SIAM J. SCI. COMPUT.
Vol. 36, No. 1, pp. A148–A167

A FAST, SIMPLE, AND STABLE CHEBYSHEV–LEGENDRE TRANSFORM USING AN ASYMPTOTIC FORMULA*

NICHOLAS HALE† AND ALEX TOWNSEND†

Abstract. A fast, simple, and numerically stable transform for converting between Legendre and Chebyshev coefficients of a degree N polynomial in $\mathcal{O}(N(\log N)^2/\log\log N)$ operations is derived. The fundamental idea of the algorithm is to rewrite a well-known asymptotic formula for Legendre polynomials of large degree as a weighted linear combination of Chebyshev polynomials, which can then be evaluated by using the discrete cosine transform. Numerical results are provided to demonstrate the efficiency and numerical stability. Since the algorithm evaluates a Legendre expansion at an $N+1$ Chebyshev grid as an intermediate step, it also provides a fast transform between Legendre coefficients and values on a Chebyshev grid.

Key words. Chebyshev, Legendre, transform, asymptotic formula, discrete cosine transform

AMS subject classifications. 65T50, 65D05, 41A60

DOI. 10.1137/130932223

1. Introduction.

Expansions of functions as finite series of orthogonal polynomials have applications throughout scientific computing, engineering, and physics [4, 8, 29]. Expansions in Chebyshev polynomials,

$$p_N(x) = \sum_{n=0}^{N} c_n^{cheb} T_n(x), \quad x \in [-1, 1], \tag{1.1}$$

where $T_n(x) = \cos(n \cos^{-1}(x))$, are often used because of their near-optimal approximation properties and associated fast algorithms [18, 35]. However, in some situations Legendre expansions,

$$p_N(x) = \sum_{n=0}^{N} c_n^{leg} P_n(x), \quad x \in [-1, 1], \tag{1.2}$$

where $P_n(x)$ is the degree n Legendre polynomial, are preferred due to their orthogonality in the standard L^2 inner product, more rapidly decaying Cauchy transform [21], or connection to spherical harmonics [28].

Unfortunately, fast algorithms are not as readily available for computing with Legendre expansions, and hence a fast transform to convert between Legendre and Chebyshev coefficients is desirable. In this paper we describe a fast, simple, and stable transform that converts between the coefficients $c_0^{leg}, \ldots, c_N^{leg}$ in (1.2) and the coefficients $c_0^{cheb}, \ldots, c_N^{cheb}$ in (1.1) in $\mathcal{O}(N(\log N)^2/\log\log N)$ operations:

*Submitted to the journal's Methods and Algorithms for Scientific Computing section August 7, 2013; accepted for publication (in revised form) November 12, 2013; published electronically February 6, 2014.

http://www.siam.org/journals/sisc/36-1/93222.html

†Mathematical Institute, University of Oxford, Oxford, OX2 6GG, UK (hale@maths.ox.ac.uk, http://people.maths.ox.ac.uk/hale/; townsend@maths.ox.ac.uk, http://people.maths.ox.ac.uk/townsend/). The first author's work was supported by The MathWorks, Inc., and King Abdullah University of Science and Technology (KAUST), award KUK-C1-013-04. The second author's work was supported by EPSRC grant EP/P505666/1 and by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC grant agreement 291068.

$$c_0^{leg}, \ldots, c_N^{leg} \quad \underset{\text{inverse transform}}{\overset{\text{forward transform}}{\rightleftarrows}} \quad \mathcal{O}(N(\log N)^2/\log\log N) \quad c_0^{cheb}, \ldots, c_N^{cheb}.$$

While there are a number of existing fast algorithms for computing such transforms, many of these require hierarchical data structures and expensive initialization procedures [2, 27], need an underlying function to evaluate [7, 17], or suffer from stability problems [20].

The algorithm we describe is based on Stieltjes' long-established asymptotic formula for Legendre polynomials [31] and can be seen as a numerically stable modification of the approach by Mori, Suda, and Sugihara [20]. As we shall explain, it can be interpreted as approximating the Legendre–Vandermonde-like matrix by a weighted linear combination of Chebyshev–Vandermonde-like matrices, whose action on vectors can be efficiently computed using the discrete cosine transform (DCT).

The outline of this paper is as follows. In the next section we discuss existing fast algorithms for the Chebyshev–Legendre transform and justify the need for a new approach. In section 3 we discuss the forward transform, first introducing the asymptotic formula of Stieltjes [31] and describing the numerically unstable algorithm of Mori, Suda, and Sugihara [20], before advocating a novel modification that leads to a fast and stable algorithm. In section 4 we describe a similar new fast algorithm for the inverse transform, and in section 5 we present numerical results for both algorithms. Finally, in section 6 we discuss applications and future work for related fast transforms.

The code used for all the numerical results in this paper is publicly available from [15]. It is also available as part of the Chebfun software system [36].

2. Existing methods.

The problem of computing coefficients in a Legendre expansion has received considerable research attention since the 1970s [11, 26]. These initial approaches required $\mathcal{O}(N^2)$ operations to compute the transform, and to the authors' knowledge the first algorithm for computing the coefficients of a Legendre expansion in less than $\mathcal{O}(N^2)$ operations is due to Orszag [25] in 1986. Later, in 1991, Alpert and Rokhlin [2] described an algorithm based on multipole-like ideas, requiring just $\mathcal{O}(N)$ operations. In 1994, Driscoll and Healy developed an $\mathcal{O}(N(\log N)^2)$ algorithm, which could be used to compute the spherical harmonic transform of band-limited functions on the 2-sphere [9]. Since then many other fast algorithms have been proposed [7, 20, 27, 38]. Figure 2.1 summarizes the main algorithms, which are briefly described below.

2.1. Approaches using asymptotic expansions.

Orzsag [25], in 1986, described a fast algorithm for eigenfunction transforms, which can be used for the computation of Legendre coefficients. The algorithm is based on a first-order WKB expansion of Legendre polynomials, but it is not considered useful in practice as the expansion converges too slowly. The algorithm we present for computing the forward transform is similar to Orszag's approach but is improved in two crucial ways: (1) we use a different asymptotic formula for $P_n(x)$ due to Stieltjes that converges more rapidly [31, 34]; and (2) we use an accompanying explicit error formula to derive the complexity and determine certain algorithmic constants of the transform.

We are not the first to use Stieltjes' asymptotic formula for computing the fast transform, as it was employed by Mori, Suda, and Sugihara [20] in 1999 to derive an algorithm requiring $\mathcal{O}(N \log N)$ operations. The algorithm described in [20] is fast and accurate for small N, but as N increases it becomes numerically unstable in floating

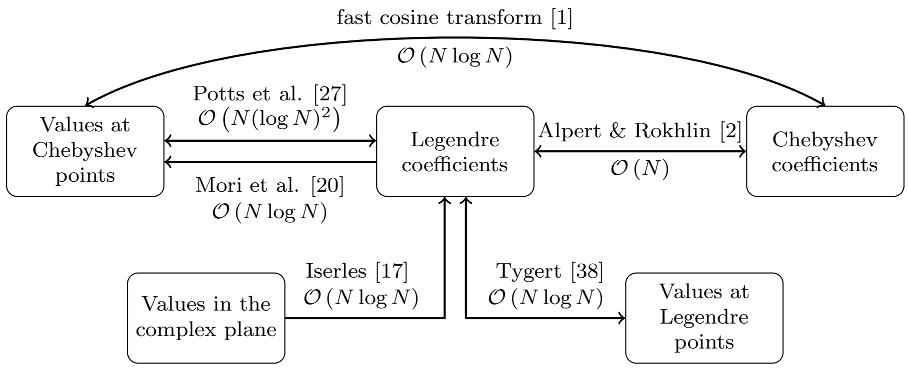

FIG. 2.1. *Existing fast algorithms related to Chebyshev–Legendre transforms.*

point arithmetic. Suda, Mori, and Sugihara were aware of the numerical instability in their algorithm and in 2002 began preparing a manuscript to fix the numerical issues. However, that work was not finished and they no longer intend to publish [32]. Furthermore, even with the unpublished modification (as noted in the manuscript), their algorithm is still unstable for large N. In this paper we present a further modification that is numerically stable for all N. In particular, in section 3 we adapt the algorithm in [20] to derive a stable transform requiring $\mathcal{O}(N(\log N)^2/\log\log N)$ operations that can transform between 1 million Legendre and Chebyshev coefficients, or more.

2.2. The fast multipole method. The fast multipole-like approach described by Alpert and Rohklin [2] transforms between Legendre and Chebyshev coefficients in $\mathcal{O}(N)$ operations. The cost of the algorithm depends on the working precision, and for double precision arithmetic they observe that after the initialization phase it is about 5.5 times the cost of a single fast Fourier transform (FFT) of the same length [2]. Although this approach is often considered state of the art, the algorithm is not widely used in practice as the initialization phase can be expensive and the hierarchical data structures required make it difficult to implement efficiently. As noted in [37], the algorithmic ideas described by Yarvin and Rokhlin in [41] are useful for converting between Chebyshev and Legendre values, and these techniques make the transform easier to implement with $\mathcal{O}(N)$ complexity and a cheaper initialization cost. Moreover, the algorithm in [24] can be used for the Chebyshev–Legendre transform, though the main advantage of their approach is its generality to the Fourier–Bessel transform and others, rather than its efficiency for the transform of interest. The algorithms for the forward and inverse transforms presented in this paper do not require an initialization phase and are sufficiently simple that they can be efficiently implemented in about 100 lines of MATLAB code (see [15, 36]).

2.3. Divide-and-conquer approaches. Potts, Steidl, and Tasche described in 1998 a fast algorithm that transforms between function values at Chebyshev points and Legendre coefficients [27]. The algorithm uses a divide-and-conquer approach and hierarchical data structures to apply the matrix-vector product involving the Legendre–Vandermonde-like matrix

$$\mathbf{P_N}(\underline{x}_N^{cheb}) = \left[P_0(\underline{x}_N^{cheb}) \mid \cdots \mid P_{N-1}(\underline{x}_N^{cheb})\right]$$

in $\mathcal{O}(N(\log N)^2)$ operations, where $P_0, \ldots, P_{N-1}$ are the first N Legendre polynomials and $\underline{x}_N^{cheb}$ denotes the vector of N Chebyshev points in decreasing order.

