 11 12 13 13
  2  5  7  9 10 12 15 11 12 13
  3  6  8 11 13 14 12 15 11 12
  3  7 10 12 14 13 14 12 15 11
  4  8 12 14 12 14 13 14 12 15
  5  9 13 12 14 12 14 13 14 12
  5 11 11 13 12 14 12 14 13 14
  6 12 11 11 13 12 14 12 14 13
  6 12 12 11 11 13 12 14 12 14
```

In principle, least-squares fitting and Padé approximation are very different techniques, since the first uses function values only at many different points, whereas the second uses values of the function and its derivatives at a single point. (These are the extreme cases of the general notion of *multipoint Padé approximation.*) In our actual computation, however, the difference is diminished, because `padeapprox` begins by computing Taylor coefficients numerically by the FFT based on samples of the function at roots of unity, a standard technique. So in fact, in this comparison, `ratinterp` and `padeapprox` both work from function values: the first from samples on $[-1,1]$, the second from samples on the unit circle. This raises the question, what is achieved by passing through the intermediate stage of Taylor coefficients? It is a fair point, and indeed, another effective approach would be to solve a rational least-squares problem on the circle directly as in Chapter 26. Explorations of this kind are presented in [Pachón 2010].

We now turn to the topic of acceleration of convergence of sequences and series. The challenge here is as follows. Suppose we know some of the initial terms of a convergent sequence,

$$s_0, s_1, s_2, s_3, \ldots \to S, \tag{28.1}$$

and we want to estimate the limit S. Equivalently, suppose we wish to estimate the limit of an infinite sum,

$$S = a_0 + a_1 + a_2 + \cdots. \tag{28.2}$$

The two problems are equivalent since we may regard (28.1) as a sequence of partial sums,

$$s_n = \sum_{k=0}^{n} a_k, \quad a_k = s_{k+1} - s_k. \tag{28.3}$$

If the sequence or series converges slowly, how might we speed it up? For example, perhaps we can afford to compute 20 terms, but this gives just 2-digit accuracy. Can we process the data further somehow to improve the accuracy to 6 digits?

There is a long history to such questions, reaching from Stirling and Euler to the recent tour de force solution of nine of the ten “SIAM 100-Digit Challenge” problems to 10,000 digits of accuracy [Bornemann et al. 2004]. It is probably fair to say that almost every method for accelerating convergence is based on the idea of embedding the sequence in an analytic function, though this may not be how the original authors conceived or described their method.

One way in which a sequence might be embedded in an analytic function is if the terms of the sequence can be regarded as values of a fixed function at different arguments. For example, suppose we define a function $f(z)$ at the points $z = 1, 2^{-1}, 2^{-2}, \ldots$ by the formula $f(2^{-k}) = s_k$. Then (28.1) becomes

$$f(1), f(2^{-1}), f(2^{-2}), \ldots \to S. \tag{28.4}$$

Does this point of view help us estimate S? The answer will probably be yes if there exists a function f that is analytic in a neighborhood of $z = 0$ and takes the given values at $z = 2^{-k}$. In such a case, to estimate S, it is enough to interpolate some of the data by a polynomial $p(z)$ and then compute $p(0)$. This is the method known as *Richardson extrapolation*, which is of great practical importance in applications.[2] In a typical application, h might be the

[2] Lewis Fry Richardson used such ideas as early as 1910, and for a systematic treatment see his charming article [Richardson 1927]. There are various earlier roots of Richardson extrapolation too, including Huygens in the 17th century.

mesh size of a numerical discretization and $f(h), f(h/2), f(h/4), \ldots$ the estimates obtained of a quantity of interest as the mesh is successively refined. Often only even powers of h appear, indicating that f is an even function, so one could take the view that the data are given at $\pm h, \pm h/2, \ldots$ and Richardson extrapolation is really Richardson interpolation. In the specific case in which $f(h)$ is an estimate of an integral by the trapezoid or rectangle rule with step length h, this becomes the quadrature method known as *Romberg quadrature.* Nor is the idea of polynomial extrapolation from data such as (28.4) limited to cases in which the sample points are related by factors of 2. If they are $1, 1/2, 1/3, \ldots,$ this is called *Salzer extrapolation* [Salzer 1955].

Often, however, the limit of a sequence or series is not in the interior of a region of analyticity of an analytic function. In such a case there may be less mileage in Richardson extrapolation, and one looks for formulations adapted to the edge of a region of analyticity. For such problems, there is a basic starting point: to insert a parameter z in (28.2) so that it becomes the series

$$S(z) = a_0 + a_1 z + a_2 z^2 + \cdots. \tag{28.5}$$

Now we have the problem of evaluating $S(1)$ for a function $S(z)$ with known Taylor coefficients. If (28.2) converges, then $z = 1$ is a point of convergence of (28.5), and if (28.2) converges more slowly than geometrically, then $z = 1$ must be on the boundary of the disk of convergence of (28.5). So by introducing a parameter z, we have converted the problem of the summation of a slowly convergent series to a problem of evaluating an analytic function at a point on the boundary of the disk of convergence of its Taylor series.

The simplest idea would be to evaluate $S(z)$ for a succession of values of z and use the identity

$$S(1) = \lim_{z\to 1} S(z),$$

where the limit is over real values of z increasing to 1. This idea is known as *Abel summation* [Hardy 1991].

A more powerful and general approach is to use rational functions, specifically Padé approximants since the data are given as Taylor coefficients. Two variants of this idea have received special attention. We could construct a sequence of type $(m, 1)$ Padé approximants, with one pole, and evaluate them at $z = 1$:

$$r_{01}(1), r_{11}(1), r_{21}(1), \ldots.$$

This is called *Aitken extrapolation* or *Aitken's* Δ^2 *method,* used by Aitken [1926] though with origins further back. Or we could work with type (n, n)

Padé approximants,

$$r_{00}(1), r_{11}(1), r_{22}(1), \ldots .$$

This is called *epsilon extrapolation* (originally for sequences) [Shanks 1955, Wynn 1956] or *eta extrapolation* (originally for series) [Bauer 1959]. An earlier appearance of essentially the same idea is due to Schmidt [1941].

Here is an example showing how powerful eta extrapolation can be for some problems. What is the value of

$$S = \sum_{n=2}^{\infty} \frac{\sin(n)}{\log(n)}?$$

The series is extremely slow to converge, as we see by taking partial sums of as many as a million terms:

```
S = @(n) sum(sin(2:n)./log(2:n));
disp('    n          S(n)')
for n = 10.^[1:6]
  fprintf('%6.1e   %10.6f\n',n,S(n))
end
```

```
    n          S(n)
1.0e+01      0.907319
1.0e+02      0.457822
1.0e+03      0.669234
1.0e+04      0.761940
1.0e+05      0.764913
1.0e+06      0.609190
```

To get 10-digit accuracy by summing the series in this fashion, we would need $10^{10000000000}$ terms! The actual answer (not known analytically) is

$$S \approx 0.68391378641828 \ldots .$$

Here are the diagonal extrapolants, that is, the results of eta extrapolation. Now we just go from 2^1 to 2^6 instead of from 10^1 to 10^6, yet we get 14 digits of accuracy instead of 1:

```
n = 2:150; c = [0 0 sin(n)./log(n)];
disp('  ( n, n)    (mu,nu)         r_nn(1)  ')
disp('  -------    -------        -----------')
for n = 2.^(1:6)
  [r,a,b,mu,nu] = padeapprox(c,n,n,0);
  fprintf('  (%2.0d,%2.0d)    (%2.0d,%2.0d)   %19.15f\n',n,n,mu,nu,r(1))
end
```

```
  ( n, n)    (mu,nu)         r_nn(1)
  -------    -------        -----------
  ( 2, 2)    ( 2, 2)     0.987966950435009
  ( 4, 4)    ( 4, 4)     0.716844624573063
  ( 8, 8)    ( 8, 8)     0.684142517808588
  (16,16)    (16,16)     0.683914509677347
  (32,32)    (32,32)     0.683913786418273
  (64,64)    (64,64)     0.683913786418274
```

The convergence is excellent. Note that we have computed Padé approximants non-robustly by specifying a tolerance of 0 to `padeapprox`. In typical applications, this use of non-robust formulas seems advantageous in extrapolation applications, though it brings a risk of sensitivity to noise. For this example, calling `padeapprox` with its default tolerance 10^{-14} leads to stagnation at type $(15, 15)$ with just 7 digits of accuracy.

This simple method of eta extrapolation, at least as implemented by Chebfun's Padé approximation code, can be encapsulated in a single Matlab command we may call `extrap`. Given a sequence $a_0, a_1, ..., a_N$, we can round $N/2$ to integers (say, round up for m and down for n) and then use `padeapprox` to compute the type (m, n) Padé approximation r. The accelerated value is then $r(1)$. Here is the code.

```
eval_at_1 = @(r) r(1); N2 = @(c) length(c)/2;
extrap = @(c) eval_at_1(padeapprox(c,ceil(N2(c)),floor(N2(c)),0));
```

The $\sin(n)/\log(n)$ example just treated is this:

```
extrap([0 0 sin(2:150)./log(2:150)])
```

```
ans =
   0.683913786418275
```

For another example, suppose we extrapolate the alternating series

$$1 - \frac{1}{2} + \frac{1}{3} - \cdots = \log(2), \tag{28.6}$$

The result is accurate to machine precision:

```
extrap((-1).^(0:30)./(1:31)), exact = log(2)
```

```
ans =
   0.693147180559945
exact =
   0.693147180559945
```

Note that here, the function f of (28.5) is $\log(1+z)$, so this example shows that eta extrapolation can be effective for functions with branch cuts as well as poles.

Another famous alternating series, which we can obtain by setting $t = 0$ in equation (9.3), is

$$1 - \frac{1}{3} + \frac{1}{5} - \cdots = \frac{\pi}{4}, \tag{28.7}$$

Again, extrapolation gives machine precision:

```
extrap((-1).^(0:30)./(1:2:61)), exact = pi/4
```

```
ans =
   0.785398163397448
exact =
   0.785398163397448
```

These examples are very impressive, but it is not always so. For example, here is what happens if we attempt to extrapolate the series

$$\zeta(2) = 1 + \frac{1}{2^2} + \frac{1}{3^2} + \cdots = \frac{\pi^2}{6}, \tag{28.8}$$

```
extrap(1./(1:30).^2), exact = pi^2/6
```

```
ans =
   1.639604727858872
exact =
   1.644934066848226
```

The convergence is very poor because in this case the function $f(z)$ of (28.5), known as the dilogarithm, has a branch point at $z = 1$ itself. As it happens, this is a case where Salzer extrapolation is effective (Exercise 28.3).

The discussion of convergence acceleration of the last five pages has little in common with the large literature of this subject, because our focus has been solely on the underlying approximations, particularly Padé approximants, and not at all on the mechanics. Our numerical illustrations have utilized the linear algebra of Chapter 27, based on the SVD and requiring $O(n^3)$ floating-point operations to compute a single estimate based on a type (n, n) approximant. The literature of convergence acceleration is quite different, for it emphasizes recurrence relations and triangular or rhomboidal arrays related to continued fractions that can be used to generate a sequence of approximations at great speed without solving matrix problems. These approaches are certainly faster, and in fact they may often be more accurate for extrapolation, though they come with a risk of sensitivity to noise and the possibility of breakdown if there is a division by 0.

A major reason why we have ignored the mechanical or implementational aspects of convergence acceleration is that these matters are complicated—and, one might say, distracting. The differences between various extrapolation algorithms in practice can be quite intricate, and in a discussion of such matters, one quickly loses sight of the underlying mathematics of approximation. For details of these aspects of convergence acceleration see surveys such as Chapter 3 of [Baker & Graves-Morris 1996], [Brezinski & Redivo Zaglia 1991], [Gragg 1972], [Joyce 1971], [Sidi 2003], [Weniger 1989], [Wimp 1981], or the appendix by Laurie in [Bornemann, et al. [2004]. Such literature also points to many further acceleration methods beyond those we have mentioned, such as Levin's sequence transformation and Brezinski's theta method.

We finish with an observation that points to exciting further territories of interest to mathematicians at least since Euler. The series (28.5) consists just of Taylor coefficients, so it is meaningful even if the radius of convergence is less than 1. Therefore our methods based on analytic continuation can sum divergent series as well as convergent ones. For example, the Taylor series

$$\frac{1}{1+z} = 1 - z + z^2 - z^3 + \cdots$$

suggests the result

$$1 - 1 + 1 - 1 + \cdots = \frac{1}{2}, \tag{28.9}$$

if we set $z = 1$. Similarly, setting $z = 2$ suggests

$$1 - 2 + 4 - 8 + \cdots = \frac{1}{3}. \tag{28.10}$$

Are these identities actually "correct"? As usual in mathematics, the answer depends on what definitions we choose. The formulas (28.9) and (28.10) are not too problematic since they correspond to Taylor series with positive radii of convergence. In more challenging cases, the series is only asymptotic. For example, what about this series with factorial coefficients considered by Euler [1760],

$$0! - 1! + 2! - 3! + \cdots = \;? \tag{28.11}$$

The factorials grow too fast to be Taylor coefficients for any function analytic in a neighborhood of $z = 0$. However, they are the asymptotic series coefficients at $z = 0$ for a function analytic in the right half-plane, namely

$$f(z) = \int_0^\infty \frac{e^{-t}}{1+zt}\, dt. \tag{28.12}$$

So a plausible candidate for the sum of (28.11) is

$$0! - 1! + 2! - 3! + \cdots = f(1) = 0.596347362\ldots. \tag{28.13}$$

Our code `extrap` makes a creditable attempt at computing this number:

