Research Article **Open Access**

T. Fukushima*

Fast computation of sine/cosine series coefficients of associated Legendre function of arbitrary high degree and order

https://doi.org/10.1515/jogs-2018-0017
Received August 27, 2018; accepted December 7, 2018

Abstract: In order to accelerate the spherical/spheroidal harmonic synthesis of any function, we developed a new recursive method to compute the sine/cosine series coefficient of the 4π fully- and Schmidt quasi-normalized associated Legendre functions. The key of the method is a set of increasing-degree/order mixed-wavenumber two- to four-term recurrence formulas to compute the diagonal terms. They are used in preparing the seed values of the decreasing-order fixed-degree, and fixed-wavenumber two- and three-term recurrence formulas, which are obtained by modifying the classic relations. The new method is accurate and capable to deal with an arbitrary high degree/order/wavenumber. Also, it runs significantly faster than the previous method of ours utilizing the Wigner d function, say around 20 times more when the maximum degree exceeds 1,000.

Keywords: associated Legendre function, Fourier series expansion, recurrence formula, spherical harmonic expansion, spheroidal harmonic expansion

1 Introduction

The spherical and spheroidal harmonic synthesis and analysis are basic mathematical tools mostly used in geodesy and geophysics as well as planetary sciences (Heiskanen and Moritz, 1967; Stacey and Davis, 2008; de Pater and Lissauer, 2010). However, it is true that their computational labor is significantly large even after recent developments in their computational procedures (Fukushima, 2012a,b, 2013, 2014, 2016). This is especially true for large values of N, the maximum degree/order of the expansion, say 21,600 or more as intended by recent studies (Rexer and Hirt, 2015a,b).

A good approach to accelerate the actual procedure of the synthesis/analysis is the utilization of the two-dimensional FFT on the unit sphere after the translation of the harmonic expansions into the Fourier transform (Colombo, 1981). Refer to Fig. 1 of Fukushima (2018) illustrating the superiority of the FFT method in the preparation of the lumped coefficients in the spherical/spheroidal harmonic synthesis. Nevertheless, for this purpose, we need a specific procedure to transform $\overline{C}_{nm}$ and $\overline{S}_{nm}$, the harmonic expansion coefficients with the 4π full normalization, into A_{km} and B_{km}, the corresponding Fourier series expansion coefficients such that (Sneeuw and Bun, 1996)

$$A_{km} = \sum_{n=\max(k,m),n-k:\text{even}}^{N} p_{nmk}\overline{C}_{nm}, \tag{1}$$

$$B_{km} = \sum_{n=\max(k,m),n-k:\text{even}}^{N} p_{nmk}\overline{S}_{nm}. \tag{2}$$

Here p_{nmk} is the sine/cosine series coefficients of $\overline{P}_{nm}(t)$, the 4π fully normalized associated Legendre function (fnALF) (Heiskanen and Moritz, 1967), such that

$$\overline{P}_{nm}(\cos\theta) = \sum_{k=0,n-k:\text{even}}^{n} p_{nmk}h_m(k\theta). \tag{3}$$

where $h_m(\psi)$ is a trigonometric function conditionally defined as

$$h_m(\psi) \equiv \begin{cases} \cos\psi, & (m:\text{even}), \\ \sin\psi, & (m:\text{odd}). \end{cases} \tag{4}$$

The computation of p_{nmk} is a classic problem (Egersdorfer and Egersdorfer, 1936). Recently, we developed an accurate method to compute p_{nmk} for arbitrary large indices, n, m, and k, say with 15 effective digits for indices up to $2^{30} \approx 10^9$ (Fukushima, 2018, §2.1). Nevertheless, we must admit that its execution speed is not so fast. This becomes prominent if compared with that of q_{nmk} (Fukushima, 2018, §2.2), the similar coefficients of the inverse transformation from A_{km} and B_{km} to $\overline{C}_{nm}$ and $\overline{S}_{nm}$, respectively.

***Corresponding Author: T. Fukushima:** National Astronomical Observatory of Japan, Graduate University of Advanced Study / SOKENDAI, 2-21-1, Ohsawa, Mitaka, Tokyo 181-8588, Japan , E-mail: Toshio.Fukushima@nao.ac.jp

Therefore, we re-examined the computation method of p_{nmk} proposed by Hofsommer and Potters (1960): a fixed-order recursive computation of p_{nmk}. Since the recurrence formula is not suitable for the forward recursion, we use it in the backward manner. For this purpose, however, we must prepare the diagonal terms, p_{nnk}, as the seed values. Gruber and Abrykosov (2016) recommended an approach to solve for them by comparison with the analytical solutions of p_{n0k} after the execution of a generic form of the recursion by regarding p_{nnk} as unknowns to be solved. Nonetheless, this formulation is difficult to be extended to arbitrary large values of n, m, and/or k.

In order to overcome this situation, we obtained a group of forward recurrence formulae to compute p_{nnk}. Combining it with the main recurrence relation of Hofsommer and Potters (1960) regarded as a backward formula after an appropriate renormalization, we developed a new recursive method to compute p_{nmk}. As will be shown in Fig. 1, the new method is as precise as the previous method of ours (Fukushima, 2018, §2.1). Also, Fig. 2 given later illustrates that the new method runs around 20 times faster than the old method if N is sufficiently large, say greater than 1,024.

Below, we explain the new method in Section 2, and present its numerical experiments in Section 3. Also, we derive the new recurrence formulae in Appendix A, extend the formulation to the case of Schmidt quasi-normalization (Winch et al., 2005) in Appendix B, describe the algorithms to implement the new method in Appendix C, and display the Fortran programs to execute the algorithm in Appendix D.

Table 1: Sample values of sine/cosine series coefficients of fnALF. Listed are the literal expression and 20 digits of all non-zero values of p_{nmk} when $0 \le m \le n \le 4$ and $0 \le k \le n$. Notice that $p_{nmk} = 0$ when (i) $n - k$ is odd, (ii) n is even, m is odd, and $k = 0$, (ii) $n > m$, and/or (iii) $n > k$.

n	m	k	p_{nmk}	
0	0	0	1	+1.0
1	0	1	$\sqrt{3}$	+1.7320508075688772935
1	1	1	$\sqrt{3}$	+1.7320508075688772935
2	0	0	$\sqrt{5}/4$	+0.5590169943749474241
2	0	2	$3\sqrt{5}/4$	+1.6770509831248422723
2	1	2	$\sqrt{15}/2$	+1.9364916731037084426
2	2	0	$\sqrt{15}/4$	+0.9682458365518542213
2	2	2	$-\sqrt{15}/4$	−0.9682458365518542213
3	0	1	$3\sqrt{7}/8$	+0.9921567416492214714
3	0	3	$5\sqrt{7}/8$	+1.6535945694153691191
3	1	1	$\sqrt{42}/16$	+0.4050462936504912644
3	1	3	$5\sqrt{42}/16$	+2.0252314682524563222
3	2	1	$\sqrt{105}/8$	+1.2808688457449497979
3	2	3	$-\sqrt{105}/8$	−1.2808688457449497979
3	3	1	$3\sqrt{70}/16$	+1.5687375497513916525
3	3	3	$-\sqrt{70}/16$	−0.5229125165837972175
4	0	0	27/64	+0.421875
4	0	2	15/16	+0.9375
4	0	4	105/64	+1.640625
4	1	2	$3\sqrt{10}/16$	+0.5929270612815711247
4	1	4	$21\sqrt{10}/32$	+2.0752447144854989366
4	2	0	$9\sqrt{5}/32$	+0.6288941186718158521
4	2	2	$3\sqrt{5}/8$	+0.8385254915624211362
4	2	4	$-21\sqrt{5}/32$	−1.4674196102342369883
4	3	2	$3\sqrt{70}/16$	+1.5687375497513916525
4	3	4	$-3\sqrt{70}/32$	−0.7843687748756958262
4	4	0	$9\sqrt{35}/64$	+0.8319487194983835060
4	4	2	$-3\sqrt{35}/16$	−1.1092649593311780080
4	4	4	$3\sqrt{35}/64$	+0.2773162398327945020

2 Method

Let us consider the numerical computation of p_{nmk}. Hinted by the recursive formulation to evaluate q_{nmk} (Fukushima, 2018, §2.2) and noting the zero value formulae of p_{nmk} expressed as

$$p_{nmk} = 0, \quad (n - k : \text{even}), \tag{5}$$

$$p_{nm0} = 0, \quad (n : \text{even};\ m : \text{odd}), \tag{6}$$

$$p_{nmk} = 0, \quad (n < m), \tag{7}$$

$$p_{nmk} = 0, \quad (n < k), \tag{8}$$

we developed a new method to compute p_{nmk} recursively, the detailed derivation of which is provided in Appendix A. Let us show its compact summary below.

