A FAST AND WELL-CONDITIONED SPECTRAL METHOD

SHEEHAN OLVER* AND ALEX TOWNSEND†

Abstract. A novel spectral method is developed for the direct solution of linear ordinary differential equations with variable coefficients. The method leads to matrices which are almost banded, and a numerical solver is presented that takes $\mathcal{O}(m^2n)$ operations, where m is the number of Chebyshev points needed to resolve the coefficients of the differential operator and n is the number of Chebyshev points needed to resolve the solution to the differential equation. We prove stability of the method by relating it to a diagonally preconditioned system which has a bounded condition number, in a suitable norm. For Dirichlet boundary conditions, this reduces to stability in the standard 2-norm.

Key words. spectral method, ultraspherical polynomials, direct solver

AMS subject classifications. 65M70, 65L10, 33C45

1. Introduction. Spectral methods are an important tool in scientific computing and engineering applications for solving differential equations (see, for instance, [3, 9, 20, 21]). Although the computed solutions can converge super-algebraically to the solution of the differential equation, conventional wisdom states that spectral methods lead to dense, ill-conditioned matrices. In this paper, we introduce a spectral method which continues to converge super-algebraically to the solution, but only requires solving an almost banded, well-conditioned linear system.

Throughout, we consider the family of linear differential equations on $[-1, 1]$:

$$\mathcal{L}u = f \qquad \text{and} \qquad \mathcal{B}u = \mathbf{c} \tag{1.1}$$

where $\mathcal{L}$ is an Nth order linear differential operator

$$\mathcal{L}u = a^N(x)\frac{\mathrm{d}^N u}{\mathrm{d}x^N} + \cdots + a^1(x)\frac{\mathrm{d}u}{\mathrm{d}x} + a^0(x)u,$$

$\mathcal{B}$ denotes K boundary conditions (Dirichlet, Neumann, etc.), $\mathbf{c} \in \mathbb{C}^K$ and $a^0, \ldots, a^N$, f are suitably smooth functions on $[-1, 1]$. We make the assumption that $a^N(x)$ does not vanish on the interval $[-1, 1]$.

Within spectral methods there is a subdivision between collocation methods and coefficient methods; the former construct matrices operating on the values of a function at, say, Chebyshev points; the latter construct matrices operating on coefficients in a basis, say, of Chebyshev polynomials. Here there is a common belief that collocation methods are more adaptable to differential equations with variable coefficients; i.e., when $a^0(x), \ldots, a^N(x)$ are not constant (see Section 9.2 of [2]). However, the spectral coefficient method that we construct is equally applicable to differential equations with variable coefficients.

For example, the first order differential equation (chosen so the entries in the resulting linear system are integers)

$$\frac{\mathrm{d}u}{\mathrm{d}x} + 4xu = 0 \qquad \text{and} \qquad u(-1) = c, \tag{1.2}$$

*School of Mathematics and Statistics, The University of Sydney, Sydney, Australia. (Sheehan.Olver@sydney.edu.au)

†Mathematical Institute, 24-29 St Giles', Oxford, England, OX1 3LB. (townsend@maths.ox.ac.uk)

results in our spectral method forming the almost banded $n \times n$ linear system

$$\begin{pmatrix} 1 & -1 & 1 & -1 & 1 & -1 & \cdots & (-1)^{n-1} \\ & 2 & & -1 & & & & \\ 2 & & 2 & & -1 & & & \\ & 1 & & 3 & & -1 & & \\ & & \ddots & & \ddots & & \ddots & \\ & & & 1 & & n-3 & & -1 \\ & & & & 1 & & n-2 & \\ & & & & & 1 & & n-1 \end{pmatrix} \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ \\ \vdots \\ \vdots \\ \\ u_{n-1} \end{pmatrix} = \begin{pmatrix} c \\ 0 \\ 0 \\ \\ \vdots \\ \vdots \\ \\ 0 \end{pmatrix}. \quad (1.3)$$

We then approximate the solution to (1.2) as

$$u(x) = \sum_{k=0}^{n-1} u_k T_k(x)$$

where T_k is the Chebyshev polynomial of degree k (of the first kind). Moreover, before solving (1.3) we use a diagonal preconditioner to scale the columns and, in this case, the preconditioned matrix system has 2-norm condition number bounded above by 53.6 for all n. The preconditioned matrix, just like (1.3), is almost banded and it can be solved in $\mathcal{O}(n)$ operations (as discussed in Section 5).

We use *boundary bordering*. That is, K rows of the linear system are used to impose the K boundary conditions. For (1.2), the last row of the linear system has been replaced and then permuted to the first row (for numerical convenience, later). Imposing the boundary condition in this way results in an error, analogous to the error in the tau method [2], which tends to zero as $n \to \infty$.

Alternatively, the boundary condition could be imposed by *basis recombination*. That is, for our simple example, by computing the coefficients in the expansion

$$u(x) = cT_0(x) + \sum_{k=1}^{\infty} \tilde{u}_k \phi_k(x)$$

where

$$\phi_k(x) = \begin{cases} T_k(x) - T_0(x) & k \text{ even} \\ T_k(x) + T_0(x) & k \text{ odd.} \end{cases}$$

The basis is chosen so that $\phi_k(-1) = 0$, and there are many other possible choices. Basis recombination is commonly used in Petrov–Galerkin spectral methods (see [11]), where the Galerkin method uses a different trial and test basis. Shen has constructed bases for second, third, fourth and higher odd order differential equations with constant coefficients so that the resulting matrices are banded or easily invertible [18, 19]. Moreover, it was shown that very specific variable coefficients preserve this matrix structure [20].

Our method will also use two different bases: the Chebyshev and ultraspherical polynomial bases, which are very similar to the bases considered in [7, 11, 18], but in our case will not depend on the boundary conditions. Basis recombination runs

counter to an important observation of Orszag [15]: Spectral methods in a convenient basis (like Chebyshev) can outperform an awkward problem-dependent basis. The problem-dependent basis may be theoretically elegant, but convenience is worth more in practice. In detail, the benefit of using boundary bordering instead of basis recombination include: (1) the solution is always computed in the convenient and orthogonal Chebyshev basis; (2) a fixed basis means we can automatically apply recurrence relations between Chebyshev polynomials (and later, *ultraspherical polynomials*); (3) we can easily incorporate variable coefficients; (4) the structure of the linear systems does not depend on the boundary condition(s), allowing for a fast, general solver (see Section 5); and (5) it seems unlikely that basis recombination can be formulated in a stable way for very high order boundary conditions.

Chebyshev polynomials or, more generally, Jacobi polynomials have been abundantly used to construct spectral coefficient methods. The tau-method can deal with variable coefficients, but results in dense matrices [14]. The spectral collocation method, as implemented in `Chebfun`, automatically resolves a solution to a differential equation, but suffers from ill-conditioning [8]. The dual-Petrov–Galerkin spectral method is efficient, but not for general variable coefficient differential equations [7, 11, 18].

We mention that the solution of linear equations with variable coefficients is absolutely critical to the solution of nonlinear problems. While we focus on linear equations in this paper, numerical experiments confirm that the approach is applicable to nonlinear problems as well, continuing to achieve stability. However, the diagonal form of variable coefficients in collocation methods may prove computationally more effective for certain problems, though at the expense of accuracy.

This paper is organised as follows. In the next section we construct the method for first order differential equations and apply it to two problems with highly oscillatory variable coefficients. In the third section we extend the approach to higher order differential equations by using ultraspherical polynomials. We also present numerical results for challenging second and higher-order differential equations. In the fourth section, we prove stability and convergence of the method in high order norms, which reduce to the standard 2-norm for Dirichlet boundary conditions. In section five, we present a fast, stable algorithm to solve the almost banded matrices in $\mathcal{O}(m^2 n)$ operations; and in the final section we describe directions for future research.

2. Chebyshev Polynomials and First Order Differential Equations. For pedagogical reasons, we begin by solving first-order differential equations of the form

$$u'(x) + a(x)u(x) = f(x) \qquad \text{and} \qquad u(-1) = c \tag{2.1}$$

where $a : [-1, 1] \to \mathbb{C}$ and $f : [-1, 1] \to \mathbb{C}$ are continuous functions with bounded variation. The continuity assumption ensures that (2.1) has a unique continuously differentiable solution on the unit interval [16]. In practice, one could solve such equations using quadrature on the integral formulation of the problem.

Suppose that $g(x)$ is a continuous function with bounded variation on $[-1, 1]$. Then g has a unique representation as a uniformly convergent Chebyshev expansion [12, Thm. 5.7]. That is,

$$g(x) = \sum_{k=0}^{\infty} g_k T_k(x) \tag{2.2}$$

where

$$g_k = \frac{2}{\pi}\int_{-1}^{1} \frac{g(x)T_k(x)}{\sqrt{1-x^2}}dx, \quad k \geq 1 \qquad \text{and} \qquad g_0 = \frac{1}{\pi}\int_{-1}^{1} \frac{g(x)}{\sqrt{1-x^2}}dx. \tag{2.3}$$

One way to approximate $g(x)$ is to truncate (2.2) after the first m terms, to obtain the polynomial

$$g_{\text{trunc}}(x) = \sum_{k=0}^{m-1} g_k T_k(x), \tag{2.4}$$

assuming we know, or can calculate, g_k. A second approach is to interpolate $g(x)$ at m Chebyshev points, to obtain the polynomial

$$g_{\text{interp}}(x) = \sum_{k=0}^{m-1} \tilde{g}_k T_k(x). \tag{2.5}$$

The coefficients $\{\tilde{g}_k\}$ and $\{g_k\}$ are related by an aliasing formula stated by Clenshaw and Curtis [5]. Usually, in practice, $g(x)$ will be many times differentiable on $[-1,1]$, and then (2.4) and (2.5) converge uniformly to $g(x)$ at an algebraic rate as $m \to \infty$. Moreover, the convergence is spectral (super-algebraic) when g is infinitely differentiable, which improves to be exponential when g is analytic in a neighborhood of $[-1,1]$. If g is entire, the convergence rate improves further to be super-exponential.

