Journal of Computational Physics **168,** 339–355 (2001)
doi:10.1006/jcph.2001.6700, available online at http://www.idealibrary.com on IDEAL®

A Performance Comparison of Associated Legendre Projections

William F. Spotz[1] and Paul N. Swarztrauber[2]

National Center for Atmospheric Research,[3] P.O. Box 3000, Boulder, Colorado 80307
E-mail: spotz@ucar.edu, pauls@ucar.edu

Received April 4, 2000; revised December 11, 2000

The associated Legendre projection provides a means for accelerating the dynamical core of a global weather or climate model. Therefore, the goal is to determine the fastest possible projection algorithm, of which this paper compares six: the standard method, which computes the projection using a forward and backward associated Legendre transform; the direct method, which uses a single projection matrix when this approach results in fewer operations; the fast multipole method; the weighted orthogonal complement method; the seminaive method; and a projection based on transforms proposed by Mohlenkamp. Timing results indicate that all the projections behave like $\mathcal{O}(N^3)$ algorithms up to at least $N = 200$ spectral truncation. For this range of resolutions, the weighted orthogonal complement has the lowest operation count, best cache utilization, and best overall timings.

Key Words: spherical harmonics; associated Legendre functions; fast multipole method; weighted orthogonal complement.

1. INTRODUCTION

The spectral transform method (STM) is a popular method for approximating the dynamics of the global atmosphere that requires $\mathcal{O}(N^3)$ operations per time step, where N is the spectral truncation of the model. Most other calculations in climate modeling and weather forecasting are $\mathcal{O}(N^2)$, with the exception of fast Fourier transforms (FFT), which are $\mathcal{O}(N^2 \log N)$. For this reason, the search for a comparable fast associated Legendre transform (the expensive component of a spherical harmonic transform) has been conducted for quite some time. Some success has been achieved [1–3], but at break-even resolutions which are too large for current global weather and climate applications.

[1] Supported by the Division of Mathematical Sciences at the National Science Foundation.

[2] Supported in part by the DOE and UCAR Cooperative Agreement for the Climate Change Prediction Program under Cooperative Agreement No. DE-FC03-97ER62402.

[3] NCAR is sponsored by the National Science Foundation.

0021-9991/01 $35.00

Recently [4], the authors solved the spherical shallow water equations (a 2-D subset of the global atmospheric dynamical equations) using an FFT-based pseudospectral method. This approach is well known to be unstable without the explicit projection of the solution onto the space of harmonic functions. With projection, the method yields (to within machine accuracy) the same results as the STM. There are two important conclusions of this work: (1) the required number of associated Legendre transforms is reduced from nine (or more) to six and (2) the associated Legendre projection operator provides a new avenue of approach in the search for a fast algorithm. Indeed, Jakob and Alpert [5] have published an asymptotically $\mathcal{O}(N^2)$ associated Legendre projection (or filter, in their terminology) that uses the fast multipole method (FMM). This was subsequently improved by Yarvin and Rokhlin [6]. More recently, the authors [7] have developed a projection algorithm that reduces the memory requirement from $\mathcal{O}(N^3)$ to $\mathcal{O}(N^2)$ and requires half the number of operations compared to standard associated Legendre transforms. For purposes of completeness, we will also consider projection algorithms composed of the latest attempts at fast associated Legendre transforms.

Here, we compare implementations of four projection algorithms to determine (a) the actual performance in terms of N for the range of resolutions used by current atmospheric models and (b) the relative performance of the algorithms. The organization of the paper is as follows: Section 2 covers background information related to the associated Legendre projection, Section 3 describes the various projection algorithms to be compared, Section 4 presents serial timing comparisons of these algorithms, Section 5 discusses parallel implementations of the projection algorithms, and conclusions are drawn in Section 6.