```
extrap((-1).^(0:10).*factorial(0:10))
```

```
ans =
   0.593294558846645
```

Summary of Chapter 28. *Rational approximations provide one of the basic technologies for analytic continuation and extrapolation. In particular, Padé approximants are the basis of standard methods of convergence acceleration for sequences and series including the Aitken Δ^2, Shanks, epsilon and eta methods.*

Exercise 28.1. Contour plot for Taylor polynomials. Draw a contour plot like the pair in this chapter for the Taylor polynomial approximants to $f(z) = \tanh(z)$. Comment on the result.

Exercise 28.2. The divergent factorial series. Compute numerically the Padé approximants $r_{33}, r_{44}, \dots, r_{77}$ for the Taylor coefficients (28.11), and show that they match $f(1)$ to better than 1%, where f is defined by (28.12). What accuracy do these approximants give for $f(1/2)$?

Exercise 28.3. Zeta function. It was noted in the text that eta extrapolation is ineffective for the series (28.8). Study the behavior of Richardson and Salzer extrapolation instead.

Exercise 28.4. Alternating square roots. (a) To 8 digits of accuracy, what do you think is the limit of $1 - 1/\sqrt{2} + 1/\sqrt{3} - \cdots$? (b) To the same accuracy, what number would you propose as a good choice for the sum of the divergent series $1 - \sqrt{2} + \sqrt{3} - \cdots$?

Exercise 28.5. Approximations to e^z. Compute type $(1, 1)$ approximations to e^z on $[-1, 1]$ by (a) Padé approximation, (b) best approximation, (c) Chebyshev–Padé approximation, (d) Carathéodory–Fejér approximation, (e) interpolation in 3 Chebyshev points, and (f) linearized least-squares approximation in a number of Chebyshev points large enough to be effectively infinite. In each case list the coefficients, measure the L^2 and L^∞ errors, and plot the error curve.

Exercise 28.6. Nonlinear least-squares approximation. Find a way to compute the true type $(1, 1)$ nonlinear least-squares approximation to e^z on $[-1, 1]$, and report the same data for this function as for the approximations of Exercise 28.7.

Exercise 28.7. An alternating series. The following identity is known:

$$1 + \frac{1}{2} - \frac{1}{3} - \frac{1}{4} + \frac{1}{5} + \frac{1}{6} - \frac{1}{7} - \cdots = \frac{\pi}{4} + \frac{1}{2}\log 2. \qquad (28.14)$$

How many digits do you get by taking $10^1, 10^2, \dots, 10^6$ terms of the series? Can you get more by extrapolation?

REFERENCES

Each reference is followed by a note highlighting a contribution of that publication that is relevant to this book. These notes are by no means comprehensive: in most cases the references include other significant contributions too. Papers listed by authors such as Cauchy, Chebyshev, Gauss, Jacobi, and Weierstrass can also be found in their collected works.

Among mathematicians of the 19th century, it is hard not to be struck by the remarkable creativity of Jacobi (1804–1851), who in his short life made key early contributions to barycentric interpolation [1825], orthogonal polynomials and Gauss quadrature [1826], and Padé approximation and rational interpolation [1846], as well as innumerable topics outside the scope of this book.

As of May 2012, the Mathematics Genealogy Project lists 8605 adademic descendents of Pafnuty Lvovich Chebyshev. For example, one chain runs Chebyshev–Lyapunov–Steklov–Smirnov–Sobolev–V. I. Lebedev, and another runs Chebyshev–Markov–Voronoy–Sierpinsky–Mazurkiewicz–Zygmund–Stein–C. Fefferman.

N. I. Achieser, On extremal properties of certain rational functions (Russian), *DAN* 18 (1930), 495–499. [Equioscillation characterization for best rational approximations.]

N. I. Achieser, *Theory of Approximation,* Dover, 1992. [Treatise by one of Chebyshev's academic great-grandsons, first published in 1956.]

V. Adamyan, D. Arov and M. Krein, Analytic properties of Schmidt pairs for a Hankel operator and the generalized Schur–Takagi problem, *Math. USSR Sb.* 15 (1971), 31–73. [Major work with a general extension of results of Carathéodory, Fejér, Schur and Takagi to rational approximation on the unit circle.]

L. Ahlfors, *Complex Analysis,* 3rd ed., McGraw-Hill, 1978. [A terse and beautiful complex analysis text by one of the masters, first published in 1953.]

N. Ahmed and P. S. Fisher, Study of algorithmic properties of Chebyshev coefficients, *Int. J. Comp. Math.* 2 (1970), 307–317. [Possibly the first paper to point out that Chebyshev coefficients can be computed by Fast Fourier Transform.]

A. C. Aitken, On Bernoulli's numerical solution of algebraic equations, *Proc. Roy. Soc. Edinb.* 46 (1926), 289–305.

B. K. Alpert and V. Rokhlin, A fast algorithm for the evaluation of Legendre expansions, *SIAM J. Sci. Stat. Comp.* 12 (1991), 158–179. [Algorithm for converting between Legendre and Chebyshev expansion coefficients.]

A. Amiraslani, *New Algorithms for Matrices, Polynomials, and Matrix Polynomials,* PhD diss., U. Western Ontario, 2006. [Algorithms related to rootfinding by values rather than Cheybyshev coefficients.]

A. Amiraslani, R. M. Corless, L. Gonzalez-Vega and A. Shakoori, Polynomial algebra by values, TR-04-01, Ontario Research Center for Computer Algebra, www.orcca.on.ca, 2004. [Outlines eigenvalue-based algorithms for finding roots of polynomials from their values at sample points rather than from coefficients in an expansion.]

A. C. Antoulas, *Approximation of Large-Scale Dynamical Systems*, SIAM, 2005. [Textbook about model reduction, a subject making much use of rational approximation.]

A. I. Aptekarev, Sharp constants for rational approximations of analytic functions, *Math. Sbornik* 193 (2002), 1–72. [Extends the result of Gonchar & Rakhmanov 1989 on rational approximation of e^x on $(-\infty, 0]$ to give the precise asymptotic form $E_{nn} \sim 2H^{n+1/2}$ first conjectured by Magnus, where H is Halphen's constant.]

T. Bagby and N. Levenberg, Bernstein theorems, *New Zeal. J. Math.* 22 (1993), 1–20. [Presentation of four proofs of Bernstein's result that best polynomial approximants to a function $f \in C([-1,1])$ converge geometrically if and only if f is analytic, with discussion of extension to higher dimension.]

G. A. Baker, Jr. and P. Graves-Morris, *Padé Approximants,* 2nd ed., Cambridge U. Press, 1996. [The standard reference on many aspects of Padé approximations and their applications.]

N. S. Bakhvalov, On the optimal speed of integrating analytic functions, *Comput. Math. Math. Phys.* 7 (1967), 63–75. [A theoretical paper that introduces the idea of going beyond polynomials to speed up Gauss quadrature by means of a change of variables/conformal map, as in Hale & Trefethen 2008.]

S. Barnett (1975a), A companion matrix analogue for orthogonal polynomials, *Lin. Alg. Applics.* 12 (1975), 197–208. [Generalization of Good's colleague matrices to orthogonal polynomials other than Chebyshev. Barnett apparently did not know that Specht 1957 had covered the same ground.]

S. Barnett (1975b), Some applications of the comrade matrix, *Internat. J. Control* 21 (1975), 849–855. [Further discussion of comrade matrices.]

Z. Battles, *Numerical Linear Algebra for Continuous Functions,* DPhil thesis, Oxford U. Computing Laboratory, 2005. [Presentation of Chebfun, including description of Chebfun's rootfinding algorithm based on recursion and eigenvalues of colleague matrices.]

Z. Battles and L. N. Trefethen, An extension of Matlab to continuous functions and operators, *SIAM J. Sci. Comp.* 25 (2004), 1743–1770. [Chebfun was conceived on December 8, 2001, and this was the first publication about it.]

F. L. Bauer, The quotient-difference and epsilon algorithms, in R. E. Langer, ed., *On Numerical Approximation,* U. Wisconsin Press, 1959, pp. 361–370. [Introduction of the eta extrapolation algorithm for series.]

R. Bellman, B. G. Kashef and J. Casti, Differential quadrature: a technique for the rapid solution of nonlinear partial differential equations, *J. Comp. Phys.* 10 (1972), 40–52. [Perhaps the first publication to give the formula for entries of a spectral differentiation matrix.]

S. N. Bernstein, Sur l'approximation des fonctions continues par des polynômes, *Compt. Rend. Acad. Sci.* 152 (1911), 502–504. [Announcement of some results proved in Bernstein 1912b.]

S. N. Bernstein (1912a), Sur les recherches récentes relatives à la meilleure approximation des fonctions continues par des polynômes, *Proc. 5th Intern. Math. Congress, v. 1*, 1912, 256–266. [Announcement of the results of Bernstein and Jackson on polynomial approximation, including a table summarizing theorems by Bernstein, Jackson and Lebesgue linking smoothness to rate of convergence.]

S. N. Bernstein (1912b), *Sur l'ordre de la meilleure approximation des fonctions continues par des polynômes de degré donné*, Mém. Acad. Roy. Belg., 1912, pp. 1–104. [Major work (which won a prize from the Belgian Academy of Sciences) establishing a number of the Jackson and Bernstein theorems on rate of convergence of best approximations for differentiable or analytic f. Bernstein's fundamental estimates for functions analytic in an ellipse appear in Sections 9 and 61.]

S. N. Bernstein (1912c), Sur la valeur asymptotique de la meilleure approximation des fonctions analytiques, *Compt. Rend. Acad. Sci.* 155 (1912), 1062–1065. [One of the first appearances of Bernstein ellipses, used here to analyze convergence of best approximations for a function with a single real singularity on the ellipse.]

S. N. Bernstein (1912d), Démonstration du théorème de Weierstrass fondée sur le calcul des probabilités, *Proc. Math. Soc. Kharkov* 13 (1912), 1–2. [Bernstein's proof of the Weierstrass approximation theorem based on Bernstein polynomials.]

S. N. Bernstein (1914a), Sur la meilleure approximation des fonctions analytiques possédant des singularités complexes, *Compt. Rend. Acad. Sci.* 158 (1914), 467–469. [Generalization of Bernstein 1912c to functions with a conjugate pair of singularities.]

S. N. Bernstein (1914b), Sur la meilleure approximation de $|x|$ par des polynômes de degrés donnés, *Acta Math.* 37 (1914), 1–57. [Investigates polynomial best approximation of $|x|$ on $[-1,1]$ and mentions as a "curious coincidence" that $nE_n \approx 1/2\sqrt{\pi}$, a value that became known as the "Bernstein conjecture," later shown false by Varga and Carpenter.]

S. N. Bernstein, Quelques remarques sur l'interpolation, *Math. Annal.* 79 (1919), 1–12. [Written in 1914 but delayed in publication by the war, this paper, like Faber 1914, pointed out that no array of nodes for interpolation could yield convergence for all continuous functions.]

S. N. Bernstein, Sur la limitation des valeurs d'un polynômes $P(x)$ de degré n sur tout un segment par ses valeurs en $(n+1)$ points du segment, *Izv. Akad. Nauk SSSR* 7 (1931), 1025–1050. [Discussion of the problem of optimal interpolation nodes, defined by minimization of the Lebesgue constant.]

S. N. Bernstein, On the inverse problem of the theory of the best approximation of continuous functions, *Sochineya* 2 (1938), 292–294. [Bernstein's lethargy theorem.]

J.-P. Berrut, Rational functions for guaranteed and experimentally well-conditioned global interpolation, *Comput. Math. Appl.* 15 (1988), 1–16. [Observes that if the barycentric formula is applied on an arbitrary grid with weights $1, -1, 1, -1, \ldots$ or $\frac{1}{2}, -1, 1, -1 \ldots$, the resulting rational interpolants are pole-free and accurate.]

J.-P. Berrut, R. Baltensperger and H. D. Mittelmann, Recent developments in barycentric rational interpolation, *Intern. Ser. Numer. Math.* 151 (2005), 27–51. [Combines conformal maps with the rational barycentric formula to get high-accuracy approximations of difficult functions.]

J.-P. Berrut, M. S. Floater and G. Klein, Convergence rates of derivatives of a family of barycentric rational interpolants, *Appl. Numer. Math.* 61 (2011), 989–1000. [Establishes convergence rates for derivatives of Floater–Hormann barycentric rational interpolants.]

J.-P. Berrut and L. N. Trefethen, Barycentric Lagrange interpolation, *SIAM Rev.* 46 (2004), 501–517. [Review of barycentric formulas for polynomial and trigonometric interpolation.]

A. Birkisson and T. Driscoll, Automatic Fréchet differentiation for the numerical solution of boundary-value problems. *ACM Trans. Math. Softw.,* to appear, 2012. [Description of Chebfun's method for solving nonlinear differential equations based on Newton or damped-Newton iteration and Automatic Differentiation.]

H.-P. Blatt, A. Iserles and E. B. Saff, Remarks on the behaviour of zeros of best approximating polynomials and rational functions, in J. C. Mason and M. G. Cox, *Algorithms for Approximation,* Clarendon Press, 1987, pp. 437–445. [Shows that the type (n, n) best rational approximations to $|x|$ on $[-1, 1]$ have all their zeros and poles on the imaginary axis and converge to x in the right half-plane and to $-x$ in the left half-plane.]

