Spherical Harmonic Transform Algorithms

John B. Drake
and
Pat Worley
and
Eduardo D'Azevedo
Oak Ridge National Laboratory

A collection of MATLAB classes for computing spherical harmonic transforms are presented and used to solve simple partial differential equations on the sphere. The spectral synthesis and analysis using fast Fourier transforms and Legendre transforms with the associated Legendre functions are presented in depth. A set of methods associated with a spectral_field class provides spectral approximation to the $\nabla\cdot$, $\nabla\times$, ∇, and ∇^2 in spherical geometry. Laplace inversion and Helmholtz equation solvers are also methods for this class. Investigation of algorithms and analysis for spherical harmonic transform options for parallel high performance computers are discussed in the context of global climate and weather models.

Categories and Subject Descriptors: F.2.1 [**Theory of Computation**]: Analysis of Algorithms—*Spherical Harmonic Transforms*; G.4 [**Software**]: Mathematical Software—*MATLAB*; G.1.8 [**Software**]: Numerical Analysis—*Partial Differential Equations*; J.2 [**Applications**]: Physical Sciences and Engineering—*Climate Modeling*

General Terms: spectral transform methods, spherical

Additional Key Words and Phrases: fluid dynamics, geophysical flow, high performance computing

1. INTRODUCTION

The spherical harmonic transform is a critical computational kernel of the dynamics portion of spectral atmospheric weather and climate codes. The announcement of sustained rates of 26.5 Tflops on the Japanese Earth Simulator (ES40, with NEC SX-6+ nodes) for an atmospheric simulation motivated a study of algorithmic options for implementing spectral transforms. The paper [Shingu et al. 2002] that won the 2002 Gordon Bell Award in Supercomputing claimed a 26.58 Tflops computational rate on the full 640 nodes of the Japanese Earth Simulator. The core of the computation of the atmospheric general circulation model (the AFES

Corresponding author is John Drake, Computer Science and Mathematics Division, Oak Ridge National Laboratory, P.O. Box 2008, MS 6367, Oak Ridge, Tennessee, 37831-6367 (drakejb@ornl.gov).
This research at Oak Ridge National Laboratory was supported by the Climate Change Prediction Program, Office of Biological and Environmental Research, U.S. Department of Energy under contract DE-AC05-84OR21400 with UT-Battelle, Inc.
Permission to make digital/hard copy of all or part of this material without fee for personal or classroom use provided that the copies are not made or distributed for profit or commercial advantage, the ACM copyright/server notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists requires prior specific permission and/or a fee.
© 200? ACM 0098-3500/200?/1200-0111 $5.00

code) that achieved this rate is a multi-level spectral transform. The dynamics part of the calculation on 640 nodes accounted for 62% of the total time with the Legendre transform alone accounting for 51.8%. The columnar physics is reported at only 12% of the total run-time. The resolution used was T1279 with 96 levels and a (3840 x 1920) horizontal grid. This is high resolution for a weather model and ultra-high for a climate model which must be integrated for years instead of days. The horizontal resolution typically used for climate simulations in the US is T85L26, which requires a (256 x 128) horizontal grid [Drake et al. 2005; Worley and Drake 2005]. The parallel algorithm used for the high resolution study and benchmarking is given in [Foster and Worley 1997]. The FFT algorithm used is given in [Temperton 1983] and is a Fortran code specifically designed for vector computation of multiple (blocked) fast Fourier transforms.

The spatial resolution of a spectral model is referred to as a truncation and specifies the number of spectral modes retained in the representation of a scalar field. Spectral methods have been applied to a wide range of fluids problems and the theory of their application is given in [Canuto et al. 1988]. For flows in a global domain, the prefered basis set for approximation of functions on the sphere, is the spherical harmonic basis. The spherical harmonic transform is used to project grid point data on the sphere onto the spectral modes in an *analysis* step and an inverse transform reconstructs grid point data from the spectral information in a *synthesis* step. The synthesis step is described equation (1). The analysis step is described by equations (2) and (3) consisting of the computation of the Fourier coefficient ξ^m and the Legendre transform which incorporates the Gaussian weights corresponding to the Gaussian latitudes $\mu_j = \sin(\theta_j)$.

$$\xi(\lambda, \mu) = \sum_{m=-M}^{M} \sum_{n=|m|}^{N(m)} \xi_n^m P_n^m(\mu) e^{im\lambda}, \tag{1}$$

$$\xi_n^m = \sum_{j=1}^{J} w_j \xi^m(\mu_j) P_n^m(\mu_j), \tag{2}$$

$$\xi^m(\mu_j) = \frac{1}{I} \sum_{i=1}^{I} \xi(\lambda_i, \mu_j) e^{im\lambda_i}. \tag{3}$$

For a Gaussian grid the triangular spectral truncation requires the number of longitudes $I \geq 3M + 1$ and number of latitudes $J = I/2$, here M refers to the modal truncation number. In what follows we will assume a triangular truncation, though extension to other truncations is straightforward.

A performance model of the parallel spectral transform can be developed to estimate the time for a multi-level calculation. The computational operation counts and communication cost estimates are based on a model in [Foster and Worley 1997] for a one dimensional decomposition and modified by Rich Loft (NCAR) to reflect a simple transpose between FFT and Legendre transform phases including vertical levels. The time for the FFT, the Legendre transform and the communication overhead are estimated using machine dependant rate constants a,b,d, and e.