First, the main recurrence formula is a decreasing-order, fixed-degree, and fixed-wavenumber three-term formula expressed as

$$p_{nmk} = k(-1)^m \alpha_{nm} p_{n,m+1,k} + \beta_{nm} p_{n,m+2,k},$$

$$(m = n - 2, n - 3, \ldots, 0), \tag{9}$$

where α_{nm} and β_{nm} are numerical coefficients defined as

$$\alpha_{nm} \equiv \sqrt{\frac{2\,(2 - \delta_{m0})}{(n - m)(n + m + 1)}}, \tag{10}$$

$$\beta_{nm} \equiv \sqrt{\frac{(2 - \delta_{m0})\,(n - m - 1)(n + m + 2)}{2(n - m)(n + m + 1)}}. \tag{11}$$

This recurrence relation is meaningful when $n - k$ is even and $0 \le k \le n$. Since $\beta_{n,n-1} = 0$, the semi-diagonal case when $m = n - 1$ is simplified as

$$p_{n,n-1,k} = k(-1)^{n-1}\alpha_{n,n-1}p_{nnk},$$

$$(n = 1, 2, \ldots;\ 0 \le k \le n;\ n - k : \text{even}). \tag{12}$$

Next, the backward recurrence formula, Eq. (9), and its special case, Eq. (12), require p_{nnk} as their seed values. They are computed by a variety of increasing-degree/order mixed-wavenumber recurrence formulae expressed as

$$p_{nn0} = \gamma_n \left(2p_{n-2,n-2,0} - p_{n-2,n-2,2}\right),$$

$$(n = 6, 8, 10, \ldots), \tag{13}$$

$$p_{nn1} = \gamma_n \left(3p_{n-2,n-2,1} - p_{n-2,n-2,3}\right),$$

$$(n = 5, 7, 9, \ldots), \tag{14}$$

$$p_{nn2} = \gamma_n \left(-2p_{n-2,n-2,0} + 2p_{n-2,n-2,2} - p_{n-2,n-2,4}\right),$$

$$(n = 6, 8, 10, \ldots), \tag{15}$$

$$p_{nnk} = \gamma_n \left(-p_{n-2,n-2,k-2} + 2p_{n-2,n-2,k} - p_{n-2,n-2,k+2}\right),$$

$$(n = 7, 8, 9, \ldots;\ 3 \le k \le n - 4;\ n - k : \text{even}), \tag{16}$$

$$p_{nn,n-2} = \gamma_n \left(-p_{n-2,n-2,n-4} + 2p_{n-2,n-2,n-2}\right),$$

$$(n = 5, 6, 7, \ldots), \tag{17}$$

$$p_{nnn} = -\gamma_n p_{n-2,n-2,n-2}, \quad (n = 5, 6, 7, \ldots), \tag{18}$$

where γ_n is a numerical coefficient defined as

$$\gamma_n \equiv \frac{1}{8}\sqrt{\frac{(2n-1)(2n+1)}{n(n-1)}}, \tag{19}$$

while $n - k$ should be even once again. Finally, the above scheme to prepare p_{nnk} demands a group of their initial values, namely p_{nnk} for (i) $0 \le n \le 4$, (ii) $0 \le k \le n$, and (iii) $n - k$ is even. They are explicitly provided in Table 1. Thus, the formulation is completed.

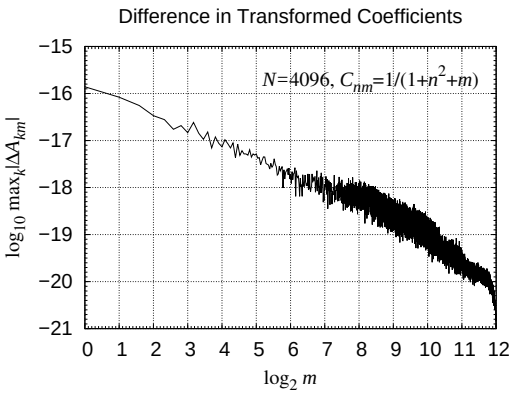

Figure 1: Difference in transformed coefficients between old and new methods. Shown are the absolute differences in A_{km} between the old method (Fukushima, 2018, §2.1) and the new method described in Sect. 2. The input data are the model spherical harmonic coefficients, $C_{nm} \equiv 1/\left(1 + n^2 + m\right)$, where $0 \le m \le n \le N = 4096$. Plotted is the maximum absolute difference of A_{km} for $0 \le k \le N$ as a function of m in a double logarithmic manner.

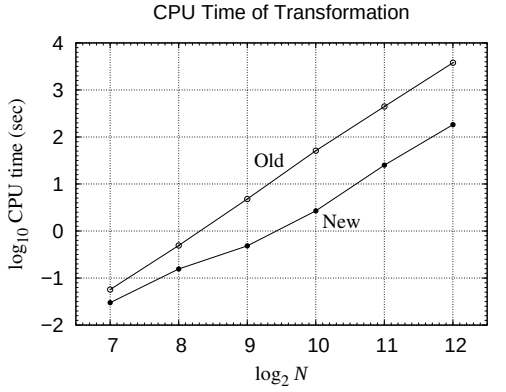

Figure 2: Maximum degree dependence of CPU time to conduct transformation from $\left(\overline{C}_{nm}, \overline{S}_{nm}\right)$ to (A_{km}, B_{km}). Displayed are the CPU times to execute the transformation from the given surface spherical harmonic coefficients, $\left(\overline{C}_{nm}, \overline{S}_{nm}\right)$, to the corresponding Fourier series coefficients on the sphere, (A_{km}, B_{km}). The CPU times are measured at a consumer PC with an Intel Core i7-4600U CPU running at 2.10 GHz clock by allowing its 2 cores and 4 threads fully employed. Compared are the computation using the old scheme (Fukushima, 2018, §2.1) and that by the new method. Both of them are roughly in proportional to N^3, where N is the maximum degree/order of the spherical harmonic coefficients to be transformed. However, the proportional coefficients are significantly different, say a factor of 20 or more when $N \ge 2^{10} = 1,024$, such that the new method is significantly faster.

3 Numerical experiments

Let us examine the computational accuracy of the new method. Refer to Fig. 1 illustrating a comparison of A_{km} obtained by the old method (Fukushima, 2018, §2.1) and the new method when $\overline{C}_{nm}$ is modelled as $\overline{C}_{nm} = 1/\left(1+n^2+m\right)$. Notice that this is a toy model without physical meaning although it has a simple analytic form and mimics Kaula's rule to some extent. At any rate, we omit the result for B_{km} since the situation is the same. Obviously, the observed difference between the two methods are negligibly small. This fact confirms the new method is of a sufficient accuracy.