H.-P. Blatt and E. B. Saff, Behavior of zeros of polynomials of near best approximation, *J. Approx. Th.* 46 (1986), 323–344. [Shows that if $f \in C([-1, 1])$ is not analytic on $[-1, 1]$, then the roots of its best approximants $\{p_n^*\}$ cluster at every point of $[-1, 1]$ as $n \to \infty$.]

H. F. Blichfeldt, Note on the functions of the form $f(x) \equiv \phi(x) + a_1 x^{n-1} + a_2 x^{n-2} + \cdots + a_n$ which in a given interval differ the least possible from zero, *Trans. Amer. Math. Soc.* 2 (1901), 100–102. [Blichfeldt proves a part of the equioscillation theorem: optimality implies equioscillation.]

M. Bôcher, Introduction to the theory of Fourier's series, *Ann. Math.* 7 (1906), 81–152. [The paper that named the Gibbs phenomenon.]

E. Borel, *Leçons sur les fonctions de variables réelles et les développements en series de polynômes,* Gauthier-Villars, Paris, 1905. [The first textbook essentially about approximation theory, including a proof of the equioscillation theorem, which Borel attributes to Kirchberger.]

F. Bornemann, D. Laurie, S. Wagon and J. Waldvogel, *The SIAM 100-Digit Challenge: A Study in High-Accuracy Numerical Computing,* SIAM, 2004. [Detailed study of ten problems whose answers are each a single number, nine of which the authors manage to compute to 10,000 digits of accuracy through the use of ingenious algorithms and acceleration methods.]

J. P. Boyd, *Chebyshev and Fourier Spectral Methods,* 2nd ed., Dover, 2001. [A 668-page treatement of the subject with a great deal of practical information.]

J. P. Boyd, Computing zeros on a real interval through Chebyshev expansion and polynomial rootfinding, *SIAM J. Numer. Anal.* 40 (2002), 1666–1682. [Proposes recursive Chebyshev expansions for finding roots of real functions, the idea that is the basis of the `roots` command in Chebfun.]

D. Braess, On the conjecture of Meinardus on rational approximation to e^x. II, *J. Approx. Th.* 40 (1984), 375–379. [Establishes an asymptotic formula conjectured by Meinardus for the best approximation error of e^x on $[-1,1]$.] CHECK

D. Braess, *Nonlinear Approximation Theory,* Springer, 1986. [Advanced text on rational approximation and other topics, with emphasis on methods of functional analysis.]

C. Brezinski, Extrapolation algorithms and Padé approximations: a historical survey, *Appl. Numer. Math.* 20 (1996), 299–318. [Historical survey.]

C. Brezinski and M. Redivo Zaglia, *Extrapolation Methods: Theory and Practice,* North-Holland, 1991. [Extensive survey.]

L. Brutman, On the Lebesgue function for polynomial interpolation, *SIAM J. Numer. Anal.* 15 (1978), 694–704. [Sharpening of a result of Erdős 1960 concerning Lebesgue constants.]

L. Brutman, Lebesgue functions for polynomial interpolation—a survey, *Ann. Numer. Math.* 4 (1997), 111–127. [Exceptionally useful survey, including detailed results on interpolation in Chebyshev points.]

P. Butzer and F. Jongmans, P. L. Chebyshev (1821–1894): A guide to his life and work, *J. Approx. Th.* 96 (1999), 111–138. [Discussion of the leading Russian mathematician of the 19th century.]

C. Canuto, M. Y. Hussaini, A. Quarteroni and T. A. Zang, *Spectral Methods: Fundamentals in Single Domains*, Springer, 2006. [A major monograph on both collocation and Galerkin spectral methods.]

C. Carathéodory and L. Fejér, Über den Zusammenhang der Extremen von harmonischen Funktionen mit ihrer Koeffizienten und über den Picard-Landauschen Satz, *Rend. Circ. Mat. Palermo* 32 (1911), 218–239. [The paper that led, together with Schur 1918, to the connection of approximation problems with eigenvalues and singular values of Hankel matrices, later the basis of the Carathéodory–Fejér method for near-best approximation.]

A. J. Carpenter, A. Ruttan and R. S. Varga, Extended numerical computations on the "1/9" conjecture in rational approximation theory, in P. Graves-Morris, E. B. Saff and R. S. Varga, eds., *Rational Approximation and Interpolation,* Lect. Notes Math. 1105, Springer, 1984. [Calculation to 40 significant digits of the best rational approximations to e^x on $(-\infty,0]$ of types $(0,0),(1,1),\dots,(30,30)$.]

A. L. Cauchy, Sur la formule de Lagrange relative à l'interpolation, *Cours d'Analyse de l'École Royale Polytechnique: Analyse algébrique,* Imprimerie Royale, Paris, 1821. [First treatment of the "Cauchy interpolation problem" of interpolation by rational functions.]

A. L. Cauchy, Sur un nouveau genre de calcul analogue au calcul infinitésimal, *Exerc. Mathématiques* 1 (1926), 11–24. [One of Cauchy's foundational texts on residue calculus, including a derivation of what became known as the Hermite integral formula.]

P. L. Chebyshev, Théorie des mécanismes connus sous le nom de parallélogrammes, *Mém. Acad. Sci. Pétersb.,* Series 7 (1854), 539–568. [Introduction of the idea of best approximation by polynomials in the supremum norm.]

P. L. Chebyshev, Sur les questions de minima qui se rattachent à la représentation approximative des fonctions, *Mém. Acad. Sci. Pétersb.* Series 7 (1859), 199–291. [Chebyshev's principal work on best approximation.]

E. W. Cheney, *Introduction to Approximation Theory,* Chelsea, 1999. [Classic approximation theory text first published in 1966.]

J. F. Claerbout, *Imaging the Earth's Interior,* Blackwell, 1985. [Text about the mathematics of migration for earth imaging by the man who developed many of these techniques, based on rational approximations of pseudodifferential operators.]

C. W. Clenshaw and A. R. Curtis, A method for numerical integration on an automatic computer, *Numer. Math.* 2 (1960), 197–205. [Introduction of Clenshaw–Curtis quadrature.]

C. W. Clenshaw and K. Lord, Rational approximations from Chebyshev series, in B. K. P. Scaife, ed., *Studies in Numerical Analysis,* Academic Press, 1974, pp. 95–113.

W. J. Cody, The FUNPACK package of special function subroutines, *ACM Trans. Math. Softw.* 1 (1975), 13–25. [Codes for evaluating special functions based on rational approximations.]

W. J. Cody, Algorithm 715: SPECFUN—A portable FORTRAN package of special function routines and test drivers, *ACM Trans. Math. Softw.* 19 (1993), 22–32. [Descendant of FUNPACK with greater portability.]

W. J. Cody, W. Fraser and J. F. Hart, Rational Chebyshev approximation using linear equations, *Numer. Math.* 12 (1968), 242–251. [Algol 60 code for best rational approximation by a variant of the Remes algorithm.]

W. J. Cody, G. Meinardus and R. S. Varga, Chebyshev rational approximations to e^{-x} in $[0,+\infty)$ and applications to heat-conduction problems, *J. Approx. Th.* 2 (1969), 50–65. [Introduces the problem of approximation of e^{-x} on $[0,\infty)$, or equivalently e^x on $(-\infty,0]$, and shows that rational best approximants converge geometrically.]

R. M. Corless and S. M. Watt, Bernstein bases are optimal, but, sometimes, Lagrange bases are better, 2004, Proc. SYNASC (Symbolic and Numeric Algorithms for Scientific Computing), Timisoara, 2004, pp. 141–152. [A contribution to polynomial rootfinding with a marvelous title.]

G. Darboux, Mémoire sur l'approximation des fonctions de très-grands nombres, et sur une classe étendue de développements en série, *J. Math. Pures Appl.* 4 (1878), 5–56. *check*

S. Darlington, Analytical approximations to approximations in the Chebyshev sense, *Bell System Tech. J.* 49 (1970), 1–32. [A precursor to the Carathéodory–Fejér method.]

P. J. Davis, *Interpolation and Approximation,* Dover, 1975. [A leading text on the subject, first published in 1963.]

P. J. Davis and P. Rabinowitz, *Methods of Numerical Integration,* 2nd ed., Academic Press, 1984. [The leading reference on numerical integration, with detailed information on many topics, first published in 1975.]

D. M. Day and L. Romero, Roots of polynomials expressed in terms of orthogonal polynomials, *SIAM J. Numer. Anal.* 43 (2005), 1969–1987. [A rediscovery of the results of Specht, Good, Barnett and others on colleague and comrade matrices.]

C. de Boor and A. Pinkus, Proof of the conjectures of Bernstein and Erdős concerning the optimal nodes for polynomial interpolation, *J. Approx. Th.* 24 (1978), 289–303. [Together with Kilgore 1978, one of the papers solving the theoretical problem of optimal interpolation.]

R. A. DeVore and G. G. Lorentz, *Constructive Approximation,* Springer, 1993. [An monograph emphasizing advanced topics.]

Z. Ditzian and V. Totik, *Moduli of Smoothness*, Springer-Verlag, New York, 1987. [Careful analysis of smoothness and its effect on polynomial approximation on an interval, including the dependence on location in the interval.]

T. A. Driscoll, F. Bornemann and L. N. Trefethen, The chebop system for automatic solution of differential equations, *BIT Numer. Math.* 48 (2008), 701–723. [Extension of Chebfun to solve differential and integral equations.]

T. A. Driscoll and N. Hale, Resampling methods for boundary conditions in spectral collocation, paper in preparation, 2012. [Introduction of spectral collocation methods based on rectangular matrices.]

M. Dupuy, Le calcul numérique des fonctions par l'interpolation barycentrique, *Compt. Rend. Acad. Sci.* 226 (1948), 158–159. [This paper is apparently the first to use the expression "barycentric interpolation" and also the first to discuss barycentric interpolation for non-equidstant points, the situation considered by Taylor 1945.]

A Dutt, M. Gu and V. Rokhlin, Fast algorithms for polynomial interpolation, integration, and differentiation, *SIAM J. Numer. Anal.* 33 (1996), 1689–1711. [Uses the Fast Multipole Method to derive fast algorithms for non-Chebyshev points.]

H. Ehlich and K. Zeller, Auswertung der Normen von Interpolationsoperatoren, *Math. Ann.* 164 (1966), 105–112. [Bound on Lebesgue constant for interpolation in Chebyshev points.]

D. Elliott, A direct method for "almost" best uniform approximation, in *Error, Approximation, and Accuracy,* eds. F. de Hoog and C. Jarvis, U. Queensland Press, St. Lucia, Queensland, 1973, 129–143. [A precursor to the Carathéodory–Fejér method.]

M. Embree and D. Sorensen, *An Introduction to Model Reduction for Linear and Nonlinear Differential Equations,* to appear. [Textbook.]

B. Engquist and A. Majda, Absorbing boundary conditions for the numerical simulation of waves, *Math. Comput.* 31 (1977), 629–651. [Highly influential paper on the use of Padé approximations to a pseudodifferential operator to develop numerical boundary conditions.]

P. Erdős, Problems and results on the theory of interpolation. II, *Acta Math. Acad. Sci. Hungar.* 12 (1961), 235–244. [Shows that Lebesgue constants for optimal interpolation points are no better than for Chebyshev points asymptotically as $n \to \infty$.]

T. O. Espelid, Extended doubly adaptive quadrature routines, Tech. Rep. 266, Dept. Informatics, U. Bergen, Feb. 2004. [Presentation of `coteda` and `da2glob` quadrature codes.]

L. Euler, *De Seriebus Divergentibus,* Novi Commentarii academiae scientiarum Petropolitanae 5, (1760) (205). [Early work on divergent series.]

L. Euler, De eximio usu methodi interpolationum in serierum doctrina, *Opuscula Analytica* 1 (1783), 157–210. [A work on various applications of interpolation, including equations related to the Newton and Lagrange formulas for polynomial interpolation.]

L. C. Evans and R. F. Gariepy, *Measure Theory and Fine Properties of Functions*, CRC Press, 1991. [Includes a definition of the total variation in the measure theoretic context.]

G. Faber, Über die interpolatorische Darstellung stetiger Funktionen, *Jahresber. Deutsch. Math. Verein.* 23 (1914), 190–210. [Shows that no fixed system of nodes for polynomial interpolation will lead to convergence for all continuous f.]

L. Fejér, Sur les fonctions bornées et intégrables, *Compt. Rend. Acad. Sci.* 131 (1900), 984–987. [Fejér, age 20, provides a new method of summing divergent Fourier series, with a new proof of the Weierstrass approximaton theorem as a corollary.] CHECK

L. Fejér, Lebesguesche Konstanten und divergente Fourierreihen, *J. f. Math.* 138 (1910), 22–53. [Shows that Lebesgue constants for Fourier projection are asymptotic to $(4/\pi^2)\log n$ as $n \to \infty$.]

L. Fejér, Ueber Interpolation, *Nachr. Gesell. Wiss. Göttingen Math. Phys. Kl.* (1916), 66–91. [Proves the Weierstrass approximation theorem by showing that Hermite–Fejér interpolants in Chebyshev points of the first kind converge for any $f \in C([-1,1])$.]

A. M. Finkelshtein, Equilibrium problems of potential theory in the complex plane, in *Orthogonal Polynomials and Special Functions*, Lect. Notes Math. 1883, pp. 79–117, Springer, 2006. [Survey article.]

M. S. Floater and K. Hormann, Barycentric rational interpolation with no poles and high rates of approximation, *Numer. Math.* 107 (2007), 315–331. [Extension of results of Berrut 1988 to a family of barycentric rational interpolants of arbitrary order.]

