Accuracy Enhancement for Higher Derivatives using Chebyshev Collocation and a Mapping Technique

Wai Sun Don
Division of Applied Mathematics
Brown University
Providence, Rhode Island 02912

Alex Solomonoff
Institute for Math and its Applications
University of Minnesota
Minneapolis, MN 55455

July 2, 1994

Running head : Accuracy Enhancement for higher Chebyshev Derivatives

Mailing Address :
Box F, 182 George Street
Div. of Applied Mathematics
Brown University
Providence, RI 02912
Tel : 401-863-2250
e-mail: wsdon@hydra.cfm.brown.edu

Subject Classification : 65D25 (Numerical Differentiation), 65F30 (Matrix Algorithms), 65G05 (Roundoff Error)

Key Words : Chebyshev collocation, Differentiation Matrix, Roundoff error, Tal-Ezer Mapping

Abstract

We study a new method in reducing the roundoff error in computing derivatives using Chebyshev collocation methods. By using a grid mapping derived by Kosloff and Tal-Ezer, and the proper choice of the parameter α, the roundoff error of the k-th derivative can be reduced from $O(N^{2k})$ to $O((N|\ln \epsilon|)^k)$, where ϵ is the machine precision and N is the number of collocation points. This drastic reduction of roundoff error makes mapped Chebyshev methods competitive with any other algorithm in computing second or higher derivatives with large N. We also study several other aspects of the mapped Chebyshev differentiation matrix. We find that 1) the mapped Chebyshev methods requires much less than π points to resolve a wave, 2) the eigenvalues are less sensitive to perturbation by roundoff error, and 3) larger time steps can be used for solving PDEs. All these advantages of the mapped Chebyshev methods can be achieved while maintaining spectral accuracy.

1 Introduction

In [5], we addressed the issue of roundoff error in computing the derivative using the Chebyshev collocation methods. For details on these methods, see Canuto, et. al. [1], or Gottlieb and Orszag [2]. We showed how to construct the Chebyshev collocation derivative with only $O(\epsilon N^2)$ roundoff error, where ϵ is the machine precision and N is the number of collocation points, by carefully constructing the entries of the derivative matrix.

There are PDEs that involve higher derivatives than the first. For example, the viscosity term in the Navier-Stokes Equation and the fourth derivative term in the Kuramoto-Sivashinsky Equation. For these problems, the roundoff error in the k-th derivative will be $O(\epsilon N^{2k})$. This can ruin the computed solution, even if k and N are not that large. This limits the applicability of Chebyshev collocation methods to certain types of PDEs and values of N.

In this paper we investigate a way of modifying the Chebyshev collocation derivative which reduces the roundoff error, and also improves its accuracy. As stated in [5], the roundoff error cannot be reduced further for basis functions based on polynomials, since the roundoff error of the Chebyshev collocation method already achieves the theoretical minimum. Therefore, to have any hope of reducing the roundoff error, the polynomial basis functions must be replaced with something else. One way to do this is to apply a coordinate transformation. More specifically, the Chebyshev collocation points $\xi_j = \cos(\pi j/N), j = 0, \ldots, N$ are mapped to a new set of points $x_j = g(\xi_j, \alpha), j = 0, \ldots, N$ with a parameter α. The transformation function $g(\xi, \alpha)$ is one-to-one and onto. The mapping is also applied to the polynomial basis functions, and so they are changed as well.

In this paper, we will concentrate on a mapping of the form $x = g(\xi, \alpha) = \sin^{-1}(\alpha\xi)/\sin^{-1}(\alpha)$. In [4], Kosloff and Tal-Ezer showed that this mapping increases the minimum spacing Δx between collocation points from $O(N^{-2})$ to $O(N^{-1})$, and argued that this reduces the spectral radius of the differentiation matrix from $O(N^2)$ to $O(N)$. Here, we will show that it has a similar effect on the roundoff error. Moreover, the mapping not only reduces the roundoff error but also requires less than π points per wave number as with standard Chebyshev methods.

In section 2, we introduce the Chebyshev collocation method and its derivative matrix. Section 3 illustrates the problem of large roundoff error when computing higher derivatives using the standard Chebyshev collocation methods. We discuss the transformation of the Chebyshev collocation points and its properties in section 4. The minimum roundoff error is estimated for the standard and mapped differentiation matrix in section 5. In section 6, some numerical results of the standard vs. mapped Chebyshev collocation methods are demonstrated. We study the effect of the mapping on the eigenvalue spectrum of the matrix with Dirichlet boundary conditions in section 7. Section 8 gives a heuristic estimate for the resolution power of the mapped Chebyshev methods. We discuss in section 9 some pitfalls and proper procedures in computing the mapped Chebyshev derivatives.

2 Chebyshev Collocation Methods

In this section, we present the Chebyshev collocation method. Consider the Chebyshev-Gauss-Lobatto collocation points,

$$x_i = \cos(\frac{\pi i}{N}), \qquad i = 0, \ldots, N,$$

which are the extrema of the Nth order Chebyshev polynomial

$$T_N(x) = \cos(N \cos^{-1} x).$$

Let $v(x)$ be a smooth function in the domain $x \in [-1, 1]$. Then $v(x)$ is interpolated by constructing the N order interpolation polynomial $g_j(x)$ such that $g_j(x_k) = \delta_{jk}$, i.e.

$$u(x) = \sum_{j=0}^{N} u_j g_j(x)\,, \tag{1}$$

where $u(x)$ is the polynomial of degree N and $u_j = v(x_j)$, $j = 0, \ldots, N$. It can be shown that

$$g_j(x) = \frac{(-1)^{j+1}(1 - x^2)\, T'_N(x)}{c_j N^2 (x - x_j)}, \qquad j = 0, \ldots, N\,, \tag{2}$$

where

$$c_j = \begin{cases} 2 & j = 0, N \\ 1 & j = 1, \ldots, N-1 \end{cases} .$$

Differentiation

The derivative of $u(x)$ at the collocation points x_j can be computed in many different ways. The most obvious way to compute the derivative is the matrix-vector multiplication. The entries of the Chebyshev derivative matrix D, are computed by taking the analytical derivative of $g_j(x)$ and evaluating it at the collocation points x_k for $j, k = 0, \ldots, N$, i.e., $D_{kj} = g'_j(x_k)$. Then the entries of the matrix are, for $j = 0, \ldots, N, k = 0, \ldots, \frac{N}{2}$

$$\begin{aligned} D_{kj} &= -\frac{1}{2}\frac{c_k}{c_j}\frac{(-1)^{j+k}}{\sin\frac{\pi}{2N}(k+j)\sin\frac{\pi}{2N}(k-j)} \quad j \neq k\,, \\ D_{kk} &= -\frac{1}{2}\frac{\cos(\frac{\pi}{N}k)}{\sin^2(\frac{\pi}{N}k)} \qquad k \neq 0\,, \\ D_{00} &= -D_{NN} = \frac{2N^2+1}{6}. \end{aligned} \tag{3}$$

and

$$D_{kj} = -D_{N-k,N-j}\,, \qquad k = \frac{N}{2} + 1, \ldots, N\,, \tag{4}$$

The derivative of $u(x_i)$ becomes ($'$ denotes derivative unless specified otherwise)

$$u'_i = D\overrightarrow{u} = \sum_{j=0}^{N} D_{ij} u_j\,, \qquad i = 0, \ldots, N\,. \tag{5}$$

There are also $O(N \log_2 N)$ algorithms for computing the derivative that involve Fast Discrete Fourier and/or Cosine Transforms. Their roundoff error properties are mostly the same as the matrix-vector multiplication. We will concentrate on the matrix-vector multiplication algorithm and only mention them occasionally.