Mathematically, we seek the Chebyshev coefficients of the truncation, (2.4), of $a(x)$ and $f(x)$ which are defined by the integrals (2.3). However, we use the coefficients in the polynomial interpolant as an approximation, since coefficients in (2.5) are faster to compute and match to machine precision for large enough m. The degree of the polynomial used to approximate $a(x)$ will later be closely related to the bandwidth of the almost banded linear system we form to solve the differential equation (2.1).

The spectral method in this paper solves for the coefficients of the solution in a Chebyshev series. In order to achieve this, we need to be able to represent differentiation, $u'(x)$, and multiplication, $a(x)u(x)$, in terms of operators on coefficients.

2.1. First Order Differentiation Operator. The derivatives of Chebyshev polynomials satisfy

$$\frac{dT_k}{dx} = \begin{cases} kC_{k-1}^{(1)} & k \geq 1 \\ 0 & k = 0 \end{cases} \tag{2.6}$$

where $C_{k-1}^{(1)}(x)$ is the Chebyshev polynomial of the second kind of degree $k-1$. We use $C^{(1)}$ instead of U to highlight the connection to ultraspherical polynomials, which are key to extending the method to higher order differential equations.

Now, we use (2.6) to derive a simple expression for the derivative of a Chebyshev series. Suppose that $u(x)$ is given by the Chebyshev series

$$u(x) = \sum_{k=0}^{\infty} u_k T_k(x). \tag{2.7}$$

Then

$$u'(x) = \sum_{k=1}^{\infty} k u_k C_{k-1}^{(1)}(x).$$

In other words, the vector of $C^{(1)}$ coefficients of the derivative is given by $\mathcal{D}_0\mathbf{u}$ where $\mathcal{D}_0$ is the differentiation operator

$$\mathcal{D}_0 = \begin{pmatrix} 0 & 1 & & & \\ & 2 & & & \\ & & 3 & & \\ & & & \ddots & \end{pmatrix} \tag{2.8}$$

and $\mathbf{u}$ is the vector of Chebyshev coefficients for $u(x)$. Note that this differentiation operator is sparse, in stark contrast to the classic differentiation operator in spectral collocation methods [2].

2.2. Multiplication Operator for Chebyshev Series. In order to handle variable coefficients of the form $a(x)u(x)$ in (2.1) we need to represent the multiplication of two Chebyshev series as an operator on coefficients. To this end, let

$$a(x) = \sum_{j=0}^{\infty} a_j T_j(x) \qquad \text{and} \qquad u(x) = \sum_{k=0}^{\infty} u_k T_k(x).$$

Multiplying these two expressions together we obtain

$$a(x)u(x) = \sum_{j=0}^{\infty}\sum_{k=0}^{\infty} a_j u_k T_j(x)T_k(x) = \sum_{k=0}^{\infty} c_k T_k(x),$$

and desire an explicit form for the c_k's. By Proposition 2.1 of [1] we have that

$$c_k = \begin{cases} a_0u_0 + \frac{1}{2}\sum_{l=1}^{\infty} a_l u_l & k = 0 \\ \frac{1}{2}\sum_{l=0}^{k-1} a_{k-l}u_l + a_0u_k + \frac{1}{2}\sum_{l=1}^{\infty} a_l u_{l+j} + \frac{1}{2}\sum_{l=0}^{\infty} a_{l+k}u_l & k \geq 1. \end{cases} \tag{2.9}$$

This can be represented by a Toeplitz plus an almost Hankel operator since $\mathbf{c} = \mathcal{M}_0[a]\mathbf{u}$ where

$$\mathcal{M}_0[a] = \frac{1}{2}\left[\begin{pmatrix} 2a_0 & a_1 & a_2 & a_3 & \dots \\ a_1 & 2a_0 & a_1 & a_2 & \ddots \\ a_2 & a_1 & 2a_0 & a_1 & \ddots \\ a_3 & a_2 & a_1 & 2a_0 & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 & \dots \\ a_1 & a_2 & a_3 & a_4 & \iddots \\ a_2 & a_3 & a_4 & a_5 & \iddots \\ a_3 & a_4 & a_5 & a_6 & \iddots \\ \vdots & \iddots & \iddots & \iddots & \iddots \end{pmatrix}\right].$$

At first glance, it appears that the multiplication operator and any truncation are dense. However, since $a(x)$ is continuous with bounded variation, we are able to uniformly approximate $a(x)$ with a finite number of Chebyshev coefficients to any desired accuracy. That is, for any $\epsilon > 0$ there exists an $m \in \mathbb{N}$ such that

$$\left\| a(x) - \sum_{k=0}^{m-1} a_k T_k(x) \right\|_{L_\infty([-1,1])} < \epsilon.$$

As long as m is large enough, to all practical purposes, we can use the truncated Chebyshev series to replace $a(x)$. Recall, in our implementation we approximate

this truncation by the polynomial interpolant of the form (2.5). Hence, the $n \times n$ principal part of $\mathcal{M}_0[a]$ is banded with bandwidth m for $n > m$. Moreover, m can be surprisingly small when $a(x)$ is analytic or many times differentiable.

There is still one ingredient absent: The operator $\mathcal{D}_0$ returns coefficients in a $C^{(1)}$ series, whereas the operator $\mathcal{M}_0[a]$ returns coefficients in a Chebyshev series. In order to correct this, we require an operator that maps coefficients in a Chebyshev series to those in a $C^{(1)}$ series.

2.3. Transformation Operator for Chebyshev Series. The Chebyshev polynomials $T_k(x)$ can be written in terms of the $C^{(1)}$ polynomials by the recurrence relation

$$T_k = \begin{cases} \frac{1}{2}\left(C_k^{(1)} - C_{k-2}^{(1)}\right) & k \geq 2 \\ \frac{1}{2}C_1^{(1)} & k = 1 \\ C_0^{(1)} & k = 0. \end{cases} \tag{2.10}$$

A more general form of (2.10), which we will use later, is given in [13]. Suppose that $u(x)$ is given by the Chebyshev series (2.7). Then, using (2.10) we have

$$\begin{aligned} u(x) = \sum_{k=0}^{\infty} u_k T_k(x) &= u_0 C_0^{(1)}(x) + \frac{1}{2}u_1 C_1^{(1)}(x) + \frac{1}{2}\sum_{k=2}^{\infty} u_k \left(C_k^{(1)}(x) - C_{k-2}^{(1)}(x)\right) \\ &= \left(u_0 - \frac{1}{2}u_2\right) C_0^{(1)}(x) + \sum_{k=1}^{\infty} \frac{1}{2}\left(u_k - u_{k+2}\right) C_k^{(1)}(x). \end{aligned}$$

Hence, the $C^{(1)}$ coefficients for $u(x)$ are $\mathcal{S}_0\mathbf{u}$ where $\mathcal{S}_0$ is the transformation operator

$$\mathcal{S}_0 = \begin{pmatrix} 1 & & -\frac{1}{2} & & & \\ & \frac{1}{2} & & -\frac{1}{2} & & \\ & & \frac{1}{2} & & -\frac{1}{2} & \\ & & & \ddots & & \ddots \end{pmatrix} \tag{2.11}$$

and $\mathbf{u}$ is the vector of Chebyshev coefficient for $u(x)$. Note that this transformation operator, and any truncation of it, is sparse and banded.

2.4. Discretization of the system. Now, we have all the ingredients to solve any differential equation of the form (2.1). Firstly, we can represent the differential operator as

$$\mathcal{L} := \mathcal{D}_0 + \mathcal{S}_0\mathcal{M}_0[a],$$

which takes coefficients in a Chebyshev series to those in a $C^{(1)}$ series. Due to this fact, the right-hand side $f(x)$ must be expressed in terms of its coefficients in a $C^{(1)}$ series. Hence, ignoring the boundary condition, we can represent the differential equation (2.1) as

$$\mathcal{L}\mathbf{u} = \mathcal{S}_0\mathbf{f}.$$

In the above, the vectors $\mathbf{u}$ and $\mathbf{f}$ are the vectors of coefficients in the Chebyshev series of the form (2.2) for $u(x)$ and $f(x)$, respectively.