Next, we compare the computational speed of the new method with the old one. Refer to Fig. 2 illustrating a comparison of the CPU time of the coefficient transformation between the old method (Fukushima, 2018, §2.1) and the new method. Obviously, Fig. 2 indicates that the new method runs around 20 times faster than the previous method of ours. What makes such a significant difference?

The answer is the difference in the complexity of the innermost loop. Indeed, the main recurrence formula of the previous method is the increasing-degree fixed-order/wavenumber three term recursion of E_{nkm} (Fukushima, 2017, Eq. (28)) written as

$$E_{nkm} = -a_{nkm}E_{n-1,km} - b_{nkm}E_{n-2,km}, \tag{20}$$

where a_{nkm} and b_{nkm} are numerical coefficient defined as

$$a_{nkm} \equiv \frac{km(2n-1)}{(n-1)\sqrt{\left(n^2-k^2\right)\left(n^2-m^2\right)}}, \tag{21}$$

$$b_{nkm} \equiv \frac{n}{n-1}\sqrt{\frac{\left[(n-1)^2-k^2\right]\left[(n-1)^2-m^2\right]}{\left(n^2-k^2\right)\left(n^2-m^2\right)}}. \tag{22}$$

Compare these expressions of a_{nkm} and b_{nkm} with those of α_{nm} and β_{nm} given in Eqs (10) and (11). In the former case, k is included inside the square roots in a complicated manner. Meanwhile, in the latter case, the coefficients α_{nm} and β_{nm} are independent with k. Namely, the wavenumber k appears as a simple multiplicative factor in the main recurrence relation, Eq (9). As a result, by setting the innermost loop as that with k, we can minimize the computational amount in the innermost loop. This results in a significant difference in the total computational labor as clearly indicated in Fig. 2.

4 Conclusion

In order to improve the slowness of the previous method of ours (Fukushima, 2018, §2.1) to compute p_{nmk}, the sine/cosine series coefficient of the 4π fully normalized associated Legendre function (ALF), we developed a new method to do the same computation. The key component of the new method is a set of newly developed recurrence formulae to obtain the diagonal coefficients, p_{nnk}. They are increasing-order, fixed-degree, and mixed-wavenumber two- to four-term formulae. Their expressions are significantly different with each other depending on the wavenumber. These results are extended to $\tilde{p}_{nmk}$, the coefficients for the Schmidt quasi-normalization of ALF. Numerical experiments revealed that the new method to compute p_{nmk} runs more than 20 times faster than the previous method (Fukushima, 2018, §2.1) and its computational speed is comparable with that of q_{nmk} (Fukushima, 2018, §2.2), which does accelerate the harmonic analysis in general. Thus, the new method will be useful in accelerating the harmonic synthesis of any function on the unit sphere.

Acknowledgement: The author thanks the anonymous referees for their valuable advices to improve the quality of the present article.

A Derivation of recurrence formulae

Let us derive the recurrence formulae presented in Section 2. For this purpose, we introduce the sine/cosine series expression of $P_{nm}(t)$, the unnormalized ALF, as

$$P_{nm}(\cos\theta) = \sum_{k=0,\ n-k:\text{even}}^{n} t_{nmk}h_m(k\theta), \tag{23}$$

where t_{nmk} is the unnormalized version of p_{nmk}. Below, we shall first obtain the recurrence formulae for t_{nmk} and next translate them into those of p_{nmk}. All the derived formulae have been literally validated by using Mathematica command sequences (Wolfram, 2003).

A.1 General and semi-diagonal terms

Once Bosch (2000, Eqs (8) and (11)) presented a pair of non-singular recurrence relations of the derivative of ALF with respect to the colatitude as

$$2\left(\frac{dP_{nm}(\cos\theta)}{d\theta}\right) = (n+m)(n-m+1)P_{n,m-1}(\cos\theta)$$

$$-P_{n,m+1}(\cos\theta), \quad (1 \le m \le n-1) \tag{24}$$

$$\frac{dP_{nn}(\cos\theta)}{d\theta} = nP_{n,n-1}(\cos\theta), \quad (n = 1, 2, \dots). \tag{25}$$

These relations are not so popular. In fact, they are missing in the standard reference books on special functions (Abramowitz and Stegun, 1964; Olver et al., 2010; Zwillinger and Moll, 2014). Also, they are not used in the standard recurrence formulations to compute the derivatives of the fnALF (Fukushima, 2012b).

In any case, let us obtain a recurrence relation of t_{nmk} by means of these recurrence formulae of $P_{nm}(\cos\theta)$. Noting the differential formula of $h_m(\psi)$ (Fukushima, 2017) rewritten as

$$\frac{dh_m(\psi)}{d\psi} = (-1)^{m-1}h_{m-1}(\psi), \tag{26}$$

we conduct the differentiation of $P_{nm}(\cos\theta)$ in terms of them as

$$\frac{dP_{nm}(\cos\theta)}{d\theta}$$

$$= (-1)^{m-1} \sum_{k=1,\ n-k:\text{even}}^{n} k t_{nmk} h_{m-1}(k\theta). \tag{27}$$

Substitute this expression of derivatives and the original expression of the point values, Eq. (23), into the above recurrence relations, Eqs (24) and (25), while noting the periodicity relation of $h_m(\psi)$ expressed as $h_{m+1}(\psi) = h_{m-1}(\psi)$. Thanks to the complete orthogonality of $h_m(k\theta)$, by comparing the both sides of the rewritten relations term by term, we obtain a pair of fixed-degree fixed-wavenumber relations of t_{nmk} as

$$2k(-1)^{m-1}t_{nmk} = (n+m)(n-m+1)t_{n,m-1,k}$$

$$-t_{n,m+1,k}, \quad (1 \le m \le n-1) \tag{28}$$

$$k(-1)^{n-1}t_{nnk} = nt_{n,n-1,k}, \quad (n = 1, 2, \dots). \tag{29}$$

Apart from the normalization constant, Eq. (28) is the same as Eq. (3.3) of Hofsommer and Potters (1960) although the derivation is significantly different. Through preliminary numerical experiments, we confirmed that the main relation, Eq. (28), is stable when used as a backward recurrence formula with respect to m. Therefore, we rewrite these relations into a decreasing-order manner as

$$t_{n,n-1,k} = \frac{k(-1)^{n-1}}{n}\, t_{nnk}, \quad (n = 1, 2, \dots). \tag{30}$$

$$t_{nmk} = \frac{2k(-1)^m t_{n,m+1,k} + t_{n,m+2,k}}{(n-m)(n+m+1)},$$

$$(m = n-2, n-3, \dots, 0). \tag{31}$$

A.2 Diagonal term

Before going further, let us write the diagonal term in a simpler form as

$$t_{nk} \equiv t_{nnk}. \tag{32}$$

This becomes the seed value for the decreasing-order recurrence formulae, Eqs (30) and (31). In order to find a formulation to prepare t_{nk}, recall that the sectorial component of the ALF is explicitly expressed (Heiskanen and Moritz, 1967, Eq. (1-57)) as

$$P_{nn}(\cos\theta) = (2n-1)!!\ \sin^n\theta, \tag{33}$$

if noting $(-1)!! = 1$. When $n \ge 2$, this expression is translated into a multiplicative recurrence relation in a leapfrog manner as

$$P_{nn}(\cos\theta) = 2r_n\,(1-\cos 2\theta)\,P_{n-2,n-2}(\cos\theta),$$

$$(n = 2, 3, \dots), \tag{34}$$

where r_n is an auxiliary coefficient defined as

$$r_n \equiv (2n-1)(2n-3)/4, \tag{35}$$

and we used a formula of the trigonometric functions, $\sin^2\theta = (1-\cos 2\theta)/2$. Hereafter, we shall split the discussion depending on the value and the parity of n.