Time for FFT $= 5a(6L+1)IJlog_2(I)$
Time for LT $= 2b(6L+1)JM^2$
Time in COMM $= dP + 2e(6L+1)J(2M+1)$
Nomenclature:
- M wave number resolution, eg. TM
- I number of longitudes ($I \geq 3M+1$)
- J number of latitudes ($J = I/2$)
- L number of vertical levels
- P number of nodes (computational unit doing FFT or LT)
- a computational rate of FFT in flops/node
- b computational rate for LT in flops/node
- d latency factor
- e bandwidth factor

Using this model with estimates of network bandwidth and the speed of a node in computing FFTs and Legendre transforms, we can project the overall, sustained computational rate of a computer for performing spherical harmonic transforms. Refinement of the performance model requires an experimental determination of the a,b,d, and e parameters. This can be done with kernel tests or by fitting performance data. The following figure shows the balance of latency (nanoseconds), bandwidth (GigaBytes per second) and computational performance (GigaFlops/sec) on the Legendre transform and FFTs, required by a system that will compute the spherical harmonic transforms for an atmospheric model with 96 levels at the T1279 resolution using 1920 processing nodes.

2. FORMULATION OPTIONS

2.1 BLAS Formulation

The spherical harmonic transform can be formulated in terms of matrix operations. This follows from the fact that it is a linear transformation of one basis representation to another. Since the Fourier transform calculation is most efficiently organized with the FFT algorithms what we describe here is matrix formulation for the Legendre transforms. A generalization to non-Gaussian grids is also possible as reported in [Swarztrauber and Spotz 2000]. For high performance hardware the efficiency of specialized matrix operations is well known and forms the basis of the LINPACK benchmark. The Basic Linear Algebra Subroutines (BLAS) have been optimized by most vendors and offer near peak rates. Since the Legendre transform can be expressed in matrix form we are led to explore the possibility of using BLAS routines for the computational kernel of the spherical harmonic transform.

2.1.1 *Li's Principle Sums.* In an unpublished study, Li [Li 1996] described the principle sums that form the Legendre transform in the synthesis and analysis phases and determined a matrix formulation that took advantage of the symmetry of the associated Legendre functions. The principle sum for the synthesis phase is

$$s_j^m = \sum_{n=m}^{N(m)} \xi_n^m P_n^m(\mu_j) \tag{4}$$

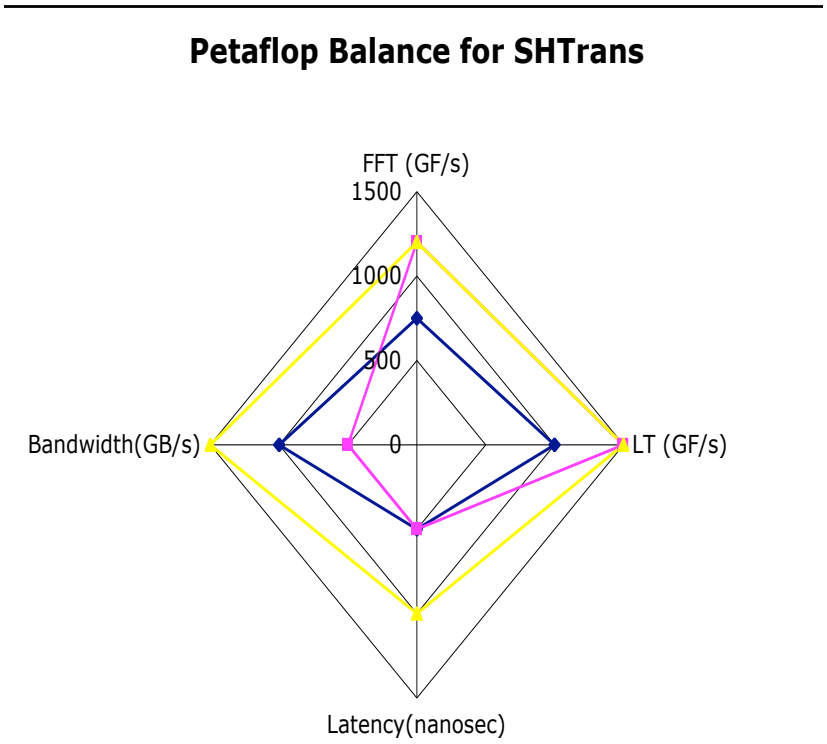


Fig. 1. System balance required for a sustained petaflop calculating T1279 transforms on 1920 processing nodes.

where the ξ_n^m is the spectral coefficient of a field. The principle sum of the analysis phase is

$$\xi_n^m = \sum_{j=1}^{J} s_j^m P_n^m(\mu_j) \tag{5}$$

where the $s_j^m = w_j \xi^m(\mu_j)$ represents the product of the Gauss weight and the m-th Fourier coefficient at latitude j.

The matrix-matrix multiplications representing these sums requires some additional notation. Let

$$\mathbf{P}_m = \begin{bmatrix} P_m^m(\mu_1) & P_{m+1}^m(\mu_1) & \cdots & \cdots & P_{N(m)}^m(\mu_1) \\ P_m^m(\mu_2) & P_{m+1}^m(\mu_2) & & & P_{N(m)}^m(\mu_2) \\ \vdots & & & \ddots & \vdots \\ \vdots & & & & \ddots \ \vdots \\ P_m^m(\mu_{J/2}) & P_{m+1}^m(\mu_{J/2}) & \cdots & \cdots & P_{N(m)}^m(\mu_{J/2}) \end{bmatrix} \tag{6}$$

be the matrix of associated Legendre functions for mode m at half of the Gauss points. Since Legendre functions are symmetric about the equator and the Gauss points are anti-symmetric, the algorithm does not require computation of the functions at all the points. The operative identities are