Tygert [38], in 2010, described a similar algorithm, noting that the Legendre–Vandermonde-like matrix can be decomposed as $\mathbf{P_N}(\underline{x}_N^{leg}) = D_{\underline{w}} U D_{\underline{s}}$, where $\underline{x}_N^{leg}$ is the vector of N Gauss–Legendre points, $D_{\underline{w}}$ is the diagonal matrix of Gauss–Legendre quadrature weights, $D_{\underline{s}}$ is the diagonal matrix of orthonormalization factors for Legendre polynomials, and U is an orthogonal matrix. Tygert then uses the fact that the orthogonal matrix U can be applied in $\mathcal{O}(N \log N)$ operations since the columns are the eigenvectors of a symmetric tridiagonal matrix [13]. The approach proposed by Tygert is more general than just a fast Legendre transform and he notes that specialized algorithms are likely to be more efficient.

2.4. Function dependent approaches. In 2011, Iserles [17] described an algorithm to compute the fast Legendre coefficients by sampling a function at points lying on a certain Bernstein ellipse in the complex plane. The algorithm requires $\mathcal{O}(N \log N)$ operations and is much simpler to implement than other approaches mentioned thus far. However, the size of the required Bernstein ellipse depends on the region of analyticity of f, making the algorithm difficult to use in a black box manner. Furthermore, it seems that in practice this algorithm suffers from numerical instability for large N ($N \geq 512$), and quadratic precision is required in the computations to get even double or single precision accuracy in the results.

More recently, De Micheli and Viano in [7] described a fast algorithm based on integral transforms, which also depends on the smoothness of the prescribed function. The algorithm we derive does not depend on the smoothness of the function and is applicable to any vector of real or complex coefficients.

3. The forward transform: Legendre to Chebyshev. For notational convenience we express (1.1) and (1.2) in the form

$$p_N(x) = \mathbf{T_N}(x)\underline{c}_N^{cheb} = \mathbf{P_N}(x)\underline{c}_N^{leg},$$

where x is an independent variable and

$$\mathbf{T_N}(x) = [T_0(x) \,|\, \ldots \,|\, T_N(x)], \qquad \mathbf{P_N}(x) = [P_0(x) \,|\, \ldots \,|\, P_N(x)] \tag{3.1}$$

are Chebyshev and Legendre *quasimatrices*,[1] i.e., the $\infty \times (N+1)$ matrices that have in their nth column the degree $n-1$ Chebyshev and Legendre polynomial, respectively. If $-1 \leq x_N < \cdots < x_0 \leq 1$ are $N+1$ distinct points in $[-1,1]$ (indexed in reverse order to simplify later notation) and $\underline{x}_N = \{x_j\}_{0 \leq j \leq N}$, then

$$p_N(\underline{x}_N) = \mathbf{T_N}(\underline{x}_N)\underline{c}_N^{cheb} = \mathbf{P_N}(\underline{x}_N)\underline{c}_N^{leg}.$$

For brevity, we refer to the Vandermonde-like matrices $\mathbf{T_N}(\underline{x}_N)$ and $\mathbf{P_N}(\underline{x}_N)$ as the Chebyshev–Vandermonde and Legendre–Vandermonde matrices, respectively. Now, since polynomial interpolants at distinct points are unique, $\mathbf{T_N}(\underline{x}_N)$ and $\mathbf{P_N}(\underline{x}_N)$ are invertible, and we may write

$$\underline{c}_N^{cheb} = \mathbf{T_N}(\underline{x}_N)^{-1}\mathbf{P_N}(\underline{x}_N)\underline{c}_N^{leg}.$$

For a general vector of distinct points $\underline{x}_N$, a naive algorithm requires $\mathcal{O}(N^2)$ operations for a matrix-vector product with $\mathbf{P_N}(\underline{x}_N)$, and $\mathcal{O}(N^3)$ operations to apply

[1] The term *quasimatrix* was coined by Stewart in [30] to describe "matrices" with columns consisting of functions.

$\mathbf{T_N}(\underline{x}_N)^{-1}$ to a vector. However, if $\underline{x}_N = \underline{x}_N^{cheb}$, i.e., the vector of $N+1$ Chebyshev–Lobatto points,

$$x_k^{cheb} = \cos(k\pi/N), \quad k = 0, \ldots, N, \tag{3.2}$$

then $\mathbf{T_N}(\underline{x}_N^{cheb})$ is the matrix representing a DCT[2] and can be applied and inverted in $\mathcal{O}(N \log N)$ operations [1, 12]. Applying $\mathbf{P_N}(\underline{x}_N^{cheb})$ to a vector in fewer than $\mathcal{O}(N^2)$ operations is less straightforward, but in the following section we describe how this can be achieved by employing a well-known asymptotic formula.

As an aside, we note that if $\mathbf{T_N}(\underline{x}_N^{cheb})^{-1}$ is not applied, then $\mathbf{P_N}(\underline{x}_N^{cheb})\underline{c}_N^{leg} = p_N(\underline{x}_N^{cheb})$ is simply $p_N(x)$ evaluated on the Chebyshev grid $\underline{x}_N^{cheb}$, which is useful for the fast evaluation of a Legendre expansion and spectral collocation methods [6].

3.1. An asymptotic formula for Legendre polynomials. In 1890, Stieltjes [31] derived the following asymptotic formula for Legendre polynomials as $n \to \infty$:

$$P_n(\cos\theta) = C_n \sum_{m=0}^{M-1} h_{m,n} \frac{\cos\big((m+n+\frac{1}{2})\theta - (m+\frac{1}{2})\frac{\pi}{2}\big)}{(2\sin\theta)^{m+1/2}} + R_{M,n}(\theta), \tag{3.3}$$

where $\theta = \cos x$ for $\theta \in (0, \pi)$, and

$$C_n = \frac{4}{\pi}\prod_{j=1}^{n} \frac{j}{j+1/2} = \sqrt{\frac{4}{\pi}}\frac{\Gamma(n+1)}{\Gamma(n+3/2)}, \tag{3.4}$$

$$h_{m,n} = \begin{cases} 1, & m = 0, \\ \prod_{j=1}^{m} \frac{(j-1/2)^2}{j(n+j+1/2)}, & m > 0. \end{cases} \tag{3.5}$$

Szegő, in his classic book on orthogonal polynomials [34], showed that the error term can be bounded by

$$|R_{M,n}(\theta)| \leq C_n h_{M,n} \frac{2}{(2\sin\theta)^{M+\frac{1}{2}}}. \tag{3.6}$$

This upper bound is sharp and a good approximate lower bound is half the upper bound, since $|R_{M,n}(\theta)|$ is less than twice the first neglected term in (3.3) [34]. The bound on $R_{M,n}(\theta)$ shows that (3.3) converges to $P_n(\cos\theta)$ as $M \to \infty$ for $\theta \in (\pi/6, 5\pi/6)$, i.e., for θ such that $|2\sin\theta| < 1$. However, as suggested by Szegő in [34] and demonstrated in [5, 16], for finite values of M this asymptotic formula can still be an excellent approximation for $\theta \notin (\pi/6, 5\pi/6)$. In practice, if n is sufficiently large, (3.3) can be used to approximate $P_n(\cos\theta)$ to double precision for almost all $\theta \in (0, \pi)$. The exact region in the (n, θ)-plane in which M terms of (3.3) approximates $P_n(\cos\theta)$ to a prescribed tolerance can be determined from (3.6), as we derive later in (3.14).

To make (3.3) amenable to evaluation using the DCT, we first note that $(m+n+\frac{1}{2})\theta - (m+\frac{1}{2}) = n\theta - (m+\frac{1}{2})(\frac{\pi}{2} - \theta)$ and rewrite the trigonometric term in (3.3) as

$$\cos\big((m+n+\tfrac{1}{2})\theta - (m+\tfrac{1}{2})\tfrac{\pi}{2}\big) = \cos\big(n\theta - (m+\tfrac{1}{2})(\tfrac{\pi}{2} - \theta)\big).$$

[2] In this paper we use the acronym DCT to refer to the discrete cosine transform of type I (DCT-I [40]) with the first and last columns of the DCT-I matrix scaled, so that it equals the matrix $\mathbf{T_N}(\underline{x}_N^{cheb})$. Moreover, the matrix $\mathbf{T_N}(\underline{x}_N^{cheb})$ is symmetric, i.e., $\mathbf{T_N}(\underline{x}_N^{cheb})^T = \mathbf{T_N}(\underline{x}_N^{cheb})$.

Applying a standard trigonometric identity to $\cos(A-B)$ we find

$$\cos\big(n\theta - (m+\tfrac{1}{2})(\tfrac{\pi}{2}-\theta)\big) = \sin(n\theta)\sin\big((m+\tfrac{1}{2})(\tfrac{\pi}{2}-\theta)\big) + \cos(n\theta)\cos\big((m+\tfrac{1}{2})(\tfrac{\pi}{2}-\theta)\big).$$

Noting that $T_n(\cos\theta) = \cos(n\theta)$ and $U_{n-1}(\cos\theta) = \sin(n\theta)/\sin\theta$ are Chebyshev polynomials of the first and second kind, respectively, we have

$$\cos\big(n\theta - (m+\tfrac{1}{2})(\tfrac{\pi}{2}-\theta)\big) = U_{n-1}(\cos\theta)\sin\big((m+\tfrac{1}{2})(\tfrac{\pi}{2}-\theta)\big)\sin\theta + T_n(\cos\theta)\cos\big((m+\tfrac{1}{2})(\tfrac{\pi}{2}-\theta)\big). \tag{3.7}$$

Finally, substituting (3.7) back into (3.3), we find the asymptotic formula (3.3) can be expressed as a weighted linear combination of Chebyshev polynomials

$$P_n(\cos\theta) = C_n \sum_{m=0}^{M-1} h_{m,n}\left(u_m(\theta)U_{n-1}(\cos\theta) + v_m(\theta)T_n(\cos\theta)\right) + R_{M,n}(\theta), \tag{3.8}$$

where

$$u_m(\theta) = \frac{\sin\big((m+\frac{1}{2})(\frac{\pi}{2}-\theta)\big)\sin\theta}{(2\sin\theta)^{m+1/2}}, \qquad v_m(\theta) = \frac{\cos\big((m+\frac{1}{2})(\frac{\pi}{2}-\theta)\big)}{(2\sin\theta)^{m+1/2}}. \tag{3.9}$$