G. B. Folland, *Introduction to Partial Differential Equations*, 2nd ed., Princeton U. Press, 1995. [An elegant introduction to PDEs published first in 1976, including the Weierstrass approximation theorem proved via the heat equation and generalized to multiple dimensions.]

B. Fornberg, Generation of finite difference formulas on arbitrarily spaced grids, *Math. Comp.* 51 (1988), 699–706. [Stable algorithm for generating finite difference formulas on arbitrary grids.]

B. Fornberg, *A Practical Guide to Pseudospectral Methods,* Cambridge U. Press, 1996. [Practically-oriented textbook of spectral collocation methods for solving ordinary and partial differential equations, based on Chebyshev interpolants.]

S. Fortune, Polynomial root finding using iterated eigenvalue computation, *Proc. 2001 Intl. Symp. Symb. Alg. Comput.*, ACM, 2001, pp. 121–128. [An eigenvalue-based rootfinding algorithm that works directly from data samples rather than expansion coefficients.]

L. Fox and I. B. Parker, *Chebyshev Polynomials in Numerical Analysis,* Oxford U. Press, 1968. [A precursor to the work of the 1970s and later on Chebyshev spectral methods.]

J. G. F. Francis, The QR transformation: a unitary analogue to the LR transformation, parts I and II, *Computer J.* 4 (1961), 256–272 and 332–345. [Introduction of the QR algorithm for numerical computation of matrix eigenvalues.]

G. Frobenius, Ueber Relationen zwischen den Näherungsbrüchen von Potenzreihen, *J. Reine Angew. Math.* 90 (1881), 1–17. [The first systematic treatment of

Padé approximation.]

M. Froissart, Approximation de Padé: application à la physique des particules élémentaires, *RCP, Programme No. 25*, v. 9, CNRS, Strasbourg (1969), pp. 1–13. [A rare publication by the mathematician and physicist after whom Froissart doublets were named (by Bessis).]

D. Gaier, *Lectures on Complex Approximation*, Birkhäuser, 1987. [A shorter book presenting some of the material considered at greater length in Smirnov & Lebedev 1968 and Walsh 1969.]

C. F. Gauss, Methodus nova integralium valores per approximationem inveniendi, *Comment. Soc. Reg. Scient. Gotting. Recent.*, 1814, pp. 39–76. [Introduction of Gauss quadrature—via continued fractions, not orthogonal polynomials.]

W. Gautschi, A survey of Gauss–Christoffel quadrature formulae, in P. L. Butzer and F. Fehér, eds., *E. B. Christoffel: The Influence of His Work in Mathematics and the Physical Sciences,* Birkhäuser, 1981, pp. 72–147. [Outstanding survey of many aspects of Gauss quadrature.]

W. Gautschi, *Orthogonal Polynomials: Computation and Approximation*, Oxford U. Press, 2004. [A monograph on orthogonal polynomials with emphasis on numerical aspects.]

K. O. Geddes, Near-minimax polynomial approximation in an elliptical region, *SIAM J. Numer. Anal.* 15 (1978), 1225–1233. [Chebyshev expansions via FFT for analytic functions on an interval.]

W. M. Gentleman (1972a), Implementing Clenshaw–Curtis quadrature, I: Methodology and experience, *Comm. ACM* 15 (1972), 337–342. [A surprisingly modern paper that includes the aliasing formula for Chebyshev polynomials.]

W. M. Gentleman (1972b), Implementing Clenshaw–Curtis quadrature, II: Computing the cosine transformation, *Comm. ACM* 15 (1972), 343–346. [First connection of Clenshaw–Curtis quadrature with FFT.]

A. Glaser, X. Liu and V. Rokhlin, A fast algorithm for the calculation of the roots of special functions, *SIAM J. Sci. Comp.* 29 (2007), 1420–1438. [Introduction of an algorithm for computation of Gauss quadrature nodes and weights in $O(n)$ operations rather than $O(n^2)$ as in Golub & Welsch 1969.]

K. Glover, All optimal Hankel-norm approximations of linear multivariable systems and their L^∞-error bounds, *Internat. J. Control* 39 (1984), 1115–1193. [Highly influential article on rational approximations in control theory.]

S. Goedecker, Remark on algorithms to find roots of polynomials, *SIAM J. Sci. Comput.* 15 (1994), 1059–1063. [Emphasizes the stability of companion matrix eigenvalues as an algorithm for polynomial rootfinding, given a polynomial expressed by its coefficients in the monomial basis.]

G. H. Golub and J. H. Welsch, Calculation of Gauss quadrature rules, *Math. Comp.* 23 (1969), 221–230. [Presentation of the famous $O(n^2)$ algorithm for Gauss quadrature nodes and weights via a tridiagonal Jacobi matrix eigenvalue problem.]

A. A. Gonchar and E. A. Rakhmanov, Equilibrium distributions and degree of rational approximation of analytic functions, *Math. USSR Sbornik* 62 (1989), 305–348. [A landmark paper, first published in Russian in 1987, that applies methods of potential theory to prove that the optimal rate of convergence for type (n,n) rational minimax approximations of e^x on $(-\infty,0]$ is $O((9.28903\ldots)^{-n})$ as $n \to \infty$.]

V. L. Goncharov, The theory of best approximation of functions, *J. Approx. Th.* 106 (2000), 2–57. [English translation of a 1945 historical survey in Russian emphasizing contributions of Chebyshev and his successors.]

P. Gonnet, S. Güttel and L. N. Trefethen, Robust Padé approximation via SVD, *SIAM Rev.*, to appear. [Introduction of the robust SVD-based algorithm for computing Padé approximants presented in Chapter 27.]

P. Gonnet, R. Pachón and L. N. Trefethen, Robust rational interpolation and least-squares, *Elect. Trans. Numer. Anal.* 38 (2011), 146–167. [A robust algorithm based on the singular value decomposition for computing rational approximants without spurious poles.]

I. J. Good, The colleague matrix, a Chebyshev analogue of the companion matrix, *Quart. J. Math.* 12 (1961), 61–68. [Together with Specht 1960, one of the two original independent discoveries that roots of polynomials in Chebyshev form can be computed as eigenvalues of colleague matrices, a term introduced here. Good recommends this approach to numerical rootfinding for functions other than polynomials too.]

D. Gottlieb, M. Y. Hussaini and S. A. Orszag, Introduction: theory and applications of spectral methods, in R. G. Voigt, D. Gottlieb and M. Y. Hussaini, *Spectral Methods for Partial Differential Equations*, SIAM, 1984. [Early survey article on spectral collocation methods, including the first publication of the formula for the entries of Chebyshev differentiation matrices.]

W. B. Gragg, The Padé table and its relation to certain algorithms of numerical analysis, *SIAM Rev.* 14 (1972), 1–62. [A careful and extensive mathematical reference on the structure and algebra of the Padé table as presented in Chapter 27, though with an emphasis on determinants.]

A. Greenbaum and L. N. Trefethen, GMRES/CR and Arnoldi/Lanczos as matrix approximation problems, *SIAM J. Sci. Comput.* 15 (1994), 359–368. [Shows that the GMRES/CR and Arnoldi/Lanczos matrix iterations are equivalent to certain polynomial approximation problems and generalizes this observation to matrix approximation problems such as "ideal GMRES".]

T. H. Gronwall, Über die Gibbssche Erscheinung und die trigonometrischen Summen $\sin x + \frac{1}{2}\sin 2x + \cdots + \frac{1}{n}\sin nx$, *Math. Ann.* 72 (1912), 228–243. [Investigates detailed behavior of Fourier approximations near Gibbs discontinuities.]

M. H. Gutknecht, Algebraically solvable Chebyshev approximation problems, in C. K. Chui, L. L. Schumaker and J. D. Ward., eds., *Approximation Theory IV*, Academic Press, 1983. [Shows that many examples of ∞-norm best approximations that can be written down explicitly correspond to Carathéodory–Fejér approximations.]

M. H. Gutknecht, In what sense is the rational interpolation problem well posed?, *Constr. Approx.* 6 (1990), 437–450. [Generalization of Trefethen & Gutknecht 1985 from Padé to multipoint Padé approximation.]

M. H. Gutknecht and L. N. Trefethen, Real polynomial Chebyshev approximation by the Carathéodory–Fejér method, *SIAM J. Numer. Anal.* 19 (1982), 358–371. [Introduction of CF approximation on an interval.]

S. Güttel, *Rational Krylov Methods for Operator Functions*, PhD dissertation, TU Bergakademie Freiberg, 2010. [Survey and analysis of advanced methods of numerical linear algebra based on rational approximations.]

N. Hale, N. J. Higham and L. N. Trefethen, Computing A^α, $\log(A)$, and related matrix functions by contour integrals, *SIAM J. Numer. Math.* 46 (2008), 2505–2523. [Derives efficient algorithms for computing matrix functions from trapezoid rule approximations to contour integrals accelerated by contour maps. These are equivalent to rational approximations.]

N. Hale and T. W. Tee, Conformal maps to multiply slit domains and applications, *SIAM J. Sci. Comput.* 31 (2009), 3195–3215. [Extension of Tee & Trefethen 2006 to new geometries and applications.]

N. Hale and A. Townsend, Fast and accurate computation of Gauss–Jacobi quadrature nodes and weights, manuscript in preparation, 2012. [Proposes an $O(n)$ algorithm based on asymptotic formulas for computing Gauss quadrature nodes and weights for large n, much faster than the Glaser–Liu–Rokhlin algorithm in a Matlab implementation.]

N. Hale and L. N. Trefethen, New quadrature formulas from conformal maps, *SIAM J. Numer. Anal.* 46 (2008), 930–948. [Shows that conformal mapping can be used to derive quadrature formulas that converge faster than Gauss, as in Bakhvalov 1967.]

N. Hale and L. N. Trefethen, Chebfun and numerical quadrature, *Science in China,* to appear, 2012. [Review of quadrature algorithms in Chebfun, including fast Gauss and Gauss–Legendre quadrature by the Glaser–Liu–Rokhlin algorithm (but not yet the Hale–Townsend algorithm) with applications to computing with functions with singularities.]

L. Halpern and L. N. Trefethen, Wide-angle one-way wave equations, *J. Acoust. Soc. Amer.* 84 (1988), 1397–1404. [Review of rational approximations to $\sqrt{1-s^2}$ on $[-1,1]$ for application to one-way wave equations.]

G. H. Halphen, Traité des fonctions elliptiques et de leurs applications, Gauthier-Villars, Paris, 1886. [A treatise on elliptic functions that contains a calculation to six digits of the number $\approx 1/9.28903$ that later became known as "Halphen's constant" in connection with the rational approximation of e^x on $(-\infty,0]$.]

P. C. Hansen, *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion,* SIAM, 1998. [A leading monograph on the treatment of rank-deficient or ill-posed matrix problems.]

G. H. Hardy, *Divergent Series,*, revised ed., Éditions Jacques Gabay, 1991. [Hardy's marvelous posthumous volume on the mathematics of divergent series, first published in 1949.]

J. F. Hart et al., *Computer Approximations*, Wiley, 1968. [A classic compendium on computer evaluation of special functions, containing 150 pages of explicit coefficients of rational approximations.]

E. Hayashi, L. N. Trefethen and M. H. Gutknecht, The CF table, *Constr. Approx.* 6 (1990), 195–223. [The most systematic and detailed treatment of the problem of rational CF approximation of a function f on the unit disk, including cases where f is just in the Wiener class or continuous on the unit circle.]

G. Heinig and K. Rost, *Algebraic Methods for Toeplitz-like Matrices and Operators,* Birkhäuser, 1984. [Analyzes rank properties of Toeplitz and Hankel matrices related to the robust Padé algorithms of Chapter 27.]

G. Helmberg and P. Wagner, Manipulating Gibbs' phenomenon for Fourier interpolation, *J. Approx. Th.* 89 (1997), 308–320. [Analyzes the overshoot in various versions of the Gibbs phenomenon for trigonometric interpolation.]

P. Henrici, *Applied and Computational Complex Analysis, vols. 1–3*, Wiley, 1974 and 1977 and 1986. [An extensive and highly readable account of applied complex analysis, full of details that are hard to find elsewhere.]

C. Hermite, Sur la formule d'interpolation de Lagrange, *J. Reine Angew. Math.* 84 (1878), 70–79. [Application of what became known as the Hermite integral formula for polynomial interpolation, which had earlier been given by Cauchy, to problems of interpolation with confluent data points.]

J. S. Hestaven, S. Gottlieb and D. Gottlieb, *Spectral Methods for Time-Dependent Problems,* Cambridge U. Press, 2007. [Well-known textbook on spectral methods.]

E. Hewitt and R. E. Hewitt, The Gibbs–Wilbraham phenomenon: an episode in Fourier analysis, *Arch. Hist. Exact Sci.* 21 (1979), 129–160. [Discussion of the complex and not always pretty history of attempts to analyze the Gibbs phenomenon.]

N. J. Higham, The numerical stability of barycentric Lagrange interpolation, *IMA J. Numer. Anal.* 24 (2004), 547–556. [Proves that barycentric interpolation in Chebyshev points is numerically stable, following earlier work of Rack & Reimer 1982.]

N. J. Higham, *Functions of Matrices: Theory and Computation,* SIAM, 2008. [The definitive treatment of the problem of computing functions of matrices as of 2008. Many of the algorithms have connections with polynomial or rational approximation.]

