Article

A High Accurate and Stable Legendre Transform Based on Block Partitioning and Butterfly Algorithm for NWP

Fukang Yin [1,*], Jianping Wu [2], Junqiang Song[3] and Jinhui Yang [4]

[1] College of Meteorology and Oceanography, National University of Defense Technology, Changsha, P.R. China, 410073; yinfukang@nudt.edu.cn
[2] College of Meteorology and Oceanography, National University of Defense Technology, Changsha, P.R. China, 410073; wjp@nudt.edu.cn
[3] College of Meteorology and Oceanography, National University of Defense Technology, Changsha, P.R. China, 410073; junqiang@nudt.edu.cn
[4] College of Meteorology and Oceanography, National University of Defense Technology, Changsha, P.R. China, 410073; yangjinhui@nudt.edu.cn
***** Correspondence: yinfukang@nudt.edu.cn; Tel.: +86-15084976786

Abstract: In this paper, we proposed a high accurate and stable Legendre transform algorithm which can reduce the potential instability for very high order at a very small increase in the computational time. The error analysis of interpolative decomposition for Legendre transform is presented. By employing block partitioning of Legendre-Vandermonde matrix and butterfly algorithm, a new Legendre transform algorithm with computational complexity $O(N(\log N)^2/\log\log N)$ in theory and less than $O(N\log^4 N)$ in practical application is obtained. Numerical results are provided to demonstrate the efficiency and numerical stability of the new algorithm.

Keywords: Legendre transform; block partitioning; interpolative decomposition; butterfly algorithm

1. Introduction

Legendre transform (LT) plays an important part in many scientific applications, such as astrophysical, numerical weather prediction and climate models. Fast Legendre transform attracts considerable interest amongst the scientific computing and numerical simulation. Scientists have paid very serious attention to develop fast Legendre transform algorithms [1-8]. The validity and reliability of these algorithms depend on whether they can keep both fast, stable and high accuracy.

Due to its numerical stability, low computational complexity and high accuracy, Tygert's algorithm (2010) [8] has been successfully implemented in IFS of ECMWF [9], YHGSM [10-12] of NUDT [13] and astrophysical [14]. In the applications of numerical weather prediction and climate models, which need many times SHT in each time step, only once precomputation is needed in first time step, then the results are stored in memory and reused in each transform. Though Tygert's algorithm (2010) is slow in terms of precomputation: $O(N^2)$ for LT and $O(N^3)$ for SHT, it doesn't have much impact on total performance. However, some unsolved issues still remain. The main issue is potential instability of interpolative decomposition (ID) [15] for very high order Legendre transform. Although, Tygert [8] points out that the reason why the butterfly procedure works so well for associated Legendre functions may be is that the associated transforms nearly weighted averages of Fourier integral operators. There are no literatures to prove that the pre-computations will compress the appropriate $n\times n$ matrix enough to enable application of the matrix to vectors using only $O(N\log N)$ floating-point operations(flops). Full numerical stability has been demonstrated both empirically and theoretically for FFT using butterfly algorithm. It is difficult to give complete and rigorous proofs of interpolative decomposition for Legendre transform as Fourier transform.

Non-oscillatory phase functions method opens up new avenues for special function transforms. The solutions of some kinds of second order differential equations can be accurately represented by non-oscillatory phase functions [16,17]. It has been proved that Legendre's differential equation [18] and its generalization Jacobi's differential equation [19] admit a non-oscillatory phase function. So non-oscillatory phase functions can be used to the expansions [19], the calculation of the roots [20] and transform [21] of special functions. Jacobi transform by non-oscillatory phase functions shows an optimal computational complexity $O(N\log^2(N)/\log\log N)$ in reference [21]. However, Legendre transform algorithm in ButterflyLab (https://github.com/ButterflyLab/ButterflyLab), which adopts interpolative butterfly factorization (IBF) [22, 23] and non-oscillatory phase functions method to evaluate the Legendre polynomials [21], does not show high accuracy as Fourier transform using IBF. So, Fast Legendre transform (FLT) based on IBF and non-oscillatory phase functions and its extension to the associated Legendre functions needs further study.

Recently, fast Legendre transform algorithm based on FFT deserved more attentions for its optimal computational complexity $O(N\log^2(N)/\log\log N)$. Hale and Townsend [24] firstly presented a fast Chebyshev-Legendre transform, and then developed a non-uniform discrete cosine transform which use a Taylor series expansion for Chebyshev polynomials about equally-spaced points in the frequency domain. Finally, Hale and Townsend [25] got an $O(N(\log N)^2/\log\log N)$ Legendre transform algorithm. Soon, fast polynomial transforms [26] based on Toeplitz and Hankel matrices was presented to accelerate the Chebyshev-Legendre transform. Although FFT-based LT has the attractive computational complexity, it needs too many times FFT which makes FFT-based LT only become more computationally efficient than LT using Dgemv when N is greater than or equal to 5000. Because the computation of associated-Legendre-Vandermonde matrices is completed in pre-computation step, it will become worse on the occasion of multiple use of FLT such as NWP, in which only once computation of associated-Legendre-Vandermonde matrices is needed for many times spectral harmonic transform (SHT).

Motivated and inspired by the ongoing research in these areas, we present a theoretical method to analyze the error of LT using butterfly algorithm, and then provide a numerically stability Legendre transform algorithm based on block partitioning and butterfly algorithm. The novel aspect is the mitigation of the potential instability of LT using butterfly algorithm at a very small increase of computational cost.

2. Mathematical preliminaries

In this section, we introduce the theorem that Legendre polynomials on equally-spaced grid can be expressed as a weighted linear combination of Chebyshev polynomials, and a partitioning of Legendre-Vandermonde matrix $\mathbf{P}_N\left(x_N^{cheb}\right)$ ($x=\underline{x}_N^{cheb}=\cos\left(\underline{\theta}_N^{cheb}\right)$). For more details, see reference [24,25].

According to Stieltjes's theory [27], Legendre polynomials can be expressed as following asymptotic formula when $n\to\infty$

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

where $\theta=\cos^{-1}x$, $\theta\in(0,\pi)$ and

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

$$h_{m,n}=\begin{cases}1, & m=0,\\ \prod_{j=1}^{m}\frac{(j-1/2)^2}{j(n+j+1/2)}, & m>0.\end{cases} \tag{3}$$

The error term in Eq. (1) can be bounded by

$$\left|R_{M,n}(\theta)\right| \le C_n h_{M,n} \frac{2}{(2\sin\theta)^{M+1/2}}, \tag{4}$$

Hale and Townsend [22] rewrote Eq. (1) as a weighted linear combination of Chebyshev polynomials

$$P_n(\cos\theta) = C_n \sum_{m=0}^{M-1} h_{m,n}\left(u_m(\theta)T_n(\sin\theta) + v_m(\theta)T_n(\cos\theta)\right) + R_{M,n}(\theta), \tag{5}$$

with $T_n(\cos\theta) = \cos(n\theta)$, $T_n(\sin\theta) = \sin(n\theta)$ and

$$u_m(\theta) = \frac{\sin\left((m+1/2)\left(\frac{\pi}{2}-\theta\right)\right)}{(2\sin\theta)^{m+1/2}}, v_m(\theta) = \frac{\cos\left((m+1/2)\left(\frac{\pi}{2}-\theta\right)\right)}{(2\sin\theta)^{m+1/2}}. \tag{6}$$

Let $x_k^{leg} = \cos\left(\theta_k^{leg}\right)$ and $\theta_0^{leg}, \ldots, \theta_{N-1}^{leg}$ are the transformed Legendre nodes, Eq. (5) can be written as

$$P_n\left(x_k^{leg}\right) = C_n \sum_{m=0}^{M-1} h_{m,n}\left(u_m\left(\theta_k^{leg}\right)T_n\left(\sin\left(\theta_k^{leg}\right)\right) + v_m\left(\theta_k^{leg}\right)T_n\left(\cos\left(\theta_k^{leg}\right)\right)\right) + R_{M,n}\left(\theta_k^{leg}\right). \tag{7}$$

3 Error analysis of Legendre transform using butterfly algorithm

The transformed Legendre nodes $\theta_0^{leg}, \ldots, \theta_{N-1}^{leg}$ can be seen as a perturbation of an equally-spaced grid $\theta_0^*, \ldots, \theta_{N-1}^*$, i.e

$$\theta_k^{leg} = \theta_k^* + \delta\theta_k, \quad 0 \le k \le N-1, \tag{8}$$

and then approximate each $x_k^{leg} = \cos\left(n\theta_k^{leg}\right)$ term by a truncated Taylor series expansion about θ_k^*. If $|\delta\theta_k|$ is small then only a few terms in the Taylor expansion are required.