Now, since $T_n(\cos\theta)$ and $U_{n-1}(\cos\theta)$ are the only terms in (3.8) that depend on both n and θ, the quasimatrix $\mathbf{P_N}(x)$ from (3.1) can be expressed in the following compact form:

$$\mathbf{P_N}(\cos\theta) = \sum_{m=0}^{M-1}\left(D_{u_m(\theta)}[0\,|\,\mathbf{U_{N-1}}(\cos\theta)] + D_{v_m(\theta)}\mathbf{T_N}(\cos\theta)\right) D_{\underline{Ch}_m} + \mathbf{R}_M(\theta), \tag{3.10}$$

where $x = \cos\theta$, $D_{u_m}(\theta)$ and $D_{v_m}(\theta)$ are the diagonal operators with $u_m(\theta)$ and $v_m(\theta)$ from (3.9) on the diagonal, $D_{\underline{Ch}_m}$ is the diagonal matrix of the pointwise product of (3.4) and (3.5) for $n = 0,\ldots,N$, and $\mathbf{R}_M(\theta) = [R_{M,0}(\theta)\,|\,\ldots\,|\,R_{M,N}(\theta)]$.

Substituting $x = \underline{x}_N^{cheb} = \cos(\underline{\theta}_N^{cheb})$ means that $\mathbf{U_{N-1}}(\underline{x}_N^{cheb})$ and $\mathbf{T_N}(\underline{x}_N^{cheb})$ are essentially discrete cosine/sine transformation matrices, which can be applied to a vector in $\mathcal{O}(N\log N)$ operations using the DCT. Since $\mathbf{P_N}(\underline{x}_N^{cheb})$ is simply a diagonally weighted linear combination of these matrices, the matrix-vector product $\mathbf{P_N}(\underline{x}_N^{cheb})\underline{c}_N^{cheb}$ in (3.10) can be evaluated in $\mathcal{O}(MN\log N)$ operations using (3.10) with an error of $\mathbf{R}_M(\underline{\theta}_N^{cheb})\underline{c}_N^{cheb}$.

3.2. Partitioning the Legendre–Vandermonde matrix for the forward transform. First, we take the unusual step of describing an unstable algorithm by Mori, Suda, and Sugihara [20] for the forward transform that we do not advocate. However, by describing this algorithm now we motivate and set the scene for its stable variant (see section 3.3).

This unstable algorithm computes $\mathbf{P_N}(\underline{x}_N^{cheb})c_N^{leg}$ by partitioning $\mathbf{P_N}(\underline{x}_N^{cheb})$ into three matrices,

$$\mathbf{P_N}(\underline{x}_N^{cheb}) = \mathbf{P_N^{\mathrm{REC}}}(\underline{x}_N^{cheb}) + \mathbf{P_N^{\mathrm{DCT}}}(\underline{x}_N^{cheb}) + \mathbf{P_N^{\mathrm{COR}}}(\underline{x}_N^{cheb}), \tag{3.11}$$

as shown in Figure 3.1 (left). Making use of DCTs, the matrix $\mathbf{P_N^{\mathrm{DCT}}}(\underline{x}_N^{cheb})$ is applied to a vector via the asymptotic formula (3.10), and in this process an unacceptably

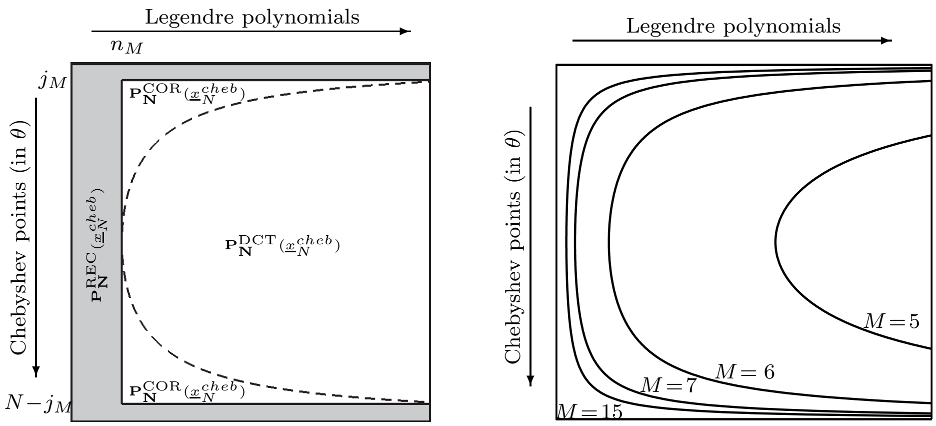

FIG. 3.1. *Left: The partition of the matrix $\mathbf{P}_{\mathbf{N}}(\underline{x}_N^{cheb})$ employed in the unstable algorithm. The dashed line indicates the boundary of the region in which the asymptotic formula can be employed without correction, and the gray region indicates the nonzero entries of the matrix $\mathbf{P}_{\mathbf{N}}^{\mathrm{REC}}(\underline{x}_N^{cheb})$. Right: The error curves $|R_{M,n}(\theta)| = \epsilon$ for $M = 5, 6, 7, 15$ and $N = 1{,}000$.*

large error for certain (n, θ) can be committed which must be corrected by a matrix $\mathbf{P}_{\mathbf{N}}^{\mathrm{COR}}(\underline{x}_N^{cheb})$. The matrix $\mathbf{P}_{\mathbf{N}}^{\mathrm{REC}}(\underline{x}_N^{cheb})$ contains all the columns and rows of $\mathbf{P}_{\mathbf{N}}(\underline{x}_N^{cheb})$ that do not intersect the error curve $|R_{M,n}(\theta)| = \epsilon$, that is,

$$\mathbf{P}_{\mathbf{N}}^{\mathrm{REC}}(\underline{x}_N^{cheb})_{ij} = \begin{cases} \mathbf{P}_{\mathbf{N}}(\underline{x}_N^{cheb})_{ij}, & 1 \le \min(i, N-i+1) \le j_M, \\ \mathbf{P}_{\mathbf{N}}(\underline{x}_N^{cheb})_{ij}, & 1 \le j \le n_M, \\ 0, & \text{otherwise.} \end{cases}$$

Here, n_M is the number of Legendre polynomials, $P_0, \ldots, P_{n_M}$, that cannot be approximated using the asymptotic formula (3.3) to a precision ϵ at $\theta = \pi/2$ ($x = 0$). In other words, $|R_{M,n}(\pi/2)| < \epsilon$ for all $n > n_M$. It can be shown, using (3.6) and its approximate sharpness, that for $n \gg M$,

$$|R_{M,n}(\pi/2)| \gtrsim C_n h_{n,M} \frac{1}{2^{M+1/2}} = \mathcal{O}\left(\frac{4\Gamma(M+1/2)^2}{\pi^{3/2}\Gamma(M+1)} \frac{n^{-M-1/2}}{2^{M+\frac{1}{2}}} \right), \tag{3.12}$$

where the last equality is the leading term in a series expansion of $R_{M,n}(\pi/2)$ for $n \gg M$. Solving (3.12) we obtain

$$n_M = \left\lfloor \frac{1}{2} \left(\epsilon \frac{\pi^{3/2}\Gamma(M+1)}{4\Gamma(M+1/2)^2} \right)^{\frac{-1}{M+\frac{1}{2}}} \right\rfloor, \tag{3.13}$$

and Table 3.1 gives some values of n_M for $3 \le M \le 15$. More generally, we can use (3.12) to derive the following *error curve*:

$$|R_{M,n}(\theta)| = \epsilon \quad \Longrightarrow \quad n \approx n_M / \sin\theta. \tag{3.14}$$

The curves clearly depend on M, and Figure 3.1 (right) depicts those for $M = 5, 7, 10, 15$ and $N = 1{,}000$. Similar figures appear in [20].

j_M gives the number of values $P_N(x_0), \ldots, P_N(x_{j_M - 1})$ that cannot be approximated by the asymptotic formula (3.3) to a tolerance of ϵ using M terms in the asymptotic formula. Thus, using (3.14), j_M is the number of points in the interval

TABLE 3.1

Algorithmic constants for $3 \leq M \leq 15$ and $\epsilon = 2.2 \times 10^{-16}$ in the regime of $N \gg n_M$ and $n \gg M$. $P_{n_M+1}(x)$ is the lowest degree Legendre polynomial that is evaluated at $x = 0$ to machine precision using M terms in the asymptotic formula, and j_M is such that $P_N(x_i)$ is evaluated to machine precision for any $j_M \leq i \leq N - j_M$.

M	3	4	5	6	7	8	9	10	11	12	13	14	15
n_M	16,072	2,053	583	252	139	90	65	50	41	35	30	27	25
j_M	5,000	658	185	80	44	28	20	15	13	11	9	8	7

$$0 \leq \theta \leq \sin^{-1}\left(\frac{n_M}{N}\right).$$

Since $\cos^{-1}(x_0), \ldots, \cos^{-1}(x_{j_M})$ are equally spaced with spacing $\pi/(N+1)$ in the θ-variable we have

$$j_M = \left\lfloor \frac{N+1}{\pi} \sin^{-1}\left(\frac{n_M}{N}\right) \right\rfloor. \tag{3.15}$$

Moreover, $\sin^{-1}(x) \approx x$ for $|x| \ll 1$, and hence for $N \gg n_M$ the parameter j_M is essentially independent of N. Table 3.1 gives the values of j_M for $3 \leq M \leq 15$.

Note that (3.13) and (3.15) for the algorithmic constants n_M and j_M assume that $n \gg M$ and $N \gg n_M$, respectively. In fact, the analysis here is intended not to be overly rigorous or technical but just to give estimates of the error curve, n_M, and j_M. A more technical analysis can be performed by taking more care when certain series expansions are employed, but this does not significantly change the practical properties of the algorithm.