N. J. Higham, The scaling and squaring method for the matrix exponential revisited, *SIAM Rev.* 51 (2009), 747–764. [Careful analysis of Matlab's method of evaluating e^A leads to several improvements in the algorithm and the recommendation to use the Padé approximant of type $(13, 13)$.]

N. J. Higham and A. H. Al-Mohy, Computing matrix functions, *Acta Numer.* 19 (2010), 159–208. [Survey includes an appendix comparing Padé and Taylor approximants for computing the exponential of a matrix.]

E. Hille, *Analytic Function Theory*, 2 vols., 2nd ed., Chelsea, 1973. [Major work first published in 1959 and 1962.]

M. Hochbruck and A. Ostermann, Exponential integrators, *Acta Numer.* 19 (2010), 209–286. [Survey of exponential integrators for the fast numerical solution of stiff ODEs and PDEs.]

G. Hornecker, Détermination des meilleures approximations rationnelles (au sens de Tchebychef) des functions réelles d'une variable sur un segment fini et des bornes d'erreur correspondantes, *Compt. Rend. Acad. Sci.* 249 (1956), 2265–2267. [Possibly the first proposal of a kind of Chebyshev–Padé approximation for intervals.]

J. P. Imhof, On the method for numerical integration of Clenshaw and Curtis, *Numer. Math.* 5 (1963), 138–141. [Shows that the Clenshaw–Curtis quadrature weights are positive.] *check*

A. Iserles, A fast and simple algorithm for the computation of Legendre coefficients, *Numer. Math.* 117 (2011), 529–553. [A fast algorithm based on a numerical contour integral over an ellipse in the complex plane.]

D. Jackson, *Über die Genauigkeit der Annäherung stetiger Funktionen durch ganze rationale Funktionen gegebenen Grades und trigonometrische Summen gegebener Ordnung,* dissertation, Göttingen, 1911. [Jackson's PhD thesis under Landau in Göttingen, which together with Bernstein's work at the same time (1912b) established many of the fundamental results of approximation theory. Despite the German, Jackson was an American from Massachusetts, like me—Harvard Class of 1908.]

D. Jackson, On the accuracy of trigonometric interpolation, *Trans. Amer. Math. Soc.* 14 (1913), 453–461. [In the final paragraph of this paper, polynomial interpolation in Chebyshev points (2.2) is considered, possibly for the first time in the literature.] *check*

C. G. J. Jacobi, *Disquisitiones Analyticae de Fractionibus Simplicibus*, dissertation, Berlin, 1825. [In his discussion of partial fractions Jacobi effectively states the "first form" of the barycentric interpolation formula.]

C. G. J. Jacobi, Über Gauss' neue Methode, die Werthe der Integrale näherungsweise zu finden, *J. Reine Angew. Math.* 1 (1826), 301–308. [This paper first invents the subject of orthogonal polynomials, then shows that Gauss quadrature can be derived in this framework.]

C. G. J. Jacobi, Über die Darstellung einer Reihe gegebener Werthe durch eine gebrochene rationale Function, *J. Reine Angew. Math.* 30 (1846), 127–156. [Jacobi's major work on rational interpolation.]

R. Jentzsch, *Untersuchungen zur Theorie analytischer Funktionen,* dissertation, Berlin, 1914. [Jentzsch, who was also a noted poet and was killed at age 27 in World War I, proves here that every point on the circle of convergence of a power series is the limit of zeros of its partial sums.]

D. C. Joyce, Survey of extrapolation processes in numerical analysis, *SIAM Rev.* 13 (1971), 435–490. [Scholarly review of a wide range of material.]

A.-K. Kassam and L. N. Trefethen, Fourth-order time-stepping for stiff PDEs, *SIAM J. Sci. Comput.* 26 (2005), 1214–1233. [Application of exponential integrator formulas to efficient numerical solution of stiff PDEs.]

T. A. Kilgore, A characterization of the Lagrange interpolating projection with minimal Tchebycheff norm, *J. Approx. Th.* 24 (1978), 273–288. [Together with de Boor & Pinkus 1978, one of the papers solving the theoretical problem of optimal interpolation.]

P. Kirchberger, *Ueber Tchebychefsche Annäherungsmethoden,* PhD thesis, Göttingen, 1902. [Kirchberger's PhD thesis under Hilbert contains apparently the first full statement and proof of the equioscillation theorem.]

P. Kirchberger, Über Tchebychefsche Annäherungsmethoden, *Math. Ann.* 57 (1903), 509–540. [Extract from his PhD thesis the year before, focusing on multivariable extensions but without the equioscillation theorem.]

A. N. Kolmogorov, A remark on the polynomials of P. L. Chebyshev deviating the least from a given function, *Uspehi Mat. Nauk* 3 (1948), 216–221 [Russian]. [Criterion for best complex approximations.]

D. Kosloff and H. Tal-Ezer, A modified Chebyshev pseudospectral method with an $O(N^{-1})$ time step restriction, *J. Comp. Phys.* 104 (1993), 457–469. [Introduces a change of variables as a basis for non-polynomial spectral methods.]

A. B. J. Kuijlaars, Convergence analysis of Krylov subspace iterations with methods from potential theory, *SIAM Rev.* 48 (2006), 3–40. [Analyzes the connection between potential theory and the roots of polynomial approximants implicitly constructed by Krylov iterations such as the conjugate gradient and Lanczos iterations.]

J. L. Lagrange, Leçons élémentaires sur les Mathématiques, Leçon V., *J. de l'École polytechnique,* Tome II, Cahier 8, pp. 274–278, Paris, 1795. [Contains what became known as the Lagrange interpolation formula, published earlier by Waring 1779 and Euler 1783.]

B. Lam, *Some Exact and Asymptotic Results for Best Uniform Approximation*, PhD thesis, U. of Tasmania, 1972. [A precursor to the Carathéodory–Fejér method.]

E. Landau, Abschätzung der Koeffizientensumme einer Potenzreihe, *Archiv Math. Phys.* 21 (1913), 42–50 and 250–255. [Investigates the norm of the degree n Taylor projection for functions analytic in the unit disk, now known as the Landau constant, showing it is asymptotic to $\pi^{-1}\log n$ as $n \to \infty$.]

H. Lebesgue, Sur l'approximation des fonctions, *Bull. Sci. Math.* 22 (1898), 278–287. [In Lebesgue's first published paper, he proves the Weierstrass approximation theorem by approximating $|x|$ by polynomials and noting that any continuous function can be approximated by piecewise linear functions.]

A. L. Levin and E. B. Saff, Potential theoretic tools in polynomial and rational approximation, in J.-D. Fournier, et al., eds., *Harmonic Analysis and Rational Approximation,* Lec. Notes Control Inf. Sci. 326/2006 (2006), 71–94. [Survey article by two of the experts.]

R.-C. Li, Near optimality of Chebyshev interpolation for elementary function computations, *IEEE Trans. Computers* 53 (2004), 678–687. [Shows that although Lebesgue constants for Chebyshev points grow logarithmically as $n \to \infty$, for many classes of functions of interest the interpolants come within a factor of 2 of optimality.]

E. L. Lindman, 'Free-space' boundary conditions for the time-dependent wave equation, *J. Comput. Phys.* 18 (1975), 66–78. [Absorbing boundary conditions based on Padé approximation of a square root function, later developed further by Engquist & Majda 1977].

G. G. Lorentz, *Approximation of Functions*, 2nd ed., Chelsea, 1986. [A readable treatment including good summaries of the Jackson theorems for polynomial and trigonometric approximation, first published in 1966.]

K. N. Lungu, Best approximations by rational functions, *Math. Notes* 10 (1971), 431–433. [Shows that the best rational approximations to a real function on an interval may be complex and hence also nonunique, with examples as simple as type $(1, 1)$ approximation of $|x|$ on $[-1, 1]$.]

H. J. Maehly and Ch. Witzgall, Tschebyscheff-Approximationen in kleinen Intervallen. II. Stetikeitssätze für gebrochen rationale Approximationen, *Numer. Math.* 2 (1960), 293–307. [Investigates well-posedness of the Cauchy interpolation problem and asymptotics of best approximations on small intervals.]

A. P. Magnus, CFGT determination of Varga's constant '1/9', unpublished manuscript, 1985. [First identification of the the exact value of Halphen's constant $C = 9.28903\ldots$ for the optimal rate of convergence $O(C^{-n})$ of best type (n, n) approximations to e^x on $(-\infty, 0]$, later proved correct by Gonchar & Rakhmanov 1989.]

A. P. Magnus and J. Meinguet, The elliptic functions and integrals of the "1/9" problem, *Numer. Alg.* 24 (2000), 117–139. [Summary of work initiated by Magnus relating potential theory, elliptic functions, and the "1/9" problem.]

J. Marcinkiewicz, Quelques remarques sur l'interpolation, *Acta Sci. Math. (Szeged)* 8 (1936–37), 127–30. [In contrast to the result of Faber 1914, shows that for any fixed continuous function f there is an array of interpolation nodes that leads to convergence as $n \to \infty$.]

A. I. Markushevich, *Theory of Functions of a Complex Variable*, 2nd ed., 3 vols., Chelsea, 1985. [A highly readable treatise on complex variables first published in

1965, including chapters on Laurent series, polynomial interpolation, harmonic functions, and rational approximation.]

J. C. Mason and D. C. Handscomb, *Chebyshev Polynomials,* Chapman and Hall/CRC, 2003. [An extensive treatment of four varieties of Chebyshev polynomials and their applications.]

G. Mastroianni and M. G. Russo, Some new results on Lagrange interpolation for bounded variation functions, *J. Approx. Th.* 162 (2010), 1417–1428. [A collection of bounds in L^p norms for both $p < \infty$ and $p = \infty$.]

G. Mastroianni and J. Szabados, Jackson order of approximation by Lagrange interpolation. II, *Acta Math. Acad. Sci. Hungar.* 69 (1995), 73–82. [Corollary 2 bounds the rate of convergence of Chebyshev interpolants for functions whose kth derivative has bounded variation.]

J. H. McCabe and G. M. Phillips, On a certain class of Lebesgue constants, *BIT* 13 (1973), 434–442. [Shows that the Lebesgue constant for polynomial interpolation in $n+1$ Chebyshev points of the second kind is bounded by that of n Chebyshev points of the first kind. The same result had been found earlier by Ehlich & Zeller 1966.]

J. H. McClellan and T. W. Parks, A personal history of the Parks–McClellan algorithm, *IEEE Sign. Proc. Mag.* 82 (2005), 82–86. [The story of the development of the celebrated filter design algorithm published in Parks & McClellan 1972.]

G. Meinardus, *Approximation of Functions: Theory and Numerical Methods,* Springer, 1967. [Classic approximation theory monograph.]

C. Méray, Observations sur la légitimité de l'interpolation, *Annal. Scient. de l'École Normale Supérieure* 3 (1884), 165–176. [Discussion of the possibility of nonconvergence of polynomial interpolants 17 years before Runge, though without so striking an example or conclusion. Méray uses just the right technique, the Hermite integral formula, which he correctly attributes to Cauchy.]

C. Méray, Nouveaux exemples d'interpolations illusoires, *Bull. Sci. Math.* 20 (1896), 266–270. [Continuation of Méray 1884 with more examples.]

S. N. Mergelyan, On the representation of functions by series of polynomials on closed sets (Russian). *Dokl. Adak. Nauk SSSR (N. S.)* 78 (1951), 405–408. Translation: *Translations Amer. Math. Soc.* 3 (1962), 287–293. [Famous theorem asserting that a function continuous on a compact set in the complex plane whose complement is connected, and analytic in the interior, can be uniformly approximated by polynomials.]

H. N. Mhaskar and D. V. Pai, *Fundamentals of Approximation Theory,* CRC/Narosa, 2000. [Extensive treatment of many topics, especially in linear approximation.]

G. Mittag-Leffler, Sur la représentation analytique des fonctions d'une variable réelle, *Rend. Circ. Mat. Palermo* (1900), 217–224. [Contains a long footnote by Phragmén explaining how the Weierstrass approximation theorem follows from the work of Runge.]

C. Moler and C. Van Loan, Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later, *SIAM Rev.* 45 (2003), 3–49. [Expanded reprinting of 1978 paper summarizing methods for computing $\exp(A)$, the best method being related to Padé approximation.]

R. de Montessus de Ballore, Sur les fractions continues algébriques, *Bull. Soc. Math. France* 30 (1902), 28–36. [Shows that type (m,n) Padé approximants to meromorphic functions converge pointwise as $m \to \infty$ in a disk about $z = 0$ with exactly n poles.]

M. Mori and M. Sugihara, The double-exponential transformation in numerical analysis, *J. Comput. Appl. Math.* 127 (2001), 287–296. [Survey of a the quadrature algorithm introduced by Takahasi & Mori 1974]

J.-M. Muller, *Elementary Functions: Algorithms and Implementation*, 2nd ed., Birkhäuser, 2006. [A text on implementation of elementary functions on computers, including a chapter on the Remez algorithm.]

Y. Nakatsukasa, Z. Bai and F. Gygi, Optimizing Halley's iteration for computing the matrix polar decomposition, *SIAM J. Matrix Anal. Appl.* 31 (2010), 2700–2720. [Introduction of an algorithm based on a rational function of high degree generated by iteration of a simple equiripple approximation.] *check*