We truncate the operators to derive a practical numerical scheme. To this end, define the $n \times \infty$ projection operator as

$$\mathcal{P}_n = (I_n, \mathbf{0}). \tag{2.12}$$

Truncating the differentiation operator to $\mathcal{P}_n \mathcal{D}_0 \mathcal{P}_n^\top$ results in an $n \times n$ matrix with a zero last row. This observation motivates us to impose the boundary condition by replacing the last row of $\mathcal{P}_n \mathcal{L} \mathcal{P}_n^\top$. We take the convention of permuting this boundary row to the first row so that the linear system is close to upper triangular. That is, in order to obtain an approximate solution to (2.1) we solve the system

$$A_n \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_{n-1} \end{pmatrix} = \begin{pmatrix} c \\ \\ \left(\mathcal{P}_{n-1} \mathcal{S}_0 \mathcal{P}_n^\top\right) \left(\mathcal{P}_n \mathbf{f}\right) \\ \\ \end{pmatrix}, \tag{2.13}$$

with

$$A_n = \begin{pmatrix} T_0(-1) & T_1(-1) & \dots & T_{n-1}(-1) \\ & \mathcal{P}_{n-1} \mathcal{L} \mathcal{P}_n^\top & & \end{pmatrix}.$$

(Note that $T_k(-1) = (-1)^k$.) The solution $u(x)$ is then approximated by the computed solution,

$$\tilde{u}(x) = \sum_{k=0}^{n-1} u_k T_k(x).$$

Remark In practice, one would typically generate the spectral system by discretizing each operator individually; i.e., instead of A_n we would have

$$\mathcal{P}_{n-1} \mathcal{D}_0 \mathcal{P}_n^\top + \left(\mathcal{P}_{n-1} \mathcal{S}_0 \mathcal{P}_n^\top\right) \left(\mathcal{P}_n \mathcal{M}[a] \mathcal{P}_n^\top\right).$$

We, however, use A_n for simplicity, and because we can generate it explicitly:

$$A_n = \mathcal{P}_{n-1} \mathcal{D}_n \mathcal{P}_n^\top + \left(\mathcal{P}_{n-1} \mathcal{S}_0 \mathcal{P}_{n+2}^\top\right) \left(\mathcal{P}_{n+2} \mathcal{M}[a] \mathcal{P}_n^\top\right).$$

We now use this method to solve two first-order differential equations.

2.5. Numerical Examples. We solve the linear system (2.13) for progressively larger n, and terminate the process when the tail of the solution falls below the relative magnitude of machine epsilon. This automatic resolution of the solution to a differential equation is similar to the one used in `Chebfun` [8]. However, `Chebfun` has to be careful how fast it increases n, due to ill-conditioning of the spectral collocation matrices. Our approach does not have this drawback.

For the first example, we consider the linear differential equation

$$u'(x) + x^3 u(x) = 100 \sin(20{,}000 x^2) \qquad u(-1) = 0 \tag{2.14}$$

which has a highly oscillatory forcing term. The exact solution is

$$u(x) = e^{-\frac{x^4}{4}} \left(\int_{-1}^{x} 100 e^{\frac{t^4}{4}} \sin(20{,}000 t^2) dt \right).$$

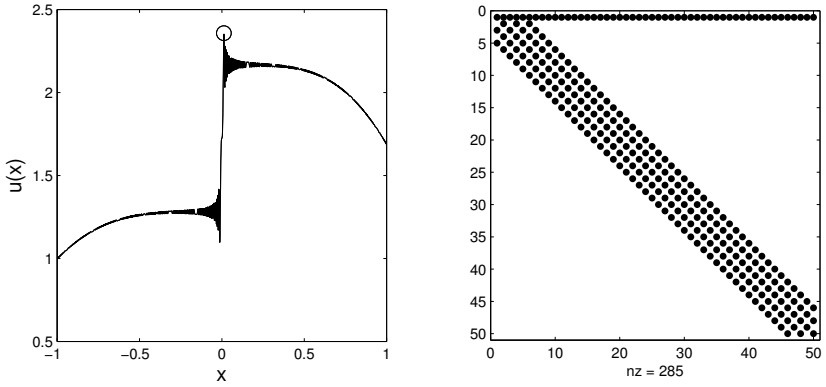


Fig. 2.1: Left: Plot of the highly oscillatory solution to (2.14) with the maximum of the solution computed and marked by a circle. Right: Spy plot of the almost banded linear system for $n = 50$.

The computed solution, which is a 20,391 degree polynomial, uniformly approximates the exact solution to ten times machine epsilon. In Figure 2.1 we plot the computed oscillatory solution, and a realization of the matrix when $n = 50$. This `spy` plot shows the almost banded structure of the linear system.

The approximate solution is expressed in terms of a Chebyshev basis, which is convenient for further manipulation, e.g. using `Chebfun`. For example, its maximum is 2.3573 (circled in Figure 2.1), its integral is 3.2879 and the equation $u(x) = 1.3$ has 113 solutions.

As a second example we consider the linear differential equation

$$u'(x) + \frac{1}{ax^2 + 1} u(x) = 0 \qquad \text{and} \qquad u(-1) = 1. \tag{2.15}$$

We take $a = 5 \times 10^4$, in which case the variable coefficient can be approximated to roughly machine precision by a polynomial of degree 7,350, and hence, for very large n, the linear system (2.13) is banded. The exact solution to (2.15) is

$$u(x) = \exp\left(-\frac{\tan^{-1}(\sqrt{a}x) + \tan^{-1}(\sqrt{a})}{\sqrt{a}} \right),$$

which can be approximated to machine precision by a Chebyshev polynomial of degree 5,377. The computed solution $\tilde{u}(x)$, is a Chebyshev polynomial of degree 5,093 and

$$\left(\int_{-1}^{1} \left(u(x) - \tilde{u}(x) \right)^2 dx \right)^{\frac{1}{2}} = 2.86 \times 10^{-15}$$

A plot of the solution and the Cauchy error are included in Figure 2.2. The Cauchy error plot confirms that the solution is, up to machine precision, independent of the number of coefficients in its expansion for $n \geq 5{,}100$.

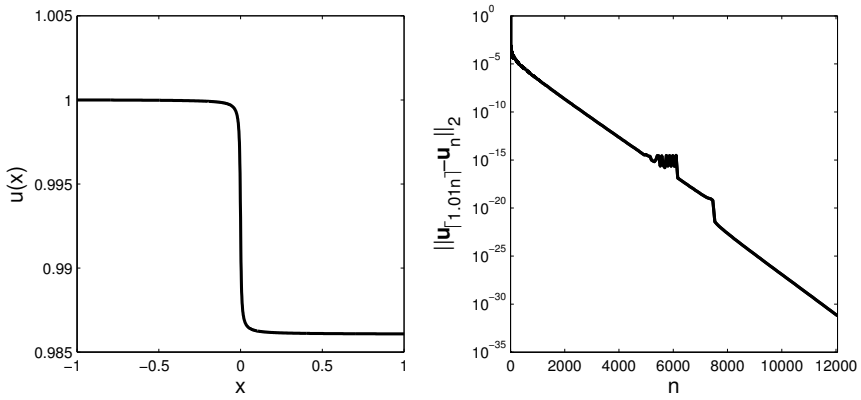


Fig. 2.2: Left: Plot of the computed solution to (2.15) with $a = 5 \times 10^4$. Right: Plot of the Cauchy error for the solution coefficients. The plot shows the 2-norm difference between the coefficients of the approximate solution when solving an $n \times n$ and $\lceil 1.01n \rceil \times \lceil 1.01n \rceil$ matrix system.

3. Ultraspherical polynomials and higher order differential equations. We now generalize the approach to higher order differential equations of the form

$$\sum_{\lambda=0}^{N} a^{\lambda}(x) \frac{d^{\lambda} u(x)}{dx^{\lambda}} = f(x) \qquad \text{on } [-1, 1], \tag{3.1}$$

with general boundary conditions $\mathcal{B}u = \mathbf{c}$. We assume that the boundary operator $\mathcal{B}$ is given in terms of the Chebyshev coefficients of u. For example, the Dirichlet boundary operator is

$$\mathcal{B} = \begin{pmatrix} T_0(-1) & T_1(-1) & T_2(-1) & \cdots \\ T_0(1) & T_1(1) & T_2(1) & \cdots \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 & -1 & \cdots \\ 1 & 1 & 1 & 1 & \cdots \end{pmatrix},$$

while the Neumann operator is

$$\mathcal{B} = \begin{pmatrix} T_0'(-1) & T_1'(-1) & T_2'(-1) & \cdots \\ T_0'(1) & T_1'(1) & T_2'(1) & \cdots \end{pmatrix} = \begin{pmatrix} 0 & 1 & -4 & \cdots & (-1)^{k+1}k^2 & \cdots \\ 0 & 1 & 4 & \cdots & k^2 & \cdots \end{pmatrix}.$$

We continue to impose that all the variable coefficients are continuous with bounded variation to ensure that each coefficient can be represented by a finite Chebyshev series.

The approach of the first order method relied on the three relations: differentiation (2.6), multiplication (2.9) and transformation (2.10). To generalize the spectral method to higher order differential equations we use similar relations, now in terms of higher order ultraspherical polynomials.

The ultraspherical (or Gegenbauer) polynomials $C_0^{(\lambda)}(x), C_1^{(\lambda)}(x), \ldots$ are a family of polynomials orthogonal with respect to the weight

$$(1 - x^2)^{\lambda - \frac{1}{2}}.$$

We will only use ultraspherical polynomials for $\lambda = 1, 2, \ldots$, defined uniquely by normalizing the leading coefficient so that

$$C_k^{(\lambda)}(x) = \frac{2^k(\lambda)_k}{k!}x^k + \mathcal{O}(x^{k-1}),$$

where $(\lambda)_k = \frac{(\lambda+k-1)!}{(\lambda-1)!}$ denotes the *Pochhammer symbol*. In particular, the ultraspherical polynomials with $\lambda = 1$ are the Chebyshev polynomials of the second kind, which we denote by $C^{(1)}$.