The Taylor series expansion of $T_n(\cos(\theta+\delta\theta)) = \cos(n(\theta+\delta\theta))$ about $\theta \in [0,\pi]$ can be expressed as

$$\begin{aligned}\cos(n(\theta+\delta\theta)) &= \cos(n\theta) + \sum_{l=1}^{\infty} \cos^{(l)}(n\theta)\frac{(n\delta\theta)^l}{l!} \\ &= \cos(n\theta) + \sum_{l=1}^{\infty} (-1)^{\lfloor (l+1)/2 \rfloor} \Phi_l(n\theta)\frac{(n\delta\theta)^l}{l!}\end{aligned}, \tag{9}$$

where

$$\Phi_l(\theta) = \begin{cases}\cos(\theta), & l \ \ even \\ \sin(\theta), & l \ \ odd\end{cases}. \tag{10}$$

Similarly $T_n(\sin(\theta+\delta\theta)) = \sin(n(\theta+\delta\theta))$ about $\theta \in [0,\pi]$ can be expressed as

$$\begin{aligned}\sin(n(\theta+\delta\theta)) &= \sin(n\theta) + \sum_{l=1}^{\infty} \sin^{(l)}(n\theta)\frac{(n\delta\theta)^l}{l!} \\ &= \sin(n\theta) + \sum_{l=1}^{\infty} (-1)^{\lfloor l/2 \rfloor} \Psi_l(n\theta)\frac{(n\delta\theta)^l}{l!}\end{aligned}, \tag{11}$$

where

$$\Psi_l(\theta) = \begin{cases}\cos(\theta), & l \ \ odd \\ \sin(\theta), & l \ \ even\end{cases}, \tag{12}$$

Substituting θ_k^* for θ in Eq. (5), one can get

$$\begin{aligned}P_n\left(\cos\left(\theta_k^*\right)\right) = {} & C_n T_n\left(\sin\left(\theta_k^*\right)\right)\sum_{m=0}^{M-1} h_{m,n} u_m\left(\theta_k^*\right) \\ & + C_n T_n\left(\cos\left(\theta_k^*\right)\right)\sum_{m=0}^{M-1} h_{m,n} v_m\left(\theta_k^*\right) + R_{M,n}\left(\theta_k^*\right)\end{aligned}. \tag{13}$$

The Taylor series expansion of $P_n\left(\cos\left(\theta_k^{leg}\right)\right)$ about θ_k^* can be expressed as

$$P_n\left(\cos\left(\theta_k^{leg}\right)\right)=P_n\left(\cos\left(\theta_k^*+\delta\theta_k\right)\right)=\sum_{l=0}^{\infty}P_n^{(l)}\left(\cos\left(\theta_k^*\right)\right)\frac{\left(\delta\theta_k\right)^l}{l!}. \tag{14}$$

According to Eq. (13), $P_n^{(l)}\left(\cos\left(\theta_k^*\right)\right)$ ($l>0$) can be written as

$$\begin{aligned}P_n^{(l)}\left(\cos\left(\theta_k^*\right)\right)= {} & C_n\sum_{m=0}^{M-1}h_{m,n}\left\{u_m\left(\theta_k^*\right)T_n^{(l)}\left(\sin\left(\theta_k^*\right)\right)+u_m^{(l)}\left(\theta_k^*\right)T_n\left(\sin\left(\theta_k^*\right)\right)\right\}\\ & +C_n\sum_{m=0}^{M-1}h_{m,n}\left\{v_m\left(\theta_k^*\right)T_n^{(l)}\left(\cos\left(\theta_k^*\right)\right)+v_m^{(l)}\left(\theta_k^*\right)T_n\left(\cos\left(\theta_k^*\right)\right)\right\},\\ & +R_{M,n}^l\left(\theta_k^*\right)\end{aligned} \tag{15}$$

Substituting Eq. (15) into Eq. (14), one can obtain

$$\begin{aligned}P_n\left(x_k^{leg}\right)= {} & C_n\sum_{m=0}^{M-1}h_{m,n}\left\{u_m\left(\theta_k^*\right)T_n\left(\sin\left(\theta_k^*\right)\right)+v_m\left(\theta_k^*\right)T_n\left(\cos\left(\theta_k^*\right)\right)\right\}\\ & +C_n\sum_{l=1}^{\infty}\frac{\left(\delta\theta_k\right)^l}{l!}\sum_{m=0}^{M-1}h_{m,n}\left\{u_m\left(\theta_k^*\right)T_n^{(l)}\left(\sin\left(\theta_k^*\right)\right)+u_m^{(l)}\left(\theta_k^*\right)T_n\left(\sin\left(\theta_k^*\right)\right)\right\}\\ & +C_n\sum_{l=1}^{\infty}\frac{\left(\delta\theta_k\right)^l}{l!}\sum_{m=0}^{M-1}h_{m,n}\left\{v_m\left(\theta_k^*\right)T_n^{(l)}\left(\cos\left(\theta_k^*\right)\right)+v_m^{(l)}\left(\theta_k^*\right)T_n\left(\cos\left(\theta_k^*\right)\right)\right\}\\ & +\sum_{l=0}^{\infty}R_{M,n}^{(l)}\left(\theta_k^*\right)\frac{\left(\delta\theta_k\right)^l}{l!}\end{aligned}. \tag{16}$$

Because

$$\begin{aligned}\sum_{l=0}^{\infty}\frac{\left(\delta\theta_k\right)^l}{l!}T_n\left(\sin\left(\theta_k^*\right)\right)\sum_{m=0}^{M-1}h_{m,n}u_m^{(l)}\left(\theta_k^*\right) & =T_n\left(\sin\left(\theta_k^*\right)\right)\sum_{m=0}^{M-1}h_{m,n}\sum_{l=0}^{\infty}u_m^{(l)}\left(\theta_k^*\right)\frac{\left(\delta\theta_k\right)^l}{l!}\\ & =T_n\left(\sin\left(\theta_k^*\right)\right)\sum_{m=0}^{M-1}h_{m,n}u_m\left(\theta_k^*+\delta\theta_k\right)\\ & =T_n\left(\sin\left(\theta_k^*\right)\right)\sum_{m=0}^{M-1}h_{m,n}u_m\left(\theta_k^{leg}\right)\end{aligned} \tag{17}$$

and

$$\begin{aligned}\sum_{l=0}^{\infty}\frac{\left(\delta\theta_k\right)^l}{l!}T_n\left(\cos\left(\theta_k^*\right)\right)\sum_{m=0}^{M-1}h_{m,n}v_m^{(l)}\left(\theta_k^*\right) & =T_n\left(\cos\left(\theta_k^*\right)\right)\sum_{m=0}^{M-1}h_{m,n}\sum_{l=0}^{\infty}v_m^{(l)}\left(\theta_k^*\right)\frac{\left(\delta\theta_k\right)^l}{l!}\\ & =T_n\left(\cos\left(\theta_k^*\right)\right)\sum_{m=0}^{M-1}h_{m,n}v_m\left(\theta_k^*+\delta\theta_k\right)\\ & =T_n\left(\cos\left(\theta_k^*\right)\right)\sum_{m=0}^{M-1}h_{m,n}v_m\left(\theta_k^{leg}\right)\end{aligned}. \tag{18}$$