Let us begin with the case when n is even. Substituting the sine/cosine series expression of the ALF, Eq. (23), into the rewritten formula, Eq. (34), we obtained a series equation expressed as

$$\sum_{k=0,\ n-k:\text{even}}^{n} t_{nk}\cos k\theta = 2r_n\,(1-\cos 2\theta)\times$$

$$\sum_{j=0,\ n-j:\text{even}}^{n-2} t_{n-2,j}\cos j\theta, \quad (n = 2, 4, \dots). \tag{36}$$

By utilizing the product-to-sum identity of the cosine functions,

$$2\cos 2\theta\cos j\theta = \cos(j+2)\theta + \cos(j-2)\theta, \tag{37}$$

and comparing the coefficients of the cosine functions of the same wavenumber in the both sides, we decompose Eq. (36) as

$$t_{nk} = \begin{cases} r_n\,(2t_{n-2,0} - t_{n-2,2}), \\ \quad (k = 0;\ n = 4, 6, \dots), \\ r_n\,(-2t_{n-2,0} + 2t_{n-2,2} - t_{n-2,4}), \\ \quad (k = 2;\ n = 6, 8, \dots), \\ r_n\,\left(-t_{n-2,k-2} + 2t_{n-2,k} - t_{n-2,k+2}\right), \\ \quad (k = 4, 6, \dots, n-4;\ n = 8, 10, \dots), \\ r_n\,(-t_{n-2,n-4} + 2t_{n-2,n-2}), \\ \quad (k = n-2;\ n = 6, 8, \dots), \\ r_n\,(-t_{n-2,n-2}), \\ \quad (k = n;\ n = 4, 6, \dots). \end{cases} \tag{38}$$

Similarly, when n is odd, we obtained another series equation expressed as

$$\sum_{k=1,\ n-k:\text{even}}^{n} t_{nk} \sin k\theta = 2r_n (1 - \cos 2\theta) \times \sum_{j=1,\ n-j:\text{even}}^{n-2} t_{n-2,j} \sin j\theta, \quad (n = 3, 5, \dots). \tag{39}$$

This time, by utilizing the product-to-sum identity of the sine functions,

$$2 \cos 2\theta \sin j\theta = \sin(j+2)\theta + \sin(j-2)\theta, \tag{40}$$

and comparing the coefficients of the sine functions of the same wavenumber in the both sides, we resolve Eq. (39) as

$$t_{nk} = \begin{cases} r_n (3t_{n-2,1} - t_{n-2,3}), \\ \qquad (k = 1;\ n = 5, 7, \dots), \\ r_n \left(-t_{n-2,k-2} + 2t_{n-2,k} - t_{n-2,k+2}\right), \\ \qquad (k = 3, 5, \dots, n-4;\ n = 7, 9, \dots), \\ r_n (-t_{n-2,n-4} + 2t_{n-2,n-2}), \\ \qquad (k = n-2;\ n = 5, 7, \dots), \\ r_n (-t_{n-2,n-2}), \\ \qquad (k = n;\ n = 3, 5, \dots). \end{cases} \tag{41}$$

Notice a delicate difference for the cases $k = 0$, 1, and 2 between Eqs (38) and (41).

A.3 4π full normalization

In the previous subsections, we obtained the recurrence relations of t_{nmk}. Let us transform them to those of p_{nmk} by a normalization. The ratio of p_{nmk} and t_{nmk} is nothing but the 4π full normalization factor as

$$\frac{p_{nmk}}{t_{nmk}} = \sqrt{(2 - \delta_{m0})(2n+1)\frac{(n-m)!}{(n+m)!}}. \tag{42}$$

Since it is independent on the wavenumber k, the translation is automatic. In fact, we rewrite Eqs (30), (31), (38), and (41) as

$$p_{nmk} = k(-1)^m \sqrt{\frac{2(2 - \delta_{m0})}{(n-m)(n+m+1)}}\, p_{n,m+1,k} + \sqrt{\frac{(2 - \delta_{m0})(n-m-1)(n+m+2)}{2(n-m)(n+m+1)}}\, p_{n,m+2,k}, \tag{43}$$

$$p_{n,n-1,k} = k(-1)^{n-1} \sqrt{\frac{2 - \delta_{n1}}{n}}\, p_{nnk}, \tag{44}$$

$$p_{nn0} = \frac{1}{8}\sqrt{\frac{(2n+1)(2n-1)}{n(n-1)}} (2p_{n-2,n-2,0} - p_{n-2,n-2,2}), \tag{45}$$

$$p_{nn1} = \frac{1}{8}\sqrt{\frac{(2n+1)(2n-1)}{n(n-1)}} (3p_{n-2,n-2,1} - p_{n-2,n-2,3}), \tag{46}$$

$$p_{nn2} = \frac{1}{8}\sqrt{\frac{(2n+1)(2n-1)}{n(n-1)}} (-2p_{n-2,n-2,0} + 2p_{n-2,n-2,2} - p_{n-2,n-2,4}), \tag{47}$$

$$p_{nnk} = \frac{1}{8}\sqrt{\frac{(2n+1)(2n-1)}{n(n-1)}} \left(-p_{n-2,n-2,k-2} + 2p_{n-2,n-2,k} - p_{n-2,n-2,k+2}\right), \tag{48}$$

$$p_{nn,n-2} = \frac{1}{8}\sqrt{\frac{(2n+1)(2n-1)}{n(n-1)}} (-p_{n-2,n-2,n-4} + 2p_{n-2,n-2,n-2}), \tag{49}$$

$$p_{nnn} = \frac{-1}{8}\sqrt{\frac{(2n+1)(2n-1)}{n(n-1)}} p_{n-2,n-2,n-2}. \tag{50}$$

These are the same as Eqs (9), (12), and (13)–(18), respectively, if noting (i) the definitions of recursion coefficients, Eqs (10), (11), and (19), and (ii) the zero value formulae, Eqs (5)–(8). Thus, the derivation is completed.

B Case of Schmidt quasi-normalization

In geomagnetism, another kind of the normalization is popular: Schmidt quasi-normalization (Winch et al., 2005). Let us denote by $\tilde{p}_{nmk}$ for the variation of p_{nmk} in that normalization. Its definition is related to t_{nmk} by following the definition of the Schmidt quasi-normalization (Winch et al., 2005, Eq. (3.2)) as

$$\frac{\tilde{p}_{nmk}}{t_{nmk}} = \sqrt{(2 - \delta_{m0})\frac{(n-m)!}{(n+m)!}}. \tag{51}$$

Then, the ratio of p_{nmk} and $\tilde{p}_{nmk}$ is written as

$$p_{nmk} = \sqrt{2n+1}\,\tilde{p}_{nmk}. \tag{52}$$

Table 2: Sample values of sine/cosine series coefficients of Schmidt quasi-normalized ALF. Same as Table 1 but for $\tilde{p}_{nmk}$, the corresponding coefficients of Schmidt quasi-normalized ALF.