N	$k=1$	$k=2$	$k=3$	$k=4$
16	0.58E-12	0.44E-10	0.14E-08	0.21E-07
32	0.83E-12	0.47E-09	0.11E-06	0.17E-04
64	0.41E-11	0.62E-08	0.55E-05	0.35E-02
128	0.17E-10	0.71E-07	0.25E-03	0.63E-00
256	0.91E-10	0.35E-04	0.51E-01	0.50E+03
512	0.35E-09	0.98E-04	0.78E+02	0.37E+05
1024	0.31E-08	0.13E-02	0.32E+03	0.51E+08

Table 1: Absolute maximum error for the k-th derivative of $u(x) = \sin(2x)$.

3 Accuracy in computing higher derivatives

In this section we show some numerical experiments which illustrate the large roundoff error of the standard Chebyshev collocation method. We computed the first four derivatives of the test function $u(x) = \sin(mx), x \in [-1, 1]$ without any mapping. The kth derivative is computed by multiplying the vector $\overrightarrow{u} = \{u(x_0), \ldots, u(x_N)\}^t$ by the derivative matrix D k times. Table 1 shows the absolute maximum error for the first four derivatives of $u(x) = \sin(2x)$ for different numbers of Chebyshev collocation points. It clearly illustrates the problem when one wishes to obtain higher derivatives using the Chebyshev collocation methods. The absolute maximum error is fine for the first derivative and acceptable for the second derivative if $N < 128$. For the third and the fourth derivatives, the error becomes unacceptable for $N > 64$ and $N > 32$, respectively.

The rate of growth of the roundoff error shown in table 1 is approximately $O(N^{2k}), k = 1, 2, 3, 4$ (See section 5 for more details). Similar conclusions can be reached for other functions, since this situation is quite generic to the Chebyshev differentiation matrix D. Since the matrix D has already achieved the minimum roundoff error possible, any further manipulation of the elements of the matrix is useless. The situation seems hopeless unless other basis functions are found. For example, we know that the Fourier collocation methods only have roundoff error growth like $O(N^k)$ for the k-th derivative. So if we modify the Chebyshev basis functions to resemble the Fourier methods in some way, then maybe we can reduce the roundoff error. However, this has to be done in such a way as to preserve the spectral accuracy that the standard Chebyshev method has. One way to achieve this is to map the Chebyshev collocation points to some other set of interpolation points. We discuss this in the next section.

4 Grid Transformation

In [4], Kosloff and Tal-Ezer derived a grid transformation that mapped the Chebyshev collocation points to a new set of interpolation points. The grid transformation was defined as

$$x = g(\xi, \alpha) = \frac{\sin^{-1}(\alpha\xi)}{\sin^{-1}\alpha} \tag{6}$$

where $\xi_j = \cos(\pi j/N), j = 0, \ldots, N$ are the Chebyshev collocation points and x_j are a new set of interpolation points with the parameter $\alpha \in [0, 1]$.

Given a function $u(x)$ and $x = g(\xi, \alpha)$, the derivative of $u(x)$ can be evaluated as

$$\frac{du}{dx} = \frac{du}{d\xi}\frac{d\xi}{dx} \tag{7}$$

where

$$\frac{d\xi}{dx} = \frac{1}{g'(\xi, \alpha)} \tag{8}$$

and

$$g'(\xi, \alpha) = \frac{\alpha}{\sin^{-1}\alpha} \frac{1}{\sqrt{1-(\alpha\xi)^2}} \tag{9}$$

So computing at the grid points $x_j = g(\xi_j, \alpha), j = 0, \ldots, N$, (5) or (7) can be rewritten as

$$\overrightarrow{u}' = \mathcal{M} D \overrightarrow{u} \tag{10}$$

where the diagonal matrix $\mathcal{M}$ has elements

$$\mathcal{M}_{jj} = \frac{1}{g'(\xi_j, \alpha)} \quad . \tag{11}$$

We define $\mathcal{D} = \mathcal{M} D$ as our new differentiation matrix.
We will now give a few important properties of the mapping $g(\xi, \alpha)$.

- the mapping derivative $g'(\xi, \alpha)$ is singular if $\alpha = 1$.
- For $\Delta x_{min} = 1 - x_1 = 1 - g(\xi_1, \alpha)$, one can show that

$$\begin{array}{lll} \alpha \to 1, & \Delta x_{min} \to & \frac{2}{N} \\ \alpha \to 0, & \Delta x_{min} \to & 1 - \cos(\frac{\pi}{N}) \quad . \end{array}$$

- if, for some positive number c [4],

$$\alpha \approx 1 - \frac{c}{N^2}$$

then

$$\Delta x_{min} \approx \frac{2}{\pi N}(\sqrt{\pi^2 + 2c} - \sqrt{2c}). \tag{12}$$

The coordinate transform results in some approximation error due to the fact that the transformation function g is not entirely smooth ($g(\xi)$ has singularities at $\xi = \pm\alpha^{-1}$). In [4, section 4.8] Kosloff and Tal-Ezer showed that if α is chosen to be

$$\alpha = \text{sech}(\frac{|\ln \epsilon|}{N}), \tag{13}$$

then the approximation error is roughly ϵ. By choosing ϵ to be the machine precision, the error of the coordinate transformation is essentially guaranteed to be harmless. With this choice for α, in the limit as $N \to \infty$, $\alpha = 1 - c/N^2$ with

$$c = \frac{1}{2}|\ln \epsilon|^2. \tag{14}$$

If we substitute this into (12) and assume that $|\ln \epsilon| \gg \pi$, this gives a minimum grid point spacing of

$$\Delta x_{min} \approx \frac{\pi}{N|\ln \epsilon|}. \tag{15}$$

N	α	$\frac{\pi}{N\lvert\ln\epsilon\rvert}$	$\Delta x_{min}(\alpha)$	$\Delta x_{min}(0)$	$\frac{\Delta x_{min}(\alpha)}{\Delta x_{min}(0)}$
16	0.25532	0.00601064	0.01964	0.0192	1.02
32	0.63778	0.00300532	0.005756	0.004815	1.19
64	0.88252	0.00150266	0.002086	0.0012045	1.73
96	0.94477	0.00100177	0.001245	0.0005354	2.32
128	0.96830	0.00075133	0.0008835	0.0003012	2.93
256	0.99191	0.00037567	0.0004067	0.0000753	5.40
512	0.99797	0.00018783	0.0001952	0.00001883	10.37
1024	0.99950	0.00009392	0.0000956	0.00000471	20.32

Table 2: $\alpha, \Delta x_{min}, \Delta x_{min}(\alpha), \Delta x_{min}(0)$ and their ratio as a function of N with $\epsilon = 6.5 \times 10^{-15}$

In table 2, α is shown as a function of N for a fixed accuracy $\epsilon = 6.5 \times 10^{-15}$, which is the machine precision of the Cray C-90. It also lists several values for equation (15), the minimum spacing with and without mapping and their relative size for each $\alpha(N)$.