Similarly, we have

$$\begin{aligned}\sum_{l=0}^{\infty}\frac{\left(\delta\theta_k\right)^l}{l!}\sum_{m=0}^{M-1}h_{m,n}u_m\left(\theta_k^*\right)T_n^{(l)}\left(\sin\left(\theta_k^*\right)\right) & =\sum_{m=0}^{M-1}h_{m,n}u_m\left(\theta_k^*\right)\sum_{l=0}^{\infty}T_n^{(l)}\left(\sin\left(\theta_k^*\right)\right)\frac{\left(\delta\theta_k\right)^l}{l!}\\ & =\sum_{m=0}^{M-1}h_{m,n}u_m\left(\theta_k^*\right)T_n\left(\sin\left(\theta_k^{leg}\right)\right)\end{aligned}, \tag{19}$$

and

$$\begin{aligned}\sum_{l=0}^{\infty}\frac{\left(\delta\theta_k\right)^l}{l!}\sum_{m=0}^{M-1}h_{m,n}v_m\left(\theta_k^*\right)T_n^{(l)}\left(\cos\left(\theta_k^*\right)\right) & =\sum_{m=0}^{M-1}h_{m,n}v_m\left(\theta_k^*\right)\sum_{l=0}^{\infty}T_n^{(l)}\left(\cos\left(\theta_k^*\right)\right)\frac{\left(\delta\theta_k\right)^l}{l!}\\ & =\sum_{m=0}^{M-1}h_{m,n}v_m\left(\theta_k^*\right)T_n\left(\cos\left(\theta_k^{leg}\right)\right)\end{aligned}. \tag{20}$$

Substituting Eq. (17) to Eq. (20) into Eq. (16), one can get

$$\begin{aligned}P_n\left(x_k^{leg}\right)= &\ C_n\sum_{m=0}^{M-1}h_{m,n}\left(u_m\left(\theta_k^{leg}\right)T_n\left(\sin\left(\theta_k^*\right)\right)+v_m\left(\theta_k^{leg}\right)T_n\left(\cos\left(\theta_k^*\right)\right)\right)\\ &+C_n\sum_{m=0}^{M-1}h_{m,n}\left\{u_m\left(\theta_k^*\right)T_n\left(\sin\left(\theta_k^{leg}\right)\right)+v_m\left(\theta_k^*\right)T_n\left(\cos\left(\theta_k^{leg}\right)\right)\right\}\\ &-C_n\sum_{m=0}^{M-1}h_{m,n}\left\{u_m\left(\theta_k^*\right)T_n\left(\sin\left(\theta_k^*\right)\right)+v_m\left(\theta_k^*\right)T_n\left(\cos\left(\theta_k^*\right)\right)\right\}\\ &+R_{M,n}\left(\theta_k^{leg}\right)\end{aligned}. \tag{21}$$

Then

$$\begin{aligned}P_n\left(x_k^{leg}\right)= &\ C_n\sum_{m=0}^{M-1}h_{m,n}\left(u_m\left(\theta_k^{leg}\right)T_n\left(\sin\left(\theta_k^*\right)\right)+v_m\left(\theta_k^{leg}\right)T_n\left(\cos\left(\theta_k^*\right)\right)\right)\\ &+C_n\sum_{m=0}^{M-1}h_{m,n}\left\{u_m\left(\theta_k^*\right)T_n\left(\sin\left(\theta_k^{leg}\right)\right)+v_m\left(\theta_k^*\right)T_n\left(\cos\left(\theta_k^{leg}\right)\right)\right\}\\ &-P_n\left(x_k^*\right)+R_{M,n}\left(\theta_k^*\right)+R_{M,n}\left(\theta_k^{leg}\right)\end{aligned}. \tag{22}$$

By truncating the second term in the right hand side of Eq. (19), it can be approximated as

$$\begin{aligned}P_n\left(x_k^{leg}\right)= &\ C_n\sum_{m=0}^{M-1}h_{m,n}\left(u_m\left(\theta_k^{leg}\right)T_n\left(\sin\left(\theta_k^*\right)\right)+v_m\left(\theta_k^{leg}\right)T_n\left(\cos\left(\theta_k^*\right)\right)\right)\\ &+C_n\sum_{l=1}^{L}\frac{\left(\delta\theta_k\right)^l}{l!}\sum_{m=0}^{M-1}h_{m,n}\left\{u_m\left(\theta_k^*\right)T_n^{(l)}\left(\sin\left(\theta_k^*\right)\right)+v_m\left(\theta_k^*\right)T_n^{(l)}\left(\cos\left(\theta_k^*\right)\right)\right\},\\ &+R_{M,n}\left(\theta_k^{leg}\right)+R_{L,M,n,\delta\theta}\end{aligned} \tag{23}$$

and then

$$\begin{aligned}P_n\left(x_k^{leg}\right)= &\ C_n\sum_{m=0}^{M-1}h_{m,n}\left(u_m\left(\theta_k^{leg}\right)T_n\left(\sin\left(\theta_k^*\right)\right)+v_m\left(\theta_k^{leg}\right)T_n\left(\cos\left(\theta_k^*\right)\right)\right)\\ &+C_n\sum_{l\ odd}^{L}\frac{\left(n\delta\theta_k\right)^l}{l!}\sum_{m=0}^{M-1}h_{m,n}\left((-1)^{\left\lfloor \frac{l}{2}\right\rfloor}u_m\left(\theta_k^*\right)T_n\left(\cos\theta_k^*\right)+(-1)^{\left\lfloor \frac{(l+1)}{2}\right\rfloor}v_m\left(\theta_k^*\right)T_n\left(\sin\theta_k^*\right)\right)\\ &+C_n\sum_{l\ even}^{L}\frac{\left(n\delta\theta_k\right)^l}{l!}\sum_{m=0}^{M-1}h_{m,n}\left((-1)^{\left\lfloor \frac{l}{2}\right\rfloor}u_m\left(\theta_k^*\right)T_n\left(\sin\theta_k^*\right)+(-1)^{\left\lfloor \frac{(l+1)}{2}\right\rfloor}v_m\left(\theta_k^*\right)T_n\left(\cos\theta_k^*\right)\right)\\ &+R_{M,n}\left(\theta_k^{leg}\right)+R_{L,M,n,\delta\theta}\end{aligned}. \tag{24}$$

Eq. (24) can be expressed in the following compact form

$$\begin{aligned}P_n\left(x_k^{leg}\right)\approx\left(U_n+V_n\right)&+\sum_{l\ odd}^{L}\frac{\left(n\delta\theta_k\right)^l}{l!}\left((-1)^{\left\lfloor \frac{l}{2}\right\rfloor}U_{nc}+(-1)^{\left\lfloor \frac{(l+1)}{2}\right\rfloor}V_{ns}\right)\\ &+\sum_{l\ even}^{L}\frac{\left(n\delta\theta_k\right)^l}{l!}\left((-1)^{\left\lfloor \frac{l}{2}\right\rfloor}U_{ns}+(-1)^{\left\lfloor \frac{(l+1)}{2}\right\rfloor}V_{nc}\right)\end{aligned}, \tag{25}$$

where

$$\begin{aligned}U_n&=C_n\sum_{m=0}^{M-1}h_{m,n}u_m\left(\theta_k^{leg}\right)T_n\left(\sin\left(\theta_k^*\right)\right),\quad V_n=C_n\sum_{m=0}^{M-1}h_{m,n}v_m\left(\theta_k^{leg}\right)T_n\left(\cos\left(\theta_k^*\right)\right)\\ U_{ns}&=C_n\sum_{m=0}^{M-1}h_{m,n}u_m\left(\theta_k^*\right)T_n\left(\sin\left(\theta_k^*\right)\right),\quad V_{ns}=C_n\sum_{m=0}^{M-1}h_{m,n}v_m\left(\theta_k^*\right)T_n\left(\sin\left(\theta_k^*\right)\right)\\ U_{nc}&=C_n\sum_{m=0}^{M-1}h_{m,n}u_m\left(\theta_k^*\right)T_n\left(\cos\left(\theta_k^*\right)\right),\quad V_{nc}=C_n\sum_{m=0}^{M-1}h_{m,n}v_m\left(\theta_k^*\right)T_n\left(\cos\left(\theta_k^*\right)\right)\end{aligned}. \tag{26}$$

So, the computation of Legendre-Vandermonde matrix can be written as

$$\begin{aligned}\mathbf{P}_{\mathrm{N}}\left(\underline{x}_N^{leg}\right)=\left(\mathbf{U}_{\mathrm{N}}+\mathbf{V}_{\mathrm{N}}\right)&+\sum_{l\ odd}^{L}\frac{\left(n\delta\theta_k\right)^l}{l!}\left((-1)^{\left\lfloor \frac{l}{2}\right\rfloor}\mathbf{U}_{\mathrm{c}}+(-1)^{\left\lfloor \frac{(l+1)}{2}\right\rfloor}\mathbf{V}_{\mathrm{s}}\right)\\ &+\sum_{l\ even}^{L}\frac{\left(n\delta\theta_k\right)^l}{l!}\left((-1)^{\left\lfloor \frac{l}{2}\right\rfloor}\mathbf{U}_{\mathrm{s}}+(-1)^{\left\lfloor \frac{(l+1)}{2}\right\rfloor}\mathbf{V}_{\mathrm{c}}\right)\\ &+\qquad\qquad\qquad\qquad\mathbf{R}_{\mathrm{total}}\end{aligned}. \tag{27}$$

The numerical stability of ID can be analyzed by Eq. (27). Since the butterfly algorithm works well for equispaced Fourier series, Legendre transform using butterfly algorithm is numerical stability with the error of $\mathbf{R}_{\text{total}}$. When L tends to infinity, the error is $R_{M,n}\left(\theta_k^{leg}\right)$.