n	m	k	$\tilde{p}_{nmk}$	
0	0	0	1	+1.0
1	0	1	1	+1.0
1	1	1	1	+1.0
2	0	0	1/4	+0.25
2	0	2	3/4	+0.75
2	1	2	$\sqrt{3}/2$	+0.866025403784438646764
2	2	0	$\sqrt{3}/4$	+0.433012701892219323382
2	2	2	$-\sqrt{3}/4$	−0.433012701892219323382
3	0	1	3/8	+0.375
3	0	3	5/8	+0.625
3	1	1	$\sqrt{6}/16$	+0.153093108923948631137
3	1	3	$5\sqrt{6}/16$	+0.765465544619743155687
3	2	1	$\sqrt{15}/8$	+0.484122918275927110647
3	2	3	$-\sqrt{15}/8$	−0.484122918275927110647
3	3	1	$3\sqrt{10}/16$	+0.592927061281571124750
3	3	3	$-\sqrt{10}/16$	−0.197642353760523708250
4	0	0	3/64	+0.046875
4	0	2	5/16	+0.3125
4	0	4	35/64	+0.546875
4	1	2	$\sqrt{10}/16$	+0.197642353760523708250
4	1	4	$7\sqrt{10}/32$	+0.691748238161832978875
4	2	0	$3\sqrt{5}/32$	+0.209631372890605284038
4	2	2	$\sqrt{5}/8$	+0.279508497187473712051
4	2	4	$-7\sqrt{5}/32$	−0.489139870078078996090
4	3	2	$\sqrt{70}/16$	+0.522912516583797217486
4	3	4	$-\sqrt{70}/32$	−0.261456258291898608743
4	4	0	$3\sqrt{35}/64$	+0.277316239832794501995
4	4	2	$-\sqrt{35}/16$	−0.369754986443726002660
4	4	4	$\sqrt{35}/64$	+0.092438746610931500665

Substitution of this relation into the expressions of p_{nmk} given in the main text produces the recurrence formulae of $\tilde{p}_{nmk}$ as

$$\tilde{p}_{nmk} = k(-1)^m \alpha_{nm} \tilde{p}_{n,m+1,k} + \beta_{nm} \tilde{p}_{n,m+2,k},$$

$$(m = n-2, n-3, \dots, 0), \tag{53}$$

$$\tilde{p}_{n,n-1,k} = k(-1)^{n-1} \alpha_{n,n-1} \tilde{p}_{nnk},$$

$$(n = 1, 2, \dots;\ 0 \le k \le n;\ n-k : \text{even}). \tag{54}$$

$$\tilde{p}_{nn0} = \tilde{\gamma}_n \left(2\tilde{p}_{n-2,n-2,0} - \tilde{p}_{n-2,n-2,2}\right),$$

$$(n = 6, 8, 10, \dots), \tag{55}$$

$$\tilde{p}_{nn1} = \tilde{\gamma}_n \left(3\tilde{p}_{n-2,n-2,1} - \tilde{p}_{n-2,n-2,3}\right),$$

$$(n = 5, 7, 9, \dots), \tag{56}$$

$$\tilde{p}_{nn2} = \tilde{\gamma}_n \left(-2\tilde{p}_{n-2,n-2,0} + 2\tilde{p}_{n-2,n-2,2} - \tilde{p}_{n-2,n-2,4}\right),$$

$$(n = 6, 8, 10, \dots), \tag{57}$$

$$\tilde{p}_{nnk} = \tilde{\gamma}_n \left(-\tilde{p}_{n-2,n-2,k-2} + 2\tilde{p}_{n-2,n-2,k} - \tilde{p}_{n-2,n-2,k+2}\right),$$

$$(n = 7, 8, 9, \dots;\ 3 \le k \le n-4;\ n-k : \text{even}), \tag{58}$$

$$\tilde{p}_{nn,n-2} = \tilde{\gamma}_n \left(-\tilde{p}_{n-2,n-2,n-4} + 2\tilde{p}_{n-2,n-2,n-2}\right),$$

$$(n = 5, 6, 7, \dots), \tag{59}$$

$$\tilde{p}_{nnn} = -\tilde{\gamma}_n \tilde{p}_{n-2,n-2,n-2}, \quad (n = 5, 6, 7, \dots), \tag{60}$$

where α_{nm} and β_{nm} are already given in Eqs (10) and (11) while $\tilde{\gamma}_n$ is defined as

$$\tilde{\gamma}_n \equiv \frac{2n-1}{8\sqrt{n(n-1)}}. \tag{61}$$

Meanwhile, the starting values needed in the recursion is listed in Table 2.

C Algorithms

C.1 Computation of sine/cosine series coefficients

Let us consider an algorithm to prepare a set of the sine/cosine series coefficients of the fnALF, p_{nmk}, for the given domain of indices, $0 \le m \le n \le N$ and $0 \le k \le n$. First of all, we note the general zero value formula, $p_{nmk} = 0$ when $n-k$ is odd. In order to save the computer memory by using this fact, we introduce a compact rewriting of p_{nmk} as

$$P_{nmj} \equiv p_{nm,\mu_n+2j}, \tag{62}$$

where $\mu_n = \left[1-(-1)^n\right]/2$ is the parity factor. First, by using the index not k but $j \equiv [k/2]_{\text{floor}}$, we explicitly evaluate the initial values, namely P_{nmj} when $0 \le m \le n \le 4$ and $0 \le j \le [n/2]_{\text{floor}}$. Next, depending on the parity of degree, n, we split the sequence to compute the diagonal and semi-diagonal terms into two parts. For this purpose, we first introduce some auxiliary quantities as

$$\ell \equiv [n/2]_{\text{floor}}, \quad J \equiv [N/2]_{\text{floor}}. \tag{63}$$

Then, the computation of the diagonal and semi-diagonal terms are conducted sequentially. In fact, they are written for even degrees, namely when $n = 6, 8, \ldots, 2J$,

$$P_{nn0} = \gamma_n \left(2P_{n-2,n-2,0} - P_{n-2,n-2,1}\right), \tag{64}$$

$$P_{nn1} = \gamma_n \left(-2P_{n-2,n-2,0} + 2P_{n-2,n-2,1} - P_{n-2,n-2,2}\right), \tag{65}$$

$$P_{nnj} = \gamma_n \left(-P_{n-2,n-2,j-1} + 2P_{n-2,n-2,j} - P_{n-2,n-2,j+1}\right), \quad (j = 2, 3, \ldots, \ell - 2), \tag{66}$$

$$P_{nn,\ell-1} = \gamma_n \left(-P_{n-2,n-2,\ell-2} + 2P_{n-2,n-2,\ell-1}\right), \tag{67}$$

$$P_{nn\ell} = -\gamma_n P_{n-2,n-2,\ell-1}, \tag{68}$$

$$P_{n,n-1,j} = -2j\alpha_n P_{nnj}, \quad (j = 0, 1, \ldots, \ell), \tag{69}$$

where γ_n and α_n are already introduced in the main text. For odd degrees, namely when $n = 5, 7, \ldots, 2J + 1$, the formulae are expressed as

$$P_{nn0} = \gamma_n \left(3P_{n-2,n-2,0} - P_{n-2,n-2,1}\right), \tag{70}$$

$$P_{nnj} = \gamma_n \left(-P_{n-2,n-2,j-1} + 2P_{n-2,n-2,j} - P_{n-2,n-2,j+1}\right), \quad (j = 1, 2, \ldots, \ell - 2), \tag{71}$$

$$P_{nn,\ell-1} = \gamma_n \left(-P_{n-2,n-2,\ell-2} + 2P_{n-2,n-2,\ell-1}\right), \tag{72}$$

$$P_{nn\ell} = -\gamma_n P_{n-2,n-2,\ell-1}, \tag{73}$$

$$P_{n,n-1,j} = +(2j + 1)\alpha_n P_{nnj}, \quad (j = 0, 1, \ldots, \ell). \tag{74}$$

Finally, according as the value and parity of degree and order, we again split the main recursion into several cases in order to reduce the chance of conditional switch maximally: when both n and m are even and $j = 0$,

$$P_{nm0} = \beta_{nm} P_{n,m+2,0}, \quad (m = n - 2, n - 4, \ldots, 0), \tag{75}$$

when both n and m are even and $j \neq 0$,

$$P_{nmj} = +2j\alpha_{nm} P_{n,m+1,j} + \beta_{nm} P_{n,m+2,j}, \quad (m = n - 2, n - 3, \ldots, 0), \tag{76}$$

when n is even, m is odd, and $j \neq 0$,

$$P_{nmj} = -2j\alpha_{nm} P_{n,m+1,j} + \beta_{nm} P_{n,m+2,j}, \quad (m = n - 2, n - 3, \ldots, 0), \tag{77}$$

when n is odd and m is even,

$$P_{nmj} = +(2j + 1)\alpha_{nm} P_{n,m+1,j} + \beta_{nm} P_{n,m+2,j}, \quad (m = n - 2, n - 3, \ldots, 0), \tag{78}$$

and when both of n and m are odd,

$$P_{nmj} = -(2j + 1)\alpha_{nm} P_{n,m+1,j} + \beta_{nm} P_{n,m+2,j}, \quad (m = n - 2, n - 3, \ldots, 0), \tag{79}$$

where β_{nm} is specified in the main text. In the above, we excluded the case when n is even, m is odd, and $j = 0$. This is because $P_{nm0} = 0$ in that case, which corresponds to one of the zero value formulae.