Importantly, ultraspherical polynomials satisfy an analogue of (2.6), as stated in the *NIST Handbook of Special Functions* [13]. That is, for $\lambda \geq 1$,

$$\frac{dC_k^{(\lambda)}}{dx} = \begin{cases} 2\lambda C_{k-1}^{(\lambda+1)} & k \geq 1 \\ 0 & k = 0. \end{cases} \tag{3.2}$$

Moreover, they also satisfy an analogue to (2.10) for $\lambda \geq 1$:

$$C_k^{(\lambda)} = \begin{cases} \frac{\lambda}{\lambda+k}\left(C_k^{(\lambda+1)} - C_{k-2}^{(\lambda+1)}\right) & k \geq 2 \\ \frac{\lambda}{\lambda+1}C_1^{(\lambda+1)} & k = 1 \\ C_0^{(\lambda+1)} & k = 0. \end{cases} \tag{3.3}$$

Suppose that $u(x)$ is represented as the Chebyshev series (2.7). Then for $\lambda = 1, 2, \ldots$ we have, by (2.6),

$$\frac{d^\lambda u(x)}{dx^\lambda} = \sum_{k=1}^{\infty} k u_k \frac{d^{\lambda-1}C_{k-1}^{(1)}(x)}{dx^{\lambda-1}}.$$

We now apply the relation (3.2) $\lambda - 1$ times to obtain

$$\frac{d^\lambda u(x)}{dx^\lambda} = 2^{\lambda-1}(\lambda-1)!\sum_{k=\lambda}^{\infty} k u_k C_{k-\lambda}^{(\lambda)}(x).$$

This means that the λ-order differentiation operator takes the form

$$\mathcal{D}_\lambda = 2^{\lambda-1}(\lambda-1)!\begin{pmatrix} \overbrace{0 \quad \cdots \quad 0}^{\lambda \text{ times}} & \lambda & & & \\ & & \lambda+1 & & \\ & & & \lambda+2 & \\ & & & & \ddots \end{pmatrix}. \tag{3.4}$$

In the process of differentiation, $\mathcal{D}_\lambda$ converts coefficients in a Chebyshev series to coefficients in a $C^{(\lambda)}$ series. Moreover, using (3.3) the operator which transforms coefficients in an ultraspherical series with parameter λ to those in a series with parameter $\lambda + 1$ is

$$\mathcal{S}_\lambda = \begin{pmatrix} 1 & & -\frac{\lambda}{\lambda+2} & & \\ & \frac{\lambda}{\lambda+1} & & -\frac{\lambda}{\lambda+3} & \\ & & \frac{\lambda}{\lambda+2} & & -\frac{\lambda}{\lambda+4} \\ & & & \ddots & & \ddots \end{pmatrix}. \tag{3.5}$$

As before, we also require a multiplication operator. Since $\mathcal{D}_\lambda$ returns coefficients in a $C^{(\lambda)}$ series we need a multiplication operator that performs multiplication between two $C^{(\lambda)}$ ultraspherical series.

3.1. Multiplication Operator for Ultraspherical Series. In order to handle the variable coefficients in (3.1), we must represent multiplication of two ultraspherical series in coefficient space. Given two functions

$$a(x) = \sum_{j=0}^{\infty} a_j C_j^{(\lambda)}(x) \qquad \text{and} \qquad u(x) = \sum_{k=0}^{\infty} u_k C_k^{(\lambda)}(x)$$

we have

$$a(x)u(x) = \sum_{j=0}^{\infty}\sum_{k=0}^{\infty} a_j u_k C_j^{(\lambda)}(x) C_k^{(\lambda)}(x). \tag{3.6}$$

To obtain a $C^{(\lambda)}$ series for $a(x)u(x)$ we use the linearization formula shown by Carlitz [4]. The linearization formula is

$$C_j^{(\lambda)}(x) C_k^{(\lambda)}(x) = \sum_{s=0}^{\min(j,k)} c_s^{\lambda}(j,k) C_{j+k-2s}^{(\lambda)}(x), \tag{3.7}$$

where

$$c_s^{\lambda}(j,k) = \frac{j+k+\lambda-2s}{j+k+\lambda-s} \frac{(\lambda)_s (\lambda)_{j-s} (\lambda)_{k-s}}{s!(j-s)!(k-s)!} \frac{(2\lambda)_{j+k-s}}{(\lambda)_{j+k-s}} \frac{(j+k-2s)!}{(2\lambda)_{j+k-2s}} \tag{3.8}$$

and $(\lambda)_k = \frac{(\lambda+k-1)!}{(\lambda-1)!}$ is the Pochhammer symbol. We substitute (3.7) into (3.6) and rearrange the summation signs to obtain

$$a(x)u(x) = \sum_{j=0}^{\infty} \left(\sum_{k=0}^{\infty} \sum_{s=\max(0,k-j)}^{k} a_{2s+j-k} c_s^{\lambda}(k, 2s+j-k) u_k \right) C_j^{(\lambda)}(x). \tag{3.9}$$

From (3.9) the (j,k) entry of the multiplication operator representing the product of $a(x)$ in a $C^{(\lambda)}$ series is

$$\mathcal{M}_{\lambda}[a]_{j,k} = \sum_{s=\max(0,k-j)}^{k} a_{2s+j-k} c_s^{\lambda}(k, 2s+j-k), \qquad j,k \geq 0.$$

Just as before, in practice, $a(x)$ will be represented by a finite series. That is,

$$a(x) \approx \sum_{j=0}^{m-1} a_j C_j^{(\lambda)}(x), \tag{3.10}$$

and with this truncation of the $C^{(\lambda)}$ series for $a(x)$ the matrix $\mathcal{P}_n \mathcal{M}_{\lambda}[a] \mathcal{P}_n^{\top}$ is banded with bandwidth m for $n > m$. The expansion (3.10) can be computed by approximating the first m Chebyshev coefficients in the Chebyshev series for $a(x)$ and then applying a truncation of the transformation operator $\mathcal{S}_{\lambda-1} \cdots S_0$.

The formula for $c_s^{\lambda}(j,k)$, (3.8), cannot be used directly to form $\mathcal{M}_{\lambda}[a]$ due to arithmetic overflow problems that arise for j or k greater than 70. Instead, we cancel

terms in the numerator and denominator of (3.8) and match up the remaining terms of similar magnitude. The following relation holds:

$$c_s^\lambda(j,k) = \frac{j+k+\lambda-2s}{j+k+\lambda-s} \times \prod_{t=0}^{s-1} \frac{\lambda+t}{1+t} \times \prod_{t=0}^{j-s-1} \frac{\lambda+t}{1+t} \times \prod_{t=0}^{s-1} \frac{2\lambda+j+k-2s+t}{\lambda+j+k-2s+t} \times \prod_{t=0}^{j-s-1} \frac{k-s+1+t}{k-s+\lambda+t}. \tag{3.11}$$

All the fractions are of magnitude $\mathcal{O}(1)$ in size, and hence $\mathcal{P}_n \mathcal{M}_\lambda[a] \mathcal{P}_n^\top$ can be formed in a completely stable way. For the purposes of computational speed, we only apply (3.11) once per entry, and use the recurrence relation

$$c_{s+1}^\lambda(j,k+2) = c_s^\lambda(j,k) \times \frac{j+k+\lambda-s}{j+k+\lambda-s+1} \times \frac{\lambda+s}{s+1} \times \times \frac{j-s}{\lambda+j-s-1} \times \frac{2\lambda+j+k-s}{\lambda+j+k-s} \times \frac{k-s+\lambda}{k-s+1}$$

to generate all the other terms required.

Remark In the special case when $\lambda = 1$, the multiplication operator $\mathcal{M}_1[a]$ can be decomposed as a Toeplitz operator plus a Hankel operator.

3.2. Discretization of the system. We now have everything in place to be able to solve higher order differential equations of the form (3.1). Firstly, we can represent the differential operator as

$$\mathcal{L} := \mathcal{M}_N[a^N]\mathcal{D}_N + \sum_{\lambda=1}^{N-1} \mathcal{S}_{N-1} \cdots \mathcal{S}_\lambda \mathcal{M}_\lambda[a^\lambda]\mathcal{D}_\lambda + \mathcal{S}_{N-1} \cdots \mathcal{S}_0 \mathcal{M}_0[a^0],$$

which takes coefficients in a Chebyshev series to those in a $C^{(N)}$ series. Due to this fact, the right hand side $f(x)$ must be expressed in terms of its coefficients in a $C^{(N)}$ series. Moreover, we impose the K boundary conditions on the solution by replacing the last K rows of $\mathcal{P}_n \mathcal{L} \mathcal{P}_n^\top$. We again take the convention of permuting these to the first K rows. That is, in order to obtain an approximate solution to (3.1) we solve the system

$$A_n \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_{n-1} \end{pmatrix} = \begin{pmatrix} \mathbf{c} \\ \\ \mathcal{P}_{n-K} \mathcal{S}_{N-1} \cdots \mathcal{S}_0 \mathbf{f} \\ \\ \end{pmatrix}, \tag{3.12}$$

where

$$A_n = \begin{pmatrix} \mathcal{B}\mathcal{P}_n^\top \\ \\ \mathcal{P}_{n-K} \mathcal{L} \mathcal{P}_n^\top \\ \\ \end{pmatrix}$$

and $\mathbf{f}$ is again a vector containing the Chebyshev coefficients of the right-hand side f. The solution $u(x)$ is then approximated by the n-term Chebyshev series:

$$u(x) \approx \sum_{k=0}^{n-1} u_k T_k(x).$$

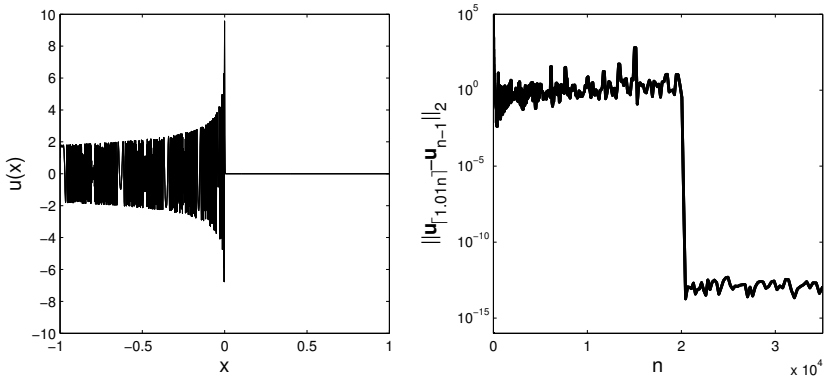


Fig. 3.1: Left: Plot of the highly oscillatory solution to (3.13) with $\epsilon = 10^{-9}$. Right: Plot of the Cauchy error for the solution coefficients. The plot shows the 2-norm difference between the coefficients of the approximate solution when solving an $n \times n$ and $\lceil 1.01n \rceil \times \lceil 1.01n \rceil$ matrix system.