Equation (15) implies that $\Delta t = O(\Delta x_{min})$ is inversely proportional to $|\ln \epsilon|$. Hence, larger Δt can only be achieved by reducing the accuracy ϵ of the interpolation. This situation might not be acceptable for those problems demanding high accuracy in the solution. However, Table 2 also clearly shows that one can still get a slightly larger time step without any degradation of accuracy of the approximation. In this paper, we are interested not on the issue of stability but on the accuracy. This allows us to fix the accuracy requirement ϵ to be the machine precision and choose α to be a function of N alone.

5 Estimates of the roundoff error

In this section we discuss two estimates of the roundoff error. In [5, section 4] we constructed the following estimate of the roundoff error: We ignored all of the rounding errors except those that occurred when the function vector u is stored in single precision. Further, we assumed that those errors occurred randomly. Specifically, we assumed that they were uncorrelated, zero-mean, and that their variance (i.e., their rms average value) was the machine precision. Under these assumptions, the average roundoff error in the i-th component of the derivative is

$$\begin{aligned} e_i^2 &\equiv \text{average}\Big\{ |(Du)_{exact,i} - (Du)_{computed,i}|^2 \Big\} \\ &= \epsilon^2 \sum_{k=0}^{N} D_{ik}^2 \\ &= \epsilon^2 (D^t D)_{ii}. \end{aligned} \tag{16}$$

The average L_2 norm of the error is

$$|e|^2 = \epsilon^2 \sum_{i,j} D_{ij}^2, \tag{17}$$

and a reasonable estimate of the maximum error is

$$e_{max} = \sup_i e_i. \tag{18}$$

We can see that if $e_{cheb,i}$ is the average roundoff error for the Chebyshev collocation derivative, then

$$e_{Tal-Ezer,i} = \mathcal{M}_{ii} e_{cheb,i} \tag{19}$$

is the corresponding error for the derivative when using the mapping. Since $\mathcal{M}_{ii}$ is small near the edges of the grid where $e_{cheb,i}$ is large, we expect the mapping to reduce the roundoff error.

Usually the error will be larger than this because of the other rounding errors that happen during the computation. Occasionally it will be smaller because of a lucky cancelation of errors.

This estimate is useful as a lower bound. If an algorithm for computing Du is able to come close to it, then there is no use in trying to improve the algorithm. Many algorithms for computing the derivative are able to approach this bound.

The other estimate for the roundoff error is a heuristic estimate based on the minimum grid point spacing. In finite difference methods with equally spaced points, the formula for the k-th derivative is

$$u_i^{(k)} = \sum_{j=0}^{k} (-1)^k \Delta x^{-k} p_{jk} u_{i+j-k/2} \tag{20}$$

where the p_{jk} are the numbers in Pascal's triangle and are $O(2^k)$ in size. We see from the size of the coefficients that rounding errors can be multiplied by roughly $(2/\Delta x)^k$. We claim that the situation is the same for spectral approximations of the derivative. Therefore a rough estimate of max norm of the error is

$$e_{max} = \epsilon(\frac{2}{\Delta x_{min}})^k. \tag{21}$$

Taking the value of Δx_{min} from equation (15), this gives

$$e_{max} \approx \epsilon(\frac{2}{\pi} N |\ln \epsilon|)^k. \tag{22}$$

The corresponding error for the unmapped Chebyshev derivative is

$$e_{cheb,max} = \epsilon(\frac{2N}{\pi})^{2k}. \tag{23}$$

6 Numerical Results on the mapped Chebyshev methods

In this section we compute the higher derivatives using the Chebyshev collocation with and without the mapping technique. We use the test function $u(x) = \sin(mx)$ for this purpose. We also tested other functions, e.g. $y = \exp(mx), \exp(-x^m), y = x^m$, with similar results. The relative and absolute error between the exact and numerical results are computed by the L_∞ (pointwise maximum) norm.

Here we note that there are two ways to computing the k-th derivative using Chebyshev collocation methods with a mapping. One way is the multiply the matrix $\mathcal{D} = \mathcal{M}D$ by the vector $\overrightarrow{u}$ k times, i.e. $\overrightarrow{u}^{(k)} = \mathcal{D}^k \overrightarrow{u}$ or $\overrightarrow{u}^{(k)} = \mathcal{D}(\mathcal{D}(\ldots(\mathcal{D}u))\ldots)$. The other way is to form an expression involving $D^l u(\xi)$ and $g^l(\xi,\alpha), l = 1,\ldots,k$ by differentiating $u(g(x))$ k times. For example,

$$\overrightarrow{u}''' = \overrightarrow{g}'^3 D\overrightarrow{u} + 3\overrightarrow{g}'\overrightarrow{g}'' D^2\overrightarrow{u} + \overrightarrow{g}''' D^3 \overrightarrow{u} \tag{24}$$

where $\overrightarrow{g}_j = g(\xi_j, \alpha)$. In numerical experiments, both techniques gave exactly the same results. In this paper, we used the first method because of its simplicity in programming.