C.2 Hints on parallel computation

Once the diagonal and semi-diagonal terms are prepared, the main recursion can be conducted in parallel with respect to n for its range, $n = 2, 3, \ldots, N$, and j for its range, $j = 0, 1, \ldots, J$. This fact will significantly accelerate the actual computation by its vector/parallel execution. Of course, the vector length of the main recursion is variable with respect to degree n, namely in proportion to $n - 1$. However, the resulting unbalance of the computational load is effectively avoided by the technique of do-loop folding (Fukushima, 2012c), namely by pairing the loops of degree, n and $N - n + 2$, and assigning the pairs to each computing unit. See also other examples of the acceleration by folding (Fukushima, 2011).

C.3 Application of X-number formulation

The recursive formulation described in the previous subsections experiences the underflow in the computation of the diagonal terms, P_{nnj}, as in the recursive computation of the fnALF (Fukushima, 2012a). In order to avoid the resulting precision degrade, we conduct the so-called X-number formulation (Fukushima, 2012a) in the recursive computation of P_{nmj}. Namely, we treat P_{nmj} as X-numbers and regard α_{nm}, β_{nm}, and γ_n as F-numbers.

C.4 Distributed summation of Fourier series coefficients

Once p_{nmk} are known, it is straightforward to transform the given spherical harmonic coefficients, $(\overline{C}_{nm}, \overline{S}_{nm})$, to the corresponding Fourier series coefficients, (A_{lm}, B_{km}), as described in the main text. However, a trick is required in its efficient implementation: a distributed summation. This is because p_{nmk} is sequentially determined for decreasing orders, $m = n, n-1, \ldots, 0$ while the summation must be conducted for increasing degrees, $n = \max(k, m), \ldots, N$, where N is the maximum degree. For this purpose, we adopt a following algorithm: to initialize all the Fourier coefficients by assigning zero values, and to increment them for each computed value of p_{nmk} as

```
A(k,m) = A(k,m) + p(n,m,k) * C(n,m);
```

```
B(k,m) = B(k,m) + p(n,m,k) * S(n,m);
```

if $m \le n$, $k \le n$, and $n - k$ is even. Actually, this process is written as a triple do-loop as implemented in `xfsh2f` as seen in Tables 8 and 9.

D Fortran programs

Here we gather a group of Fortran subroutines to execute the new method presented in the main text. First of all, Table 3 shows `pinit`, a primitive subroutine to return P_{nmj} when $0 \le n \le 4$. Next come `dpeven` and `dpodd` listed in Tables 4 and 5, respectively. They return the double precision (DP) X-number vectors representing P_{nnj} and $P_{n,n-1,j}$, the diagonal and semi-diagonal transformation coefficients, for even and odd degree/order n, respectively. In using them, we assume that the DP X-number vector of the two-step-previous values, $P_{n-2,n-2,j}$, is provided externally if $n \ge 2$. This is the reason why, in `xfsh2f`, we separated the diagonal computation into even and odd sequences of n, which can be conducted in parallel. At any rate, apart from the special value formulae for $0 \le n \le 4$, the sequential calls of `dpeven` for $n = 6, 8, \ldots$ and those of `dpodd` for $n = 5, 7, \ldots$ provide a set of the diagonal and semi-diagonal terms. Thirdly, Tables 6 and 7 illustrate `gpeven` and `gpodd`. They return a vector of P_{nmj} with respect to j when n is even and odd, respectively. The programs require two previous vectors, $P_{n,m+1,j}$ and $P_{n,m+2,j}$. Finally, we prepared Tables 8 and 9 listing `xfsh2f`, a Fortran subroutine to transform the 4π fully normalized spherical harmonic coefficients, $\overline{C}_{nm}$ and $\overline{S}_{nm}$, into the corresponding Fourier coefficients, A_{km} and B_{km}. It internally calls `pinit`, `dpeven`, `dpodd`, `gpeven`, and `gpodd`.

Table 3: Fortran subroutine to return P_{nmj} when $n \le 4$. It returns `p(j)` $= P_{nmj}$ as a vector with respect to j. This is a straightforward implementation using Table 1.

```
subroutine pinit(n,m,p)
integer n,m;real*8 p(0:*)
if(n.eq.0) then
    p(0)=1.d0
elseif(n.eq.1) then
    p(0)=+1.7320508075688773d0
elseif(n.eq.2) then
    if(m.eq.0) then
        p(0)=+0.5590169943749474d0
        p(1)=+1.6770509831248423d0
    elseif(m.eq.1) then
        p(0)=0.d0
        p(1)=+1.9364916731037084d0
    elseif(m.eq.2) then
        p(0)=+0.9682458365518542d0
        p(1)=-0.9682458365518542d0
    endif
elseif(n.eq.3) then
    if(m.eq.0) then
        p(0)=+0.9921567416492215d0
        p(1)=+1.6535945694153691d0
    elseif(m.eq.1) then
        p(0)=+0.4050462936504913d0
        p(1)=+2.0252314682524563d0
    elseif(m.eq.2) then
        p(0)=+1.2806888457449498d0
        p(1)=-1.2806888457449498d0
    elseif(m.eq.3) then
        p(0)=+1.5687375497513917d0
        p(1)=-0.5229125165837972d0
    endif
elseif(n.eq.4) then
    if(m.eq.0) then
        p(0)=+0.421875d0
        p(1)=+0.9375d0
        p(2)=+1.640625d0
    elseif(m.eq.1) then
        p(0)=0.d0
        p(1)=+0.5929270612815711d0
        p(2)=+2.0752447144854989d0
    elseif(m.eq.2) then
        p(0)=+0.6288941186718159d0
        p(1)=+0.8385254915624211d0
        p(2)=-1.4674196102342370d0
    elseif(m.eq.3) then
        p(0)=0.d0
        p(1)=+1.5687375497513917d0
        p(2)=-0.7843687748756958d0
    elseif(m.eq.4) then
        p(0)=+0.8319487194983835d0
        p(1)=-1.1092649593311780d0
        p(2)=+0.2773162398327945d0
    endif
endif
return;end
```