I. P. Natanson, *Constructive Theory of Functions,* 3 vols., Frederick Ungar, 1964 and 1965. [This major work by a scholar in Leningrad gives equal emphasis to algebraic and trigonometric approximation.]

D. J. Newman, Rational approximation to $|x|$, *Mich. Math. J.* 11 (1964), 11–14. [Shows that whereas polynomial approximants to $|x|$ on $[-1,1]$ converge at the rate $O(n^{-1})$, for rational approximants the rate is $O(\exp(-C\sqrt{n}))$.]

D. J. Newman, Rational approximation to e^{-x}, *J. Approx. Th.* 10 (1974), 301–303. [Shows by a lower bound 1280^{-n} that type (n,n) rational approximants to e^{x} on $(-\infty, 0]$ can converge no faster than geometrically as $n \to \infty$ in the supremum norm.]

J. Nuttall, The convergence of Padé approximants of meromorphic functions, *J. Math. Anal. Appl.* 31 (1970), 147–153. [Shows that type (n,n) Padé approximants to meromorphic functions converge in measure as $n \to \infty$, though not pointwise.]

H. O'Hara and F. J. Smith, Error estimation in the Clenshaw–Curtis quadrature formula, *Comput. J.* 11 (1968), 213–219. [Early paper arguing that Clenshaw–Curtis and Gauss quadrature have comparable accuracy in practice.]

A. V. Oppenheim, R. W. Schafer and J. R. Buck, *Discrete-time Signal Processing*, Prentice Hall, 1999. [A standard textbook on the subject, which is tightly connected with polynomial and rational approximation.]

S. A. Orszag (1971a), Galerkin approximations to flows within slabs, spheres, and cylinders, *Phys. Rev. Lett.* 26 (1971), 1100–1103. [Orszag's first publication on Chebyshev spectral methods.]

S. A. Orszag (1971b), Accurate solution of the Orr–Sommerfeld stability equation, *J. Fluid Mech.* 50 (1971), 689–703. [The most influential of Orszag's early papers on Chebyshev spectral methods.]

R. Pachón, *Algorithms for Polynomial and Rational Approximation in the Complex Domain,* DPhil thesis, U. of Oxford, 2010. [Includes chapters on rational best approximants, interpolants, and Chebyshev–Padé approximants and their application to exploration of functions in the complex plane.]

R. Pachón, P. Gonnet and J. Van Deun, Fast and stable rational interpolation in roots of unity and Chebyshev points, *SIAM J. Numer. Anal.*, to appear. [Linear algebra formulation of the rational interpolation problem in a manner closely suited to computation.]

R. Pachón, R. B. Platte and L. N. Trefethen, Piecewise-smooth chebfuns, *IMA J. Numer. Anal.* 30 (2010), 898–916. [Generalization of Chebfun from single to multiple polynomial pieces, including edge detection algorithm to determine breakpoints.]

R. Pachón and L. N. Trefethen, Barycentric-Remez algorithms for best polynomial approximation in the chebfun system, *BIT Numer. Math.* 49 (2009), 721–741. [Chebfun implementation of Remez algorithm for computing polynomial best approximations.]

H. Padé, Sur la représentation approchée d'une fonction par des fractions rationelles, *Annales Sci. de l'École Norm. Sup.* 9 (1892) (supplément), 3–93. [The first of many publications by Padé on the subject that became known as Padé approximation, with discussion of defect and block structure including a number of explicit examples.]

T. W. Parks and J. H. McClellan, Chebyshev approximation for nonrecursive digital filters with linear phase, *IEEE Trans. Circuit Theory* CT-19 (1972), 189–194. [Proposes what became known as the Parks–McClellan algorithm for digital filter design, based on a barycentric formulation of the Remez algorithm for best approximation by trigonometric polynomials.]

B. N. Parlett and C. Reinsch, Handbook series linear algebra: balancing a matrix for calculation of eigenvalues and eigenvectors, *Numer. Math.* 13 (1969), 293–304. [Introduction of the technique of balancing a matrix by a diagonal similarity transformation that is crucial to the success of the QR algorithm.]

K. Pearson, *On the Construction of Tables and on Interpolation I. Uni-variate Tables,* Cambridge U. Press, 1920. [Contains as an appendix a fascinating annotated bibliography of 50 early contributions to interpolation. Pearson's annotations are not always as polite as my own, with comments like "Not very adequate" and "A useful, but somewhat disappointing book."]

O. Perron, *Die Lehre von den Kettenbrüchen,* 2nd ed., Teubner, 1929. [This classic monograph on continued fractions, first published in 1913, was perhaps the first to identify the problem of spurious poles or Froissart doublets in Padé approximation. At the end of §78 a function is constructed whose type $(m, 1)$ Padé approximants have poles appearing infinitely often on a dense set of points in the complex plane.]

P. P. Petrushev and V. A. Popov, *Rational Approximation of Real Functions,* Cambridge U. Press, 1987. [Detailed presentation of a great range of results known up to 1987.]

R. Piessens, Algorithm 473: Computation of Legendre series coefficients [C6], *Comm. ACM* 17 (1974), 25–25. [$O(n^2)$ algorithm for converting from Chebyshev to Legendre expansions.]

A. Pinkus, Weierstrass and approximation theory, *J. Approx. Th.* 107 (2000), 1–66. [Detailed discussion of Weierstrass's nowhere-differentiable function and of the Weierstrass approximation theorem and its many proofs and generalizations.]

R. B. Platte, L. N. Trefethen and A. B. J. Kuijlaars, Impossibility of fast stable approximation of analytic functions from equispaced samples, *SIAM Rev.* 53 (2011), 308–318. [Shows that any exponentially convergent scheme for approximating analytic functions from equispaced samples in an interval must be exponentially ill-conditioned as $n \to \infty$; thus no approximation scheme can eliminate the Gibbs and Runge phenomena completely.]

G. Pólya, Über die Konvergenz von Quadraturverfahren, *Math. Z.* 37 (1933), 264–286. [Proves that a family of interpolating quadrature rules converges for all continuous integrands if and only if the sums of the absolute values of the weights are uniformly bounded; proves further that Newton–Cotes quadrature approximations do not always converge as $n \to \infty$, even if the integrand is analytic.]

Ch. Pommerenke, Padé approximants and convergence in capacity, *J. Math. Anal. Appl.*, 41 (1973), 775–780. [Sharpens Nuttall's result on convergence of Padé approximants in measure to convergence in capacity.]

J. V. Poncelet, Sur la valeur approchée linéaire et rationelle des radicaux de la forme $\sqrt{a^2+b^2}, \sqrt{a^2-b^2}$ etc., *J. Reine Angew. Math.* 13 (1835), 277–291. [Perhaps the very first discussion of minimax approximation.]

D. Potts, G. Steidl and M. Tasche, Fast algorithms for discrete polynomial transforms, *Math. Comp.* 67 (1998), 1577–1590. [Algorithms for converting between Chebyshev and Legendre expansions.]

M. J. D. Powell, *Approximation Theory and Methods,* Cambridge U. Press, 1981. [Approximation theory text with a computational emphasis, particularly strong on the Remez algorithm and on splines.]

H. A. Priestley, *Introduction to Complex Analysis,* 2nd ed., Oxford U. Press, 2003. [Well known introductory complex analysis textbook first published in 1985.]

I. E. Pritsker and R. S. Varga, The Szegő curve, zero distribution and weighted approximation, *Trans. Amer. Math. Soc.* 349 (1997), 4085–4105. [Analysis of the Szegő curve using methods of potential theory.]

P. Rabinowitz, Rough and ready error estimates in Gaussian integration of analytic functions, *Comm. ACM* 12 (1969), 268–270. [Derives tight bounds on accuracy of Gaussian quadrature by simple arguments.]

H.-J. Rack and M. Reimer, The numerical stability of evaluation schemes for polynomials based on the Lagrange interpolation form, *BIT* 22 (1982), 101–107. [Proof of stability for barycentric polynomial interpolation in well-distributed point sets, later developed further by Higham 2004.]

T. Ransford, *Potential Theory in the Complex Plane,* Cambridge U. Press, 1995. [Perhaps the only book devoted to this subject.]

T. Ransford, Computation of logarithmic capacity, *Comput. Meth. Funct. Th.* 10 (2010), 555–578. [An algorithm for computing capacity of a set in the complex plane, with examples.]

E. Remes, Sur un procédé convergent d'approximations successives pour déterminer les polynômes d'approximation, *Compt. Rend. Acad. Sci.* 198 (1934), 2063–2065. [One of the original papers presenting the Remez algorithm.] *check*

E. Remes, Sur le calcul effectif des polynômes d'approximation de Tchebichef, *Compt. Rend. Acad. Sci.* 199 (1934), 337–340. [The other original paper presenting the Remez algorithm.] *check*

E. Y. Remes, On approximations in the complex domain, *Dokl. Akad. Nauk SSSR* 77 (1951), 965–968 [Russian]. *check*

E. Ya. Remez, General computational methods of Tchebycheff approximation, Atomic Energy Commission Translation 4491, Kiev, 1957, pp. 1–85. *check*

J. R. Rice, *The Approximation of Functions,* Addison-Wesley, 1964 and 1969. [Two volumes, the first linear and the second nonlinear.]

L. F. Richardson, The deferred approach to the limit. I—single lattice. *Phil. Trans. Roy. Soc. A* (1927), 299–349. [Systematic discussion of Richardson extrapolation, emphasizing discretizations with $O(h^2)$ error behavior.]

M. Richardson and L. N. Trefethen, A sinc function analogue of Chebfun, *SIAM J. Sci. Comput.* 33 (2011), 2519–2535. [Presents a "Sincfun" software analogue of Chebfun for dealing with functions with endpoint singularities via variable transformation and sinc function interpolants.]

F. Riesz, Über lineare Funktionalgleichungen, *Acta Math.* 41 (1918), 71–98. [First statement of the general existence result for best approximation from finite-dimensional linear spaces.]

M. Riesz, Über einen Satz des Herrn Serge Bernstein, *Acta. Math.* 40 (1916), 43–47. [Gives a new proof of a Bernstein inequality based on the barycentric formula for Chebyshev points, in the process deriving the barycentric coefficients $(-1)^j$ half a century before Salzer 1972.] *check*

T. J. Rivlin, *An Introduction to the Approximation of Functions,* Dover, 1981. [Appealing short textbook originally published in 1969.]

T. J. Rivlin, *Chebyshev Polynomials: From Approximation Theory to Algebra and Number Theory,* 2nd ed., Wiley, 1990. [Classic book on Chebyshev polynomials and applications, with first edition in 1974.]

J. D. Roberts, Linear model reduction and solution of the algebraic Riccati equation by use of the sign function, *Internat. J. Control* 32 (1980), 677–687. [This article connecting rational functions with the sign function was written in 1971 as Technical Report CUED/B-Control/TR13 of the Cambridge University Engineering Dept.]

W. Rudin, *Principles of Mathematical Analysis,* 3rd ed., McGraw-Hill, 1976. [Influential textbook first published in 1953.]

P. O. Runck, Über Konvergenzfragen bei Polynominterpolation mit äquidistanten Knoten. II, *J. Reine Angew. Math.* 210 (1962), 175–204. [Analyzes the Gibbs overshoot for two varieties of polynomial interpolation of a step function.]

C. Runge (1885a), Zur Theorie der eindeutigen analytischen Functionen, *Acta Math.* 6 (1885), 229–244. [Publication of Runge's theorem: a function analytic on a compact set in the complex plane whose complement is connected can be uniformly approximated by polynomials. This was later generalized by Mergelyan.]

C. Runge (1885b), Über die Darstellung willkürlicher Functionen, *Acta Math.* 7 (1885), 387–392. [Shows that a continuous function on a finite interval can be uniformly approximated by rational functions. It was later noted by Phragmén and Mittag-Leffler that this and the previous paper by Runge together imply the Weierstrass approximation theorem.]

C. Runge, Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten, *Z. Math. Phys.* 46 (1901), 224–243. [Méray had pointed out that polynomial interpolants might fail to converge, but it was this paper that focussed on equispaced sample points, showed that divergence can take place even in the interval of interpolation, and identified the "Runge region" where analyticity is required for convergence.]

A. Ruttan, The length of the alternation set as a factor in determining when a best real rational approximation is also a best complex rational approximation, *J. Approx. Th.* 31 (1981), 230–243. [Shows that complex best approximations are always better than real ones in the strict lower-right triangle of a square block of the Walsh table.]

A. Ruttan and R. S. Varga, A unified theory for real vs. complex rational Chebyshev approximation on an interval, *Trans. Amer. Math. Soc.* 312 (1989), 681–697. [Shows that type $(m, m+2)$ complex rational approximants to real functions can be up to 3 times as accurate as real ones.]

E. B. Saff, An extension of Montessus de Ballore's theorem on the convergence of interpolating rational functions, *J. Approx. Th.* 6 (1972), 63–67. [Generalizes the de Montessus de Ballore theorem from Padé to multipoint Padé approximation.]