We compute the resulting vector $\mathbf{P_N}(\underline{x}_N^{cheb})c_N^{leg}$ by applying the three matrices in (3.11) separately. The matrix-vector product $\mathbf{P}_\mathbf{N}^{\mathrm{REC}}(\underline{x}_N^{cheb})\underline{c}_N^{leg}$ can be computed in $\mathcal{O}(N)$ operations because the matrix $\mathbf{P}_\mathbf{N}^{\mathrm{REC}}(\underline{x}_N^{cheb})$ has fewer than $(2j_M + n_M)N = \mathcal{O}(N)$ nonzero entries. These entries cannot be computed via the asymptotic formula (3.3) because for these (n, θ) we have $|R_{M,n}(\theta)| > \epsilon$. Instead, we use the well-known three-term recurrence relation satisfied by Legendre polynomials [22]:

$$P_{n+1}(x) = \left(2 - \frac{1}{n+1}\right) x P_n(x) - \left(1 - \frac{1}{n+1}\right) P_{n-1}(x), \quad n \geq 1. \tag{3.16}$$

In this way, $\mathbf{P}_\mathbf{N}^{\mathrm{REC}}(\underline{x}_N^{cheb})\underline{c}_N^{leg}$ is computed without explicitly forming $\mathbf{P}_\mathbf{N}^{\mathrm{REC}}(\underline{x}_N^{cheb})$.

For the matrix-vector product $\mathbf{P}_\mathbf{N}^{\mathrm{DCT}}(\underline{x}_N^{cheb})\underline{c}_N^{leg}$ we write $\mathbf{P}_\mathbf{N}^{\mathrm{DCT}}(\underline{x}_N^{cheb})$ as the weighted sum of Chebyshev–Vandermonde matrices given in (3.10). However, since the first n_M columns and first and last j_M rows of $\mathbf{P}_\mathbf{N}^{\mathrm{DCT}}(\underline{x}_N^{cheb})$ are zero we must restrict the DCTs when applying the matrices $\mathbf{U_{N-1}}(\underline{x}_N^{cheb})$ and $\mathbf{T_N}(\underline{x}_N^{cheb})$. One can think of this as pre- and postmultiplying the Chebyshev–Vandermonde matrices by identity matrices with the first and last j_M and first n_M entries on the diagonal zeroed, respectively. As before, each multiplication by the Chebyshev–Vandermonde matrices can be computed in $O(N \log N)$ operations using the DCT.

Unfortunately, in computing $\mathbf{P}_\mathbf{N}^{\mathrm{DCT}}(\underline{x}_N^{cheb})\underline{c}_N^{leg}$ we have employed the asymptotic formula in a region where $|R_{M,n}(\theta)| > \epsilon$, and we must correct for this. To do so, we construct a correction matrix $\mathbf{P}_\mathbf{N}^{\mathrm{COR}}(\underline{x}_N^{cheb})$ (see Figure 3.1 (left)), with each nonzero entry equal to the true value of a Legendre polynomial minus the erroneous evaluation via the asymptotic formula (3.3). Thus, to compute each entry of $\mathbf{P}_\mathbf{N}^{\mathrm{COR}}(\underline{x}_N^{cheb})$ we

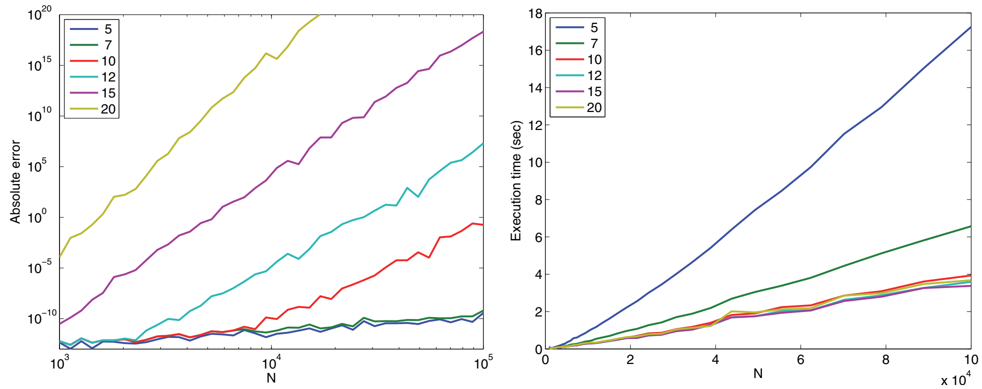

FIG. 3.2. *Left: The maximum error in the computed coefficients $\underline{c}_N^{cheb}$ for various M. For every $M \geq 1$ there is an integer $N_{\max}$ such that the algorithm is numerically unstable for $n > N_{\max}$. This situation can be remedied using the algorithm described in section* 3.3. *Right: The execution times for the same values of M and $10^3 \leq N \leq 10^5$. By selecting a small value of M one can ensure that the instability occurs only at a large N, but then the algorithm is much less efficient.*

evaluate the Legendre polynomial using the three-term recurrence (3.16) and subtract the value obtained from the asymptotic formula in the form (3.3). Fortunately, as can be derived by the analysis in [20], the matrix $\mathbf{P}_{\mathbf{N}}^{\mathrm{COR}}(\underline{x}_N^{cheb})$ contains $\mathcal{O}(N \log N)$ nonzero entries, and thus the correction vector $\mathbf{P}_{\mathbf{N}}^{\mathrm{COR}}(\underline{x}_N^{cheb})\underline{c}_N^{leg}$ can be computed in $\mathcal{O}(N \log N)$ operations. Since each of the matrices on the right-hand side of (3.2) can be applied in $\mathcal{O}(N \log N)$ operations, so can $\mathbf{P}_N(\underline{x}_N^{cheb})$, and hence the entire forward transformation.

The major problem with this algorithm, as described, is that for any M the transform becomes numerically unstable for sufficiently large N. The reason for this is cancellation error in floating point arithmetic. For large N the asymptotic formula can erroneously evaluate to arbitrarily large values outside the dashed line in Figure 3.1 (left), which means the entries in $\mathbf{P}_{\mathbf{N}}^{\mathrm{COR}}(\underline{x}_N^{cheb})$ lose all precision. This effect appears in practice, and in Figure 3.2 (left) we show the absolute error in the computed Chebyshev coefficients for various values of M between 5 and 20 with $1{,}000 \leq N \leq 10{,}000$. It is the cancellation error in computing the entries of $\mathbf{P}_{\mathbf{N}}^{\mathrm{COR}}(\underline{x}_N^{cheb})$ that makes the algorithm numerically unstable and therefore, for large N, it is not as useful as one might hope for computing the forward transform.

3.3. Block partitioning for numerical stability. Now we describe the algorithm that we do advocate for the forward transform based on a different partitioning of the Legendre–Vandermonde matrix $\mathbf{P}_{\mathbf{N}}(\underline{x}_N^{cheb})$. The algorithm is numerically stable and computes the vector $\mathbf{T}_{\mathbf{N}}(\underline{x}_N^{cheb})^{-1}\mathbf{P}_{\mathbf{N}}(\underline{x}_N^{cheb})\underline{c}_N^{leg}$ in $\mathcal{O}(N(\log N)^2/\log\log N)$ operations.

We partition the matrix $\mathbf{P}_{\mathbf{N}}(\underline{x}_N^{cheb})$ into $K+1$ submatrices, where K grows like $\mathcal{O}(\log N/\log\log N)$, in such a way that each submatrix can be applied to the vector c_N^{leg} in at most $\mathcal{O}(N \log N)$ operations. In particular, we partition $\mathbf{P}_{\mathbf{N}}(\underline{x}_N^{cheb})$ as

$$\mathbf{P}_{\mathbf{N}}(\underline{x}_N^{cheb}) = \mathbf{P}_{\mathbf{N}}^{\mathrm{REC}}(\underline{x}_N^{cheb}) + \sum_{k=1}^{K} \mathbf{P}_{\mathbf{N}}^{(k)}(\underline{x}_N^{cheb}), \tag{3.17}$$

where, for $k = 1, \ldots, K$, we have

$$\mathbf{P}_{\mathbf{N}}^{(k)}(\underline{x}_N^{cheb}) = \begin{cases} \mathbf{P}_{\mathbf{N}}(\underline{x}_N^{cheb})_{ij}, & i_k \le i \le N - i_k, \quad \alpha^k N \le j \le \alpha^{k-1} N, \\ 0 & \text{otherwise,} \end{cases}$$

$\alpha = \mathcal{O}(1/\log N)$ (see (3.19) for the precise definition), and

$$i_k = \left\lfloor \frac{N+1}{\pi} \sin^{-1}\left(\frac{n_M}{\alpha^k N}\right) \right\rfloor. \tag{3.18}$$

The value of i_k is the row index such that the submatrix $\mathbf{P_N}^{(k)}(\underline{x}_N^{cheb})$ nearly touches the error curve (3.14) and hence has a similar form as j_M in (3.15) with N replaced by $\alpha^k N$. Note that there is no need for a correction matrix $\mathbf{P}_{\mathbf{N}}^{\mathrm{COR}}(\underline{x}_N^{cheb})$ in this algorithm and that the matrix $\mathbf{P}_{\mathbf{N}}^{\mathrm{REC}}(\underline{x}_N^{cheb})$ has more nonzero entries than in section 3.2, as shown in Figure 3.3 for $K = 3$. We remark that since K grows relatively slowly with N, we require $N \ge 100{,}000$ for our algorithm to use $K \ge 3$ and $N \ge 10^6$ for $K \ge 4$.

This partitioning separates the matrix $\mathbf{P_N}(\underline{x}_N^{cheb})$ into submatrices $\mathbf{P}_{\mathbf{N}}^{(k)}(\underline{x}_N^{cheb})$ whose nonzero entries can be computed by using the asymptotic formula without correction and in such a way that the cost of computing $\mathbf{P_N}(\underline{x}_N^{cheb})\underline{c}_N^{leg}$ is minimal. The minimal cost is achieved, up to a constant, by balancing the cost of computing $\mathbf{P}_{\mathbf{N}}^{\mathrm{REC}}(\underline{x}_N^{cheb})\underline{c}_N^{leg}$ with the cost of the K matrix-vector products $\mathbf{P}_{\mathbf{N}}^{(k)}(\underline{x}_N^{cheb})\underline{c}_N^{leg}$ for $k = 1, \ldots, K$.