Table 4: Fortran subroutine to return P_{nnj} and $P_{n,n-1,j}$ when degree n is even and $n \geq 6$. The returned values are (xp(j), ip(j)) and (xp1(j), ip1(j)), double precision X-number vectors representing P_{nnj} and $P_{n,n-1,j}$, respectively. We assume the availability of $P_{n-2,n-2,j}$ as (xpold(j), ipold(j)). The subroutine internally calls xnorm and xlsum2 listed in Tables 7 and 8 of Fukushima (2012a), respectively.

```
subroutine dpeven(n,xpold,xp,xp1,ipold,ip,ip1)
integer n,ipold(0:*),ip(0:*),ip1(0:*)
integer jx,jxm2,jxm1,n2,j,itemp,jm1,jp1
real*8 xpold(0:*),xp(0:*),xp1(0:*)
real*8 gamma,gamma2,xtemp,alpha2
jx=n/2;jxm2=jx-2;jxm1=jx-1;n2=n*2
gamma=sqrt(dble(n2+1)*dble(n2-1)/ &
     (dble(n)*dble(n-1)))*0.125d0
gamma2=gamma*2.d0
call xlsum2(gamma2,xpold(0),-gamma,xpold(1), &
     xp(0),ipold(0),ipold(1),ip(0))
call xlsum2(-gamma2,xpold(0),gamma2,xpold(1), &
     xtemp,ipold(0),ipold(1),itemp)
call xlsum2(1.d0,xtemp,-gamma,xpold(2),xp(1), &
     itemp,ipold(2),ip(1))
do j=2,jxm2
     jm1=j-1;jp1=j+1
     call xlsum2(-gamma,xpold(jm1),gamma2, &
          xpold(j),xtemp,ipold(jm1),ipold(j),itemp)
     call xlsum2(1.d0,xtemp,-gamma,xpold(jp1), &
          xp(j),itemp,ipold(jp1),ip(j))
enddo
call xlsum2(-gamma,xpold(jxm2),gamma2,xpold(jxm1), &
     xp(jxm1),ipold(jxm2),ipold(jxm1),ip(jxm1))
xp(jx)=-gamma*xpold(jxm1);ip(jx)=ipold(jxm1)
call xnorm(xp(jx),ip(jx))
alpha2=sqrt(2.d0/dble(n))*2.d0
xp1(0)=0.d0;ip1(0)=0
do j=1,jx
     xp1(j)=-dble(j)*alpha2*xp(j);ip1(j)=ip(j)
     call xnorm(xp1(j),ip1(j))
enddo
return;end
```

Table 5: Fortran subroutine to return P_{nnj} and $P_{n,n-1,j}$ when n is odd and $n \geq 5$. Same as Table 4 but when n is odd and $n \geq 5$.

```
subroutine dpodd(n,xpold,xp,xp1,ipold,ip,ip1)
integer n,ipold(0:*),ip(0:*),ip1(0:*)
integer jx,jxm2,jxm1,n2,j,itemp,jm1,jp1
real*8 xpold(0:*),xp(0:*),xp1(0:*)
real*8 gamma,gamma2,xtemp,alpha
jx=(n-1)/2;jxm2=jx-2;jxm1=jx-1;n2=n*2
gamma=sqrt(dble(n2+1)*dble(n2-1)/ &
     (dble(n)*dble(n-1)))*0.125d0
gamma2=gamma*2.d0
call xlsum2(gamma*3.d0,xpold(0),-gamma,xpold(1), &
     xp(0),ipold(0),ipold(1),ip(0))
do j=1,jxm2
     jm1=j-1;jp1=j+1
     call xlsum2(-gamma,xpold(jm1),gamma2, &
          xpold(j),xtemp,ipold(jm1),ipold(j),itemp)
     call xlsum2(1.d0,xtemp,-gamma,xpold(jp1), &
          xp(j),itemp,ipold(jp1),ip(j))
enddo
call xlsum2(-gamma,xpold(jxm2),gamma2,xpold(jxm1), &
     xp(jxm1),ipold(jxm2),ipold(jxm1),ip(jxm1))
xp(jx)=-gamma*xpold(jxm1);ip(jx)=ipold(jxm1)
call xnorm(xp(jx),ip(jx))
alpha=sqrt(2.d0/dble(n))
do j=0,jx
     xp1(j)=dble(2*j+1)*alpha*xp(j);ip1(j)=ip(j)
     call xnorm(xp1(j),ip1(j))
enddo
return;end
```

Table 6: Fortran subroutine to return P_{nmj} when n is even. The returned values are (xp0(j), ip0(j)), a double precision X-number vector representing P_{nmj}. We assume that $P_{n,m+1,j}$ and $P_{n,m+2,j}$ are externally provided as (xp1(j), ip1(j)) and (xp2(j), ip2(j)), respectively. The subroutine internally calls xnorm and xlsum2 provided in Tables 7 and 8 of Fukushima (2012a).

```
subroutine gpeven(jmax,n,m,xp2,xp1,xp0,ip2,ip1,ip0)
integer jmax,n,m,ip2(0:*),ip1(0:*),ip0(0:*),m1,m2,modd
real*8 xp2(0:*),xp1(0:*),xp0(0:*),u,alpha2,beta
m1=m+1;m2=m+2;modd=m-int(m/2)*2
if(m.eq.0) then
    u=sqrt(0.5d0/(dble(n)*dble(n+1)))
else
    u=sqrt(1.d0/(dble(n-m)*dble(n+m1)))
endif
alpha2=4.d0*u;beta=sqrt(dble(n-m1)*dble(n+m2))*u
xp0(0)=beta*xp2(0);ip0(0)=ip2(0)
call xnorm(xp0(0),ip0(0))
if(modd.eq.0) then
    do j=1,jmax
        call xlsum2(dble(j)*alpha2,xp1(j),beta, &
            xp2(j),xp0(j),ip1(j),ip2(j),ip0(j))
    enddo
else
    do j=1,jmax
        call xlsum2(-dble(j)*alpha2,xp1(j),beta, &
            xp2(j),xp0(j),ip1(j),ip2(j),ip0(j))
    enddo
endif
return;end
```

Table 7: Fortran subroutine to return P_{nmj} when n is odd. Same as Table 6 but when n is odd.

```
subroutine gpodd(jmax,n,m,xp2,xp1,xp0,ip2,ip1,ip0)
integer jmax,n,m,ip2(0:*),ip1(0:*),ip0(0:*),m1,m2,modd
real*8 xp2(0:*),xp1(0:*),xp0(0:*),u,alpha,beta
m1=m+1;m2=m+2;modd=m-int(m/2)*2
if(m.eq.0) then
    u=sqrt(0.5d0/(dble(n)*dble(n+1)))
else
    u=sqrt(1.d0/(dble(n-m)*dble(n+m1)))
endif
alpha=2.d0*u;beta=sqrt(dble(n-m1)*dble(n+m2))*u
if(modd.eq.0) then
    do j=0,jmax
        call xlsum2(dble(2*j+1)*alpha,xp1(j),beta, &
            xp2(j),xp0(j),ip1(j),ip2(j),ip0(j))
    enddo
else
    do j=0,jmax
        call xlsum2(-dble(2*j+1)*alpha,xp1(j),beta, &
            xp2(j),xp0(j),ip1(j),ip2(j),ip0(j))
    enddo
endif
return;end
```