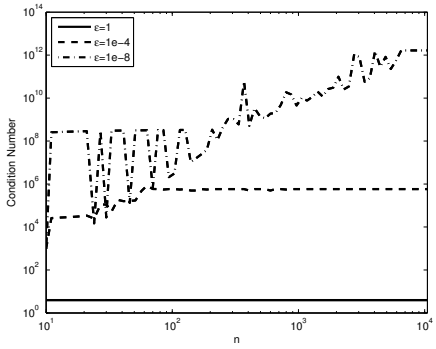


Fig. 3.2: Plot of the condition number of the spectral systems that our method forms against the size of the matrix system, for: (solid) $\epsilon = 1 \times 10^{-8}$, (dashed) $\epsilon = 1 \times 10^{-4}$ and (dot-dashed) $\epsilon = 1$. The plot demonstrates boundedness of the 2-norm condition number. The observed error out performs what is suggested by the size of the constants.

3.3. Numerical Examples. For the first example we consider the Airy differential equation

$$\epsilon u''(x) - xu(x) = 0 \qquad \text{and} \qquad u(-1) = \mathrm{Ai}\left(-\sqrt[3]{\frac{1}{\epsilon}}\right),\ u(1) = \mathrm{Ai}\left(\sqrt[3]{\frac{1}{\epsilon}}\right) \tag{3.13}$$

where $\mathrm{Ai}(\cdot)$ is the Airy function of the first kind.

In Figure 3.1 we take $\epsilon = 10^{-9}$ and plot the computed solution which is a poly-

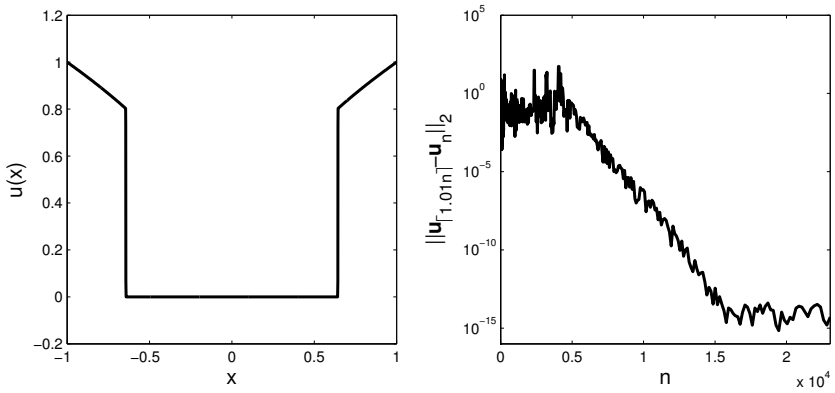


Fig. 3.3: Left: Plot of the solution to the boundary layer problem (3.14) with $\epsilon = 10^{-7}$. Right: Plot of the Cauchy error in the 2-norm for the solution coefficients.

nomial of degree 20,003. The exact solution to (3.13) is the scaled Airy function,

$$u(x) = \mathrm{Ai}\left(\sqrt[3]{\frac{1}{\epsilon}}x\right).$$

Letting $\tilde{u}(x)$ denote the computed solution, we have

$$\left(\int_{-1}^{1} \left(u(x) - \tilde{u}(x)\right)^2\right)^{1/2} = 1.07 \times 10^{-11}.$$

The magnitude of this L_2 error is due to other numerical issues related to the size of the approximation and the high oscillations in the true solution, and the Cauchy error plot in Figure 3.1 indicates that the solution coefficients themselves are resolved to about machine precision, for $n \geq 20{,}000$. In Figure 3.2 we show numerical evidence that the condition number of the linear systems we form are bounded for all n. Later, for $\epsilon = 1$, we show in Figure 5.2 that the derivatives of the solution are also well approximated.

For the second example we consider the boundary layer problem,

$$\epsilon u''(x) - 2x\left(\cos(x) - \frac{8}{10}\right)u'(x) + \left(\cos(x) - \frac{8}{10}\right)u(x) = 0 \tag{3.14}$$

with boundary conditions

$$u(-1) = u(1) = 1.$$

Perturbation theory shows that the solution has two boundary layers at $\pm\cos^{-1}(.8)$ both of width $\mathcal{O}(\epsilon^{1/4})$. In Figure 3.3 we take $\epsilon = 10^{-7}$. The computed solution is of degree 15,394, and it is, again, confirmed by the Cauchy error plot that the solution is well-resolved.

For the last example we consider the high order differential equation

$$u^{(10)}(x) + \cosh(x)u^{(8)}(x) + x^2u^{(6)}(x) + x^4u^{(4)}(x) + \cos(x)u^{(2)}(x) + x^2u(x) = 0$$

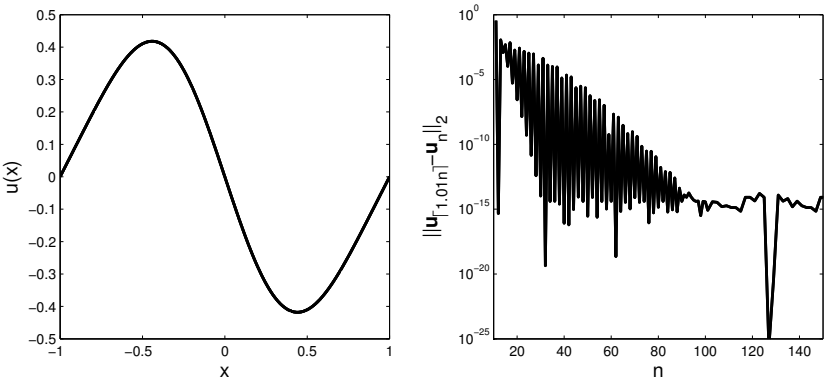


Fig. 3.4: Left: Plot of the solution to the 10th order differential equation. Right: Plot of the Cauchy error in the 2-norm for the solution coefficients.

with boundary conditions

$$u(-1) = u(1) = 0,\ u'(-1) = u'(-1) = 1,\ u^{(k)}(\pm 1) = 0,\ 2 \le k \le 4.$$

This is far from a practical example, and an exact solution seems difficult (if not impossible) to come by. Instead, we note that if $u(x)$ is the solution then it is odd; that is, $u(x) = -u(-x)$. Our method does not impose such a condition and therefore, we can use it along with the Cauchy error to gain confidence in the computed solution. The computed solution $\tilde{u}(x)$ is of degree 55 and plotted in Figure 3.4. Moreover, the computed solution is odd to about machine precision,

$$\left(\int_{-1}^{1} \left(\tilde{u}(x) + \tilde{u}(-x) \right)^2 \right)^{1/2} = 1.252 \times 10^{-14}.$$

4. Stability and convergence. The 2-norm condition number of a matrix $A \in \mathbb{C}^{n \times n}$ is defined as

$$\kappa(A) = \|A\|_2 \left\|A^{-1}\right\|_2.$$

In numerical experiments, $\kappa\left(A_n\right)$ grows proportionally with n, which is significantly better than the typical growth of $\mathcal{O}(n^{2N})$ in the condition number for the standard tau and collocation methods (see section 4.3 of [3]). However, the accuracy seen in practice even outperforms this: the backward error is consistent with a numerical method with bounded condition number. Later, we will show that a trivial, diagonal preconditioner results in a linear system with bounded condition number. Before that, we state the following lemma that explains the stability we observe in practice when solving ill-conditioned systems with Gaussian elimination. In section 5 we will present a stable algorithm that can solve the almost banded systems (with bandwidth m) with only $\mathcal{O}(m^2 n)$ operations.

LEMMA 4.1. *Suppose D is a diagonal matrix where all its entries are powers of the machine base. Then Gaussian elimination with partial pivoting (GEP) applied to solve $A\mathbf{c} = \mathbf{b}$ is stable if GEP applied to solve $AD\mathbf{q} = \mathbf{b}$ is stable and $\|D\|_\infty = \mathcal{O}(1)$.*

Proof. The diagonal matrix D, applied on the right of A scales the columns by its diagonal entries and this scaling is done without error. With partial pivoting, the choice of pivot is not affected by exact column scaling and hence, the computed unit lower triangular matrix $\tilde{L}^{-1}$ is exactly the same in the computed LU decompositions of A and AD. Moreover, the permutation matrices are identical. Therefore, the right-hand side after applying these matrices is the same, and thus we are solving

$$(U + \delta U)(\mathbf{c} + \delta\mathbf{c}) = \mathbf{r} + \delta\mathbf{r} \text{ and}$$
$$(U + \delta U)D(\mathbf{q} + \delta\mathbf{q}) = \mathbf{r} + \delta\mathbf{r},$$

where $\delta\mathbf{r}$, δU and $\delta\mathbf{q}$ are small by assumption.