E. B. Saff and A. D. Snider, *Fundamentals of Complex Analysis with Applications to Engineering, Science, and Mathematics,* 3rd ed., Prentice Hall, 2003. [Widely used introductory complex analysis textbook.]

E. B. Saff and V. Totik, *Logarithmic Potentials with External Fields,* Springer, 1997. [Presentation of connections between potential theory and rational approximation.]

E. B. Saff and R. S. Varga (1978a), Nonuniqueness of best complex rational approximations to real functions on real intervals, *J. Approx. Th.* 23 (1978), 78–85. [Rediscovery of results of Lungu 1971.]

E. B. Saff and R. S. Varga (1978b), On the zeros and poles of Padé approximants to e^z. III, *Numer. Math.* 30 (1978), 241–266. [Analysis of the curves in the complex plane along which poles and zeros of these approximants cluster.]

T. W. Sag and G. Szekeres, Numerical evaluation of high-dimensional integrals, *Math. Comp.* 18 (1964), 245–253. [Introduction of changes of variables that can speed up Gauss and other quadrature formulas, even in one dimension.]

Salazar Celis, *check*

H. E. Salzer, A simple method for summing certain slowly convergent series, *J. Math. Phys.* 33 (1955), 356–359. ["Salzer's method" for acceleration of convergence, based on interpreting a sequence of values as samples of a function $f(x)$ at $x_n = n^{-1}$.]

H. E. Salzer, Lagrangian interpolation at the Chebyshev points $x_{n,\nu} = \cos(\nu\pi/n)$, $\nu = 0(1)n$; some unnoted advantages, *Computer J.* 15 (1972), 156–159. [Barycentric formula for polynomial interpolation in Chebyshev points.]

H. E. Salzer, Rational interpolation using incomplete barycentric forms, *Z. Angew. Math. Mech.* 61 (1981), 161–164. [One of the first publications to propose the use of rational interpolants defined by barycentric formulas.]

T. Schmelzer and L. N. Trefethen, Evaluating matrix functions for exponential integrators via Carathéodory–Fejér approximation and contour integrals, *Elect. Trans. Numer. Anal.* 29 (2007), 1–18. [Fast methods based on rational approximations for evaluating the φ functions used by exponential integrators for solving stiff ODEs and PDEs.]

J. R. Schmidt, On the numerical solution of linear simultaneous equations by an iterative method, *Philos. Mag.* 32 (1941), 369–383. [Proposal of what became known as the epsilon or eta algorithm some years before Shanks 1955, Wynn 1956, and Bauer 1959.]

C. Schneider and W. Werner, Some new aspects of rational interpolation, *Math. Comp.* 47 (1986), 285–299. [Extension of barycentric formulas to rational interpolation.]

A. Schönhage, Fehlerfortpflanzung bei Interpolation, *Numer. Math.* 3 (1961), 62–71. [Independent rediscovery of results close to those of Turetskii 1940 concerning Lebesgue constants for equispaced points.]

A. Schönhage, Zur rationalen Approximierbarkeit von e^{-x} über $[0,\infty)$, *J. Approx. Th.* 7 (1973), 395–398. [Proves that in maximum-norm approximation of e^x on $(-\infty,0]$ by inverse-polynomials $1/p_n(x)$, the optimal rate is $O(3^{-n})$.]

I. Schur, Über Potenzreihen, die im Innern des Einheitskreises beschränkt sind, *J. Reine Angew. Math.* 148 (1918), 122–145. [Solution of the problem of

Carathéodory and Fejér via the eigenvalue analysis of a Hankel matrix of Taylor coefficients.]

D. Shanks, Non-linear transformations of divergent and slowly convergent sequences, *J. Math. Phys.* 34 (1955), 1–42. [Introduction of Shanks' method for convergence acceleration by Padé approximation, closely related to the epsilon algorithm of Wynn 1956.]

J. Shen, T. Tang and L.-L. Wang, *Spectral Methods: Algorithms, Analysis and Applications,* Springer, 2011. [Systematic presentation of spectral methods including convergence theory.]

B. Shiffman and S. Zelditch, Equilibrium distribution of zeros of random polynomials, *Int. Math. Res. Not.* 2003, no. 1. [Shows that polynomials given by expansions in orthogonal polynomials with random coefficients have roots clustering near the support of the orthogonality measure.] *check*

A. Sidi, *Practical Extrapolation Methods,* Cambridge U. Press, 2003. [Extensive treatment of methods for acceleration of convergence.]

G. A. Sitton, C. S. Burrus, J. W. Fox and S. Treitel, Factoring very-high-degree polynomials, *IEEE Signal Proc. Mag.*, Nov. 2003, 27–42. [Discussion of rootfinding for polynomials of degree up to one million by the Lindsey–Fox algorithm.]

V. I. Smirnov and N. A. Lebedev, *Functions of a Complex Variable: Constructive Theory*, MIT Press, 1968. [Major survey of problems of polynomial and rational approximation in the complex plane.]

F. Smithies, *Cauchy and the Creation of Complex Function Theory*, Cambridge U. Press, 1997. [Detailed account of Cauchy's almost single-handed creation of this field during 1814–1831.]

M. A. Snyder, *Chebyshev Methods in Numerical Approximation,* Prentice Hall, 1966. [An appealing short book emphasizing rational as well as polynomial approximations.]

W. Specht, Die Lage der Nullstellen eines Polynoms. III, *Math. Nachr.* 16 (1957), 369–389. [Development of comrade matrices, whose eigenvalues are roots of polynomials expressed in bases of orthogonal polynomials.]

W. Specht, Die Lage der Nullstellen eines Polynoms. IV, *Math. Nachr.* 21 (1960), 201–222. [The final page considers colleague matrices, the special case of comrade matrices for Chebyshev polynomials. These ideas were developed independently by Good 1961.]

H. Stahl, The convergence of Padé approximants to functions with branch points, *J. Approx. Th.* 91 (1997), 139–204. [Generalizes the Nuttall–Pommerenke theorem on convergence of type (n, n) Padé approximants to the case of functions f with branch points.]

H. Stahl, Spurious poles in Padé approximation, *J. Comp. Appl. Math.* 99 (1998), 511–527. [Defines and analyzes what it means for a pole of a Padé approximant to be spurious.]

H. Stahl, Best uniform rational approximation of $|x|$ on $[-1,1]$, *Russian Acad. Sci. Sb. Math.* 76 (1993), 461–487. [Proof of the conjecture of Varga, Ruttan and Carpenter that best rational approximations to $|x|$ on $[-1,1]$ converge at the rate $\sim 8\exp(-\pi\sqrt{n})$.]

H. R. Stahl, Best uniform rational approximation of x^α on $[0,1]$, *Acta Math.* 190 (2003), 241–306. [Generalization of the results of the paper above to approximation of x^α on $[0,1]$, completing earlier investigations of Ganelius and Vyacheslavov.]

H. Stahl and T. Schmelzer, An extension of the '1/9'-problem, *J. Comput. Appl. Math.* 233 (2009), 821–834. [Announcement of numerous extensions of the "9.28903" result of Gonchar & Rakhmanov 1989 for type (n,n) best approximation of e^x on $(-\infty,0]$, showing that the same rate of approximation applies on compact sets in the complex plane and on Hankel contours, and that "9.28903" is also achieved on $(-\infty,0]$ in type $(n,n+k)$ approximation of e^x or of related functions such as φ functions for exponential integrators.]

K.-G. Steffens, *The History of Approximation Theory: From Euler to Bernstein*, Birkhäuser, 2006. [Discussion of many people and results, especially of the St. Petersburg school, by a student of Natanson.]

E. M. Stein and R. Shakarchi, *Real Analysis: Measure Theory, Integration, and Hilbert Spaces,* Princeton U. Press, 2005. [A leading textbook.]

F. Stenger, Explicit nearly optimal linear rational approximation with preassigned poles, *Math. Comput.* 47 (1986), 225–252. [Construction of rational approximants by a method related to sinc expansions.]

F. Stenger, *Numerical Methods Based on Sinc and Analytic Functions,* Springer, 1993. [Comprehensive treatise by the leader in sinc function algorithms.]

F. Stenger, *Sinc Numerical Methods,* CRC Press, 2010. [A handbook of sinc methods and their implementation in the author's software package Sinc-Pack.]

T. J. Stieltjes (1884a), Note sur quelques formules pour l'évaluation de certaines intégrales, *Bull. Astr. Paris* 1 (1884), 568–569. *check*

T. J. Stieltjes (1884b), Quelques recherches sur la théorie des quadratures dites mécaniques, *Ann. Sci. École Norm. Sup.* 1 (1884), 409–426. [Proves that Gauss quadrature converges for any Riemann integrable integrand.] *check*

T. J. Stieltjes, Sur les polynômes de Jacobi, *Compt. Rend. Acad. Sci.* 199 (1885), 620–622. [Shows that the roots of $(x^2-1)P_{n-1}^{(1,1)}(x)$ are Fekete points (minimal-energy points) in $[-1,1]$.]

J. Szabados, Rational approximation to analytic functions on an inner part of the domain of analyticity, in A. Talbot, ed., *Approximation Theory,* Academic Press, 1970, pp. 165–177. [Shows that for some functions analytic in a Bernstein ρ-ellipse, type (n,n) rational best approximations are essentially no better than degree n polynomial best approximations.]

G. Szegő, Über eine Eigenschaft der Exponentialreihe, *Sitzungsber. Berl. Math.*

Ges. 23 (1924), 50–64. [Shows that as $n \to \infty$, the zeros of the normalized partial sums $s_n(nz)$ of the Taylor series of e^z approach the Szegő curve in the complex z-plane defined by $|ze^{1-z}| = 1$ and $|z| \le 1$.]

G. Szegő, *Orthogonal Polynomials,* Amer. Math. Soc., 1985. [A classic monograph by the master, including chapters on polynomial interpolation and quadrature, first published in 1939.]

E. Tadmor, The exponential accuracy of Fourier and Chebyshev differencing methods, *SIAM J. Numer. Anal.* 23 (1986), 1–10. [Presents theorems on exponential accuracy of Chebyshev interpolants of analytic functions and their derviatives.]

T. Takagi, On an algebraic problem related to an analytic theorem of Carathéodory and Fejér and on an allied theorem of Landau, *Japan J. Math.* 1 (1924), 83–91 and ibid., 2 (1925), 13–17. [Beginnings of the generalization of Carathéodory & Fejér 1911 and Schur 1918 to rational approximation.]

H. Takahasi and M. Mori, Estimation of errors in the numerical quadrature of analytic functions, *Applicable Anal.* 1 (1971), 201–229. [Relates the accuracy of a quadrature formula to the accuracy of an associated rational function as an approximation to $\log((z+1)/(z-1))$ on a contour enclosing $[-1,1]$.]

H. Takahasi and M. Mori, Double exponential formulas for numerical integration, *Publ. RIMS, Kyoto U.* 9 (1974), 721–741. [Introduction of the double exponential or tanh-sinh quadrature rule, in which Gauss quadrature is transformed by a change of variables to another formula that can handle endpoint singularities.]

A. Talbot, The uniform approximation of polynomials by polynomials of lower degree, *J. Approx. Th.* 17 (1976), 254–279. [A precursor to the Carathéodory–Fejér method.]

F. D. Tappert, The parabolic approximation method, in J. B. Keller and J. S. Papadakis, eds., *Wave Propagation and Underwater Acoustics,* Springer, 1977, pp. 224–287. [Describes techniques for one-way acoustic wave simulation in the ocean, based on polynomial and rational approximations of a pseudodifferential operator.]

R. Taylor and V. Totik, Lebesgue constants for Leja points, *IMA J. Numer. Anal.* 30 (2010), 462–486. [Proves that for general sets in the complex plane, the Lebesgue constants associated with Leja points grow subexponentially.]

W. J. Taylor, Method of Lagrangian curvilinear interpolation, *J. Res. Nat. Bur. Stand.* 35 (1945), 151–155. [The first use of the barycentric interpolation formula, for equidistant points only and without the term "barycentric", which was introduced by Dupuy 1948.]

T. W. Tee and L. N. Trefethen, A rational spectral collocation method with adaptively transformed Chebyshev grid points, *SIAM J. Sci. Comp.* 28 (2006), 1798–1811. [Numerical solution of differential equations with highly nonuniform solutions using Chebyshev–Padé approximation, conformal maps, and spectral methods based on rational barycentric interpolants, as advocated by Berrut and coauthors.]

H. Tietze, Eine Bemerkung zur Interpolation, *Z. Angew. Math. Phys.* 64 (1917), 74–90. [Investigates the Lebesgue function for equidistant points, showing the local maxima decrease monotonically from the outside of the interval toward the middle.]

A. F. Timan, A strengthening of Jackson's theorem on the best approximation of continuous functions by polynomials on a finite interval of the real axis, *Doklady Akad. Nauk SSSR* 78 (1951), 17–20. [A theorem on polynomial approximation that recognizes the greater approximation power near the ends of the interval.]

A. F. Timan, *Theory of Approximation of Functions of a Real Variable,* Dover, 1994. [First published in Russian in 1960.]

K.-C. Toh and L. N. Trefethen, Pseudozeros of polynomials and pseudospectra of companion matrices, *Numer. Math.* 68 (1994), 403–425. [Analysis of stability of companion matrix eigenvalues as an algorithm for polynomial rootfinding, given a polynomial expressed by its coefficients in the monomial basis.]

L. Tonelli, I polinomi d'approssimazione di Tschebychev, *Annali di Mat.* 15 (1908), 47–119. [Extension of results on real best approximation to the complex case.]