2. BACKGROUND

2.1. *Spherical Harmonic Transforms*

A smooth spherical function $f(\lambda, \theta)$, where $0 \le \lambda < 2\pi$ is the longitude and $-\pi/2 \le \theta \le \pi/2$ is the latitude, can be approximated to spectral order N by an expansion of spherical harmonics given by

$$f(\lambda, \theta) = \sum_{m=0}^{N} \sum_{n=m}^{N} \bar{P}_n^m(\theta)(a_{m,n} \cos m\lambda + b_{m,n} \sin m\lambda), \tag{1}$$

where $\bar{P}_n^m(\theta)$ are the normalized associated Legendre functions

$$\bar{P}_n^m(\theta) = \frac{1}{2^n n!}\left[\frac{2n+1}{2}\frac{(n-m)!}{(n+m)!}\right]^{1/2} \cos m\theta \frac{d^{n+m}}{dx^{n+m}}(x^2-1)^n,$$

with $x = \sin\theta$.

The computation of $a_{m,n}$ and $b_{m,n}$ (or harmonic analysis) in (1) consists of two phases. First, a Fourier transform yields the Fourier functions

$$a_m(\theta) = \int_0^{2\pi} f(\lambda, \theta) \cos m\lambda \, d\lambda, \tag{2a}$$

$$b_m(\theta) = \int_0^{2\pi} f(\lambda, \theta) \sin m\lambda \, d\lambda, \tag{2b}$$

followed by an associated Legendre transform to obtain

$$a_{m,n} = \int_{-\pi/2}^{\pi/2} a_m(\theta) \bar{P}_n^m(\theta) \cos\theta \, d\theta. \tag{3}$$

An analogous equation provides $b_{m,n}$ in terms of $b_m(\theta)$. All subsequent equations for $b_{m,n}$ and $b_m(\theta)$ will not be written, because they have the same form as the corresponding equations for $a_{m,n}$ and $a_m(\theta)$.

The computation of $f(\lambda, \theta)$ (or harmonic synthesis) in (1) also occurs in two phases. First, compute

$$a_m(\theta) = \sum_{n=m}^{N} a_{m,n} \bar{P}_n^m(\theta), \tag{4}$$

followed by

$$f(\lambda, \theta) = \sum_{m=0}^{N} (a_m(\theta) \cos m\lambda + b_m(\theta) \sin m\lambda). \tag{5}$$

Equations (2a) and (2b) are forward Fourier transforms and (5) is a backward Fourier transform. Neither are considered in this work because fast algorithms exist for these operations. Equations (3) and (4) are forward and backward associated Legendre transforms, respectively, which do not in general have fast algorithms and are consequently the focus of the work presented here.

2.2. *Associated Legendre Transforms on Gauss Grids*

We discretize $f(\lambda, \theta)$ by choosing λ_i, $i = 1 \ldots N_{\text{lon}}$ and θ_j, $j = 1 \ldots N_{\text{lat}}$. If λ_i is equally spaced, we can utilize fast Fourier transforms for (2a), (2b), and (5), making this the obvious choice. The θ_j are most often chosen such that $x_j = \sin\theta_j$, where the x_j are the Gauss–Legendre quadrature points on $[-1, 1]$. Thus the discrete form of (3) for the Gauss grid becomes

$$a_{m,n} = \sum_{j=1}^{N_{\text{lat}}} a_m(\theta_j) \bar{P}_n^m(\theta_j) w_j, \tag{6}$$

where w_j are the Gauss weights on $[-1, 1]$. In matrix notation, the backward and forward associated Legendre transforms, (4) and (6) respectively, can be written