$$P_n^m(\mu_{J+1-j}) = P_n^m(-\mu_j) = (-1)^{n-m} P_n^m(\mu_j). \tag{7}$$

Introducing a vector notation for the spectral coefficients,

$$\mathbf{x}_m = \begin{bmatrix} \xi_m^m \\ \xi_{m+1}^m \\ \vdots \\ \xi_{N(m)}^m \end{bmatrix} \tag{8}$$

and

$$\tilde{\mathbf{x}}_m = \begin{bmatrix} \xi_m^m \\ -\xi_{m+1}^m \\ \vdots \\ (-1)^{N(m)-m} \xi_{N(m)}^m \end{bmatrix} \tag{9}$$

the first principle sum can be represented in a matrix-matrix multiplication formulation as

$$\begin{bmatrix} s_1^m & s_J^m \\ s_2^m & s_{J-1}^m \\ \vdots & \vdots \\ s_{J/2}^m & s_{J/2+1}^m \end{bmatrix} = \mathbf{P}_m [\mathbf{x}_m \tilde{\mathbf{x}}_m]. \tag{10}$$

In this equation the s_j^m's and the ξ_n^m's are complex while the $\mathbf{P}$ matrix is real. Computational performance may be enhanced by explicitly separating the complex vectors into real and imaginary parts forming input and output matrices with four columns instead of two. The inverse transform (analysis phase) involves two steps. First, a matrix-matrix multiply step uses the transpose of the Legendre matrix,

$$\begin{bmatrix} \tau_1^m & \tilde{\tau}_J^m \\ \tau_2^m & \tilde{\tau}_{J-1}^m \\ \vdots & \vdots \\ \tau_{J/2}^m & \tilde{\tau}_{J/2+1}^m \end{bmatrix} = (\mathbf{P}_m)^T \begin{bmatrix} s_1^m & s_J^m \\ s_2^m & s_{J-1}^m \\ \vdots & \vdots \\ s_{J/2}^m & s_{J/2+1}^m \end{bmatrix}. \tag{11}$$

The intermediate quantities, τ_n^m and $\tilde{\tau}_n^m$, are then used to compute the spectral coefficients,

$$\xi_n^m = \tau_n^m + (-1)^{n-m}\tilde{\tau}_n^m. \tag{12}$$

The formulation has been implemented in MATLAB, where a fast BLAS is available when matrix notation is used. This allows us to test the formulation as well as the assumptions of advantage with specialized library routines. The MATLAB code and class structure used to express the formulation are described in section 4.

By timing the computational portions of the transforms we note that a considerable amount of time is spent in packing and unpacking Fourier and spectral coefficients and very little time in the matrix multiply and FFT. Both of these computational steps are highly optimized in MATLAB, using FFTW for the FFTs and LAPACK for the matrix multiply. This is very similar to the situation with using math libraries on supercomputers since these are highly optimized but may in fact require incompatible storage orders.

2.2 Exposed Loop Formulation

A formulation that does not require explicit data movement to accommodate special purpose routines, leaves much to the compiler. A good compiler will recognize BLAS constructs and take appropriate action depending on the size of loops, etc.

2.2.1 *Odd-Even Constructs.* The basic computational loops as given in Ren-Cang Lis formulation are the same, but we split them in such away as to exploit the symmetry of the Legendre functions. This is done by splitting into odd and even modes. The first sum can be written in two parts for $(1 \leq j \leq J/2)$,,

$$s_j^m = \sum_{n=m}^{N(m)} \xi_n^m P_n^m(\mu_j) \tag{13}$$

and

$$s_{J+1-j}^m = \sum_{n=m,2}^{N(m)} \xi_n^m P_n^m(\mu_j) - \sum_{m+1,2}^{N(m)} \xi_n^m P_n^m(\mu_j). \tag{14}$$

The second sum is represented in different ways when $(n-m)$ is odd or even,

$$\xi_n^m = \sum_{j=1}^{J/2} (s_j^m + s_{J+1-j}^m) P_n^m(\mu_j) \quad , mod_2(n-m) = 0, \tag{15}$$

$$\xi_n^m = \sum_{j=1}^{J/2} (s_j^m - s_{J+1-j}^m) P_n^m(\mu_j) \quad , mod_2(n-m) = 1 \tag{16}$$

2.3 Legendre Functions On-the-Fly Formulation

2.3.1 *Recursions.* In [4], a 4 term recursion is given for computing the normalized associated Legendre functions. Since the recursion can be applied to an entire column of the $\mathbf{P}_m$ matrix with vector operations, it may be advantageous to compute the functions on the fly. This has the added advantage of reducing the storage required for the spectral transform from $O(M^3)$ to $O(M^2)$.

The normalized Legendre functions are defined by

$$P_n^m(\theta) \equiv \frac{1}{2^n n!}\sqrt{\frac{(2n+1)(n-m)!}{2(n+m)!}}\cos^m(\theta)\frac{d^{n+m}}{d\mu^{n+m}}(\mu^2-1)^n, \tag{17}$$

where $\mu = \sin\theta$. The four-term recursion starts from pre-computed and stored values of the matrices $\mathbf{P}_0$ and $\mathbf{P}_1$. Denoting a single column of the $\mathbf{P}_m$ matrix by

$$\vec{P}_n^m = \begin{bmatrix} P_n^m(\mu_1) \\ \vdots \\ P_n^m(\mu_{J/2}) \end{bmatrix}, \tag{18}$$

the recurrence in [4, equation D.1] is given by

$$\vec{P}_{n+1}^{m+1} = \frac{1}{a_{n+m+1}}(b_n a_{n+m+1}\vec{P}_{n-1}^{m-1} - a_{n-m+1}\vec{P}_{n+1}^{m-1} + b_n a_{n-m-1}\vec{P}_{n-1}^{m+1}). \tag{19}$$

The coefficients of the recursion are given explicitly by $a_n = \sqrt{n(n+1)}$ and $b_n = \sqrt{\frac{(2n+3)}{(2n-1)}}$. Using this recursion the columns of the other matrices can be computed using vector operations. These can be computed on the fly as part of the m-loop of the Legendre transforms in either formulation. Since the recurrence can be split into odd and even modes it fits well with the open loop formulation without requiring either duplicate computation or intermediate storage.