	No Mapping			with Mapping		
	MV	FFT	CFT	MV	FFT	CFT
N	First Derivative					
16	0.58E-12	0.11E-11	0.95E-12	0.13E-11	0.44E-12	0.12E-11
32	0.83E-12	0.18E-11	0.58E-11	0.85E-12	0.15E-11	0.12E-10
64	0.41E-11	0.66E-11	0.79E-10	0.23E-11	0.28E-11	0.23E-10
128	0.17E-10	0.71E-10	0.46E-09	0.68E-11	0.15E-10	0.24E-09
256	0.91E-10	0.23E-09	0.44E-08	0.39E-10	0.15E-10	0.72E-09
512	0.35E-09	0.58E-09	0.69E-07	0.72E-10	0.78E-10	0.55E-08
1024	0.31E-08	0.54E-08	0.38E-06	0.83E-10	0.85E-10	0.16E-07
N	Second Derivative					
16	0.44E-10	0.12E-09	0.48E-10	0.10E-09	0.55E-10	0.91E-10
32	0.47E-09	0.58E-09	0.22E-08	0.20E-09	0.45E-09	0.32E-08
64	0.62E-08	0.47E-08	0.11E-06	0.20E-08	0.95E-09	0.17E-07
128	0.71E-07	0.33E-06	0.24E-05	0.13E-07	0.29E-07	0.46E-06
256	0.35E-05	0.36E-05	0.91E-04	0.21E-06	0.22E-07	0.28E-05
512	0.98E-05	0.49E-04	0.59E-02	0.33E-06	0.72E-06	0.47E-04
1024	0.13E-02	0.20E-02	0.13E+00	0.21E-05	0.15E-05	0.28E-03
N	Third Derivative					
16	0.14E-08	0.72E-08	0.11E-08	0.47E-08	0.31E-08	0.39E-08
32	0.11E-06	0.11E-06	0.45E-06	0.44E-07	0.79E-07	0.51E-06
64	0.55E-05	0.27E-05	0.91E-04	0.93E-06	0.35E-06	0.69E-05
128	0.25E-03	0.95E-03	0.76E-02	0.15E-04	0.32E-04	0.51E-03
256	0.51E-01	0.36E-01	0.12E+01	0.55E-03	0.33E-04	0.68E-02
512	0.78E+00	0.28E+01	0.31E+03	0.10E-02	0.37E-02	0.23E+00
1024	0.32E+03	0.44E+03	0.28E+05	0.27E-01	0.14E-01	0.28E+01
N	Fourth Derivative					
16	0.21E-07	0.30E-06	0.10E-06	0.15E-06	0.11E-06	0.11E-06
32	0.17E-04	0.16E-04	0.63E-04	0.55E-05	0.95E-05	0.58E-04
64	0.35E-02	0.13E-02	0.52E-01	0.30E-03	0.10E-03	0.20E-02
128	0.63E+00	0.21E+01	0.17E+02	0.12E-01	0.25E-01	0.40E+00
256	0.50E+03	0.28E+03	0.11E+05	0.98E+00	0.15E+00	0.11E+02
512	0.37E+05	0.11E+06	0.11E+08	0.23E+01	0.13E+02	0.83E+03
1024	0.51E+08	0.65E+08	0.41E+10	0.21E+03	0.93E+02	0.20E+05

Table 3: Absolute maximum error for the first four derivative of $u(x) = \sin(2x)$.

In Table 3, we computed the maximum absolute error of the 1st, 2nd, 3rd and 4th derivative of the function $\sin(2x)$. The computations used a Cray C90, whose machine precision is about 6.5×10^{-15} in single precision. We used three different algorithms, namely, Matrix-Vector Multiplication, FFT-Recursion and CFT-Recursion with and without the mapping $g(\xi, \alpha)$. The Cosine transform-recursion algorithm is computed using the forward and inverse cosine transform subroutines FCR and FCRINV from LARCLIB. This is a library local to NASA Langley Research Center, but the source code is available. Both CFT and FFT algorithm are based on the same RFFTMLT subroutine in the Cray Scientific Library Routine. The Matrix-Vector Multiplication algorithm (MV) used the MXM subroutine in the same library package.

Based on Table 3, We can see that the solutions using the mapped Chebyshev methods are uniformly better than standard one for all four differentiations. Another observation is that the CFT algorithm is uniformly worse than both the MV and FFT algorithm in either the standard or mapped category, especially for third and fourth derivative and large N. Moreover, the Matrix-Vector Multiply (MV) algorithm and FFT Recursion algorithm (FFT) had a similar order of error. We will concentrate our discussion on the data obtained by MV algorithm alone.

In Figure 1 we compare the roundoff error of the mapped Chebyshev derivative with the error estimates in equations (18) and (22) for the test function $u(x) = \cos(4x) + \sin(4x)$. The vertical axis of the graph is (Max error)/ϵ. We see that the actual error comes fairly close to the lower bound. We can also see that the heuristic error estimate is fairly close to real error and the lower bound, although the dependence on k is not exactly right.

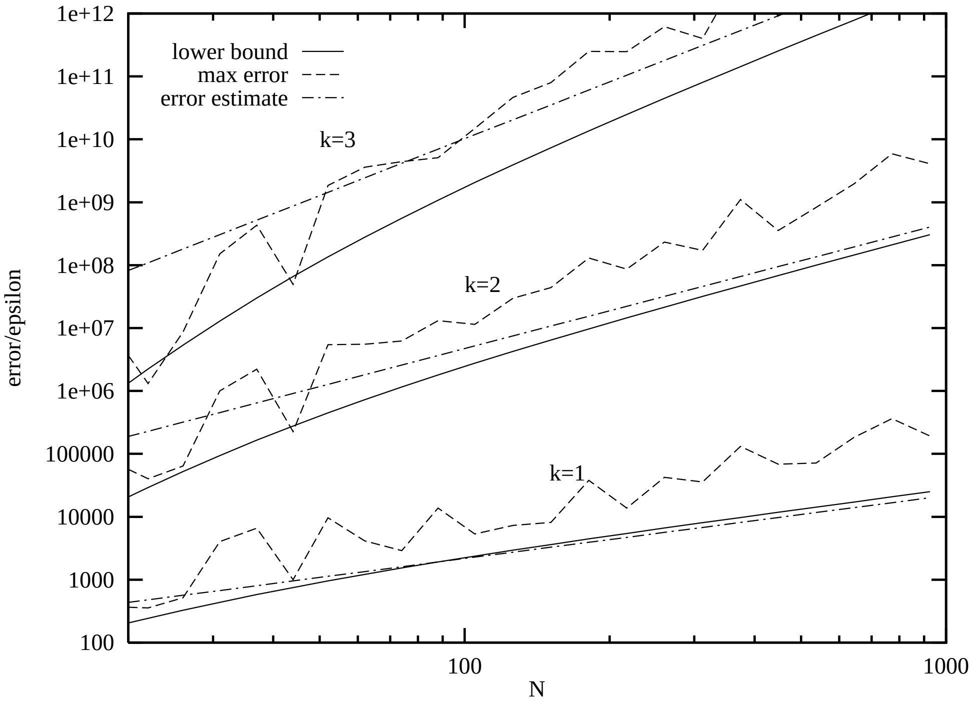

Figure 1: Maximum error vs. number of grid points for k-th derivative. Lower bound is equation (18) and error estimate is equation (22). Error has been scaled by machine precision, i.e., the vertical axis is (max err)/ϵ, with $\epsilon = 6.5 \times 10^{-15}$.

N	$k=1$	$k=2$	$k=3$	$k=4$
	No Mapping			
16	0.13E-07	0.22E-05	0.18E-03	0.89E-02
32	0.47E-11	0.93E-09	0.33E-06	0.58E-04
64	0.63E-10	0.85E-07	0.68E-04	0.38E-01
128	0.13E-09	0.50E-06	0.11E-02	0.18E+01
256	0.39E-09	0.95E-05	0.11E+00	0.95E+03
512	0.10E-08	0.62E-04	0.26E+01	0.83E+05
1024	0.69E-08	0.31E-02	0.72E+03	0.12E+09
	With Mapping			
16	0.50E-07	0.83E-05	0.63E-03	0.30E-01
32	0.54E-11	0.29E-08	0.62E-06	0.83E-04
64	0.17E-10	0.65E-08	0.11E-05	0.50E-03
128	0.17E-10	0.15E-07	0.14E-04	0.11E-01
256	0.54E-10	0.33E-06	0.87E-03	0.15E+01
512	0.13E-09	0.89E-06	0.38E-02	0.13E+02
1024	0.78E-09	0.16E-04	0.17E+00	0.12E+04

Table 4: Absolute maximum error for the k-th derivative of $u(x) = \exp(x^k/\delta) + \cos(mx), m = k = 2, \delta = 0.9$.