$$\mathbf{a}_m^F = \mathbf{P}_m \mathbf{a}_m^S, \tag{7}$$

$$\mathbf{a}_m^S = \mathbf{P}_m^T \mathbf{W} \mathbf{a}_m^F, \tag{8}$$

where $\mathbf{a}_m^F = [a_m(\theta_1) \ldots a_m(\theta_{N_{\text{lat}}})]^T$ are the discrete Fourier functions, $\mathbf{a}_m^S = [a_{m,m} \ldots a_{m,N}]^T$ are the spherical harmonic coefficients, $\mathbf{W}$ is the $N_{\text{lat}} \times N_{\text{lat}}$ diagonal weight matrix such that $(\mathbf{W})_{j,j} = w_j$, and $\mathbf{P}_m$ is the $N_{\text{lat}} \times (N - m + 1)$ backward transform matrix given by

$$\mathbf{P}_m = \begin{bmatrix} \bar{P}_m^m(\theta_1) & \cdots & \bar{P}_N^m(\theta_1) \\ \vdots & & \vdots \\ \bar{P}_m^m(\theta_{N_{\text{lat}}}) & \cdots & \bar{P}_N^m(\theta_{N_{\text{lat}}}) \end{bmatrix}.$$

Note that we have two distinct upper summation bounds, N and N_{lat}. A uniform resolution of waves on the sphere is achieved via a triangular truncation or equivalently by the restriction $N < \min(N_{\text{lat}}, N_{\text{lon}}/2)$. Often, it is desirable for N to be even smaller. As an example, for nonlinear PDEs with quadratic terms, the 2/3 rule $N = [\min(2N_{\text{lat}}, N_{\text{lon}}) - 1]/3$ is used to prevent aliasing errors. Other truncations are possible as well. For this reason, in the future we will substitute N_{tr} for N to indicate a (possibly) truncated expansion.

2.3. *Associated Legendre Transforms on Arbitrary Grids*

Machenhauer [8] extended discrete associated Legendre transforms to equally spaced grids θ_j in such a way that they could be computed in the same number of operations as the Gaussian distributions. In [7], this capability was extended to arbitrary grids as follows. The backward transform matrix $\mathbf{P}_m$ is defined as before, but the $N_{\text{lat}} \times N_{\text{lat}}$ weight matrix is no longer diagonal and is different for even and odd m. For even m, $\mathbf{W}_0 = (\mathbf{P}_0\mathbf{P}_0^T)^{-1}$, where $\mathbf{P}_0$ is constructed for $N_{\text{tr}} = N_{\text{lat}} - 1$. For odd m, $\mathbf{W}_1 = (\tilde{\mathbf{P}}_1\tilde{\mathbf{P}}_1^T)^{-1}$, where $\tilde{\mathbf{P}}_1$ is a rank-one augmentation of $\mathbf{P}_1$; see [7] for details. If we define and store the forward transform matrix

$$\mathbf{Z}_m^T = \mathbf{P}_m^T\mathbf{W}_\ell, \tag{9}$$

where $\ell = 0$ (1) if m is even (odd), then the Gaussian and arbitrary grid transforms can be computed in the same number of operations, at the expense of doubling the storage requirements. Thus the forward transform (8) becomes

$$\mathbf{a}_m^S = \mathbf{Z}_m^T\mathbf{a}_m^F, \tag{10}$$

and the backward transform (7) is equally valid on a Gauss or arbitrary grid. Note that the forward transform matrix defined in (9) is equivalent to the forward transform matrix defined by Machenhauer [8] for the case when θ_j is an equally spaced grid.

2.4. *The Associated Legendre Projection*

The vector of discrete Fourier functions $\mathbf{a}_m^F$ has constant length N_{lat}, while the vector of spherical harmonic coefficients $\mathbf{a}_m^S$ has the usually shorter (never longer) length $N_{\text{tr}} - m + 1$. Thus the computation of $\mathbf{a}_m^S$ in (10) followed by the recomputation of $\mathbf{a}_m^F$ in (7) will usually result in the loss of degrees of freedom for all cases except $m = 0$ when $N_{\text{tr}} = N_{\text{lat}} - 1$. Specifically, this operation will perform a weighted least squares projection [9] of the original $\mathbf{a}_m^F$ onto the space of the associated Legendre functions with triangular truncation N_{tr}, and can be written