An discussion of performance of these options on vector and scalar processors was given in [D'Azevedo 2004].

3. ALTERNATIVE ALGORITHMS

A number of papers propose alternative algorithms for the spherical harmonic transform. In this section we will briefly survey these in an effort to gauge their appropriate use for high resolution climate and weather modeling. The possibility of replacing the transform kernel with a lower operation count algorithm, or of finding a more efficient method of computation is what motivates the survey. Our conclusion is that new formulations of the larger dynamical problem and other means of evaluating derivatives for the partial differential equations do indeed offer more promising algorithms. These new algorithms represent the progress in the numerical analysis of spectral methods that has occurred over the last decade but has not yet found its way into production use.

3.1 THE FAST SPHERICAL HARMONIC ALGORITHMS

Driscoll and Healy [Driscoll and Healy, Jr. 1989] introduced the first exact fast spherical harmonic transform in 1989. Subsequent refinements [Healy, Jr. et al. 1996; Inda et al. 2001; Mohlenkamp 1999] have resulted in parallel versions with reasonable accuracy, stability and performance. The asympototic operation count for the fast algorithm is $M^2(\log 2M)^2$. A cross-over point for performance in comparision with the direct method occurs at M=128 with a savings by a factor of three at M=512. For M=1279 and other high resolution cases, the operation count of the transform can be reduced dramatically.

An approximate fast transform has been developed using the ideas in [Boyd 1992]. The new algorithm [Suda and Takami 2002] is based on fast polynomial interpolation accelerated by the Fast Multipole Method. The asymptotic operation count is $M^2 \log M$, an improvement on the Driscoll and Healy algorithm. The crossover point with the direct method is observed at M=512. A speedup factor of 1.8 is observed at M=1365.

3.2 FFT2D Derivative Evaluations

The main use of the spectral method is in the highly accurate and efficient representation of differential terms in numerical approximations to partial differential equations. For the flow equations of the atmosphere used in weather and climate modeling, the spherical harmonic spectral representation leads to diagonal forms for the Laplacian operator. The solution to an Helmoltz equation is the critical step in a semi-implicit time discretization that effectively filters gravity waves and stablizes long time integrations. The spectral methods offer an important advantage for the solution of this Helmholtz equation over more standard grid point (finite difference, control volume and finite element) methods.

An alternative way to evaluate the differential terms uses a 2-D (lon-lat) FFT where the latitude direction is taken on great circles. The paper [W.F. Spotz and Swarztrauber 1998] studies Merilee's pseudospectral model and finds a FFT based algorithm that exactly matches results from a spherical harmonic transform model. The evaluation of derivatives can be accomplished with $I^2 \log I$ operations. The argument is made that, with the addition of a fast projection algorithm to ensure all modes remain in the spherical harmonic space, the FFT2D based model is the fastest spherical harmonic algorithm. This method was coupled with two and three level semi-Lagrangian time-stepping methods for a very efficient, high order solution algorithm [Layton and Spotz 2002]. The key new element that makes these algorithms attractive is the existence of a fast spherical harmonic projection which stabilizes the Fourier method.

3.3 Fast Spherical Harmonic Projection Algorithms

The projection of a function onto the spherical harmonic modes (analysis) followed by the inverse transformation back to grid point space is a filter of the original function. If there is no need to use the spectral coefficients to evaluate derivatives, then the forward and inverse transforms can be combined algebraically to produce an explicit projection operator. Several algorithms have been studied in [Spotz and Swarztrauber 2001] and compared in single processor and parallel implementations.

A more interesting method for evaluating the projection utilizes a Fast Multipole Method. The multipole projection was proposed as an approximate, fast projection method. It is based on an application of the Christoffel-Darboux formula relating the sum of products of associated Legendre functions. Several variants have been developed including [Holmes et al. 1996; Yarvin and Rokhlin 1998]. The study in [Spotz and Swarztrauber 2001] proposes several other algorithms and concludes that the Weighted Orthogonal Complement(WOC) algorithm developed in [Swarztrauber and Spotz 2000] is the most efficient in terms of operation count, cache utilization and overall performance for the resolutions studied. This algorithm for Legendre projection is faster than the multipole projection method in its known

implementations to date.

4. MATLAB SPHERICAL HARMONIC CLASSES

The MATLAB class structures are defined in a subdirectory usually called *my_classes*. Following ESMF style conventions [Collins et al. 2005], the methods and data structures for the spectral transform fall into two classes: the *@gauss_grid* and the *@spectral_field*. The *gauss_grid* class defines the evenly spaced longitudinal points and the latitudinal points corresponding to the Gauss points of the spherical grid. In addition, the geometric information, such as the radius of the sphere and the values of the associated Legendre functions at the points are stored. The standard *get*,*set* and *display* methods are provided. In the *private* methods are *grule*, that calculates the Gauss points and weights, and *shtraninit*, that computes the $\mathbf{P}_m$ matrix to initialize the spherical harmonic transform. The initialization of a *@gauss_grid* object, specifies the number of latitudinal points, nj. A triangular truncation of the spectral coefficients is assumed, so for example, if $nj = 32$, the spectral transform is initialized for T21 spectral fields.