As we show later, the matrix $\mathbf{P}_{\mathbf{N}}^{\mathrm{REC}}(\underline{x}_N^{cheb})$ contains $\mathcal{O}(KN/\alpha)$ nonzero entries and hence can be applied to a vector in $\mathcal{O}(KN/\alpha)$ operations. In a similar fashion to the algorithm described in section 3.2, we compute the nonzero entries of $\mathbf{P}_{\mathbf{N}}^{\mathrm{REC}}(\underline{x}_N^{cheb})$ by using the three-term recurrence relation (3.16).

The other matrices $\mathbf{P}_{\mathbf{N}}^{(k)}(\underline{x}_N^{cheb})$ for $k = 1, \ldots, K$ are applied to a vector in $\mathcal{O}(N \log N)$ operations by employing the asymptotic formula (3.10) evaluated via the DCT. Notice that the nonzero entries of $\mathbf{P}_{\mathbf{N}}^{(k)}(\underline{x}_N^{cheb})$ form a rectangular submatrix of $\mathbf{P_N}(\underline{x}_N^{cheb})$, and therefore the matrix-vector product $\mathbf{P}_{\mathbf{N}}^{(k)}(\underline{x}_N^{cheb})\underline{c}_N^{leg}$ can be computed

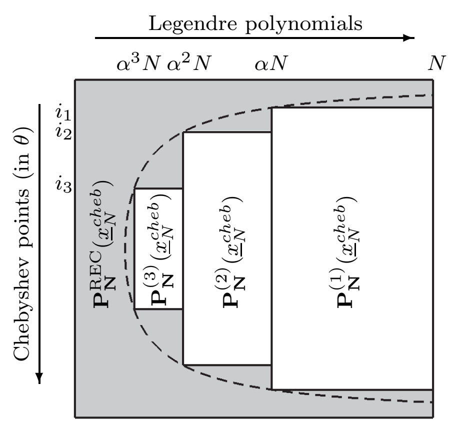

FIG. 3.3. *Partitioning of the matrix $\mathbf{P_N}(\underline{x}_N^{cheb})$ employed to compute the forward transform. The dashed line indicates the boundary of the region in which the asymptotic formula can be used without correction, and the gray region indicates the nonzero entries of the matrix $\mathbf{P}_{\mathbf{N}}^{\mathrm{REC}}(\underline{x}_N^{cheb})$. The diagram is not drawn to scale, and in practice the submatrix $\mathbf{P}_{\mathbf{N}}^{\mathrm{REC}}(\underline{x}_N^{cheb})$ occupies just a tiny proportion of $P_N(\underline{x}_N^{cheb})$.*

by restricting the DCTs when applying the matrices $\mathbf{U_{N-1}}(\underline{x}_N^{cheb})$ and $\mathbf{T_N}(\underline{x}_N^{cheb})$. Again, one can think of this as pre- and postmultiplying the Chebyshev–Vandermonde matrices by identity matrices with certain entries on the diagonal zeroed out.

We now tidy up some unfinished business and detail how to partition the matrix $\mathbf{P_N}(\underline{x}_N^{cheb})$ in (3.17) and analyze the complexity of the resulting algorithm. First, the number of nonzero entries in $\mathbf{P}_\mathbf{N}^{\rm REC}(\underline{x}_N^{cheb})$ can be calculated by artificially cutting it up into rectangular regions, and to leading order it has

$$\begin{aligned} 2\left(\sum_{k=1}^{K-1}\alpha^k N\left(i_{k+1}-i_k\right)\right) &\approx 2\left(\sum_{k=1}^{K-1}\alpha^k N\frac{N}{\pi}\left(\sin^{-1}\left(\frac{n_M}{\alpha^{k+1}N}\right)-\sin^{-1}\left(\frac{n_M}{\alpha^k N}\right)\right)\right) \\ &\approx \frac{2}{\pi}KN\left(\frac{1}{\alpha}-1\right) \end{aligned}$$

nonzero entries, where the last approximation uses $\sin^{-1}(x)\approx x$ for $|x|\ll 1$. Therefore, the leading order cost of computing $\mathbf{P}_\mathbf{N}^{(k)}(\underline{x}_N^{cheb})\underline{c}_N^{leg}$ is $\mathcal{O}(KN/\alpha)$ operations, and we want to balance this with the $\mathcal{O}(KN\log N)$ cost of computing $\mathbf{P}_\mathbf{N}^{\rm REC}(\underline{x}_N^{cheb})\underline{c}_N^{leg}$ for $k=1,\dots,K$. To balance we should select α such that $KN/\alpha = KN\log N$, i.e., $\alpha=\mathcal{O}\left(1/\log N\right)$. In practice, we have found that $\alpha=(1/\log(N/n_M))$ is a good choice for large N, and to avoid this becoming too close to 1 when N is small we take

$$\alpha=\min(1/\log(N/n_M),1/2). \tag{3.19}$$

Moreover, this discussion also determines K since we need to partition the matrix $\mathbf{P_N}(\underline{x}_N^{cheb})$ into $K+1$ parts so that $\alpha^{K+1}N<n_M$, and therefore we have

$$K=\mathcal{O}\left(\log N/\log\log N\right).$$

Putting this together, we have described an algorithm for the forward transform that requires $\mathcal{O}\left(KN\log N\right)$ operations, i.e., $\mathcal{O}\left(N(\log N)^2/\log\log N\right)$ operations. Furthermore, since the algorithm only employs the asymptotic formula for (n,θ), where $|R_{M,n}(\theta)|<\epsilon$, the transform is numerically stable. Additionally, the block partitioning means almost all computations can be vectorized, and since each of the $K+1$ matrix-vector multiplications as well as the DCTs in the asymptotic formula are independent, the algorithm is trivially parallelizable.

4. The inverse transform: Chebyshev to Legendre. The inverse transform converts a vector of Chebyshev coefficients, $\underline{c}_N^{cheb}$, to a vector of Legendre coefficients, $\underline{c}_N^{leg}$. Similarly to the forward transform, it can be represented by a matrix-vector product involving Chebyshev– and Legendre–Vandermonde matrices:

$$\underline{c}_N^{leg}=\mathbf{P_N}(\underline{x}_N^{cheb})^{-1}\mathbf{T_N}(\underline{x}_N^{cheb})\underline{c}_N^{cheb}. \tag{4.1}$$

We begin with the integral definition for the Legendre coefficients, i.e.,

$$c_n^{leg}=\frac{1}{\|P_n\|_2^2}\int_{-1}^{1}p_N(x)P_n(x)dx,\qquad 0\le k\le N, \tag{4.2}$$

where $P_n(x)$ is the degree n Legendre polynomial and $p_N(x)=\mathbf{T_N}(x)\underline{c}_N^{cheb}$ is the polynomial with Chebyshev coefficients $\underline{c}_N^{cheb}$ as in (1.1). Since $p_N(x)$ is a polynomial of degree at most N, for any $0\le n\le N$ the integrand, $p(x)P_n(x)$, is a polynomial of degree at most $2N$, and the $(2N+1)$-point Clenshaw–Curtis quadrature rule is exact for all the integrals in (4.2). Therefore, the Legendre coefficients satisfy the discrete sums

$$c_n^{leg} = \frac{1}{\|P_n\|_2^2} \sum_{j=0}^{2N} w_j p_N(x_j) P_n(x_j), \qquad 0 \le n \le N,$$

where $-1 \le x_{2N} < \cdots < x_0 \le 1$ and $w_{2N}, \ldots, w_0$, are the Clenshaw–Curtis quadrature nodes and weights, respectively. Again, we have indexed the nodes in reverse order for easier notation later. Note that these Clenshaw–Curtis nodes are just the Chebyshev points from (3.2) (with N replaced by $2N$) and hence $\underline{x}_{2N}^{cheb} = (x_0, \ldots, x_{2N})^T$. Moreover, denote by $\underline{w}_{2N}$ the vector of Clenshaw–Curtis weights, $\underline{w}_{2N} = (w_0, \ldots, w_{2N})^T$, which can be computed in $O(N \log N)$ operations using the algorithm of Waldvogel [39], and $\underline{s}_{2N}$, the orthonormalization vector for Legendre polynomials, $\underline{s}_{2N} = \left(\|P_0\|_2^{-2}, \ldots, \|P_{2N}\|_2^{-2}\right)^T$. With this notation, (4.2) takes the following compact form:

$$\begin{aligned} \underline{c}_N^{leg} &= \left[I_{N+1} \,|\, \mathbf{0}_N\right] D_{\underline{s}_{2N}} \mathbf{P_{2N}}(\underline{x}_{2N}^{cheb})^T D_{\underline{w}_{2N}}\, p_N(\underline{x}_{2N}^{cheb}) \\ &= \left[I_{N+1} \,|\, \mathbf{0}_N\right] D_{\underline{s}_{2N}} \mathbf{P_{2N}}(\underline{x}_{2N}^{cheb})^T D_{\underline{w}_{2N}}\, \mathbf{T_{2N}}(\underline{x}_{2N}^{cheb}) \begin{bmatrix} I_{N+1} \\ \mathbf{0}_N \end{bmatrix} \underline{c}_N^{cheb}, \end{aligned} \tag{4.3}$$

where $D_{\underline{w}_{2N}}$ and $D_{\underline{s}_{2N}}$ are diagonal matrices with diagonal entries $\underline{w}_{2N}$ and $\underline{s}_{2N}$, respectively.

The vector of Legendre coefficients $\underline{c}_N^{leg}$ in (4.3) has been expressed in terms of a matrix-vector product involving $\mathbf{P_{2N}}(\underline{x}_{2N}^{cheb})^T$, whereas the original relation (4.1) involves the inverse $\mathbf{P_N}(\underline{x}_N^{cheb})^{-1}$. Therefore, at the cost of doubling the size of the Legendre–Vandermonde matrices, we are able to employ the same asymptotic formula (3.3), as before. Note that the premultiplication by $\left[I_{N+1} \,|\, \mathbf{0}_N\right]$ in (4.3) means that only the first $N+1$ rows of $\mathbf{P_{2N}}(\underline{x}_{2N}^{cheb})^T$ are required in practice.