6.1 Accuracy for Functions with Boundary Layers

It is well known that the Chebyshev method has good resolution power for functions with boundary layers. How well does the mapped Chebyshev method do on such functions? We tested functions of the type

$$u(x) = \exp(x^k/\delta) + \cos(mx) \quad .$$

The gradient of the boundary layer can be adjusted by varying the parameter $\delta > 0$. The smaller the δ becomes, the steeper the gradient will be near the boundary. The other parameters k and m were set to be $m = k = 2$. We computed the derivatives for two different $\delta = 0.9$ and $\delta = 0.3$.

Table 4 showed the absolute maximum error for case of $\delta = 0.9$. The gradient in this case is not very large. The data showed that both algorithms performed equally well for small N. However, for large $N > 32$ and higher derivatives, the mapped Chebyshev method outperforms the unmapped one.

Table 5 shows the absolute and relative maximum error for case of $\delta = 0.3$. As seen in Table 5, the unmapped Chebyshev method performs better for small $N \leq 32$. We speculate that this is because the mapped Chebyshev method will need more points to resolve such high gradient near the boundary. It is quite self-evident that once the function is well resolved at $N \geq 64$, the mapped Chebyshev methods again perform better. For higher derivatives, since the error is masked by the large functional value, we examined the error in the relative sense for the third and fourth derivatives. The relative accuracy using the mapping is rather good. The fourth derivative with $N = 512$ has only .15% error with the mapping rather than 430% error of the standard Chebyshev methods.

	Absolute Error				Relative Error	
N	$k=1$	$k=2$	$k=3$	$k=4$	$k=3$	$k=4$
	No Mapping					
16	0.91E-03	0.16E+00	0.13E+02	0.66E+03	0.11E-02	0.62E-02
32	0.98E-10	0.72E-08	0.32E-05	0.60E-03	0.17E-09	0.55E-08
64	0.72E-09	0.12E-05	0.10E-02	0.59E+00	0.85E-07	0.55E-05
128	0.17E-08	0.66E-05	0.14E-01	0.19E+02	0.11E-05	0.17E-03
256	0.78E-08	0.20E-03	0.25E+01	0.21E+05	0.21E-03	0.19E+00
512	0.40E-08	0.27E-03	0.12E+02	0.47E+06	0.95E-03	0.43E+01
1024	0.68E-07	0.20E-01	0.33E+04	0.34E+09	0.32E-01	0.32E+04
	With Mapping					
16	0.15E-02	0.25E+00	0.19E+02	0.98E+03	0.16E-02	0.89E-02
32	0.13E-07	0.71E-05	0.18E-02	0.28E+00	0.15E-06	0.26E-05
64	0.19E-09	0.54E-07	0.12E-04	0.11E-01	0.98E-09	0.10E-06
128	0.16E-08	0.17E-05	0.13E-02	0.81E+00	0.65E-07	0.74E-05
256	0.85E-09	0.32E-05	0.11E-01	0.21E+02	0.89E-06	0.19E-03
512	0.37E-08	0.19E-04	0.66E-01	0.16E+03	0.55E-05	0.15E-02
1024	0.29E-08	0.63E-04	0.69E+00	0.56E+04	0.58E-04	0.51E-01

Table 5: Absolute and relative maximum error for the k-th derivative of $u(x) = \exp(x^k/\delta) + \cos(mx), m = k = 2, \delta = 0.3$.

7 Eigenvalue Spectrum of the Differentiation Matrix

In this section, we analyze the eigenvalue spectrum of both the standard and mapped Chebyshev Matrices D and $\mathcal{D}$ respectively, subject to the Dirichlet boundary conditions $u(1) = 0$. It is equivalent to deleting the first row and column of the matrix D and $\mathcal{D}$. Their eigenvalues are computed by the IMSL subroutine EVLRG.

It is an important issue because the (time-) stability condition (Δt fixed, $t \to \infty$) of any explicit time marching scheme, for example, Runge-Kutta or Adams-Bashford scheme, is controlled by its largest eigenvalues.

The eigenvalues of D and $\mathcal{D}$ can be separated into two groups; Those with large imaginary parts, i.e. larger than $O(N)$ which we will call outliers, and all the others. The outliers are stable to perturbations of the matrix. In [3], Trefethen and Trummer found that the non-outlying eigenvalues of D were extremely sensitive to perturbations of the matrix elements, and simply storing the elements of the matrix in single or double precision would completely change the spectrum of the matrix. The changed eigenvalues would have large negative real parts of order $O(N^2)$, as $N \to \infty$. However, they showed that for any given precision ϵ, then for non-outlying eigenvalues λ with negative real part $\mathrm{Re}\lambda < -\frac{1}{2}|\ln \epsilon|$, all relative precision of those eigenvalues are lost. In [3], they simulated the perturbation by lowering the precision of the arithmetic. Here the perturbation comes from representing the matrix elements in finite precision.

For $N = 64$, figures (2.a) and (2.b), show the eigenvalue spectrum of the standard Chebyshev differentiation matrix D in 32 and 64 bits precision arithmetic respectively. Both cases exhibited spurious eigenvalues with large negative real part in the form of an arc to the left of $\frac{1}{2}\ln \epsilon$. However, the largest negative real part computed by 32 bits precision is much larger than those computed with 64 bits precision. For any fixed N, the only way to eliminate those spurious eigenvalues is to use much higher precision arithmetic. This is undesirable and impractical for the numerical solution of PDEs.

However, the eigenvalues of the mapped Chebyshev differentiation matrix $\mathcal{D}$ with the Dirichlet boundary conditions behave very differently. For $N = 64$, figures (2.c) and (2.d), showed the eigenvalue spectrum of the standard Chebyshev differentiation matrix $\mathcal{D}$ in 32 and 64 bits precision arithmetic respectively. For the 32-bit precision matrix, there are a few eigenvalues of $\mathcal{D}$ to the left of the line $Re\lambda = \frac{1}{2}\ln\epsilon$. in the form of a small arc. For the 64-bits precision, there are none. There are a few (less than 10) eigenvalues that have negative real part λ that are less than $\frac{1}{2}\ln\epsilon$. Hence we can conclude that the stability of the numerical scheme based on the mapped Chebyshev collocation methods is much less sensitive to the precision of the arithmetic than the unmapped method.