Spherical harmonic transforms act on fields that are defined on spherical Gaussian grids. The *@spectral_field* class defines such a field. The analysis method, that computes spectral coefficients from grid point values of a field, is *shtrana*. The inverse transform (synthesis), that computes grid point values from given spectral coefficients, is called *shtrans*. The Legendre transforms (forward and backward) are *private* methods. Algorithmic options such as discussed above may be included here, but the default is the open loop formulation. The MATLAB complex FFTW is used by the *shtrana(s)* methods to compute the real transforms. In addition to the basic *set, get* and *display* methods, a variety of differential operators are implemented. The method library that uses spherical transforms to discretize differential operators in spherical geometry includes:

div - the spherical (horizontal) divergence operator, $\nabla\cdot$	$\frac{1}{a}\left[\frac{1}{1-\mu^2}\frac{\partial U}{\partial\lambda}+\frac{\partial V}{\partial\mu}\right]$
curl - the curl operator, $k\cdot\nabla\times$	$\frac{1}{a}\left[\frac{1}{1-\mu^2}\frac{\partial V}{\partial\lambda}-\frac{\partial U}{\partial\mu}\right]$
grad - the spherical (horizontal) gradient operator, ∇	$\left(\frac{1}{a}\frac{\partial}{\partial\lambda},\frac{1-\mu^2}{a}\frac{\partial}{\partial\mu}\right)$
del2 - the spherical (horizontal) Laplacian operator, ∇^2	$\frac{1}{a^2}\left[\frac{1}{1-\mu^2}\frac{\partial^2}{\partial\lambda^2}+\frac{\partial}{\partial\mu}\left((1-\mu^2)\frac{\partial}{\partial\mu}\right)\right]$
del2inv - the inverse Laplacian operator (Laplace equation solution)	
helmholtz - the solution of Helmholtz equation $k^2g+\nabla^2g=f$.	
UVinv - the inversion operator for the vorticity, divergence and velocity	

To calculate the divergence of (U,V) the spectral coefficients are summed with the derivative, $H_n^m = (1-\mu^2)\frac{dP_n^m}{d\mu}$ of the spherical harmonic, according to the formula,

$$div(U,V)_n^m = \frac{1}{a}\sum_{j=1}^{J}\left[imU^m(\mu_j)P_n^m(\mu_j) - V^m(\mu_j)H_n^m(\mu_j)\right]\frac{w_j}{(1-\mu_j^2)}. \qquad (20)$$

The U^m denotes the Fourier coefficient of the $U = u\cos\theta$ field. Note that normally multiplication by $\cos\theta$ is necessary so that the vector field is differentiable at the poles. All FFTs are computed in MATLAB using the native routines. This leads to some inefficiency due to use of complex transforms for real fields, (see Exercise 3.6 of [Trefethen 2000]).

The curl of (U, V) is calculated in spectral space from the formula

$$curl(U,V)_n^m = -\frac{1}{a}\sum_{j=1}^{J}\left[imV^m(\mu_j)P_n^m(\mu_j) + U^m(\mu_j)H_n^m(\mu_j)\right]\frac{w_j}{(1-\mu_j^2)} \tag{21}$$

The gradient operator is not invariant under choice of coordinate system. The operator is calculated in the (λ, μ) system as

$$\nabla_{(\lambda,\mu)}\phi = (\frac{1}{a}\frac{\partial\phi}{\partial\lambda}, \frac{1-\mu^2}{a}\frac{\partial\phi}{\partial\mu}) \tag{22}$$

To convert to θ-coordinates we have $\nabla_{(\lambda,\mu)} = \cos\theta\nabla_{(\lambda,\theta)}$. The computation of the components uses the Legendre synthesis with H rather than P in the summation of cofficients.

$$\left\{\frac{1}{a}\frac{\partial\phi}{\partial\lambda}\right\}_n^m = \frac{1}{a}\sum_m\sum_n im\phi_n^m P_n^m(\mu)e^{im\lambda} \tag{23}$$

and

$$\left\{\frac{1-\mu^2}{a}\frac{\partial\phi}{\partial\mu}\right\}_n^m = \frac{1}{a}\sum_m\sum_n \phi_n^m H_n^m(\mu)e^{im\lambda}. \tag{24}$$

Note that the first component computation could be done in Fourier space.

Similarly the Laplacian is calculated in spectral space using the eignfunction relationship for the operator with the spherical harmonics,

$$\nabla^2\psi_n^m = -\frac{n(n+1)}{a^2}\psi_n^m. \tag{25}$$

This relation is inverted for the solution of Laplace's equation using *del2inv*. A similar derivation applies for the Helmholtz equation.

The inversion of the (U, V) relationship with vorticity, $\xi = \mathbf{k}\cdot\nabla\times\mathbf{v}$, and divergence, $\delta = \nabla\cdot\mathbf{v}$, is given by the sums,

$$U(\lambda_i,\mu_j) = -\sum_{m=-M}^{M}\sum_{\substack{n=|m|\\ n\neq 0}}^{N(m)}\frac{a}{n(n+1)}\left[im\delta_n^m P_n^m(\mu_j) - \xi_n^m H_n^m(\mu_j)\right]e^{im\lambda_i}$$

$$V(\lambda_i,\mu_j) = -\sum_{m=-M}^{M}\sum_{\substack{n=|m|\\ n\neq 0}}^{N(m)}\frac{a}{n(n+1)}\left[im\xi_n^m P_n^m(\mu_j) + \delta_n^m H_n^m(\mu_j)\right]e^{im\lambda_i}$$

A *test* method is provided along with a rudimentary *plot* method for spectral fields as a unit test for each method of the *spectral_field* class.