4.1. The transpose of the asymptotic formula. To apply the transposed Legendre–Vandermonde matrix, $\mathbf{P_{2N}}(\underline{x}_{2N}^{cheb})^T$, we transpose the asymptotic formula for quasimatrices (3.10):

$$\begin{aligned} \mathbf{P_{2N}}(\cos\theta)^T = &\sum_{m=0}^{M-1} D_{\underline{Ch}_m} \left([0 \,|\, \mathbf{U_{2N-1}}(\cos\theta)]^T D_{u_m(\theta)} + \mathbf{T_{2N}}(\cos\theta)^T D_{v_m(\theta)}\right) \\ &+ \mathbf{R}_M(\theta)^T, \end{aligned} \tag{4.4}$$

where $x = \cos\theta$. Thus, when $x = \underline{x}_{2N}^{cheb} = \cos(\underline{\theta}_{2N}^{cheb})$, the relation (4.4) expresses $\mathbf{P_{2N}}(\underline{x}_{2N}^{cheb})^T$ as a weighted sum of transposed Chebyshev–Vandermonde matrices $\mathbf{U_{2N-1}}(\underline{x}_{2N}^{cheb})^T$ and $\mathbf{T_{2N}}(\underline{x}_{2N}^{cheb})^T$. Since we have indexed the Chebyshev points in decreasing order, the Chebyshev–Vandermonde matrix $\mathbf{T_{2N}}(\underline{x}_{2N}^{cheb})$ is symmetric, i.e.,

$$\mathbf{T_{2N}}(\underline{x}_{2N}^{cheb})^T = \mathbf{T_{2N}}(\underline{x}_{2N}^{cheb}),$$

and can be applied to a vector in the same way as before.

For $[0 \,|\, \mathbf{U_{2N-1}}(\underline{x}_{2N}^{cheb})]^T$ we use the *conversion matrix* [23],

$$S_{2N-1} = \begin{pmatrix} 1 & 0 & -\frac{1}{2} & & & & \\ & \frac{1}{2} & 0 & -\frac{1}{2} & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \frac{1}{2} & 0 & -\frac{1}{2} \\ & & & & \frac{1}{2} & 0 \\ & & & & & \frac{1}{2} \end{pmatrix} \in \mathbb{R}^{2N\times 2N},$$

which converts Chebyshev coefficients in a series of $T_0,\dots,T_{2N-1}$ to coefficients in a series with $U_0,\dots,U_{2N-1}$ so that

$$\mathbf{U_{2N-1}}(\underline{x}_{2N}^{cheb})S_{2N-1} = \mathbf{T_{2N-1}}(\underline{x}_{2N}^{cheb}). \tag{4.5}$$

Using (4.5), we then have

$$[\mathbf{0}\,|\,\mathbf{U_{2N-1}}(\underline{x}_{2N}^{cheb})]^T = \begin{bmatrix} \mathbf{0} \\ \mathbf{U_{2N-1}}(\underline{x}_{2N}^{cheb})^T \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ S_{2N-1}^{-T}\mathbf{T_{2N-1}}(\underline{x}_{2N}^{cheb}) \end{bmatrix}. \tag{4.6}$$

Hence, we can apply $[0\,|\,\mathbf{U_{2N-1}}(\underline{x}_{2N}^{cheb})]^T$ to a vector in $\mathcal{O}(N\log N)$ operations by using the DCT and solving a lower triangular linear system with two nonzero diagonals in $\mathcal{O}(N)$ operations. Therefore, each of the terms in the asymptotic formula can be applied in $\mathcal{O}(N\log N)$ operations, and the doubling of the Chebyshev grid means the implied constant is around a factor of two larger than the forward transform.

4.2. Block partitioning for computing the inverse transform. As with the forward transform, we partition the transposed Legendre–Vandermonde matrix so that we employ the asymptotic formula (4.4) only for entries for which it gives an accurate approximation. In fact, the block partitioning is almost identical, since the error curve (3.14) is essentially the same. In particular, we partition $\mathbf{P_{2N}}(\underline{x}_{2N}^{cheb})^T$ so that

$$\mathbf{P_{2N}}(\underline{x}_{2N}^{cheb})^T = \mathbf{P}_{\mathbf{2N}}^{\mathrm{REC}}(\underline{x}_{2N}^{cheb})^T + \sum_{k=1}^{K}\mathbf{P}_{\mathbf{2N}}^{(k)}(\underline{x}_{2N}^{cheb})^T,$$

which can be seen in Figure 4.1, where i_k' is simply (3.18) with one N replaced by $2N$.

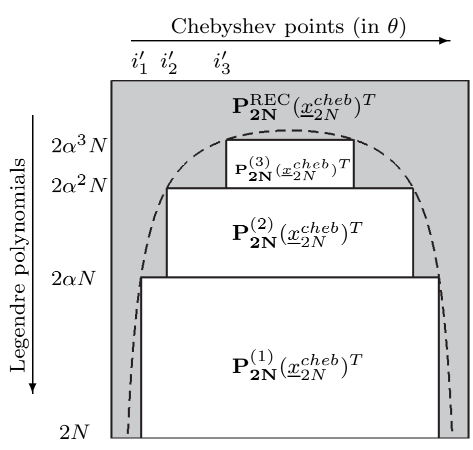

FIG. 4.1. *Partitioning of the matrix $\mathbf{P_{2N}}(\underline{x}_{2N}^{cheb})^T$ in the algorithm for the inverse transform. The dashed line indicates the boundary of the region in which the asymptotic formula can be employed without correction, and the gray region indicates the nonzero entries of the matrix $\mathbf{P}_{\mathbf{2N}}^{\mathrm{REC}}(\underline{x}_{2N}^{cheb})^T$. Again, this diagram is not drawn to scale and in practice the gray region represents a tiny proportion of the matrix $\mathbf{P_{2N}}(\underline{x}_{2N}^{cheb})^T$.*

As in section 3.3, to balance the computation costs we require $\alpha = \mathcal{O}(1/\log N)$, and hence $K = \mathcal{O}(\log N/\log\log N)$. To apply the matrix $\mathbf{P}_{\mathbf{2N}}^{\mathrm{REC}}(\underline{x}_{2N}^{cheb})^T$ to a vector we use the three-term recurrence relation (3.16) to compute its $\mathcal{O}(KN/\alpha)$ nonzero entries. For the matrix-vector products involving each $\mathbf{P}_{\mathbf{2N}}^{(k)}(\underline{x}_{2N}^{cheb})^T$, we use the transposed asymptotic formula (4.4) evaluated using the DCT and the relationship (4.6). Hence, in total, the numerically stable algorithm described for the inverse transform requires $\mathcal{O}(N(\log N)^2/\log\log N)$ operations to convert N Chebyshev coefficients to Legendre coefficients.

5. Numerical results. Here we no longer consider the unstable algorithm for the forward transform, described in section 3.2, and instead concentrate on the algorithms that we advocate for the forward and inverse transforms. All numerical experiments were performed on a single core of a 2011 1.8-GHz Intel Core i7 MacBook Air with MATLAB 2013a. Execution times should be considered as approximate. The accuracy results are determined by comparing to an extended precision multiplication of the vector of coefficients by the transformation matrices L^n and M^n from [2]. For timing comparisons, we compare against the direct multiplication of a vector by $\mathbf{P}_N(\underline{x}_N^{cheb})$ computed in MATLAB via the three-term recurrence relation.

5.1. Numerical results for the forward transform. In our implementation we use $M = 10$ for all N, though the efficiency of the algorithm for the forward transform is not particularly sensitive to the choice of M (see Figure 5.1 (left)). For $M = 10$ we find our algorithm is faster than the direct $O(N^2)$ computation when $N \geq 512$ and that $N = 10^6$ takes 31.2 seconds.

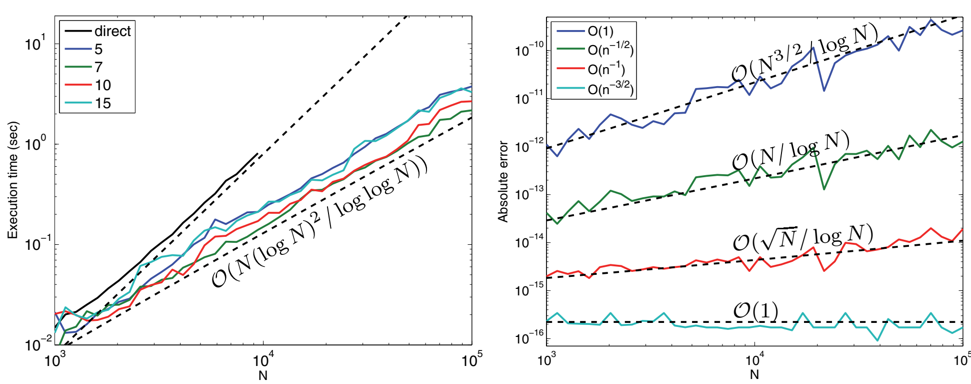

FIG. 5.1. *Left: Execution times for the forward transform for* $10^3 \leq N \leq 10^5$ *and* $M = 5, 7, 10, 15$ *in* MATLAB. *Right: For* $M = 10$*, the absolute error in the Chebyshev coefficients after converting* $N+1$ *Legendre coefficients to Chebyshev with different decay rates: no decay (blue),* $N^{-1/2}$ *(green),* N^{-1} *(red), and* $N^{-3/2}$ *(cyan).*

The dominating computational cost of our algorithm is in computing the DCT. Unfortunately, MATLAB does not natively support a DCT command,[3] but we can achieve the same transform via a FFT as follows [12, 36]:

[3]Note that the Signal Process Toolbox in MATLAB does supply a DCT command, but this utilizes the FFT in a similar way to the `dct1` code given above.

```
function v = dct1(c)
%DCT1   Compute a (scaled) DCT of type 1 using the FFT.
% DCT1(C) = T_N(X_N)*C, where X_N = cos(pi*(0:N))/N and T_N(X) = [T_0,
% T_1, ..., T_N](X). T_k is the kth 1st-kind Chebyshev polynomial.
  N = size(c, 1);                 % Number of terms.
  ii = N-1:-1:2;                  % Indicies of interior coefficients.
  c(ii) = 0.5*c(ii);              % Scale interior coefficients.
  v = ifft([c ; c(ii)]);          % Mirror coefficients and call FFT.
  v = (N-1)*[ 2*v(N) ; v(ii) + v(2*N-ii) ; 2*v(1) ]; % Re-order.
  v = flipud(v);                  % Flip the order.
end
```

However, the vector is doubled in length before applying the FFT and this means that this is twice as expensive as a DCT. We expect that the execution times of our algorithm would improve by nearly a factor of two if MATLAB allowed direct access to the FFTW DCT routines [10]. In Figure 5.1 (left) we show the execution times of the forward transform for $M = 5, 7, 10, 15$ and $10^3 \leq N \leq N^5$.