Table 8: Fortran subroutine to transform $\left(\overline{C}_{nm}, \overline{S}_{nm}\right)$, the 4π fully normalized spherical harmonic coefficients of a given function defined on the spherical surface, to (A_{km}, B_{km}), the corresponding Fourier series coefficients of the function. In the program, (i) x2f and xnorm are the Fortran function/subroutine to handle X-numbers (Fukushima, 2012a, tables 6 and 7), and (ii) pinit, dpeven, dpodd, gpeven, and gpodd are the Fortran subroutines listed in Tables 3–7, respectively (continuing).

```
subroutine xfsh2f(nmax,c,s,a,b)
integer nmax,m,k,n,jmax,n1,NX;parameter (NX=2200)
real*8 c(0:NX,0:NX),s(0:NX,0:NX)
real*8 a(0:NX,0:NX),b(0:NX,0:NX)
integer ipold(0:NX),ip(0:NX),ip0(0:NX)
integer ip1(0:NX),ip2(0:NX)
real*8 xpold(0:NX),xp(0:NX),xp0(0:NX)
real*8 xp1(0:NX),xp2(0:NX),pj,x2f
do m=0,nmax
    do k=0,nmax
        a(k,m)=0.d0;b(k,m)=0.d0
    enddo
enddo
do n=0,4,2
    jmax=int(n/2)
    do m=0,n
        call pinit(n,m,xp)
        do j=0,jmax
            k=2*j;a(k,m)=a(k,m)+xp(j)*c(n,m)
            b(k,m)=b(k,m)+xp(j)*s(n,m)
        enddo
    enddo
enddo
do j=0,jmax
    ip(j)=0
enddo
do n=6,nmax,2
    do j=0,jmax
        xpold(j)=xp(j);ipold(j)=ip(j)
    enddo
    jmax=int(n/2);n1=n-1
    call dpeven(n,xpold,xp,xp1,ipold,ip,ip1)
    do j=0,jmax
        k=2*j;pj=x2f(xp(j),ip(j))
        a(k,n)=a(k,n)+pj*c(n,n)
        b(k,n)=b(k,n)+pj*s(n,n)
        pj=x2f(xp1(j),ip1(j))
        a(k,n1)=a(k,n1)+pj*c(n,n1)
        b(k,n1)=b(k,n1)+pj*s(n,n1)
        xp2(j)=xp(j);ip2(j)=ip(j)
    enddo
    do m=n-2,0,-1
        call gpeven(jmax,n,m,xp2,xp1,xp0,ip2,ip1,ip0)
        do j=0,jmax
            k=2*j;pj=x2f(xp0(j),ip0(j))
            a(k,m)=a(k,m)+pj*c(n,m)
            b(k,m)=b(k,m)+pj*s(n,m)
            xp2(j)=xp1(j);ip2(j)=ip1(j)
            xp1(j)=xp0(j);ip1(j)=ip0(j)
        enddo
    enddo
enddo
```

Table 9: Fortran subroutine to transform $\left(\overline{C}_{nm}, \overline{S}_{nm}\right)$ to (A_{km}, B_{km}) (continued).

```
do n=1,3,2
    jmax=int((n−1)/2)
    do m=0,n
        call pinit(n,m,xp)
        do j=0,jmax
            k=2*j+1;a(k,m)=a(k,m)+xp(j)*c(n,m)
            b(k,m)=b(k,m)+xp(j)*s(n,m)
        enddo
    enddo
enddo
do j=0,jmax
    ip(j)=0
enddo
do n=5,nmax,2
    do j=0,jmax
        xpold(j)=xp(j);ipold(j)=ip(j)
    enddo
    jmax=int((n−1)/2);n1=n−1
    call dpodd(n,xpold,xp,xp1,ipold,ip,ip1)
    do j=0,jmax
        k=2*j+1;pj=x2f(xp(j),ip(j))
        a(k,n)=a(k,n)+pj*c(n,n)
        b(k,n)=b(k,n)+pj*s(n,n)
        pj=x2f(xp1(j),ip1(j))
        a(k,n1)=a(k,n1)+pj*c(n,n1)
        b(k,n1)=b(k,n1)+pj*s(n,n1)
        xp2(j)=xp(j);ip2(j)=ip(j)
    enddo
    do m=n−2,0,−1
        call gpodd(jmax,n,m,xp2,xp1,xp0,ip2,ip1,ip0)
        do j=0,jmax
            k=2*j+1;pj=x2f(xp0(j),ip0(j))
            a(k,m)=a(k,m)+pj*c(n,m)
            b(k,m)=b(k,m)+pj*s(n,m)
            xp2(j)=xp1(j);ip2(j)=ip1(j)
            xp1(j)=xp0(j);ip1(j)=ip0(j)
        enddo
    enddo
enddo
return;end
```

References

Abramowitz M. and Stegun I. A. (eds), 1964, Handbook of mathematical functions with formulae, graphs, and mathematical tables. NBS Appl. Math. Ser. 55, NBS, Washington DC.

Bosch W., 2000, On the computation of derivatives of Legendre functions. Phys. Chem. Earth, A, 25:655–659.

Colombo O.L., 1981, Numerical methods for harmonic analysis on the sphere. Rep 310, Dept. Geod. Sci., Ohio State Univ., Columbus.

de Pater I. and Lissauer J.J., 2010, Planetary Sciences, 2nd ed. Cambridge Univ. Press, London.

Egersdorfer R. and Egersdorfer L., 1936, Formeln und Tabellen der zugeordneten Kugelfunktionen 1. Art von $n = 1$ bis $n = 20$. Reichs. für Wett. Wiss., I:18–47.

Fukushima T., 2011, Efficient parallel computation of all-pairs N-body acceleration by do loop folding. Astron. J., 142:18–22.

Fukushima T., 2012a, Numerical computation of spherical harmonics of arbitrary degree and order by extending exponent of floating point numbers. J. Geod., 86:271–285.

Fukushima T., 2012b, Numerical computation of spherical harmonics of arbitrary degree and order by extending exponent of floating point numbers: II first-, second-, and third-order derivatives. J. Geod., 86:1019–1028.

Fukushima T., 2012c, Parallel computation of satellite orbit acceleration. Comp. Geosci., 49:1–9.

Fukushima T., 2013, Recursive computation of oblate spheroidal harmonics of the second kind and their first-, second-, and third-order derivatives. J. Geod., 87:303–309.

Fukushima T., 2014, Prolate spheroidal harmonic expansion of gravitational field. Astron. J., 147:152–160.

Fukushima T., 2016, Numerical computation of point values, derivatives, and integrals of associated Legendre function of the first kind and point values and derivatives of oblate spheroidal harmonics of the second kind of high degree and order. Proc. IAG Symp., 143:192–197.

Fukushima T., 2017, Rectangular rotation of spherical harmonic expansion of arbitrary high degree and order. J. Geod., 91:995–1011.

Fukushima T., 2018, Transformation between surface spherical harmonic expansion of arbitrary high degree and order and double Fourier series on sphere. J. Geod., 92:123–130.

Gruber C. and Abrykosov O., 2016, On computation and use of Fourier coefficients for associated Legendre functions. J. Geod., 90:525–535.

Heiskanen W.A. and Moritz H., 1967, Physical Geodesy. Freeman, San Francisco.

Hofsommer D.J. and Potters M.L., 1960, Table of Fourier coefficients of associated Legendre functions. Proc. KNAW ser. A Math. Sci., 63:460–466.

Olver F. W. J., Lozier D. W., Boisvert R. F., and Clark C. W. (eds), 2010, NIST Handbook of Mathematical Functions. Cambridge Univ Press, Cambridge.

Rexer M. and Hirt C., 2015a, Ultra-high degree surface spherical harmonic analysis using the Gauss-Legendre and the Driscoll/Healy quadrature theorem and application to planetary topography models of Earth, Mars and Moon. Surv. Geophys., 36:803–830.

Rexer M. and Hirt C., 2015b, Spectral analysis of the Earth's topographic potential via 2D-DFT: a new data-based degree variance model to degree 90,000. J. Geod., 89:887–909.

Sneeuw N.J. and Bun R., 1996, Global spherical harmonic computation by two-dimensional Fourier methods. J. Geod., 70:224–232.

Stacey F.D. and Davis P.M., 2008, Physics of the Earth, 4th ed. Cambridge Univ. Press, Cambridge.

Winch D.E., Ivers D.J., Turner J.P.R., and Stening R.J., 2005, Geomagnetism and Schmidt quasi-normalization. Geophys. J. Int., 160:487–504.

Wolfram S., 2003, The Mathematica Book, 5th ed. Wolfram Research Inc./Cambridge Univ. Press, Cambridge.

Zwillinger D. and Moll V. (eds), 2014, (Gradshteyn and Ryzhik) Table of integrals, series, and products, 8th ed. Acad Press, Amsterdam.