Initialization of a spectral field requires a previous *gauss_grid* object. The *get* and *set* methods can be used to initialize the grid point values or the spectral coeffecients.

The following are plots from the *test* method using field values of the spherical harmonic, $Y_3^7(\lambda, \theta)$. To invoke this method from MATLAB requires that the *my_classes* subdirectory be added to the MATLAB path and that a *gauss_grid* and *spectral_field* have been initialized. For example,

```
% addpath ~/Matlab/my_classes
% G = gauss_grid('T42',64);
% f = spectral_field('test function',G);
% test(f);
```

To take maximum advantage of the matrix formulations, MATLAB will use optimized versions of the BLAS if certain environment variables are set when MATLAB is invoked. For example, an alias may be useful such as

```
alias matlab='export BLAS_VERSION=atlas_P4.so;
export LD_ASSUME_KERNEL=2.3.98;
/usr/share/linux.x86/Matlab6/bin/matlab'
```

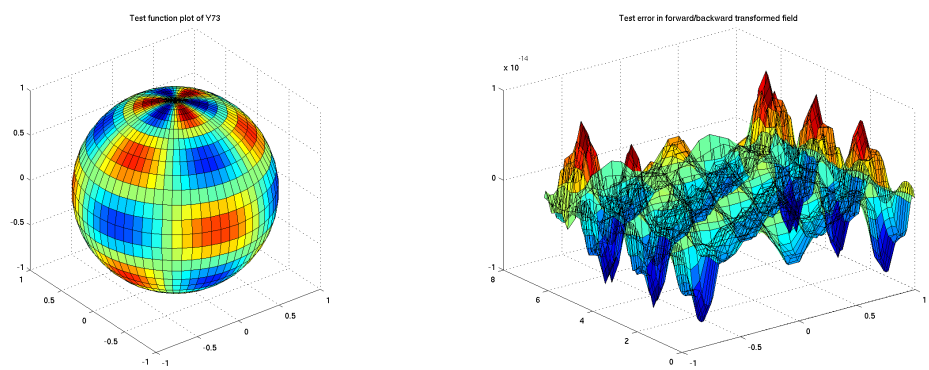


Fig. 2. MATLAB output from *test*.

Here are several of the key programs for the spherical harmonic transform as implemented in MATLAB.

5. BAROTROPIC VORTICITY EQUATION: MATLAB EXAMPLE

To demonstrate the use of the spectral transform in dynamics equations for the atmosphere, the simplest setting is a barotropic vorticity equation. The velocity is related to a horizontal stream function by $\mathbf{v} = \mathbf{k} \times \nabla\psi$. This stream function is calculated by inverting the elliptic equation

$$\nabla^2\psi = \xi, \tag{26}$$

```
function f = shtrana(f)
% SHTRANA Spectral analysis of a spectral_grid field
% Compute the spectral coefficients from the field grid
% point values using a spherical harmonic transform analysis
%————————————————————————————————
% Input:
% f - spectral_field class object
% In particular:
% f.gp - field grid point values
% f.G - field gauss_grid
% Method of spectral_field class: Sept 2005
%—-————————————-————————-
    ni = get(f.G,'ni'); nj = get(f.G,'nj');
    mm = get(f.G,'mm'); nn = get(f.G,'nn'); kk = get(f.G,'kk');
    wg = get(f.G,'wg'); P = get(f.G,'P'); gp = get(f,'gp');
%————————————————————————————————
    xf =fft(gp,ni)/ni; %note normalization for MATLAB FFT
%order ni= 2n real transform coming out as r0,r1,r2...i2,i1
%multiply Fourier coefficients by the Gauss weights
    for j=1:nj
        xf(:,j)=wg(j)*xf(:,j);
    end
    f.sc = legtranOLa(xf,nj,mm,nn,kk,P); % inverse Legendre
```

Fig. 3. MATLAB implementation of spherical harmonic transform analysis

where ξ is the (absolute) horizontal vorticity, $\xi = \mathbf{k} \cdot curl\mathbf{v}$. The governing equation of motion is given in terms of the potential vorticity

$$\frac{d\eta}{dt} = 0. \tag{27}$$

The potential vorticity is related to the absolute vorticity by $\eta = \xi + f$, where $f = 2\Omega \sin\theta$ is the Coriolis term. The potential vorticity equation is written in advective form with the material derivative

$$\frac{d\eta}{dt} \equiv \frac{\partial \eta}{\partial t} + \mathbf{v} \cdot \nabla \eta. \tag{28}$$

The time integration algorithm directly treats the material derivative operator using a semi-Lagrangian transport scheme. Particle tracking to determine "departure points" uses MATLAB *@slt_grid* class that extends the Gaussian grid with extra halo points for interpolation methods.