LEMMA 1: For any $L \geq 1$ and $n \geq 0$ [25]

$$R_{L,n,\delta\theta} := \max_{\theta\in[0,\pi]}\left|\cos\left(n\left(\theta+\delta\theta\right)\right)-\sum_{l=0}^{L-1}\cos^{(l)}\left(n\theta\right)\frac{\left(\delta\theta\right)^l}{l!}\right| \leq \frac{\left(n\left|\delta\theta\right|\right)^L}{L!}. \tag{28}$$

LEMMA 2: For any $L \geq 1$ and $n \geq 0$, the error bound of Eq. (22) is

$$R \leq \frac{2C_n h_{M,n}}{L!}\left(\frac{\left(n\left|\delta\theta_k\right|\right)^L}{\left(2\sin\theta_k^{cheb}\right)^{M+\frac{1}{2}}}+\frac{1}{\left(2\sin\theta_k^{leg}\right)^{M+\frac{1}{2}}}\right), \tag{29}$$

Proof.

$$\begin{aligned}
\left|R_{M,L,n,\delta\theta_k}\right| &= \left|\sum_{l=0}^{L}\left(-1\right)^{\left\lfloor\frac{(l+1)}{2}\right\rfloor}C_n\sum_{m=0}^{M-1}h_{m,n}\left(u_m\left(\theta_k^*\right)\Psi_l\left(n\theta_k^*\right)+v_m\left(\theta_k^*\right)\Phi_l\left(n\theta_k^*\right)\right)\frac{\left(n\delta\theta_k\right)^l}{l!}\right| \\
&\leq \left|\frac{\left(n\left|\delta\theta_k\right|\right)^L}{L!}C_n\sum_{m=0}^{M-1}h_{m,n}\left|\left(u_m\left(\theta_k^*\right)\Psi_l\left(n\theta_k^*\right)+v_m\left(\theta_k^*\right)\Phi_l\left(n\theta_k^*\right)\right)\right|\right| \\
&\leq C_n h_{M,n}\frac{2}{\left(2\sin\theta_k^*\right)^{M+1/2}}\frac{\left(n\left|\delta\theta_k\right|\right)^L}{L!}
\end{aligned} \tag{30}$$

Finally, one can get the total upper error bound

$$R = \left|R_{M,L,n,\delta\theta_k}+R_{M,n}\left(\theta_k^{leg}\right)\right| \leq \frac{2C_n h_{M,n}}{L!}\left(\frac{\left(n\left|\delta\theta_k\right|\right)^L}{\left(2\sin\theta_k^{cheb}\right)^{M+\frac{1}{2}}}+\frac{1}{\left(2\sin\theta_k^{leg}\right)^{M+\frac{1}{2}}}\right). \tag{31}$$

4 Block partitioning of the Legendre-Vandermonde matrix

It can be found that the matrix $\mathbf{P}_{\mathrm{N}}\left(x_N^{leg}\right)$ can be considered as a perturbation of matrix $\mathbf{P}_{\mathrm{N}}\left(x_N^{cheb}\right)$ from Eq. (24). The block partitioning of $\mathbf{P}_{\mathrm{N}}\left(x_N^{leg}\right)$ can be performed by using the same method as $\mathbf{P}_{\mathrm{N}}\left(x_N^{cheb}\right)$ in the paper of Hale and Townsend [24]. So the matrix $\mathbf{P}_{\mathrm{N}}\left(x_N^{leg}\right)$ is partitioned as

$$\mathbf{P}_{\mathrm{N}}\left(x_N^{leg}\right) = \mathbf{P}_{\mathrm{N}}^{\mathrm{REC}}\left(x_N^{leg}\right)+\sum_{k=1}^{K}\mathbf{P}_{\mathrm{N}}^{(k)}\left(x_N^{leg}\right). \tag{32}$$

This partitioning separates the matrix $\mathbf{P}_{\mathrm{N}}\left(x_N^{leg}\right)$ into block $\mathbf{P}_{\mathrm{N}}^{\mathrm{REC}}\left(\underline{x}_N^{leg}\right)$ and K sub-matrices $\mathbf{P}_{\mathrm{N}}^{(k)}\left(x_N^{leg}\right)$. Block $\mathbf{P}_{\mathrm{N}}^{\mathrm{REC}}\left(x_N^{leg}\right)$ contains the columns and rows of $\mathbf{P}_{\mathrm{N}}\left(x_N^{leg}\right)$ which can't be computed by using Eq. (24).

$$\mathbf{P}_{\mathrm{N}}^{\mathrm{REC}}\left(x_N^{leg}\right)_{ij} = \begin{cases}\mathbf{P}_{\mathrm{N}}\left(x_N^{leg}\right)_{ij}, & 1 \leq \min\left(i, N-i+1\right) \leq j_M, \\ \mathbf{P}_{\mathrm{N}}\left(x_N^{leg}\right)_{ij}, & 1 \leq j \leq n_M, \\ 0, & otherwsie\end{cases}, \tag{33}$$

where

$$n_M = \left\lfloor \frac{1}{2}\left(\varepsilon\frac{\pi^{3/2}\Gamma\left(M+1\right)}{4\Gamma\left(M+1/2\right)}\right)^{\frac{-1}{M+\frac{1}{2}}}\right\rfloor, \tag{34}$$

and

$$j_M = \left\lfloor \frac{N+1}{\pi} \sin^{-1}\left(\frac{n_M}{N}\right) \right\rfloor, \tag{35}$$

$$\mathbf{P}_{\mathrm{N}}^{(k)}\left(x_N^{leg}\right)_{ij} = \begin{cases} \mathbf{P}_{\mathrm{N}}\left(x_N^{leg}\right)_{ij}, & i_k \le i \le N - j_k,\ \ \alpha^k N \le j \le \alpha^{k-1} N \\ 0, & otherwsie \end{cases}, \tag{36}$$

where $\alpha = O(1/\log N)$ and $i_k = \left\lfloor \frac{N+1}{\pi} \sin^{-1}\left(\frac{n_M}{\alpha^k N}\right) \right\rfloor$.

$$\mathbf{P}_{\mathrm{N}}\left(\underline{x}_N^{leg}\right)\underline{c}_N^{leg} = \mathbf{P}_{\mathrm{N}}^{\mathrm{REC}}\left(\underline{x}_N^{leg}\right)\underline{c}_N^{leg} + \sum_{k=1}^{K} \mathbf{P}_{\mathrm{N}}^{(k)}\left(\underline{x}_N^{leg}\right)\underline{c}_N^{leg}, \tag{37}$$

Nonzero entries of $\mathbf{P}_{\mathrm{N}}^{(k)}\left(x_N^{leg}\right)$ can be accurately expressed by the asymptotic formula which means that the butterfly compression to $\mathbf{P}_{\mathrm{N}}^{(k)}\left(x_N^{leg}\right)$ is stable and accurate. The matrix-vector product $\mathbf{P}_{\mathrm{N}}^{(k)}\left(\underline{x}_N^{leg}\right)\underline{c}_N^{leg}$ can be evaluated by the butterfly algorithm, so $\sum_{k=1}^{K} \mathbf{P}_{\mathrm{N}}^{(k)}\left(\underline{x}_N^{leg}\right)\underline{c}_N^{leg}$ can be computed in $O(KN\log N)$ operations. By restricting $\mathbf{P}_{\mathrm{N}}^{\mathrm{REC}}\left(\underline{x}_N^{leg}\right)$ has fewer than $O(KN\log N)$ nonzero entries, the matrix-vector product $\mathbf{P}_{\mathrm{N}}^{\mathrm{REC}}\left(\underline{x}_N^{leg}\right)\underline{c}_N^{leg}$ can be computed in $O(KN\log N)$ operations. Finally, the optimal computational cost is achieved. Let $n_m = \min(n_M, N-1)$, the parameters α and K are defined as

$$\alpha = \begin{cases} \min\left(1/\log(N/n_m), 1/2\right), & \text{for small } N \\ 1/\log(N/n_m), & \text{for large } N \end{cases}$$

and $K = O(\log N/\log\log N)$, respectively. In the practical application, only parameters N, n_m, α and K are used to obtain information such as starting row/column index and offset for all blocks.