We now show the error in backward substitution is small. We first note that

$$c_k = d_k q_k \text{ and } c_k + \delta c_k = d_k(q_k + \delta q_k),$$

and hence

$$\delta c_k = d_k \delta q_k$$

is small. □

If the column scaling is not a power of the machine base then the effect on the choice of pivots is hard to quantify [10].

4.1. A diagonal preconditioner and compactness. Through-out this section we assume that $a^N(x) = 1$ (otherwise, divide through by the coefficient on the highest order term). We make the restriction that $K = N$; that is, the Nth order differential equation has exactly N boundary conditions. When $K \leq N$ it is more appropriate to choose a non-diagonal preconditioner, but we do not analysis that situation here.

We show that there exists a diagonal preconditioner so that the preconditioned system has bounded condition number in higher order norms (Definition 4.2). For Dirichlet boundary conditions the preconditioned system has bounded condition number in the 2-norm. Any differential equation with Robin boundary conditions can be written as a system of differential equations with Dirichlet conditions, and effectively solved with bounded 2-norm condition number. However, this is undesirable because (i) reformulating the differential equation is difficult to automate; and (ii) to compute n coefficients of the desired solution, linear systems much larger than $n \times n$ have to be solved.

Define the diagonal preconditioner by,

$$\mathcal{R} = \frac{1}{2^{N-1}(N-1)!}\text{diag}\Big(\overbrace{1,\ldots,1}^{N \text{ times}}, \frac{1}{N}, \frac{1}{N+1}, \ldots\Big).$$

In practice, we observe that many other diagonal preconditioners also give a bounded condition number, and it is straightforward to construct such a preconditioner using only powers of the machine base. Moreover, it is likely that there are preconditioners which give much better practical bounds on the backward error.

The analysis will follow from the fact that, on suitably defined spaces,

$$\begin{pmatrix}\mathcal{B}\\ \mathcal{L}\end{pmatrix}\mathcal{R} = I + \mathcal{K},$$

for a compact operator $\mathcal{K}$. To this aim, we need to be precise on which spaces these operators act on. Since we are working in coefficient space, we will consider the problems as defined in ℓ^2_λ spaces:

DEFINITION 4.2. *$\ell^2_\lambda \subset \mathbb{C}^\infty$ is the Banach space with norm*

$$\|\mathbf{u}\|_{\ell^2_\lambda} = \sqrt{\sum_{k=0}^{\infty} |u_k|^2 (k+1)^{2\lambda}} < \infty.$$

LEMMA 4.3. *Suppose that $\mathcal{B} : \ell^2_D \to \mathbb{C}^K$ is bounded. Then*

$$\begin{pmatrix} \mathcal{B} \\ \mathcal{L} \end{pmatrix} \mathcal{R} = I + \mathcal{K}$$

where $\mathcal{K} : \ell^2_\lambda \to \ell^2_\lambda$ is compact for $\lambda = D-1, D, \ldots$.

Proof. Note that

$$\begin{pmatrix} \mathcal{B} \\ \mathcal{L} \end{pmatrix} = \begin{pmatrix} 2^{N-1}(N-1)!\mathcal{P}_N \\ \mathcal{D}_N \end{pmatrix} + \begin{pmatrix} \mathcal{B} - 2^{N-1}(N-1)!\mathcal{P}_N \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ \mathcal{S}_{N-1}\mathcal{M}_{N-1}[a^{N-1}]\mathcal{D}_{N-1} + \cdots + \mathcal{S}_{N-1}\cdots\mathcal{S}_1\mathcal{M}_1[a^1]\mathcal{D} + \mathcal{S}_{N-1}\cdots\mathcal{S}_0\mathcal{M}[a^0] \end{pmatrix}$$

where $\mathcal{P}_N : \ell^2_\lambda \to \mathbb{C}^N$ is again the $N \times \infty$ projection operator (2.12).

We first remark that $\mathcal{R} : \ell^2_\lambda \to \ell^2_{\lambda+1}$. Therefore, $\mathcal{B}\mathcal{R} : \ell^2_\lambda \to \mathbb{C}^N$ is bounded for $\lambda = D-1, D, \ldots$. Furthermore, we remark that $\mathcal{S}_k : \ell^2_\lambda \to \ell^2_{\lambda+1}$ are bounded for $k = 1, 2, \ldots$. Finally, $\mathcal{D}_N : \ell^2_\lambda \to \ell^2_{\lambda-1}$, and it follows that

$$\begin{pmatrix} 2^{N-1}(N-1)!\mathcal{P}_N \\ \mathcal{D}_N \end{pmatrix} \mathcal{R} = I : \ell^2_\lambda \to \ell^2_\lambda.$$

Since $\begin{pmatrix} \mathcal{B} - 2^{N-1}(N-1)!\mathcal{P}_N \\ 0 \end{pmatrix} \mathcal{R} : \ell^2_\lambda \to \ell^2_\lambda$ for $\lambda = D-1, D, \ldots$ is bounded and has finite rank, it is compact. Note that $\mathcal{R}$ is compact as an operator $\mathcal{R} : \ell^2_\lambda \to \ell^2_\lambda$, as are $\mathcal{S}_{N-1}, \ldots, \mathcal{S}_1$ (since $\mathcal{S}_k\mathcal{R}^{-1} : \ell^2_\lambda \to \ell^2_\lambda$ are bounded and $\mathcal{R}$ is compact). It follows that the last term

$$\begin{pmatrix} 0 \\ \mathcal{S}_{N-1}\cdots\mathcal{S}_0\mathcal{M}[a^0] \end{pmatrix} \mathcal{R} : \ell^2_\lambda \to \ell^2_\lambda$$

is compact, since $\mathcal{M}[a^0]$ and $\mathcal{S}_0$ are also bounded. Finally, each of the intermediate terms are compact since $\begin{pmatrix} 0 \\ \mathcal{D}_k \end{pmatrix} \mathcal{R} : \ell^2_\lambda \to \ell^2_\lambda$ is bounded and $\mathcal{S}_k$ are compact. ∎

The compactness of $\mathcal{K}$ allows us to show well-conditioning and convergence.

LEMMA 4.4. *Suppose that $\begin{pmatrix} \mathcal{B} \\ \mathcal{L} \end{pmatrix} : \ell^2_{\lambda+1} \to \ell^2_\lambda$ is an invertible operator for some $\lambda \in \{D-1, D, \ldots\}$. Then, as $n \to \infty$,*

$$\|A_n R_n\|_{\ell^2_\lambda} = \mathcal{O}(1) \text{ and } \|(A_n R_n)^{-1}\|_{\ell^2_\lambda} = \mathcal{O}(1),$$

for the diagonal matrix $R_n = \mathcal{P}_n \mathcal{R} \mathcal{P}_n^\top$ and truncated spectral matrix

$$A_n = \mathcal{P}_n \begin{pmatrix} \mathcal{B} \\ \mathcal{L} \end{pmatrix} \mathcal{P}_n^\top.$$

Proof. Since $\mathcal{R} : \ell^2_\lambda \to \ell^2_{\lambda+1}$ is trivially invertible, we have that $I + \mathcal{K} : \ell^2_\lambda \to \ell^2_\lambda$ is invertible. The lemma follows since $\mathcal{K}$ is compact, $A_n R_n = A_n \mathcal{P}_n \mathcal{R} \mathcal{P}_n^\top = I + \mathcal{P}_n \mathcal{K} \mathcal{P}_n^\top$, and $\mathcal{P}_n^\top \mathcal{P}_n \mathcal{K} \mathcal{P}_n^\top \mathcal{P}_n$ converges in norm to $I + \mathcal{K}$. ∎

4.2. Convergence. We furthermore have convergence, at exactly the same rate that the Chebyshev expansion converges to the true solution.

THEOREM 4.5. *Suppose* $\mathbf{f} \in \ell^2_{\lambda-N+1}$ *for some* $\lambda \in \{D-1, D, \ldots\}$, *and that* $\begin{pmatrix} \mathcal{B} \\ \mathcal{L} \end{pmatrix} : \ell^2_{\lambda+1} \to \ell^2_\lambda$ *is an invertible operator. Define*

$$\mathbf{u}_n = A_n^{-1} \begin{pmatrix} \mathbf{c} \\ \mathcal{P}_{n-N} \mathcal{S}_{N-1} \cdots \mathcal{S}_0 \mathbf{f} \end{pmatrix}.$$

Then

$$\|\mathbf{u} - \mathcal{P}_n^\top \mathbf{u}_n\|_{\ell^2_{\lambda+1}} \le C \|\mathbf{u} - \mathcal{P}_n^\top \mathcal{P}_n \mathbf{u}\|_{\ell^2_{\lambda+1}} \to 0.$$

Proof.
Note that $\begin{pmatrix} \mathbf{c} \\ \mathcal{S}_{N-1} \cdots \mathcal{S}_0 \mathbf{f} \end{pmatrix} \in \ell^2_\lambda$.
Let $\mathbf{v}_n = R_n^{-1} \mathbf{u}_n$ and $\mathbf{v} = \mathcal{R}^{-1} \mathbf{u}$, so that

$$\mathbf{v}_n = (A_n R_n)^{-1} \mathcal{P}_n (I + \mathcal{K}) \mathbf{v}.$$

Moreover,

$$\mathcal{P}_n \mathbf{v} = (A_n R_n)^{-1} \mathcal{P}_n (I + \mathcal{K}) \mathcal{P}_n^\top \mathcal{P}_n \mathbf{v}$$