An important observation is that even for $N = 64$, the largest eigenvalues are considerably smaller in the mapped case than in the unmapped case. Also the largest eigenvalue of the single-precision-mapped matrix is smaller than the largest eigenvalue for the double-precision-mapped matrix. The largest eigenvalue of D is $(-91.9, \pm 351.977)$ and its absolute value is $e_1 = 363.777$. The largest eigenvalue of $\mathcal{D}$ is $(-52.1, \pm 207.987)$ and its absolute value is $e_2 = 214.426$. The ratio $\frac{e_1}{e_2} = 1.696$, which is very close to the value 1.732 in Table 2 for $N = 64$.

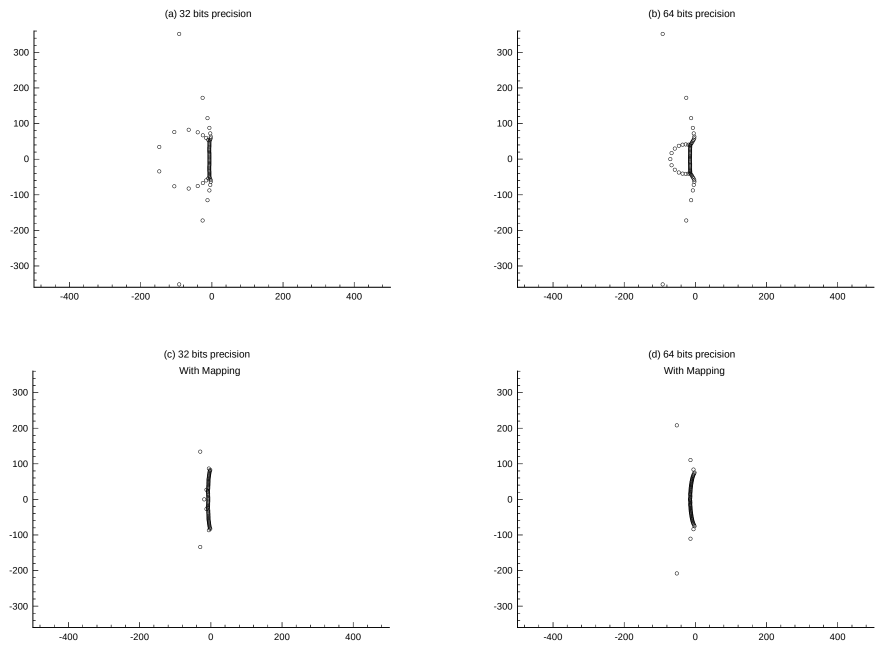

Figure 2: Eigenvalues spectrum of the standard and mapped Chebyshev differentiation matrix D (a,b) and $\mathcal{D}$ (c,d) respectively, in 32 (a,c) and 64 (b,d) bits precision arithmetic. $N = 64$, the α of the mapping computed using the different values of ϵ for the 32-bit and 64-bit precision matrices.

As seen in table 6, the most negative eigenvalue of $\mathcal{D}$ lying on the real axis grows like $O(N^{1.25})$, at least for the values of N in the table. This is substantially smaller than the $O(N^2)$ growth seen when using the standard differentiation matrix D.

This result of $O(N^{1.25})$ is actually quite surprising, for the following reason: recall that the trace

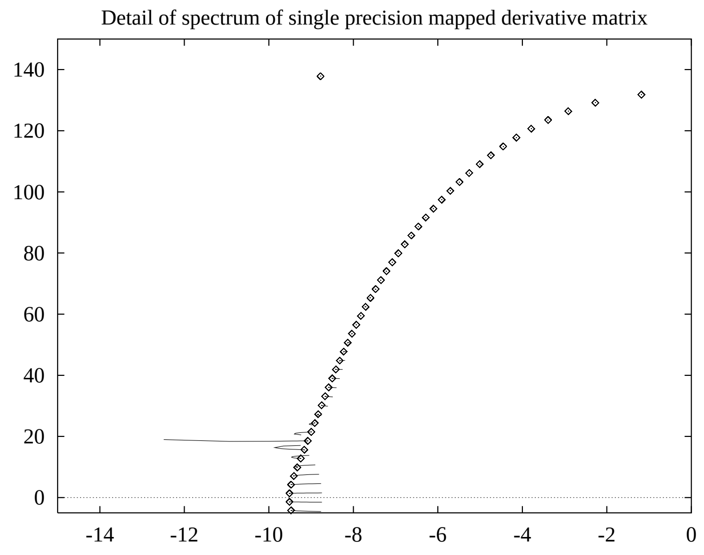

Figure 3: Trajectories of some eigenvalues of the mapped Chebyshev derivative matrix as the size of perturbations of the matrix increases from $\epsilon_{double\ precision}$ to $\epsilon_{single\ precision}$ The α is computed according to a 32-bit precision ϵ, $N = 96$. All computations are in IEEE double precision.

N	64	128	256	512	1024
λ_0	-15.1	-36.5	-92.0	-212.6	-480.0
r		1.27	1.33	1.20	1.18

Table 6: The most negative eigenvalue of $\mathcal{D}$ lying on the real axis, λ_0 and the local exponent of its growth, i.e., $O(N^r)$.

of a matrix is defined as $\mathrm{tr}(M) = \sum_i M_{ii}$, and that $\mathrm{tr}(M) = \sum_i \lambda_i$, where $\{\lambda_i\}$ are the eigenvalues of M.

Both D and $\mathcal{D}$ are anti-centrosymmetric (ACS) matrices, i.e., they satisfy equation (4). The trace of an ACS matrix can easily be shown to be zero. Suppose we modify these matrices by throwing away the first row and column, and call these matrices $\tilde{D}$ and $\tilde{\mathcal{D}}$. Then

$$\mathrm{tr}(\tilde{D}) = -D_{00} = -\frac{2N^2+1}{6}, \tag{25}$$

and

$$\begin{aligned} \mathrm{tr}(\tilde{\mathcal{D}}) &= -\mathcal{D} && (26)\\ &= -\mathcal{M}_{00} D_{00} && (27)\\ &= -\frac{\sin^{-1}\alpha\,\sqrt{1-\alpha^2}}{\alpha}\,\frac{2N^2+1}{6}. && (28)\end{aligned}$$

Since $\mathcal{M}_{00}$ is supposed to be small, the mapping forces the eigenvalues of $\mathcal{D}$ to be closer to the imaginary axis than those of D. Letting $\alpha = \mathrm{sech}(|\ln\epsilon|/N)$, and taking the limit as $N \to \infty$,

$$\mathrm{tr}(\tilde{\mathcal{D}}) \approx -\frac{\pi}{6} N |\ln\epsilon| + \frac{1}{3}|\ln\epsilon|^2. \tag{29}$$

The trace of $\tilde{\mathcal{D}}$ is real, so $\sum_i \mathrm{Im}\lambda_i = 0$, and

$$\mathrm{tr}(\tilde{\mathcal{D}}) = \sum_i \mathrm{Re}\lambda_i. \tag{30}$$

Also, the eigenvalues of $\tilde{\mathcal{D}}$ are experimentally always found to be in the left half-plane. If some of them were not, it would make $\mathcal{D}$ useless for solving parabolic PDEs, so for the rest of the section, we will assume that

$$\mathrm{Re}\lambda_i \le 0, \quad \forall i \tag{31}$$

Now we can use our knowledge of $\mathrm{tr}(\tilde{\mathcal{D}})$ to say some things about the spectrum of $\tilde{\mathcal{D}}$.