The barotropic vorticity solution algorithm also follows the ESMF style [Collins et al. 2005] specification of a gridded component model with a *begin, run, finalize* steps. In brief, the solution algorithm is as follows:

(1) Begin method
    (a) integration control initialization
    (b) gauss_grid initialization
    (c) slt_grid initialization

```
function f = shtrans(f)
% SHTRANA Spectral synthesis to a spectral_grid field
% Compute the grid point values from the spectral coefficients using
% a spherical harmonic transform synthesis
%-----------------------------------------
% Input:
% f - spectral_field class object
% In particular:
% f.gp - field grid point values
% f.G - field gauss_grid
% Output:
% f.sc - Output array of complex spectral coeffiecients (n,m)
% Local
% xf - matrix of Fourier coefficients ordered (m,j)
% Method of spectral_field class: Sept 2005
%---------------------------------
    ni = get(f.G,'ni'); nj = get(f.G,'nj');
    mm = get(f.G,'mm'); nn = get(f.G,'nn'); kk = get(f.G,'kk');
    wg = get(f.G,'wg'); P = get(f.G,'P');
%-----------------------------------------
    xf = legtranOLs(f.sc,nj,mm,nn,kk,P); % inverse Legendre transform
%make xf into a hermetian array for real transform back
    for m=1:ni/2
        xf(ni-m+1,:) = conj(xf(m+1,:));
    end
    f.gp =real(ifft(xf,ni))*ni; % inverse Fourier transform, note normalization
```

Fig. 4. MATLAB implementation of spherical harmonic synthesis

(d) spectral_field initialization (prognostic initial conditions)
(e) spectral_field initialization (diagonsitic initial conditions)

(2) Run method (loop until done)
(a) slt_particle tracking and departure point calcuation
(b) slt_interpolation of η at departure points
(c) slt_interpolation of right hand side at departure points
(d) update η to new time level
(e) UVinv invert diagnostic relation to get new velocity $\mathbf{v}$
(f) del2inv invert Laplace operator to get stream function ψ

(3) Finalize method

The example MATLAB program *BV.m* integrates the barotropic vorticity equation for three days starting from a Rossby-Haurwitz wave four initial condition. The following plots show the final solution using a T10 spectral truncation for the *gauss_grid*.

6. CONCLUSIONS

A set of algorithms and MATLAB classes for computing spherical harmonic transforms was described. This set of classes enables a number of algorithmic studies and

```
function x = legtranOLa(s,nj,mm,nn,kk,P)
% Compute a Legendre transform analysis
% Input:
% s - complex Fourier coeffecients ordered (m,j)
% nj = number of Gauss latitudes
% mm, nn,kk are the truncation parameters
% P - associated Legendre functions ordered (j,n,m)
% Output:
% x - matrix of spectral coefficients ordered (n,m)
%————————————————————————————
% Based on Spherical harmonic transform formulation with open loops (OL)
%—-————————————————————————
    njo2=nj/2;
    x = zeros(nn+1,mm+1);
    for m=0:mm
        j=1:njo2;
        for n=m:2:nn
            x(n+1,m+1) = x(n+1,m+1) + (s(m+1,j) + s(m+1,nj-j+1))*P(j,n+1,m+1);
        end
        for n=m+1:2:nn
            x(n+1,m+1) = x(n+1,m+1) + (s(m+1,j) - s(m+1,nj-j+1))*P(j,n+1,m+1);
        end
    end
```

Fig. 5. MATLAB implementation of Open Loop Legendre analysis

computational experiments important for the development of weather and climate models and high performance computers. The typical use of the spectral transform is illustrated using the differential operator methods of these classes to solve the barotropic vorticity equation.

The MATLAB programs (.m files) are freely available and may be obtained from `http://www.csm.ornl.gov/~bbd/SHTrans`.

ACKNOWLEDGMENTS

The authors wish to thank the DOE SciDAC program for continuing interest in the relationship of mathematics and computation to the sciences. This research is sponsored by the Office of Advanced Scientific Computing Research and the Office of Biological and Environmental Research; U.S. Department of Energy. The work was performed at the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC under Contract No. De-AC05-00OR22725.

REFERENCES

Boyd, J. 1992. Multipole expansions and pseudospectral cardinal functions: A new generalization of the fast fourier transform. *J. Comp. Phys. 103,* 1, 184–186.

Canuto, C., Hussaini, M., Quarteroni, A., and Zang, T. 1988. *Spectral Methods in Fluid Dynamics.* Springer-Verlag.

Collins, N., Theurich, G., DeLuca, C., Suarez, M., Trayanov, A., Balaji, V., Li, P., Yang,

```
function s = legtranOLs(x,nj,mm,nn,kk,P)
% Compute a Legendre transform synthesis – OPEN LOOP formulation
% Input:
% x - complex spectral coeffecients (n,m)
% nj = number of Gauss latitudes
% mm, nn,kk are the truncation parameters
% P - associated Legendre functions ordered (j,n,m)
% Output:
% s - matrix of Fourier coefficients ordered (m,j)
%————————————————————————————
% Based on Spherical harmonic transform formulation as an open loop
%—-————————————————————————
    njo2=nj/2;
    ni=2*nj;
%
    s=zeros(mm+1,nj);
%
    for m=0:mm
        for j=1:njo2
            for n = m:nn
                s(m+1,j) = s(m+1,j) + x(n+1,m+1)*P(j,n+1,m+1); %first half
            end
            for n=m:2:nn
                s(m+1,nj-j+1) = s(m+1,nj-j+1) + x(n+1,m+1)*P(j,n+1,m+1); %second even
            end
            for n=m+1:2:nn
                s(m+1,nj-j+1) = s(m+1,nj-j+1) - x(n+1,m+1)*P(j,n+1,m+1); %second odd
            end
        end
    end
```

Fig. 6. MATLAB implementation of Open Loop Legendre synthesis

W., Hill, C., and da Silva, A. 2005. Design and implementation of components in the earth system modeling framework. *Int. J. of High Perf. Comput. Appl. 19,* 3, 341–350.