Thus

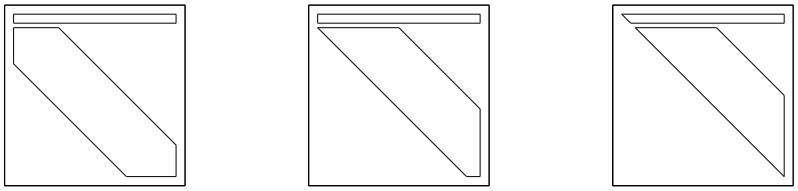
$$\begin{aligned} \mathbf{v} - \mathcal{P}_n^\top \mathbf{v}_n &= \mathbf{v} - \mathcal{P}_n^\top \mathcal{P}_n \mathbf{v} + \mathcal{P}_n^\top (A_n R_n)^{-1} \mathcal{P}_n (I + \mathcal{K}) \mathcal{P}_n^\top \mathcal{P}_n \mathbf{v} \\ &\qquad - \mathcal{P}_n^\top (A_n R_n)^{-1} \mathcal{P}_n (I + \mathcal{K}) \mathbf{v} \\ &= (I - \mathcal{P}_n^\top (A_n R_n)^{-1} \mathcal{P}_n (I + \mathcal{K}))(\mathbf{v} - \mathcal{P}_n^\top \mathcal{P}_n \mathbf{v}) \end{aligned}$$

Moreover, since $\|\mathcal{R}^{-1} \mathbf{u}\|_{\ell^2_\lambda} \le \frac{1}{N} \|\mathbf{u}\|_{\ell^2_{\lambda+1}}$ and $\|\mathcal{R} \mathbf{v}\|_{\ell_{\lambda+1}} \le (N+1) \|\mathbf{v}\|_{\ell^2_\lambda}$ we have,

$$\begin{aligned} \|\mathbf{u} - \mathcal{P}_n^\top \mathbf{u}_n\|_{\ell^2_{\lambda+1}} &\le (N+1) \|\mathbf{v} - \mathcal{P}_n^\top \mathbf{v}_n\|_{\ell^2_\lambda} \\ &\le (N+1) \left[1 + \left\| (A_n R_n)^{-1} \right\|_{\ell^2_\lambda} \left(1 + \|\mathcal{K}\|_{\ell^2_\lambda} \right) \right] \left\| \mathbf{v} - \mathcal{P}_n^\top \mathcal{P}_n \mathbf{v} \right\|_{\ell^2_\lambda} \\ &\le C \left\| \mathbf{u} - \mathcal{P}_n^\top \mathcal{P}_n \mathbf{u} \right\|_{\ell^2_{\lambda+1}}. \end{aligned}$$

Lastly, since $\mathbf{u} \in \ell^2_{\lambda+1}$ we know that $\left\| \mathbf{u} - \mathcal{P}_n^\top \mathcal{P}_n \mathbf{u} \right\|_{\ell^2_{\lambda+1}} \to 0$ as $n \to \infty$. ∎

5. Fast Linear Algebra for Almost Banded Matrices. The spectral method we have described requires the solution of a linear system $Ax = b$ where $A \in \mathbb{C}^{n \times n}$. The matrix A is banded, with lower bandwidth of $m_1 = \mathcal{O}(m)$ and upper bandwidth of $m_2 = \mathcal{O}(m)$, except for the first K dense boundary rows. Here, $m_1 \ge K$ and $m_2 \ge -m_1$. For simplicity we assume that $m_2 \ge 0$. The typical structure of A is



(a) Original Matrix (b) After Left Givens Rotations (c) Upper Triangular

Fig. 5.1: Typical structure of the matrices while solving spectral systems

shown diagrammatically in Figure 5.1a. Here, we describe a stable algorithm to solve $Ax = b$ in $\mathcal{O}(m^2 n)$ operations and with space requirement $\mathcal{O}(mn)$.

Since A is nearly upper triangular, apart from m_1 subdiagonals, a first guess at how to solve $Ax = b$ would be to compute a QR factorization by applying Givens rotations on the left. However, whether we apply Givens rotations on the left or the right, the resulting upper triangular part will be dense because of the boundary rows. However, by applying a partial factorization on the left followed by another partial factorization on the right, we can obtain a factorization $A = QRP^\star$ where P and Q are orthogonal and R is upper triangular with no more non-zero entries than A.

We first apply Givens rotations on the left of A to introduce zero entries in the m_1th subdiagonal. These Givens rotations introduce zeros starting in the $(m_1, 1)$ entry and successively eliminating down the subdiagonal to the $(n, n - m_1 + 1)$ entry. Each Givens rotation applied on the left of A performs a linear combination of two neighboring rows. One of the rows contains the non-zero entry to be eliminated and the other is the row immediately above this. After the first $n - m_1$ Givens rotations, the resulting matrix has a zero m_1th subdiagonal and, typically, non-zero entries along the $(m_2 + 1)$th superdiagonal have been introduced. In the same way, Givens rotations are then used to eliminate, in turn, the $(m_1 - 1)$th, $(m_1 - 2)$th, ..., $(K + 1)$th subdiagonals. The typical structure of the resulting matrix is shown in Figure 5.1b and has lower bandwidth of K and an upper bandwidth of $m_2 + m_1 - K$, except for the first K non-zero boundary rows.

In a similar fashion, we now apply Givens rotations on the right. The first sequence of Givens rotations on the right eliminates the Kth subdiagonal by starting in the last row and successively eliminating to the Kth row. Each Givens rotation applied on the right performs a linear combination of two neighboring columns. One of the columns contains the non-zero entry to be eliminated and the other is the column immediately to the right. After the first $n - K$ Givens rotations the resulting matrix has a zero Kth subdiagonal and, typically, non-zero entries along the $(m_2 + m_1 - K + 1)$th superdiagonal have been introduced. Givens rotations are then used to eliminate, in turn, the $(K - 1)$th, $(K - 2)$th,..., 1st subdiagonals. The resulting matrix is upper triangular, with upper bandwidth of $m_1 + m_2$ (except for the first K rows) and shown diagrammatically in Figure 5.1c. Each Givens rotation, whether applied on the left or right, eliminated one zero in a subdiagonal and introduced, at most, one non-zero in a superdiagonal. In particular, the upper triangular matrix has no more non-zeros than the original matrix A.

The factorization can be written as $A = QRP^\star$, where $Q, P \in \mathbb{C}^{n \times n}$ are orthogo-

nal and $R \in \mathbb{C}^{n\times n}$ is upper triangular. To reduce storage requirements we overwrite A with Q, R and P as we perform the factorization using the same trick as described by Demmel [6]. Note that, if we were computing $Ax = b$ just once, we would not store any information about Q, because we can apply the left Givens rotations directly to the vector b, as we go. However, in practice we progressively increase n, until the solution of the differential equation is well-resolved, and in this case we are able to reapply the left Givens rotation used on the smaller system to the new larger system.

Solving the system $Ax = b$ is computed with the following four steps:

1. Factorize $A = QRP^\star$ by applying left and right Givens rotations.
2. Compute $Q^\star b$.
3. Solve $Ry = Q^\star b$ for y by backwards substitution.
4. Compute $x = Py$.

The factorization requires $\mathcal{O}(mn)$ Givens rotations with each one performing a linear combination between $\mathcal{O}(m)$ non-zero entries. Hence, this factorization can be computed in $\mathcal{O}(m^2n)$ operations. Backwards substitution takes $\mathcal{O}(m^2n)$ operations, since R is banded except for the first $K = \mathcal{O}(1)$ rows. Moreover, $Q^\star b$ and Py are computed by applying $\mathcal{O}(mn)$ Givens rotations to vectors and hence, can be computed in $\mathcal{O}(mn)$ operations. In total, the almost banded linear systems that our spectral method forms can be solved in $\mathcal{O}(m^2n)$ operations.

Storage requirement is asymptotically minimal since A has $\mathcal{O}(mn)$ non-zero entries, and R has no more than A. The total space requirement is $\mathcal{O}(mn)$, which is asymptotically the same as storing the original matrix A.

5.1. Linear algebra stability in higher order norms. We first remark that if $\mathcal{B}$ is a bounded operator from $\ell^2_1 \to \ell^2_0$, such as Dirichlet boundary conditions, the results of Section 4 prove that the *preconditioned* linear system has bounded 2-norm condition number as $n \to \infty$. Because the $QRP^\star$ decomposition is computed using Givens rotations, which are stable in ℓ^2_0 [10], as is backward substitution, we see that the linear algebra scheme applied to the preconditioned operator is stable, and has $\mathcal{O}(m^2n)$ complexity. We remark that a variant of Lemma 4.1 can be proved showing that this stability property holds for the non-preconditioned version as well.

The results of Section 4 also show convergence and well-conditioning in higher order norms. One would expect numerical round-off in $QRP^\star$ decomposition to destroy this convergence property. However, in practice, this is not the case. In Figure 5.2, we solve the standard Airy equation as a two point boundary value problem: $u'' = xu, u(-1) = \mathrm{Ai}(-1), u(1) = \mathrm{Ai}(1)$. We witness convergence in higher order norms as well. In other words, the computed solution has a fast decaying tail, and all of the *absolute* error is in the low order coefficients.