Let us consider two extreme arrangements of eigenvalues which satisfy equations (30) and (31). First, suppose all of the eigenvalues but one lay on the imaginary axis. Then the real part of the remaining one would be $\mathrm{tr}(\tilde{\mathcal{D}})$. This maximizes $\max_i \mathrm{Re}(-\lambda_i)$. Next, suppose that all of the eigenvalues have the same real part, so $\mathrm{Re}\lambda_i = \mathrm{tr}(\tilde{\mathcal{D}})/N, \quad \forall i$. This minimizes $\max_i \mathrm{Re}(-\lambda_i)$. The reality falls between these extremes, so

$$\begin{aligned} \max_i \mathrm{Re}(-\lambda_i) &\le -\mathrm{tr}(\tilde{\mathcal{D}}) \approx \frac{\pi}{6} N|\ln\epsilon| && (32)\\ &\ge -\frac{\mathrm{tr}(\tilde{\mathcal{D}})}{N} \approx \frac{\pi}{6}|\ln\epsilon|. && (33)\end{aligned}$$

If the $O(N^{1.25})$ growth rate for the eigenvalues of $\tilde{\mathcal{D}}$ lying on the real axis shown in table 6 holds for all N then for some large enough N, it will exceed the limit given by equation (32). Therefore the $O(N^{1.25})$ growth rate cannot be the asymptotic rate as $N \to \infty$. The asymptotic growth rate of the real part of any eigenvalue of $\mathcal{D}$ can be at most $O(N)$. In table 6, we see that r is decreasing. Presumably, as $N \to \infty$, r approaches 1.

In the case of the Legendre derivative matrix, Trefethen and Trummer found that the sensitivity of the eigenvalues of that matrix effectively raised its spectral radius from $O(N)$ to $O(N^2)$. In infinite precision, the spectrum of the Legendre matrix is similar to that of the Chebyshev matrix, except that it doesn't have the $O(N^2)$ outliers. Heuristically, we expect that the mapping reduces the spectral radius of $\tilde{\mathcal{D}}$ to $O(N|\ln \epsilon|)$. A reasonable thing to fear is that eigenvalue sensitivity might create eigenvalues with large negative real parts, increasing the spectral radius above $O(N|\ln \epsilon|)$. This cannot happen, at least not by generating eigenvalues with large negative real parts. Equation (32) prevents any eigenvalue from having real part bigger than $O(N|\ln \epsilon|)$.

¿From equation (33), at least some of the eigenvalues have to be to the left of the line $\mathrm{Re}\lambda = -\frac{1}{2}|\ln \epsilon|$. It is not clear if this means that some of the eigenvalues have to be changed by roundoff error, since the eigenvalues farthest to the left appear to be the outliers, which are stable anyway.

In Figure 3 we show how the eigenvalues of the mapped derivative matrix change as the matrix perturbations increase in size. We computed the eigenvalues of the matrix

$$\tilde{\mathcal{D}}_t = \tilde{\mathcal{D}} + tE \tag{34}$$

where the entries of the matrix E are random numbers, uniformly distributed in the interval $[-\frac{1}{2}, \frac{1}{2}]$. the curves shown in Figure 3 are the paths taken by the eigenvalues of $\tilde{\mathcal{D}}_t$ in the complex plane as t went from from 10^{-15} to 5×10^{-8}. The mapping used a value of α consistent with $\epsilon = 5 \times 10^{-8}$. All of the computations were done in IEEE double precision using the LAPACK subroutine DGEEVX. The ends of the curves with the diamond are the small-t ends. Only the non-outlying eigenvalues with non-negative imaginary part are shown. It can be clearly seen that at least a few of the eigenvalues are effected by roundoff error. The diamonds that don't appear to have a curve attached to them are eigenvalues which are not affected by perturbations.

This contradicts the statement in [4], that the eigenvalues of the mapped Chebyshev matrix will not be affected by the roundoff error. They are, but much less than for the unmapped matrix.

8 Resolution power of the mapped Chebyshev methods

The standard Chebyshev method requires at least π points per wave before it begins to resolve an oscillatory function like $\sin(m\pi x)$. How many points does the mapped Chebyshev method require? In the limit as $\alpha \to 1$, the grid points become equally spaced, the Chebyshev polynomials become cosine functions, and resolution requires two points/wave. However α, as a function of N, never reaches one, so its not clear what really happens.

We claim (without proof) that the grid begins to resolve a function when the density of points reaches two points per wave in the center of the domain, where the density of points is lowest. This implies that

$$\frac{N}{g'(0,\alpha)} > \frac{m}{\pi} \tag{35}$$

if the function $u(x) = \cos(mx)$ is to be resolved. In the limit as $N \to \infty$, this means that $2 + \pi|\ln \epsilon|/(4N)$ points per wave are required.

We computed the normalized L_2 Error of the function $\cos(mx)$ for many m and N in figure 4. The x axis represents the inverse of the wave number m/π which is normalized by the factor $N/(\pi g'(0,\alpha))$. ¿From the previous section, we have $\alpha = \mathrm{sech}((\ln \epsilon)/N)$ and $g'(0,\alpha) = \alpha/\sin^{-1}\alpha$.

Examining the data carefully, for any fixed wave number m/π, the error decays exponentially fast to ϵ as N increases and as soon as $N/(mg'(0,\alpha)) > 1$. This implies that a minimum of $g'(0,\alpha)\pi$ points is needed to resolve a wave. Table 7 shows some typical values of $g'(0,\alpha)\pi$ for various sample N. ϵ is fixed at 6.5×10^{-15}. For large N, only about two points per wave are needed. This is close to the performance of Fourier methods.