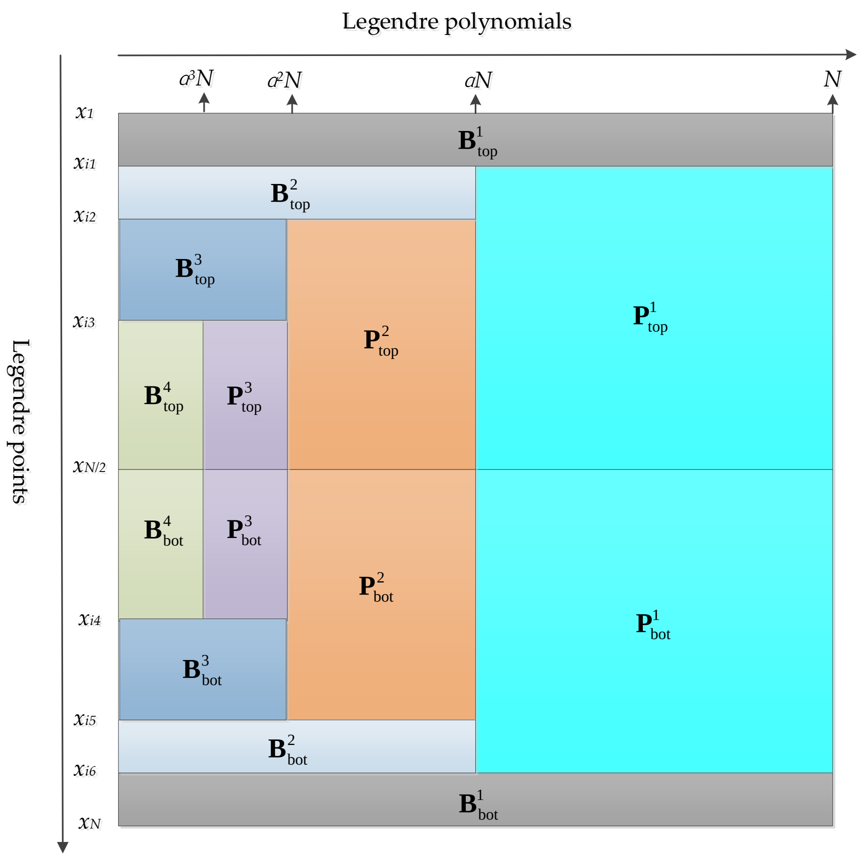


Figure 1. Partitioning of the Legendre-Vandermonde matrix for N=1024 (in which matrix **B** is the boundary parts can't be accurately expressed by the asymptotic formula while matrix **P** is the internal parts can. There are 2(K+1) sub-matrices of **B** and 2K sub-matrices of **P**. According to the symmetric or anti-symmetric property of Legendre polynomials, we only need to consider K+1 sub-matrices of **B** and K sub-matrices of **P** on top).

Figure 1 shows the partitioning of the Legendre-Vandermonde matrix for N=1024. The Legendre-Vandermonde matrix is divided into boundary (denoted by symbol **B**) and internal (denoted by symbol **P**) parts. The boundary parts include the elements which can't be accurately expressed by the asymptotic formula. There are 2(K+1) sub-matrices of **B** and 2K sub-matrices of **P**. According to the symmetric or anti-symmetric property of Legendre polynomials, only K+1 sub-matrices of **B** and **K** sub-matrices of **P** on the top are used. Table 1 presents a summary of Legendre transform algorithm using block butterfly algorithm. Direct computation part and butterfly multiplication part is cost $O(KN\log N)$ operations, respectively.

Table 1. Pseudocode for Legendre transform using block butterfly algorithm.

```
Block Butterfly Algorithm For Legendre Transform
Input $N$ and $\underline{c}_N^{leg}$ to compute $\underline{v}_N^{leg} = \mathbf{P}_\mathrm{N}\left(\underline{x}_N^{leg}\right)\underline{c}_N^{leg}$
Pre-computation Part
    Block Partitioning: $\mathbf{B}_\mathrm{top}^1, \mathbf{B}_\mathrm{top}^2, \ldots, \mathbf{B}_\mathrm{top}^{k+1}$ and $\mathbf{P}_\mathrm{top}^1, \mathbf{P}_\mathrm{top}^2, \ldots, \mathbf{P}_\mathrm{top}^{k}$
    extract symmetric part $\mathbf{B}_\mathrm{tops}^1, \mathbf{B}_\mathrm{tops}^2, \ldots, \mathbf{B}_\mathrm{tops}^{k+1}$, $\mathbf{P}_\mathrm{tops}^1, \mathbf{P}_\mathrm{tops}^2, \ldots, \mathbf{P}_\mathrm{tops}^{k}$ and anti-symmetric part
    $\mathbf{B}_\mathrm{topa}^1, \mathbf{B}_\mathrm{topa}^2, \ldots, \mathbf{B}_\mathrm{topa}^{k+1}$ $\mathbf{P}_\mathrm{topa}^1, \mathbf{P}_\mathrm{topa}^2, \ldots, \mathbf{P}_\mathrm{topa}^{k}$
    for i=1,2,…,k
        call butterfly_compression( $\mathbf{P}_\mathrm{tops}^i$ )   ! Symmetric Part
        call butterfly_compression( $\mathbf{P}_\mathrm{topa}^i$ )   ! Anti-Symmetric Part
    end for
Direct Computation Part:
    for i=1,2,…,k+1
        call dgemv( $\mathbf{B}_\mathrm{tops}^i$ )   ! Symmetric Part
        call dgemv( $\mathbf{B}_\mathrm{topa}^i$ )   ! Anti-Symmetric Part
    end for
Butterfly Multiplication Part:
    for i=1,2,…,k
        call butterfly_multiply()   ! Symmetric Part
        call butterfly_multiply()   ! Anti-Symmetric Part
    end for
Combine the results of symmetric and anti-symmetric part to get $\underline{v}_N^{leg}$
```

Parameters CMAX and EPS need for butterfly matrix compression are still needed in block butterfly algorithm. Cmax is the number of columns in each sub-matrix on level 0, EPS is desired precision in interpolative decomposition [13]. A dimensional thresh value DIMTHESH [13] is also needed in Legendre transform calls to activate FLT when wavenumber (m) less and equal to NSMAX-2DIMTHESH+3 (NSMAX is truncation order). Block butterfly algorithm is equivalent to Tygert's algorithm (2010) when no block partition is used, so two dimensional thresh values could be introduced to include Tygert's algorithm (2010) and LT using DGEMM for further reducing the computational complexity. To facilitate comparison with Tygert's algorithm, only one dimensional thresh value is used and set to 200 in the rest of the paper.

5. Results

In this section, all tests are performed on the MilkyWay-2 super computer (see Liao et al. [28] for more details) which installed in NUDT. Each compute node possesses 64GB of memory. The CPU model name is Intel(R)Xeon(R) CPU E5-2692V2 @2.2GHz. A private 32KB L1 instruction cache, a 32KB L1 data cache, a 256KB L2 cache, and a 30720KB L3 cache are used. ID software package developed by Martinsson et al. [29] for low rank approximation of matrices is employed to perform interpolative decompositions for all tests. ID package can be downloaded from Mark Tygert's homepage (http://tygert.com/software.html).

Hereafter, LT using matrix-matrix multiplication, Tygert's algorithm (2010) and block butterfly algorithm are named as LT0, LT1 and LT2, respectively. Furthermore, spherical harmonic transform (SHT) using LT0, LT1 and LT2 are noted as SHT0, SHT1 and SHT2, respectively.

Figures 2, 3 and 4 show the errors of LT with CMAX=64 in log10 form for EPS=1.0E-05, EPS=1.0E-07 and EPS=1.0E-10, respectively. It can be found that both maximum error and root-mean-square error of LT2 are improved by about one order magnitude than LT1.