Convergence in higher order norms implies convergence of derivatives, and this is verified by differentiating the computed solution by applying $\mathcal{S}_0^{-1}\mathcal{D}_1$ repeatedly. (We note that the banded, upper triangular nature of $\mathcal{S}_0$ means that its inverse applied to a vector is computable in $\mathcal{O}(n)$ time; after all, this is precisely the classical differentiation operator for Chebyshev expansions.) We compare the computed derivatives with the true derivatives of the Airy function $\mathrm{Ai}^{(p)}$ at a single point, $x = -1/2$, and see convergence for $p = 1, 5$ and 20.

We note that the stability of the $QRP^\star$ algorithm in higher order norms appears to follow from Q being almost banded: its banded along the superdiagonal, and decays spectrally along the subdiagonals. We will not attempt to prove this here as it seems likely that it only holds true *generically*; hence, a differential equation could possibly be constructed which breaks this property.

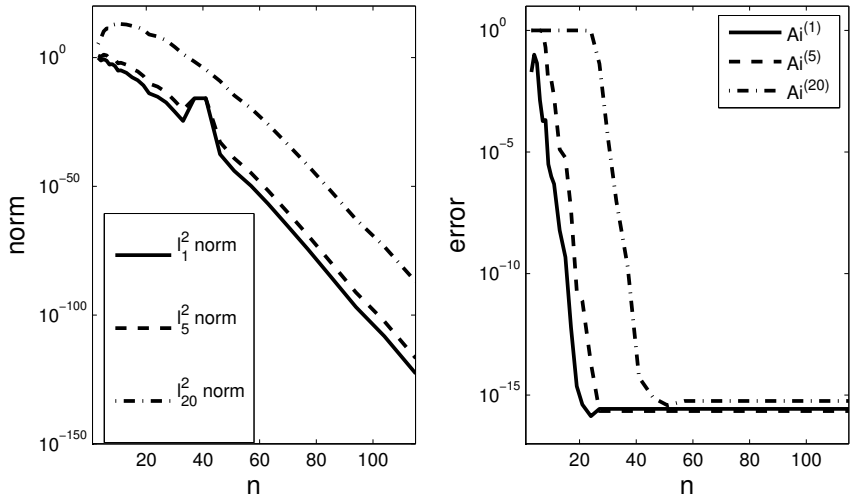


Fig. 5.2: Left: Plot of the Cauchy error of the solution coefficients measured in the: (solid) ℓ_1^2-norm, (dashed) ℓ_5^2-norm, (dot-dashed) ℓ_{20}^2-norm. Right: Plot of the relative error in derivatives of the solution at $x = -\frac{1}{2}$ for the (solid) 1st, (dashed) 5th, (dot-dashed) 20th derivative.

Since ℓ_λ^2 is a Hilbert space, Givens rotations can be constructed with the relevant inner product, resulting in orthogonal operations (i.e., with condition number one) in ℓ_λ^2. The stability of such an algorithm in ℓ_λ^2 follows immediately. With this modification, we have an $\mathcal{O}(m^2 n)$ stable algorithm which is guaranteed to converge in higher order norms.

Remark While we have discussed the convergence in higher order norms with Dirichlet boundary conditions, the exact same logic applies to the convergence and stability observed with higher order boundary conditions.

6. Future work. A straightforward extension of this work is to exotic boundary conditions. For example, we can impose a condition on the value of the solutions integral using the boundary operator

$$(\mu_0, \mu_1, \ldots)$$

where μ_k are the Clenshaw–Curtis weights [5], computable in $\mathcal{O}(n \log n)$ time [22]. Other boundary conditions, such as Robin conditions or conditions involving values or derivatives inside $(-1, 1)$, are equably imposable.

One item for future work is to investigate the method on linear integro-differential equations, such as

$$a(x)u'(x) + b(x)u(x) + \int_{-1}^{x} c(s)u(s)ds = f(x). \tag{6.1}$$

In order to preserve the almost banded structure of the linear system the solution $u(x)$ could be represented in a $C^{(1)}$ series; instead of a Chebyshev series. The integral

recurrence relation

$$\int_{-1}^{x} U_n(s)ds = \frac{T_{n+1}(x)}{n+1} - \frac{T_{n+1}(-1)}{n+1}, \qquad n \geq 0$$

means that the integration operator can be represented by

$$\mathcal{I}_1 = \begin{pmatrix} 1 & -\frac{1}{2} & \frac{1}{3} & \cdots \\ 1 & & & \\ & \frac{1}{2} & & \\ & & \ddots & \end{pmatrix}.$$

Similar, almost banded systems can be constructed and used to solve (6.1).

A second extension of this work is to nonlinear differential equations, of the form

$$\mathcal{B}u = \mathbf{0} \qquad \text{and} \qquad \mathcal{L}u + g(u) = f.$$

It is straightforward to incorporate the approach of this paper into an infinite-dimensional Newton iteration, as in [8]. The Newton iteration takes the form

$$u_{k+1} = u_k + \begin{pmatrix} \mathcal{B} \\ \mathcal{L} + g'(u_k) \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{0} \\ \mathcal{L}u_k + g(u_k) - f \end{pmatrix}.$$

Since the linear operator we invert involves the solution itself, it is unlikely that the bandedness of its discretization for large n would be useful, hence the resulting algorithm would not be faster than what exists. However, the well-conditioning of the linear operator is preserved, and would likely translate to better accuracy of the resulting algorithm for nonlinear problems.

Lastly an exciting generalization of this work would be to higher dimensions, where the density of matrices has inhibited the usefulness of spectral methods. Constructing a method on rectangular domains should be straightforward, using tensor products of ultraspherical polynomials. Indeed, a similar approach for Helmholtz equation (i.e., a constant coefficient PDE) was used in [18]. Using the theory of [17], there are potentially generalization of ultraspherical polynomials to deltoid domains. What is less clear is how the results would be generalizable to more general domains, in particular, to a triangle.

Acknowledgments. We thank P. Gonnet, discussions with whom led to the observation of well-conditioning of coefficient methods, which initiated the research of this paper. We also thank the rest of the Chebfun team, including T. Driscoll, N. Hale and L. N. Trefethen for valuable feedback.

REFERENCES

[1] G. Baszenski and M. Tasche, Fast polynomial multiplication and convolutions related to the discrete cosine transform, *Lin. Alg. Appl.*, 252 (1997), pp. 1–25.
[2] J. P. Boyd, *Chebyshev and Fourier Spectral Methods*, Dover Publications, (2001).
[3] C. Canuto, *Spectral Methods: Fundamentals in Single Domains*, Springer, (2006).
[4] L. Carlitz, The product of two ultraspherical polynomials, *Proc. Glasgow Math. Assoc.*, 5 (1961), pp. 76–79.
[5] C. W. Clenshaw and A. R. Curtis, A method for numerical integration on an automatic computer, *Numer. Math.*, 2 (1960) pp. 197–205.
[6] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, (1997).

[7] E. H. Doha and W. M. Abd-Elhameed, Efficient spectral ultraspherical-dual-Petrov-Galerkin algorithms for the direct solution of $(2n + 1)$th-order linear differential equations, *Math. Comp. Simul.*, 79 (2009), pp. 3221–3242.

[8] T. A. Driscoll, F. Bornemann and L. N. Trefethen, The chebop system for automatic solution of differential equations, *BIT Numer. Math.*, 48 (2008), pp. 701–723.

[9] B. Fornberg, *A Practical Guide to Pseudospectral Methods*, Cambridge University Press, (1998).

[10] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, SIAM, (2002).

[11] H. Ma and W. Sun, A Legendre–Petrov–Galerkin and Chebyshev collocation method for third-order differential equations, *SIAM J. Numer. Anal.*, (2000), pp. 1425–1438.

[12] J. C. Mason and D. C. Handscomb, *Chebyshev Polynomials*, Chapman & Hall/CRC, (2003).

[13] F. W. J. Olver, D. W. Lozier, R. F. Boisvert and C. W. Clark, *NIST Handbook of Mathematical Functions*, Cambridge University Press, (2010).

[14] E. L. Ortiz, The tau method, *SIAM J. Numer. Anal.*, (1969), pp. 480–492.

[15] S. A. Orszag, Accurate solution of the Orr–Sommerfeld stability equation Journal of Fluid Mechanics, 50 , pp. 689–703.

[16] P. D. Ritger and N. J. Rose, *Differential Equations with Applications*, Dover Publications, (2000).

[17] B. N. Ryland and H. Z. Munthe-Kaas, On multivariate Chebyshev polynomials and spectral approximations on triangles, in: *Spectral and High Order Methods for Partial Differential Equations*, Springer Berlin Heidelberg, (2011), pp. 19–41.

[18] J. Shen, Efficient spectral-Galerkin method II. Direct solvers of second- and fourth-order equations using Chebyshev polynomials, *SIAM J. Sci. Comput.*, 16 (1995), pp. 74–87.

[19] J. Shen, A new dual-Petrov–Galerkin method for third and higher odd-order differential equations: application to the KdV equation, *SIAM J. Numer. Anal.*, 41 (2004), pp. 1595–1619.

[20] J. Shen, T. Tang and L. L. Wang, *Spectral Methods: Algorithms, Analysis and Applications*, Springer, (2009).

[21] L. N. Trefethen, *Spectral Methods in* Matlab, SIAM, (2000).

[22] J. Waldvogel, Fast construction of the Fejér and Clenshaw–Curtis quadrature rules, *BIT Numer. Math.*, 46 (2006), pp. 195–202.