To get an idea of the accuracy we can expect in the forward transform, suppose the entries of the vector $\underline{c}_N^{leg}$ decay like n^{-r}, i.e., $(\underline{c}_N^{leg})_n = \mathcal{O}(n^{-r})$, and define $D_r = \operatorname{diag}(1^{-r}, \ldots, N^{-r})$. Then we have

$$\underline{c}_N^{leg} = D_r \hat{\underline{c}}_N^{leg},$$

where the vector $\hat{\underline{c}}_N^{leg}$ has no decay, and hence

$$\left\| \mathbf{P_N}(\underline{x}_N^{cheb}) \underline{c}_N^{leg} \right\|_\infty \leq \left\| \mathbf{P_N}(\underline{x}_N^{cheb}) D_r \right\|_\infty \left\| \hat{\underline{c}}_N^{leg} \right\|_\infty .$$

Since $\max_{x\in[-1,1]} P_n(x) = P_n(1) = 1$, the infinity norm of the matrix $\mathbf{P_N}(\underline{x}_N^{cheb}) D_r$ is the absolute sum of the first row, and hence

$$\left\| \mathbf{P_N}(\underline{x}_N^{cheb}) D_r \right\|_\infty = \sum_{n=1}^{N} n^{-r} = H_{N,r} = \begin{cases} N, & r = 0, \\ \mathcal{O}(\sqrt{N}\,), & r = 1/2, \\ \mathcal{O}(\log N), & r = 1, \\ \mathcal{O}(1), & r > 1, \end{cases}$$

where $H_{N,r}$ is the generalized harmonic number [3, Theorem 3.2]. Therefore, we expect the absolute error of the forward transform to grow something like $H_{N,r}$.

To investigate the actual error we observe in computing the forward transform we take random vectors[4] of Legendre coefficients that have entries decaying like n^{-r} with $r = 0, 1/2, 1, 3/2$. Figure 5.1 (right) shows the maximum absolute error in the computed Chebyshev coefficients. We observe an error growth of $\mathcal{O}(N^{3/2-r}/\log N)$ for $r = 0, 1/2, 1$ in the computed vector $\underline{c}_N^{cheb}$ and no error growth for $r = 3/2$. For $r = 0, 1/2, 1$, this observed error growth seems to be $\sqrt{N}/\log N$ times worse than $H_{N,r}$, and we cannot explain this mysterious factor. However, for suitably decaying

[4]Vectors are generated with the MATLAB command `randn` with the random number generator `mt19937ar` and `rng(1)`, and decay is introduced by scaling the nth entry by n^{-r}.

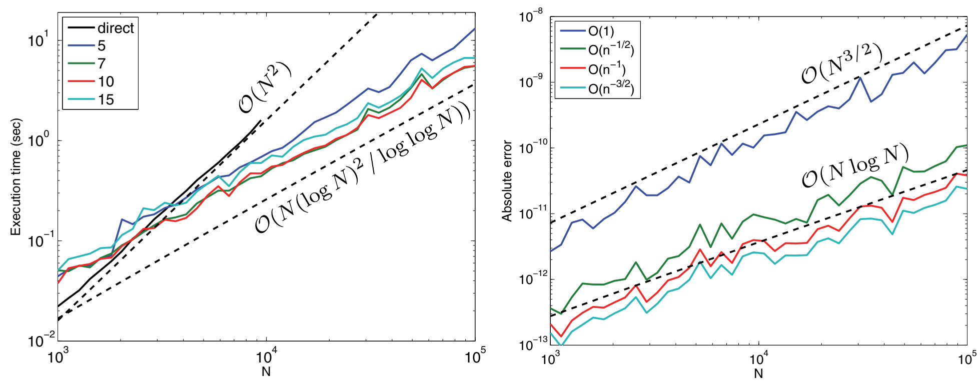

FIG. 5.2. *Left: Execution times for the inverse transform for* $10^3 \leq N \leq 10^5$ *and* $M = 5, 7, 10, 15$ *in* MATLAB. *Right: For* $M = 10$*, the absolute error in the* N *computed Legendre coefficients after converting Chebyshev coefficients with different decay rates: no decay (blue),* $n^{-1/2}$ *(green),* n^{-1} *(red), and* $n^{-3/2}$ *(cyan).*

vectors $\underline{c}_N^{leg}$, i.e., entries that decay like $n^{-3/2}$ or faster, the error in $\mathbf{P_N}(\underline{x}_N^{cheb})\underline{c}_N^{leg}$ remains bounded with N. Often, in practice, a Legendre expansion approximates a smooth function and hence the coefficients decay sufficiently. Note that the vectors of coefficients taken in this experiment are not derived directly from functions but instead mimic the algebraic decay that is expected from algebraically smooth functions. In particular, the observed absolute error does not include an error from truncating an infinite Chebyshev series of a function.

5.2. Numerical results for the inverse transform. Our implementation of the inverse transform also uses 10 terms in the asymptotic formula (4.4), but as before the efficiency of the algorithm is not particularly sensitive to the value of M (Figure 5.2 (left)). The cost of the inverse transform is approximately twice that of the forward transform because the algorithm for the inverse evaluates DCTs on vectors of twice the length.

In Figure 5.2 (right) we repeat the same accuracy experiment for the inverse transform. This time we observe an error growth of $\mathcal{O}(N^{3/2})$ for $r = 0$ and $\mathcal{O}(N \log N)$ for $r = 1/2, 1, 3/2$. An error growth of $\mathcal{O}(N)$ for any $r \geq 1/2$ is due to the orthogonalization scaling factor, $\|P_n\|_2^{-2}$, that appears in the integral definition (4.2) of the Legendre coefficients. Conventionally, Legendre polynomials are scaled so that $P_n(1) = 1$ for all n, and for this choice $\|P_n\|_2^{-2} = n + 1/2$. Our algorithm computes the coefficients in the orthonormal Legendre basis to essentially machine precision, but to convert them to the standard Legendre basis the nth coefficient is multiplied by $n + 1/2$. In addition, we again observe a mysterious log factor that we cannot explain.

To further investigate the error, we check that the forward and inverse transformations are numerically inverses of each other. We take random vectors and introduce decay by scaling them by n^{-r} with $r = 0, 1/2, 1, 3/2$, in the same way as in the previous experiments. In Figure 5.3 (left) we show the absolute error between the original vector and the vector converted to Legendre coefficients (forward transform) and then back. In Figure 5.3 (right) we reverse the order of the transforms, applying the inverse transform first, and check the absolute error between the original vector and the vector converted to Chebyshev coefficients and then back.

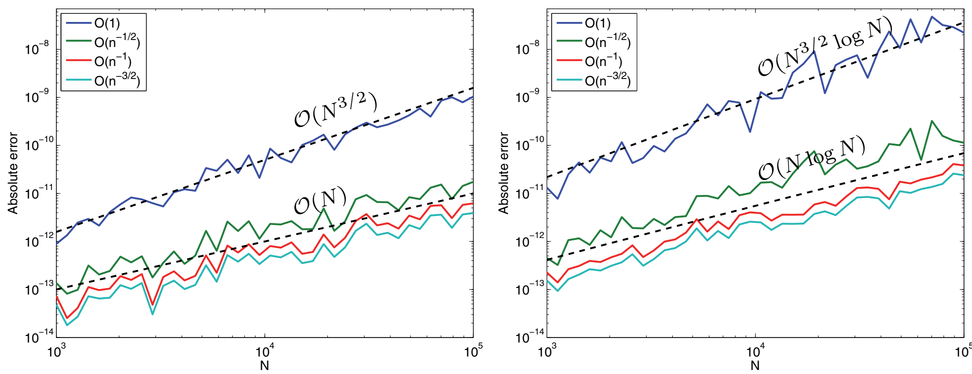

FIG. 5.3. *Left: The absolute error in N Chebyshev coefficients after converting them to Legendre coefficients and then back. Right: The absolute error in N Legendre coefficients after converting them to Chebyshev coefficients and then back. Here, we take random vectors that have been scaled to impose decay: no decay (blue), $n^{-1/2}$ (green), n^{-1} (red), and $n^{-3/2}$ (cyan).*

6. Extensions and conclusion. We have presented a fast Chebyshev–Legendre transform based on the asymptotic formula (3.3), and here we give possible extensions to other related fast transforms and some applications.

Fast evaluation of Legendre expansions. The forward transform converts Legendre coefficients to Chebyshev coefficients, and as an intermediary step of the transform the Legendre expansion of degree N (1.2) is evaluated at the vector of Chebyshev points $\underline{x}_N^{cheb}$. More precisely, $\underline{x}_N^{cheb}$ is the vector of Chebyshev points of the second kind. This immediately gives a fast algorithm for evaluating a Legendre expansion of degree N at $\underline{x}_N^{cheb}$ in $\mathcal{O}(N(\log N)^2/\log\log N)$ operations. A similar fast transform can be derived that evaluates a Legendre expansion at any other set of Chebyshev points:

first kind: $x_k = \cos\left(\frac{(k+\frac{1}{2})\pi}{N+1}\right)$, $0 \le k \le N$, the roots of T_{N+1};
second kind: $x_k = \cos\left(\frac{k\pi}{N}\right)$, $0 \le k \le N$, the roots of U_{N-1} and ± 1;
third kind: $x_k = \cos\left(\frac{k\pi}{N+\frac{1}{2}}\right)$, $0 \le k \le N$, the roots of V_N and 1 (see [22]);
fourth kind: $x_k = \cos\left(\frac{(k+\frac{1}{2})\pi}{N+\frac{1}{2}}\right)$, $0 \le k \le N$, the roots of W_N and -1 (see [22]).