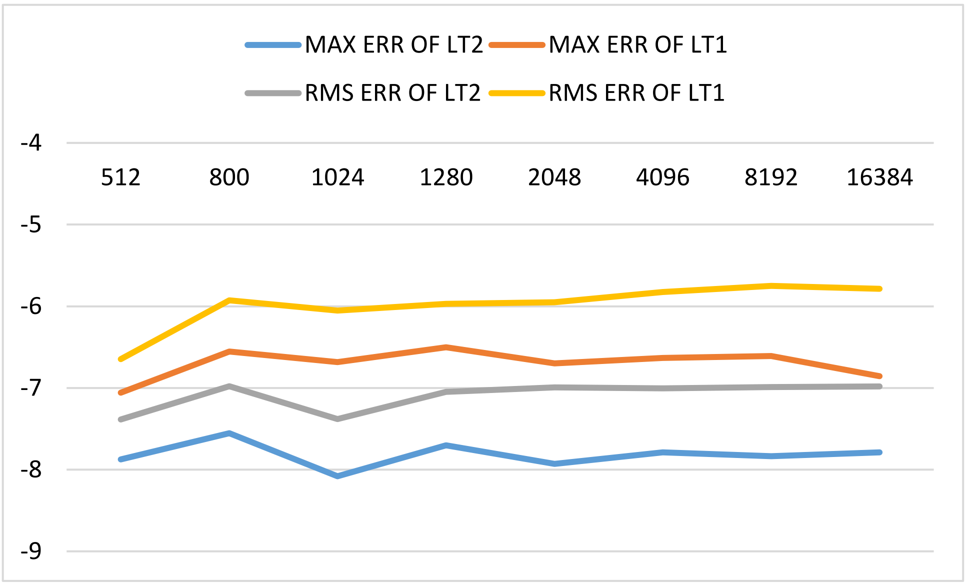


Figure 2. Errors of LT in log10 form with EPS=1.0E-05 and CMAX=64.

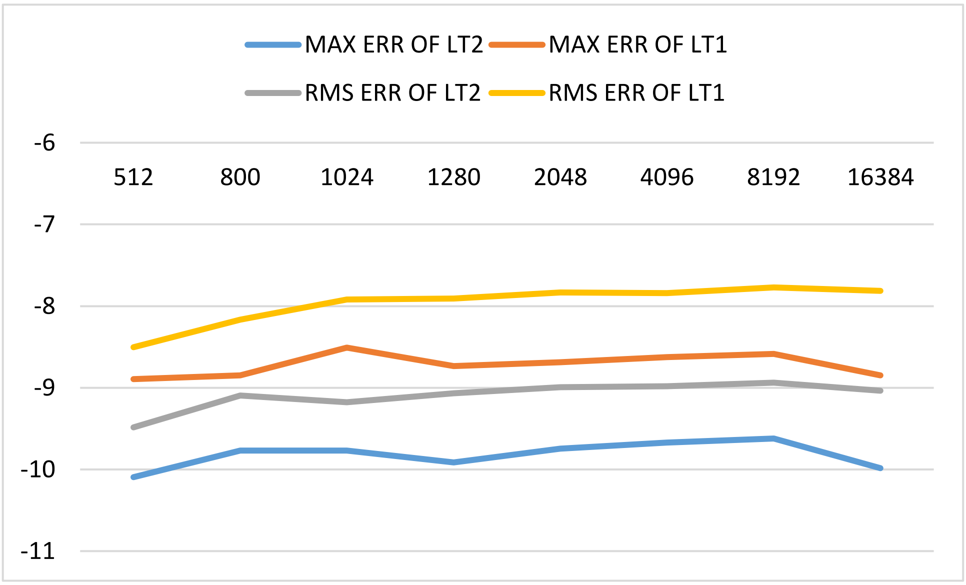
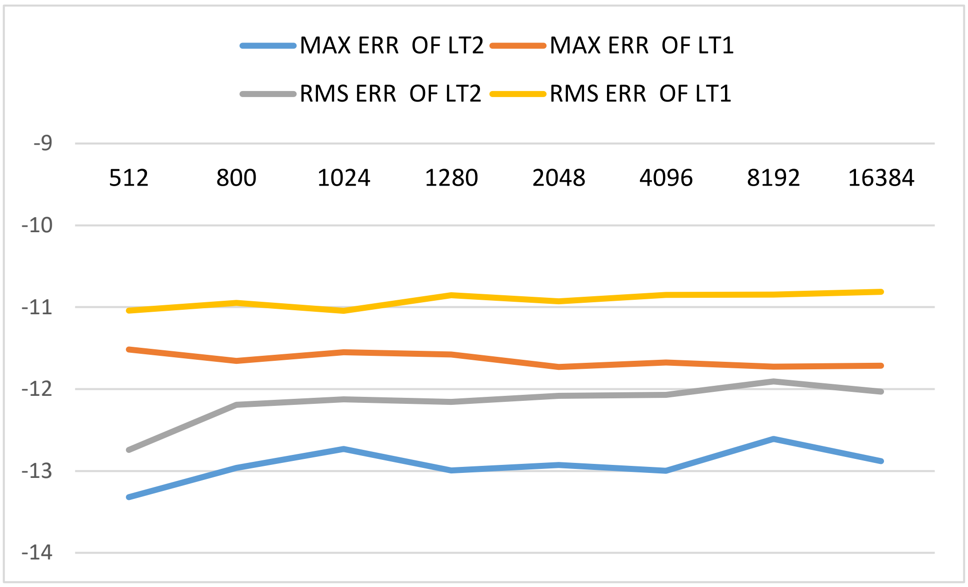
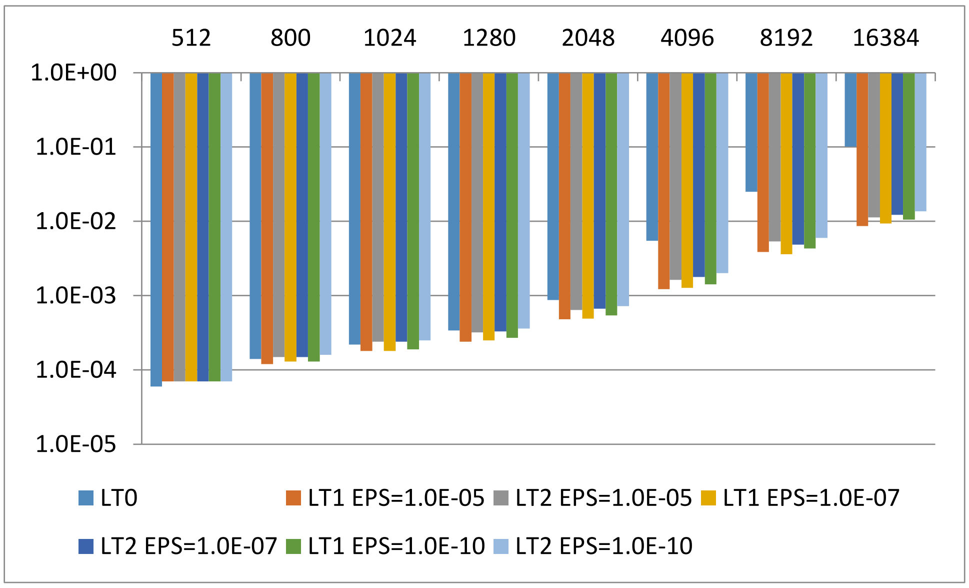


Figure 3. Errors of LT in log10 form with EPS=1.0E-07 and CMAX=64.



224

225 **Figure 4.** Errors of LT in log10 form with EPS=1.0E-10 and CMAX=64.



226

227 **Figure 5.** Computational time for different Legendre transform algorithms (LT0 is the algorithm using
228 DGEMM, LT1 is the butterfly algorithm and LT2 is the proposed method, Unit: second).