D'Azevedo, E. 2004. Performance of the spherical harmonic transform on modern architectures. In *Proceedings of the SIAM Conference on Parallel Processing for Scientific Computing*. SIAM.

Drake, J., Jones, P., and Carr, G. 2005. Overview of the software desing and parallell algorithms of the ccsm. *Int. J. High Perf. Comput. Appl. 19,* 3, 177–186.

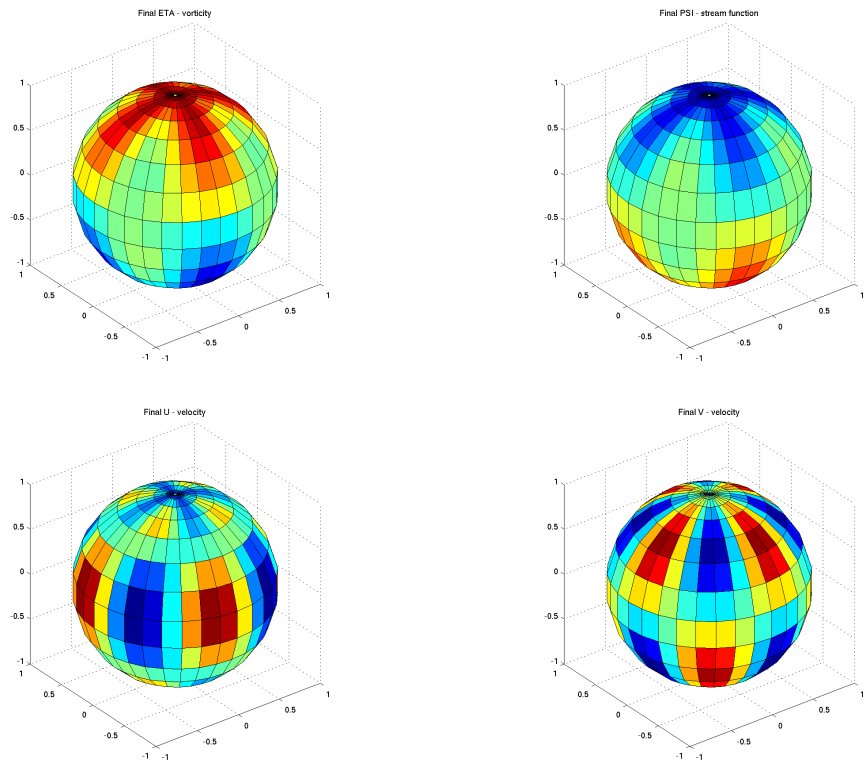
Driscoll, J. and Healy, Jr., D. 1989. Computing fourier transforms and convolutions on the 2-sphere. In *Proceedings of the 30th IEEE Symposium on Foundations of Computer Science*. IEEE, 344–349.

Foster, I. and Worley, P. 1997. Parallel algorithms for the spectral transform method. *SIAM J. Sci. Stat. Comput. 18,* 3, 806–837.

Healy, Jr., D., Rockmore, D., and Moore, S. 1996. An fft for the 2-sphere and applications. In *Proceedings ICASSP 96*. 1323–1326.

Holmes, J., Wang, Z., Drake, J., Lyon, B., and Chen, W.-T. 1996. A fast multipole transformation for global climate calculations. Tech. Rep. ORNL TM-13135.

Inda, M., Bisseling, R., and Maslen, D. 2001. On the efficient parallel computation of legendre transforms. *SIAM J. Sci. Comput. 23,* 1, 271–303.



Layton, A. T. and Spotz, W. F. 2002. A semi-lagrangian double fourier method for the shallow water equations on the sphere. *J. Comp. Phys. 189,* 1, 180–196.

Li, R.-C. 1996. An efficient implementation of a spectral transform method for solving the shallow water equations. *unpublished paper and personal communication.*

Mohlenkamp, M. 1999. A fast transform for spherical harmonics. *J. Fourier Anal. Appl. 5(2/3)*, 159–184.

Shingu, S., Takahara, H., Fuchigami, H., Yamada, M., Tsuda, Y., Ohfuchi, W., Sasaki, Y., Kobayashi, K., Hagiwara, T., Habata, S., Yokokawa, M., Itoh, H., and Otsuka, K. 2002. A 26.58 terflop global atmospheric simulation with the spectral transform method on the earth simulator. *IEEE.*

Spotz, W. F. and Swarztrauber, P. 2001. A performance comparison of associated legendre projections. *J. Comp. Phys. 168*, 339–355.

Suda, R. and Takami, M. 2002. A fast spherical harmonic transform algorithm. *Math. Comp. 7*, 703–715.

Swarztrauber, P. N. and Spotz, W. F. 2000. Generalized discrete spherical harmonic transforms. *J. Comp. Phys. 145*, 213–230.

Temperton, C. 1983. Fast mixed-radix real fourier transform. *J. Comp. Phys. 52*, 340–350.

Trefethen, L. N. 2000. *Spectral Methods in MATLAB.* SIAM.

W.F. Spotz, M. T. and Swarztrauber, P. 1998. Fast shallow-water equation solvers in latitude-longitude coordinates. *J. Comp. Phys. 145*, 432–444.

Worley, P. H. and Drake, J. 2005. Software design for performance portability in the community atmosphere model. *Int. J. High Perf. Comput. Appl. 19,* 3, 187–201.

Yarvin, N. and Rokhlin, V. 1998. A generalized one-dimensional fast multipole method with applications to filtering of spherical harmonics. *J. Comp. Phys. 159,* 2, 594–609.

Received February 200?; revised August 200? and November 200?; accepted December 200?.