L. N. Trefethen, Chebyshev approximation on the unit disk, in H. Werner et al., eds., *Constructive Aspects of Complex Analysis*, D. Riedel, 1983. [An introduction to several varieties of CF approximation.]

L. N. Trefethen, Square blocks and equioscillation in the Padé, Walsh, and CF tables, in P. R. Graves-Morris, et al., eds., *Rational Approximation and Interpolation,* Lect. Notes in Math., v. 1105, Springer, 1984. [Shows that square block structure in all three tables of rational approximations arises from equioscillation-type characterizations involving the defect.]

L. N. Trefethen, *Spectral Methods in MATLAB*, SIAM, 2000. [Matlab-based textbook on spectral methods for ODEs and PDEs.]

L. N. Trefethen, Is Gauss quadrature better than Clenshaw–Curtis?, *SIAM Rev.* 50 (2008), 67–87. [Shows by considering approximation properties in the complex plane that for most functions, the Clenshaw–Curtis and Gauss formulas have comparable accuracy.]

L. N. Trefethen, Householder triangularization of a quasimatrix, *IMA J. Numer. Anal.* 30 (2010), 887–897. [Extends the Householder triangularization algorithm to quasimatrices, i.e., "matrices" whose columns are functions rather than vectors.]

L. N. Trefethen and D. Bau, III, *Numerical Linear Algebra,* SIAM, 1997. [A standard text, with a section "When vectors become continuous functions" at p. 52 that foreshadows Chebfun computation with quasimatrices.]

L. N. Trefethen and M. H. Gutknecht (1983a), Real vs. complex rational Chebyshev approximation on an interval, *Trans. Amer. Math. Soc.* 280 (1983), 555–561. [Shows that type (m, n) complex rational approximations to a real function on an interval may be arbitrarily much better than real ones, for $n \geq m+3$.]

L. N. Trefethen and M. H. Gutknecht (1983b), The Carathéodory–Fejér method for real rational approximation, *SIAM J. Numer. Anal.* 20 (1983), 420–436. [Introduction of real rational CF approximation, and first numerical computation of the constant 9.28903... for minimax rational approximation of e^x on $(-\infty, 0]$.]

L. N. Trefethen and M. H. Gutknecht, On convergence and degeneracy in rational Padé and Chebyshev approximation, *SIAM J. Math. Anal.* 16 (1985), 198–210. [Proves theorems to the effect that the Padé approximation operator is continuous, and Padé approximants are limits of best approximants on regions shrinking to a point, provided that the defect is 0.]

L. N. Trefethen and M. H. Gutknecht, Padé, stable Padé, and Chebyshev–Padé approximation, in J. C. Mason and M. G. Cox, *Algorithms for Approximation,* Clarendon Press, 1987, pp. 227–264. [Reduces the problem of Chebyshev–Padé approximation to the problem of stable Padé approximation, that is, Padé approximation with a constraint on location of poles.]

L. N. Trefethen and L. Halpern, Well-posedness of one-way wave equations and absorbing boundary conditions, *Math. Comput.* 47 (1986), 421–435. [Shows that approximations from two diagonals of the Padé table must be used in these applications; polynomial and other approximations are ill-posed.]

L. N. Trefethen and J. A. C. Weideman, Two results concerning polynomial interpolation in equally spaced points, *J. Approx. Th.* 65 (1991), 247–260. [Discussion of the size of Lebesgue constants and "6 points per wavelength" for polynomial interpolation in equispaced points.]

L. N. Trefethen, J. A. C. Weideman and T. Schmelzer, Talbot quadratures and rational approximations, *BIT Numer. Math.* 46 (2006), 653–670. [Shows how integrals approximated by the trapezoid rule correspond to rational approximations in the complex plane, with particular attention to the approximation of e^x on $(-\infty, 0]$.]

A. H. Turetskii, The bounding of polynomials prescribed at equally distributed points, *Proc. Pedag. Inst. Vitebsk* 3 (1940), 117–127 (Russian). [Derivation of the $\sim 2^n/en \log n$ asymptotic size of Lebesgue constants for equispaced polynomial interpolation. This paper went largely unnoticed for fifty years and the main result was rediscovered by Schönhage 1961.]

Ch.-J. de la Vallée Poussin, Note sur l'approximation par un polynôme d'une fonction dont la derivée est à variation bornée, *Bull. Acad. Belg.* 1908, 403–410. CHECK

Ch. de la Vallée Poussin, Sur les polynômes d'approximation et la représentation approchée d'un angle, *Acad. Roy. de Belg., Bulletins de la Classe des Sci.* 12 (1910). CHECK

Ch. J. de la Vallée Poussin, *Leçons sur l'approximation des fonctions d'une variable réelle,* Gauthier-Villars, Paris, 1919. CHECK

J. Van Deun and L. N. Trefethen, A robust implementation of the Carathéodory–Fejér method, *BIT Numer. Math.* 51 (2011), 1039–1050. [Twenty-five years after the original theoretical papers, a paper describing the practical details behind the Chebfun `cf` command.]

R. S. Varga and A. J. Carpenter, On the Bernstein conjecture in approximation theory, *Constr. Approx.* 1 (1985), 333–348. [Shows that degree n best polynomial approximants to $|x|$ have asymptotic accuracy $0.280\ldots n^{-1}$ rather than $0.282\ldots n^{-1}$.]

R. S. Varga, A. Ruttan and A. J. Carpenter, Numerical results on best uniform rational approximation of $|x|$ on $[-1,1]$, *Math. USSR Sbornik* 74 (1993), 271–290. [High-precision numerical calculations lead to the conjecture that best rational approximations to $|x|$ on $[-1,1]$ converge asymptotically at the rate $\sim 8\exp(-\pi\sqrt{n})$, proved by Stahl 1993.]

N. S. Vyacheslavov, On the uniform approximation of $|x|$ by rational functions, *Sov. Math. Dokl.* 16 (1975), 100–104. [Sharpens the result of Newman 1964 by showing that rational approximations to $|x|$ on $[-1,1]$ converge at the rate $O(\exp(-\pi\sqrt{n}))$.]

J. Waldvogel, Fast construction of the Fejér and Clenshaw–Curtis quadrature rules, *BIT Numer. Math.* 46 (2006), 195–202. [Presentation of $O(n\log n)$ algorithms for finding nodes and weights.]

H. Wallin, On the convergence theory of Padé approximants, in *Linear Operators and Approximation*, Internat. Ser. Numer. Math. 20 (1972), pp. 461–469. [Shows that there exists an entire function f whose (n,n) Padé approximants are unbounded for all $z \neq 0$.]

J. L. Walsh, The existence of rational functions of best approximation, *Trans. Amer. Math. Soc.* 33 (1931), 668–689. [Shows that there exists a best rational approximation of type (m,n) to a given continuous function f, not just on an interval such as $[-1,1]$ but also on more general sets in the complex plane.]

J. L. Walsh, On approximation to an analytic function by rational functions of best approximation, *Math. Z.* 38 (1934), 163–176. [Perhaps the first discussion of what is now called the Walsh table, the table of best rational approximations to a given function f for various types (m,n).]

J. L. Walsh, The analogue for maximally convergent polynomials of Jentzsch's theorem, *Duke Math. J.* 26 (1959), 605–616. [Shows that every point on the boundary of a region of convergence of a sequence of polynomial approximations is the limit of zeros of its partial sums.]

J. L. Walsh, *Interpolation and Approximation by Rational Functions in the Complex Domain*, 5th ed., American Mathematical Society, 1969. [An encyclopedic but hard-to-read treatise on all kinds of material related to polynomial and rational approximation in the complex plane, originally published in 1935.]

H. Wang and S. Xiang, On the convergence rates of Legendre approximation, *Math. Comp.* 81 (2012), 861–877. [Theorem 3.1 connects barycentric interpolation weights $\{\lambda_k\}$ and Gauss–Legendre quadrature weights $\{w_k\}$.]

R. C. Ward, Numerical computation of the matrix exponential with accuracy estimate, *SIAM J. Numer. Anal.* 14 (1977), 600–610. [Presentation of a scaling-and-squaring algorithm for computing the exponential of a matrix by Padé approximation, a form of which is used by Matlab's `expm` command.]

E. Waring, Problems concerning interpolations, *Phil. Trans. R. Soc.* 69 (1779), 59–67. [Presents the Lagrange interpolation formula 16 years before Lagrange.]

G. A. Watson, *Approximation Theory and Numerical Methods,* Wiley, 1980. [Textbook with special attention to L_1 and L_p norms.]

M. Webb, Computing complex singularities of differential equations with Chebfun, *SIAM Undergrad. Research Online,* submitted, 2012. [Exploration of rational approximation for locating complex singularities of numerical solutions to ODE problems including Lorenz and Lotka–Volterra equations.]

M. Webb, L. N. Trefethen and P. Gonnet, Stability of barycentric interpolation formulas, *SIAM J. Sci. Comp.*, submitted, 2011. [Confirming the theory of Higham 2004, shows that the "type 2" barycentric interpolation formula can be dangerously unstable if used for extrapolation outside the data interval.]

J. A. C. Weideman, Computing the dynamics of complex singularities of nonlinear PDEs, *SIAM J. Appl. Dyn. Syst.* 2 (2003), 171–186. [Applies Padé approximation to computed solutions of nonlinear time-dependent PDEs to estimate locations of moving poles and other singularities.]

J. A. C. Weideman and S. C. Reddy, A MATLAB differentiation matrix suite, *ACM Trans. Math. Softw.* 26 (2000), 465–519. [A widely-used collection of Matlab programs for generating Chebyshev, Legendre, Laguerre, Hermite, Fourier, and sinc spectral differentiation matrices of arbitrary order.]

J. A. C. Weideman and L. N. Trefethen, The kink phenomenon in Fejér and Clenshaw–Curtis quadrature, *Numer. Math.* 107 (2007), 707–727. [Analysis of the effect that as n increases, Clenshaw–Curtis quadrature initially converges at the same rate as Gauss rather than half as fast as commonly supposed.]

J. A. C. Weideman and L. N. Trefethen, Parabolic and hyperbolic contours for computing the Bromwich integral, *Math. Comput.* 76 (2007), 1341–1356. [Derivation of geometrically-convergent "Talbot contour" type rational approximations for problems related to e^x on $(-\infty, 0]$.]

K. Weierstrass, Über continuierliche Functionen eines reellen Arguments, die für keinen Werth des letzteren einen bestimmten Differentialquotienten besitzen, *Königliche Akademie der Wissenschaften,* 1872. [Weierstrass's publication of an example (which he had lectured on a decade earlier) of a continuous, nowhere-differentiable function.]

K. Weierstrass, Über die analytische Darstellbarkeit sogenannter willkürlicher Functionen einer reellen Veränderlichen, *Sitzungsberichte der Akademie zu Berlin*, 633–639 and 789–805, 1885. [Presentation of the Weierstrass approximation theorem.]

B. D. Welfert, Generation of pseudospectral differentiation matrices. I, *SIAM J. Numer. Anal.* 34 (1997), 1640–1657. [Derivation of stable recursive formulas for computation of derivatives of interpolants.]

E. J. Weniger, Nonlinear sequence transformations for the acceleration of convergence and the summation of divergent series, *Computer Phys. Rep.* 10 (1989), 189–371 (also available as arXiv:math/0306302v1, 2003). [Extensive survey.]

H. Werner, On the rational Tschebyscheff operator, *Math. Z.* 86 (1964), 317–326. [Shows that the operator mapping a real function $f \in C[-1, 1]$ to its best real rational approximation of type (m, n) is continuous if and only if f is itself rational of type (m, n) or its best approximation has defect 0 ("nondegenerate").]

H. Wilbraham, On a certain periodic function, *Cambridge and Dublin Math. J.* 3 (1848), 198–201. [Analyzes the Gibbs phenomenon fifty years before Gibbs.]

J. H. Wilkinson, The perfidious polynomial, in G. H. Golub, ed., *Studies in Numerical Analysis*, Math. Assoc. Amer., 1984. [Wilkinson's major work on polynomials was in the 1960s, but this entertaining review, which won the Chauvenet Prize of the Mathematical Association of America in 1987, remains noteworthy not least because of its memorable title.]

J. Wimp, *Sequence Transformations and their Applications,* Academic Press, 1981. [Monograph on many methods for acceleration of convergence.]

C. Winston, On mechanical quadratures formulae involving the classical orthogonal polynomials, *Ann. Math.* 35 (1934), 658–677. [States a general connection between Gauss–Jacobi quadrature weights and the Lagrange polynomials.] *check*

P. Wynn, On a device for computing the $e_m(S_n)$ transformation, *Math. Comp.* 10 (1956), 91–96. [Wynn's first of many papers about the epsilon algorithm for acceleration of convergence of sequences.]

S. Xiang and F. Bornemann, On the convergence rates of Gauss and Clenshaw–Curtis quadrature for functions of limited regularity, archive 1203.2445v1, 2012. *check*

K. Zhou, J. C. Doyle and K. Glover, *Robust and Optimal Control*, Prentice Hall, 1996. [A leading textbook on optimal control, with special attention to approximation issues.]

W. P. Ziemer, *Weakly Differentiable Functions*, Springer, 1989. [Includes a definition of total variation in the measure theoretic context.]