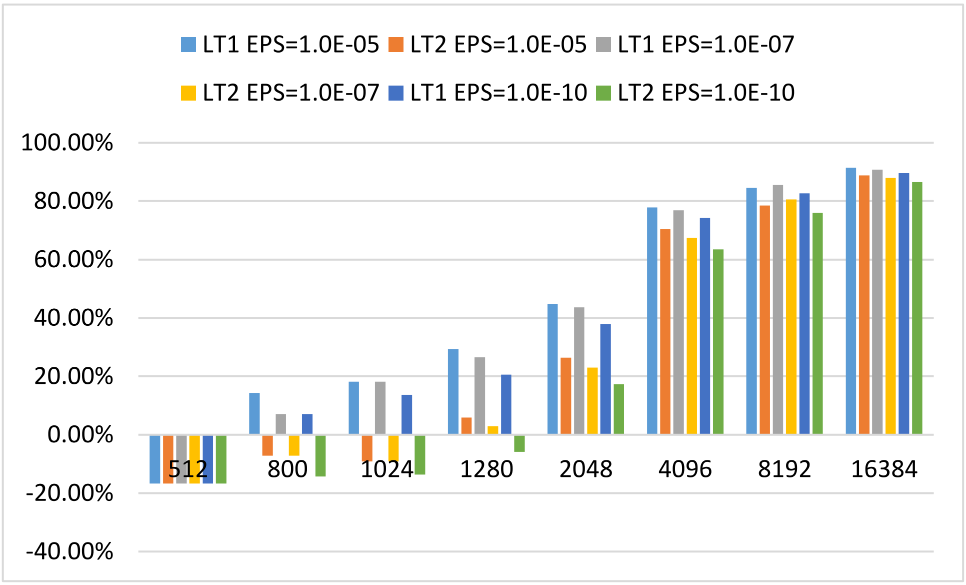


Figure 6. Speedup of LT1 and LT2 with CMAX=64 compare to LT0.

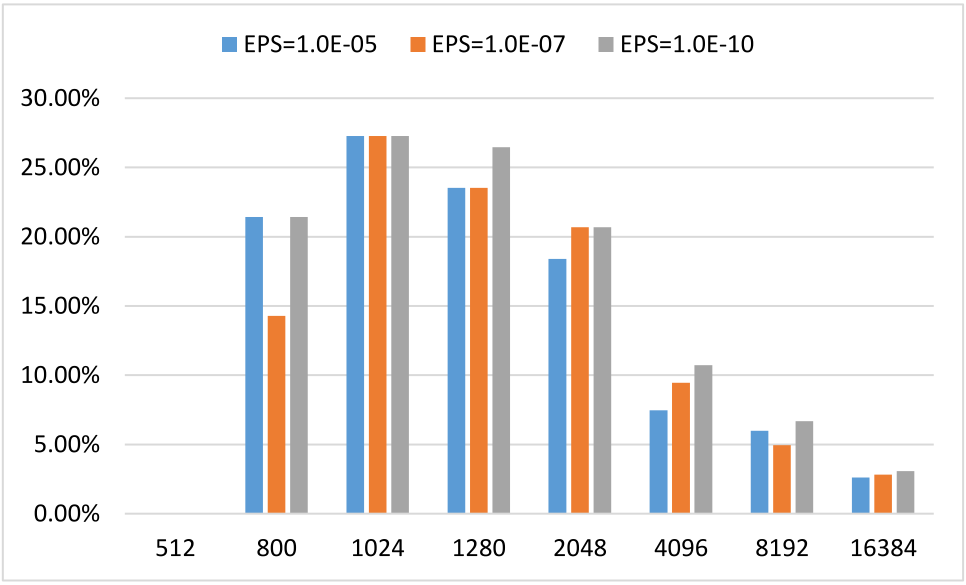


Figure 7. Loss speedup of LT2 with CMAX=64 compare to LT1.

Figure 5 shows the computational time for different LT algorithms. The speedup and loss speedup of LT2 with CMAX=64 are demonstrated in Figure 6 and 7, respectively. From Figs 6 and 7, LT2 begins to show faster than LT0 when N=2048 and achieves more than 26%, 22%, 17% reduction in elapsed time for EPS=1.0E-5, EPS=1.0E-7 and EPS=1.0E-10. LT2 has achieved more than 17%, 63%, 75% and 86% reduction in elapsed time for a run of *N*2048, *N*4096, *N*8192 and *N*16384, respectively. The loss of speedup is less than 21%, 11%, 7% and 4% for *N*=2048, 4096, 8192 and 16384, respectively. According to the results of Yin [13], the potential instability of interpolative decomposition only exists in the case of very high order. So the presented method can alleviate the potential instability of interpolative decomposition at a very small computational cost.

Figures 8 and 9 show the computational time of LT scaled by $N\log^3 N$ and $N\log^4 N$, respectively. The computational complexity of LT2 is less than $O(N\log^4 N)$ which appears to a little bigger. The boundary blocks which can't be accurately expressed by the asymptotic formula and the internal blocks with dimension less that dimensional thresh value result in the increase of the computational complexity.

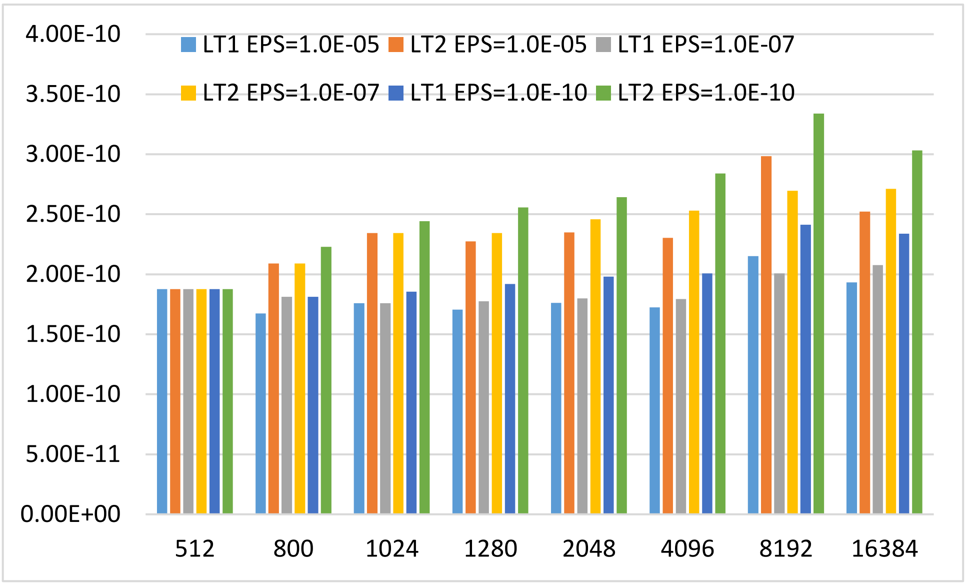


Figure 8. Computational time scaled by Nlog^{3}N with CMAX=64.

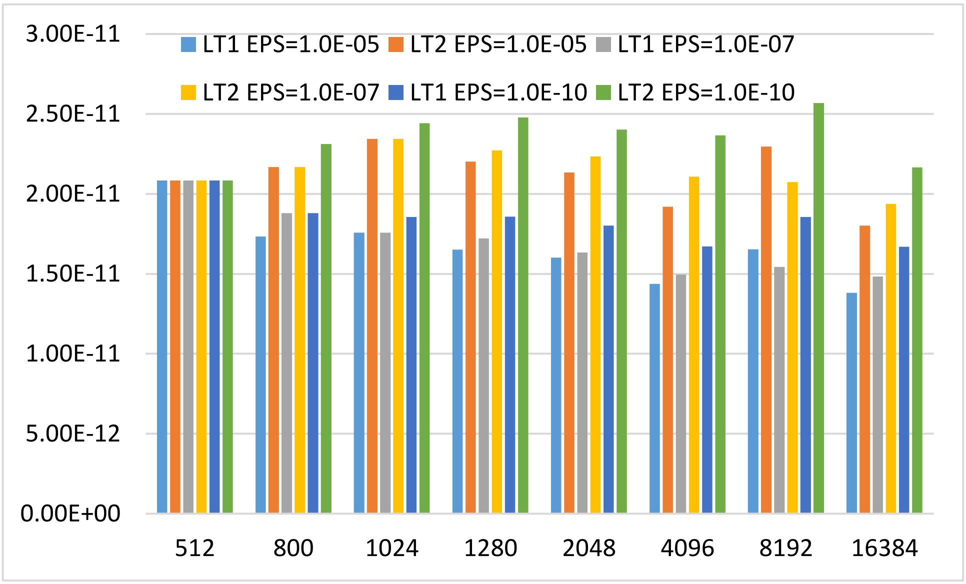


Figure 9. Computational time scaled by $N\log^4 N$ with CMAX=64.