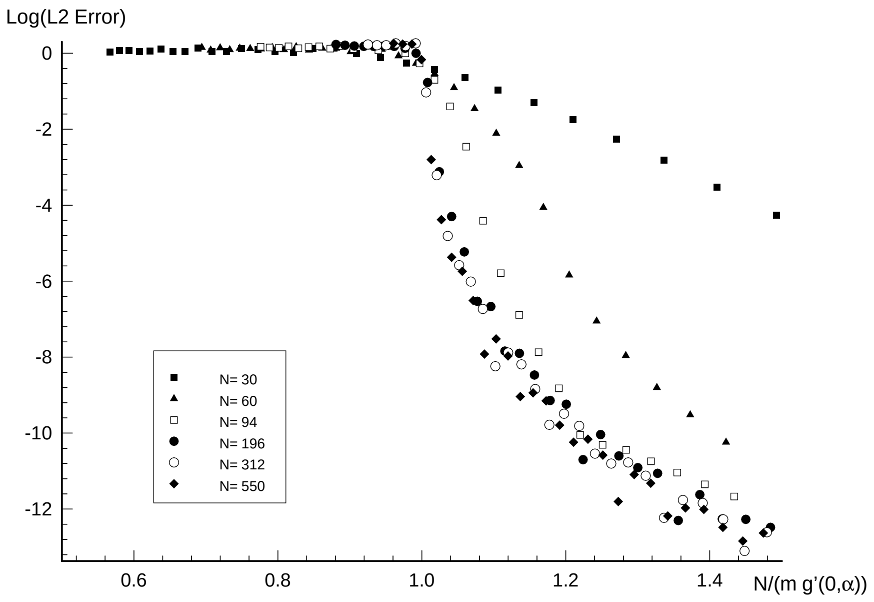

Figure 4: The normalized L_2 Error of the function $\cos(mx)$ for many m and N

N	32	64	128	256	512	1024
$g'(0, \alpha)\pi$	2.90	2.56	2.31	2.16	2.08	2.04

Table 7: Number of points/wave vs. N

9 How not to compute the derivative

Since computing the k-th derivative with k and N large can result in very large roundoff errors, it is worthwhile to work extra-hard to minimize them. In this section we discuss several non-obvious sources of roundoff error and show how to avoid them.

9.1 Ill-conditioning of the computations associated with the mapping

Recall that the grid transformation function is

$$x = g(\xi) = \beta^{-1}\sin^{-1}(\alpha\xi) \tag{36}$$

where $\beta = \sin^{-1}\alpha$. and then the mapped grid points are $x_i = g(\xi_i)$, where $\xi_i = \cos(\pi i/N)$ are the unmapped Chebyshev grid points. The derivative of $g(\xi)$ is

$$g'(\xi) = \frac{\alpha}{\beta\sqrt{1-\alpha^2\xi^2}}. \tag{37}$$

When $|\alpha\xi|$ is near 1, $|g'(\xi)|$ is large. This can be a problem. The roundoff errors associated with storing the ξ_i in single precision will result in errors in x_i of size $O(\epsilon g'(\xi_i))$. The largest values of $|\xi_i|$ is 1, and as $N \to \infty$, $\alpha \to 1$. The computation of the transformation derivative $\mathcal{M}_{ii}$ is also ill-conditioned. To avoid excessive errors resulting from these problems, both x_i and $\mathcal{M}_{ii}$ need to be computed in extended precision before being stored in single precision. Failure to do this can result in loss of about 1 digit of precision for N larger than about 100.

9.2 Ill-conditioning of the matrix multiply used to form D^k

The most obvious way to obtain the k-th derivative matrix D^k is by constructing the first derivative matrix D and doing $k-1$ matrix multiplications. This is a bad idea; the matrix multiplication is ill-conditioned. The sum that must be evaluated to obtain the entries of the 2nd derivative matrix is

$$D^{(2)}_{ij} = (-1)^{i+j}\frac{c_i}{c_j}\sum_{k=0}^{N}\frac{1}{s(i+k)\,s(i-k)\,s(k+j)\,s(k-j)}$$

where $s(i) = \sin\frac{\pi}{2N}(i)$, and ignoring the $k=i$ and $k=j$ terms in the sum. The terms in the sum change sign twice; when $k=i$ and again when $k=j$. This is a problem. It means that the partial sums can be much larger than the final sum. The roundoff error is basically proportional to the size of the partial sums, see [6]. This can be partially fixed by adding up the terms in the sum in random order, rather than in order of increasing k. But a better idea is simply to use closed-form expressions for the entries of the matrix. Of course, there *are* no closed-form expressions for the entries of the matrix $(\mathcal{M}D)^k$. This is a strong argument in favor of using the long form of the mapped derivative matrix, i.e., according to equation (24).

9.3 Cosine transform Algorithms

There are other algorithms for computing this matrix-vector product. One involves doing a discrete cosine transform on u, doing $O(N)$ operations on the transformed data, and then an inverse cosine transform. One way of computing the cosine transform is to symmetrically extend u into a vector of length $2n$ and do an FFT on the longer vector.

Two faster alternative are the algorithms described in Dollimore [8], and in Cooley, Lewis, and Welch [7]. They both involves an $O(N)$ preprocessing step, an FFT on a vector of length n, and an $O(N)$ post-processing step. Unfortunately the pre- and post-processing steps are ill-conditioned, and these algorithms have $O(N\epsilon)$ roundoff error, compared to $O(\epsilon)$ roundoff error for the symmetric extension algorithm. When used to compute the Chebyshev derivative, this gives $O(N^3\epsilon)$ roundoff error rather than $O(N^2\epsilon)$.

Swarztrauber [9] describes a cosine transform which appears to be free of roundoff error problems, and at least as fast as the Dollimore algorithm. However, there are many publicly-available subroutines which encode the Dollimore algorithm, and we have not been able to find any examples of Swarztrauber's algorithm.

Acknowledgments

The authors would like to thank Prof. Alvin Bayliss for bringing this subject to the authors' attention. Computing time on the Cray YMP was provided by the US Army Corps of Civil Engineering, Waterway Experiment Station, High Performance Computer Center. The first author would like to acknowledge the support of this research by AFOSR grant 93-1-0090, DARPA grant N00014-91-J-4016 and NSF grant DMS-92-11820.

References

[1] C. Canuto, A. Quarteroni, M. Y. Hussaini and T. Zang, *Spectral Methods in Fluid Mechanics*, Springer-Verlag, New York, 1988.

[2] D. Gottlieb and S. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications*, SIAM, Philadelphia, 1977.

[3] L. N. Trefethen and M. R. Trummer, *An Instability Phenomenon in Spectral Methods*, SIAM J. Numer. Anal., 24 (1987), pp. 1008-1023.

[4] D. Kosloff and H. Tal-Ezer, Modified Chebyshev Pseudospectral Methods With $O(N^{-1})$ Time Step Restriction, J. of Comp. Phy., 104, 2 (1993), pp. 457-469

[5] W. S. Don and A. Solomonoff, Accuracy and Speed in Computing the Chebyshev Collocation Derivative, Submitted to SIAM J. Sci. Comp.

[6] N.J. Higham, "The Accuracy of Floating Point Summation", SIAM J. Sci. Comp., **14**(4), pp. 783-799, 1993.

[7] J.W. Cooley, P.A.W. Lewis, and P.D. Welch, "The Fast Fourier Transform Algorithm: Programming considerations in the Calculation of Sine, Cosine, and Laplace Transforms", J. Sound Vib. **12**(3), pp. 315-337, 1970.

[8] J. Dollimore, "Some Algorithms for use with the Fast Fourier Transform", J. Inst. Math. Appl., **12**, pp. 115-117, 1973.

[9] Paul N. Swarztrauber, "Symmetric FFTs", Math. Comp. **47**(175), pp. 323-346, July 1986.