These algorithms would still employ a DCT but of a different type: DCT-I for the second kind, DCT-III for the first kind, DCT-V for the third kind, and DCT-VII for the fourth kind (see [40] for further details).

Best least-squares approximation. The best least-squares approximation of degree $n \ge 0$ to a Chebyshev expansion,

$$p(x) = \sum_{j=0}^{N} c_j T_j(x), \qquad x \in [-1,1],$$

can be computed by truncating its Legendre expansion after $n+1$ terms. The forward Chebyshev–Legendre transform can be used to computed the best least-squares approximation in $\mathcal{O}(N(\log N)^2/\log\log N)$ operations. First, the vector of Chebyshev coefficients $(c_0,\ldots,c_N)^T$ are converted to Legendre coefficients using the forward transform, and then this vector is truncated to length $n+1$. If the Chebyshev coefficients of the best least-squares approximation are required, for example, in the `polyfit` command in Chebfun, the $n+1$ Legendre coefficient can be converted back to Chebyshev by the inverse transform.

Fast Chebyshev–Jacobi transform. Legendre polynomials are a special case of Jacobi polynomials, and Hahn [14, 22] gives a more general asymptotic formula that remains remarkably similar to (3.3). We expect that this asymptotic formula also leads to a fast Chebyshev–Jacobi transform with approximately the same methodology. However, we have not yet been able to find an accompanying explicit error formula, which is useful for deriving the details of the algorithm.

Fast spherical harmonic transform. The spherical harmonic expansion of a function takes the form

$$f(\theta,\phi) = \sum_{n=0}^{\infty}\sum_{m=-n}^{n} \alpha_n^m P_n^{|m|}(\cos\theta)e^{im\phi},$$

where (θ,ϕ) are spherical coordinates parameterizing the surface of the sphere embedded in $\mathbb{R}^3$ and $P_n^{|m|}$ is the associated Legendre polynomial of degree n and order $|m|$ [28]. There are many algorithms for the fast spherical harmonic transform [19, 28, 33], but it may also be possible to derive an algorithm that has a similar flavor to this paper. It would be interesting to consider the advantages, if any, of a fast algorithm for this transform based on an asymptotic formula for $P_n^{|m|}$, and we leave this for future work.

Conclusion. We have presented an $\mathcal{O}(N(\log N)^2/\log\log N)$ algorithm for the Chebyshev–Legendre transform, which is faster than the direct approach for $N \geq 512$. We block partitioned the Legendre–Vandermonde-like matrix to ensure that the asymptotic formula was evaluated only when it was valid, thus ensuring stability of the algorithm, and the use of DCTs allowed fast evaluation. If the coefficients in a truncated series expansion decay faster than $n^{-3/2}$, then the forward transform has no error growth with N and the inverse transform has an error growth of $\mathcal{O}(N\log N)$. Our publicly available MATLAB implementation [15] can convert between as many as 1 million Legendre coefficients and Chebyshev coefficients with high accuracy.

Acknowledgments. We would like to thank Nick Trefethen for his personal, intellectual, and financial support throughout this work and Anthony Austin for his assistance in translating a significant portion of a paper in Japanese [20]. We would also like to thank André Weideman, Sheehan Olver, and the Chebfun team for stimulating discussions, and we are grateful to Reiji Suda for privately communicating an unpublished manuscript.

REFERENCES

[1] N. Ahmed, T. Natarajan, and K. R. Rao, *Discrete cosine transform*, IEEE Trans. Comput., 100 (1974), pp. 90–93.
[2] B. K. Alpert and V. Rokhlin, *A fast algorithm for the evaluation of Legendre expansions*, SIAM J. Sci. Statist. Comput., 12 (1991), pp. 158–179.
[3] T. M. Apostol, *Introduction to Analytic Number Theory*, Undergrad. Texts Math., Springer, New York, 1976.
[4] K. Atkinson and W. Han, *Spherical Harmonics and Approximations on the Unit Sphere: An Introduction*, Lecture Notes in Math. 2044, Springer, New York, 2012.

[5] I. Bogaert, B. Michiels and J. Fostier, $\mathcal{O}(1)$ *computation of Legendre polynomials and Gauss–Legendre nodes and weights for parallel computing*, J. Sci. Comput., 34 (2012), pp. C83–C101.

[6] J. P. Boyd, *Chebyshev and Fourier Spectral Methods*, Dover Publications, New York, 2001.

[7] E. De Micheli and G. A. Viano, *The expansion in Gegenbauer polynomials: A simple method for the fast computation of the Gegenbauer coefficients*, J. Comput. Phys., 239 (2013), pp. 112–122.

[8] W. S. Don and D. Gottlieb, *The Chebyshev–Legendre method: Implementing Legendre methods on Chebyshev points*, SIAM J. Numer. Anal., 31 (1994), pp. 1519–1534.

[9] J. R. Driscoll and D. M. Healy, *Computing Fourier transforms and convolutions on the 2-sphere*, Adv. in Appl. Math., 15 (1994), pp. 202–250.

[10] M. Frigo and S. G. Johnson, *The design and implementation of FFTW*3, Proc. IEEE, 93 (2005), pp. 216–231.

[11] N. Gallagher, G. Wise, and J. Allen, *A novel approach for the computation of Legendre polynomial expansions*, IEEE Trans. Acoustic Speech Signal Process., 26 (1978), pp. 105–106.

[12] W. M. Gentleman, *Implementing Clenshaw–Curtis quadrature* II*: Computing the cosine transformation*, Commun. ACM, 15 (1972), pp. 343–346.

[13] M. Gu and S. C. Eisenstat, *A divide-and-conquer algorithm for the symmetric tridiagonal eigenproblem*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 172–191.

[14] E. Hahn, *Asymptotik bei Jacobi-Polynomen und Jacobi-Funktionen*, Math. Z., 171 (1980), pp. 201–226.

[15] N. Hale and A. Townsend, LEG2CHEB, http://github.com/nickhale/leg2cheb (2013).

[16] N. Hale and A. Townsend, *Fast and accurate computation of Gauss–Legendre and Gauss–Jacobi quadrature nodes and weights*, SIAM J. Sci. Comput., 35 (2013), pp. A652–A672.

[17] A. Iserles, *A fast and simple algorithm for the computation of Legendre coefficients*, Numer. Math., 117 (2011), pp. 529–553.

[18] J. C. Mason and D. C. Handscomb, *Chebyshev Polynomials*, Taylor & Francis, New York, 2002.

[19] M. J. Mohlenkamp, *A fast transform for spherical harmonics*, J. Fourier Anal. Appl., 5 (1999), pp. 159–184.

[20] A. Mori, R. Suda, and M. Sugihara, *An improvement on Orszag's fast algorithm for Legendre polynomial transform*, Trans. Info. Process. Soc. Japan, 40 (1999), pp. 3612–3615.

[21] S. Olver, *A general framework for solving Riemann–Hilbert problems numerically*, Numer. Math., 122 (2012), pp. 305–340.

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

[23] S. Olver and A. Townsend, *A fast and well-conditioned spectral method*, SIAM Rev., 55 (2013), pp. 462–489.

[24] M. O'Neil, F. Woolfe, and V. Rokhlin, *An algorithm for the rapid evaluation of special function transforms*, Appl. Comput. Harmon. Anal., 28 (2010), pp. 203–226.

[25] S. A. Orszag, *Fast eigenfunction transforms*, in Science and Computers, Academic Press, New York, 1986, pp. 13–30.

[26] R. Piessens, *Computation of Legendre series coefficients*, Comm. ACM, 17 (1974), p. 25.

[27] D. Potts, G. Steidl, and M. Tasche, *Fast algorithms for discrete polynomial transforms*, Math. Comp., 67 (1998), pp. 1577–1590.

[28] V. Rokhlin and M. Tygert, *Fast algorithms for spherical harmonic expansions*, SIAM J. Sci. Comput., 27 (2006), pp. 1903–1928.

[29] J. Shen, *Efficient spectral-Galerkin method* I. *Direct solvers of second- and fourth-order equations using Legendre polynomials*, SIAM J. Sci. Comput., 15 (1994), pp. 1489–1505.

[30] G. W. Stewart, *Afternotes Goes to Graduate School*, SIAM, Philadelphia, 1998.

[31] T. J. Stieltjes, *Sur les polynômes de Legendre*, Ann. Fac. Sci. Toulouse, 4 (1890), G1–G17.

[32] R. Suda, *Private communication*, University of Tokyo, 2013.

[33] R. Suda and M. Takami, *A fast spherical harmonics transform algorithm*, Math. Comput., 71 (2002), pp. 703–715.

[34] G. Szegő, *Orthogonal Polynomials*, American Mathematical Society, Providence, RI, 1939.

[35] L. N. Trefethen, *Approximation Theory and Approximation Practice*, SIAM, Philadelphia, 2013.

[36] L. N. Trefethen et al., Chebfun Version 4.2 (revision 2925), http://www.maths.ox.ac.uk/chebfun (2013).

[37] M. Tygert, *Fast algorithms for spherical harmonic expansions,* II, J. Comput. Phys., 227 (2008), pp. 4260–4279.

[38] M. Tygert, *Recurrence relations and fast algorithms*, Appl. Comput. Harmon. Anal., 28 (2010), pp. 121–128.
[39] J. Waldvogel, *Fast construction of the Fejér and Clenshaw–Curtis quadrature rules*, BIT, 43 (2003), pp. 1–18.
[40] Z. Wang and B. R. Hunt, *The discrete W transform*, Appl. Math. Comput., 16 (1985), pp. 19–48.
[41] N. Yarvin and V. Rokhlin, *A generalized one-dimensional fast multipole method with application to filtering of spherical harmonics*, J. Comput. Phys., 147 (1998), pp. 594–609.