Legendre-Vandermond matrix is divided into boundary blocks and internal blocks. Boundary blocks which can't be accurately expressed by the asymptotic formula cause instability of interpolative decompositions and are not suitable for interpolation decomposition. The matrix-vector multiplication based on butterfly algorithm is faster than BLAS function DGEMV only when the dimension of matrix is greater than or equal to 512. Internal blocks with lower matrix dimension adopt direct matrix-vector multiplication instead of butterfly algorithm. The number of nonzero elements of boundary blocks, internal blocks which do not participate in interpolation decomposition cause the increase of the computational cost compare to Tygert's algorithm. Therefore, through reasonable partitioning, the theoretical computational complexity of the proposed method can reach the optimal computational complexity $O(N\log^2(N)/\log\log N)$.

6. Conclusions

In this paper, a high accurate and stable Legendre transform algorithm is proposed. A block partitioning based on asymptotic formula is employed to mitigate the potential instability of

Legendre transform using butterfly algorithm. The Legendre-Vandermonde matrix is divided into one block $\mathbf{P}_{\mathrm{N}}^{\mathrm{REC}}\left(\underline{x}_N^{leg}\right)$ and K sub-matrices $\mathbf{P}_{\mathrm{N}}^{(k)}\left(x_N^{leg}\right)$. Instead of FFT method, butterfly algorithm is employed to compute $\mathbf{P}_{\mathrm{N}}^{(k)}\left(x_N^{leg}\right)\underline{c}_N^{leg}$. Numerical results demonstrate that the proposed method improves stability by about one order magnitude than Tygert's algorithm (2010) while only sacrifices less than 7% speedup for very high order ($N\geqslant$4096) Legendre transform.

Although the complexity of proposed method is a little greater than Tygert's algorithm (2010), the proposed method is equivalent to Tygert's algorithm (2010) when no block partition is used. In the application of NWP, an additional dimensional thresh value could be introduced to include Tygert's algorithm (2010) for further reducing the computational complexity.

In future, we will study more optimal block partition method to improve the computational performance while still keep stability and make a detail analysis about spectral harmonic transform using proposed method for very high resolution, especially its performance in the reduction of potential numerical instability for resolution T7999.

Author Contributions: Conceptualization, Fukang Yin and Jianping Wu; Formal analysis, Fukang Yin; Funding acquisition, Fukang Yin; Methodology, Fukang Yin and Junqiang Song; Supervision, Junqiang Song; Validation, Jianping Wu and Jinhui Yang; Writing – original draft, Fukang Yin; Writing – review & editing, Jinhui Yang.

Funding: This research was funded by the National Natural Science Foundation of China (Grant 41705078) and partly supported by the National Natural Science Foundation of China (Grants 61379022 and 41605070).

Acknowledgments: The author acknowledges Yingzhou Li (Duke University) and Haizhao Yang (National University of Singapore) for providing ButterflyLab for reference. The author would also like to thank three anonymous reviewers for their insightful and constructive comments, which help to improve the quality of this paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Suda, R. and M. Takami, A fast spherical harmonics transform algorithm, Math. Comput. 71(2002) 703-715.
2. Kunis, S. and D. Potts, Fast spherical Fourier algorithms, J. Comput. Appl. Math. 161(2003) 75-98.
3. Suda, R. Stability analysis of the fast Legendre transform algorithm based on the fast multipole method, Proceedings of the Estonian Academy of Sciences Physics Mathem 53(2004) 107-115.
4. Suda, R. Fast Spherical Harmonic Transform Routine FLTSS Applied to the Shallow Water Test Set, Mon. Weather Rev. 133(2005) 634-648.
5. Rokhlin, V. and M. Tygert, Fast Algorithms for Spherical Harmonic Expansions, SIAM J. Sci. Comput. 27(2005) 1903-1928.
6. Tygert, M. Recurrence relations and fast algorithms, Appl. Comput. Harmon. A. 28(2006) 121-128.
7. Tygert, M. Fast algorithms for spherical harmonic expansions, II, J. Comput. Phys. 227(2008) 4260-4279.
8. Tygert, M. Short Note: Fast algorithms for spherical harmonic expansions, III, J. Comput. Phys. 229(2010) 6181-6192.
9. Wedi N P, Hamrud M, Mozdzynski G. A Fast Spherical Harmonics Transform for Global NWP and Climate Models, Mon. Weather Rev. 141(2013) 3450-3461.
10. Wu, J. P., J. Zhao, J. Q. Song, and W. M. Zhang, Preliminary design of dynamic framework for global non-hydrostatic spectral model, Comput. Eng. Design. 32(2011) 3539-3543.
11. Yang, J., J. Song, J. Wu, K. J. Ren, and H. Leng, A high-order vertical discretization method for a semi-implicit mass-based non-hydrostatic kernel, Quart. J. Roy. Meteor. Soc., 141(2015) 2880-2885.
12. Yang, J., J. Song, J. Wu, F. Ying, J. Peng, and H. Leng, A semi-implicit deep-atmosphere spectral dynamical kernel using a hydrostatic-pressure coordinate, Quart. J. Roy. Meteor. Soc. 143(2017) 2703-2713.
13. Yin, F., G. Wu, J. Wu, J. Zhao and J. Song, Performance evaluation of the fast spherical harmonic transform algorithm in the yin–he global spectral model, Mon. Weather Rev. 146(2018) 3163-3182.
14. Seljebotn, D. S. Wavemoth -- Fast spherical harmonic transforms by butterfly matrix compression, Astrophysical Journal Supplement, 199(2012) 537-546.
15. Cheng, H., Z. Gimbutas, P. Martinsson, and V. Rokhlin, On the compression of low rank matrices, SIAM J. Sci. Comput. 26(2005) 1389-1404.

16. Zhu Heitman, James Bremer, and Vladimir Rokhlin, On the existence of nonoscillatory phase functions for second order ordinary differential equations in the high-frequency regime, J. Comput. Phys. 290(2015) 1-27.
17. James Bremer and Vladimir Rokhlin, Improved estimates for nonoscillatory phase functions, Discrete Cont. Dyn-A, 36(2016) 4101-4131.
18. James Bremer and Vladimir Rokhlin, On the nonoscillatory phase function for Legendre's differential equation, J. Comput. Phys. 350(2017) 326-342.
19. James Bremer and Haizhao Yang, 2019: Fast algorithms for Jacobi expansions via nonoscillatory phase functions, IMA J. Numer. Anal., arXiv:1803.03889 [math.NA], https://doi.org/10.1093/imanum/drz016.
20. Andreas Glaser, Xiangtao Liu, and Vladimir Rokhlin, A fast algorithm for the calculation of the roots of special functions, SIAM J. Sci. Comput. 29(2019) 1420-1438.
21. Bremer, James, Pang, Qiyuan, Yang, Haizhao, 2019: Fast Algorithms for the Multi-dimensional Jacobi Polynomial Transform, arXiv:1901.07275 [math.NA].
22. Hale, N. and Townsend, A. A fast, simple, and stable Chebyshev-Legendre transform using an asymptotic formula, SIAM J. Sci. Comput. 36(2014) 148-167.
23. Candès, Emmanuel, Demanet L, Ying L . A Fast Butterfly Algorithm for the Computation of Fourier Integral Operators. Multiscale Modeling & Simulation, 2009, 7(4):1727-1750.
24. Ying L and Haizhao Yang, interpolative butterfly factorization, SIAM J. Sci. Comput., 39(2), A503-A531.
25. Hale, N. and Townsend, A. A fast FFT-based discrete Legendre transform, IMA J. Numer. Anal. 36(2015) 1670-1684.
26. Townsend, A., Webby, M., and Olver, S., Fast polynomial transforms based on Toeplitz and Hankel matrices, Math. Comput. 87(2018) 1913-1934
27. StieltjesT. J. 1890: Sur les polynômes de Legendre, Ann. Fac. Sci. Toulouse, 4, G1-G17.
28. Liao, X., L. Xiao, C. Yang, and Y. Lu, MilkyWay-2 supercomputer: System and application, Front. Comput. Sci. 8(2014) 345-356.
29. Martinsson P.G., V. R., Y. Shkolnisky, M. Tygert 2008. ID: a software package for low rank approximation of matrices via interpolative decompositions, version 0.2. http://cims.nyu.edu/~tygert/id_doc.pdf.