$$\tilde{\mathbf{a}}_m^F = \mathbf{P}_m\mathbf{Z}_m^T\mathbf{a}_m^F, \tag{11}$$

$$= \mathbf{F}_m\mathbf{a}_m^F, \tag{12}$$

where $\mathbf{F}_m = \mathbf{P}_m\mathbf{Z}_m^T$ is the $N_{\text{lat}} \times N_{\text{lat}}$ associated Legendre projection matrix for zonal wave number m, and $\tilde{\mathbf{a}}_m^F$ is the vector of projected discrete Fourier functions.

This projection operator is useful [4, 10] for stabilizing spherical shallow water models with nonspherical harmonic numerical approximations. In its "naive" form (11), the projection costs the same as two associated Legendre transforms. A three-variable shallow water model requires the equivalent of six associated Legendre transforms for stabilization,

as compared to nine transforms for the most efficient spectral transform model [11]. This alone represents a speedup that is compounded using the accelerated projection algorithms compared in this paper.

3. PROJECTION ALGORITHMS

The application for which the following projection algorithms are intended is an atmospheric model with equally spaced latitude points. Because the point distribution is symmetric about the equator, and the associated Legendre functions are either symmetric or antisymmetric about the equator, all the algorithms take advantage of the well-known symmetry optimization. Namely, the $\mathbf{a}_m^F$ are decomposed into even and odd components before the computations, resulting in two sets of computations with a quarter of the original number of operations, giving an overall savings of a factor of two.

3.1. *The Standard Algorithm*

The standard algorithm explicitly computes $\mathbf{a}_m^S$ from (10) and then $\tilde{\mathbf{a}}_m^F$ from (7). The theoretical mult/add operation count (and the standard against which we will measure other projections) for this algorithm is

$$C_{\text{Std}}(m) = N_{\text{lat}}(N_{\text{tr}} - m + 1),$$

which is plotted in Fig. 1 and labeled "Std." Here we present the operation counts for three projections, as functions of m, to clearly reveal contributions to the total operation counts.

3.2. *The Direct Algorithm*

The direct algorithm was proposed by Yarvin and Rokhlin [6] as a reference comparison for the FMM. It computes the projection two different ways: by direct computation of (11) for large m, or of (12) for small m. The crossover occurs at $m = N_{\text{tr}} + 1 - N_{\text{lat}}/2$. The resulting operation count for the direct method is

$$C_{\text{D}}(m) = \begin{cases} N_{\text{lat}}^2/2, & m \le N_{\text{tr}} + 1 - N_{\text{lat}}/2, \\ N_{\text{lat}}(N_{\text{tr}} - m + 1), & m > N_{\text{tr}} + 1 - N_{\text{lat}}/2. \end{cases} \tag{13}$$

The expression $C_{\text{D}}(m)$ is plotted in Fig. 1 and labeled "D."

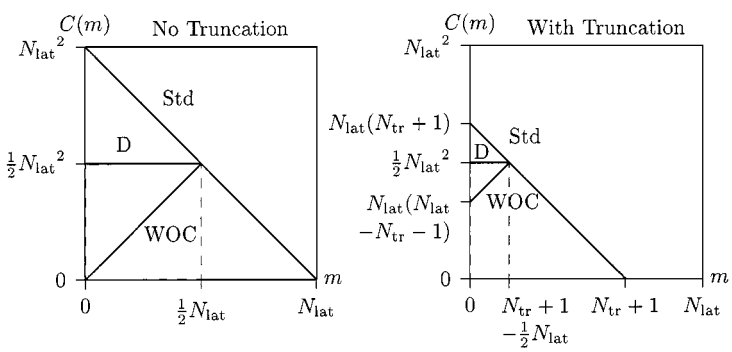

FIG. 1. Operation counts for $\mathcal{O}(N^3)$ algorithms. "Std" refers to the standard algorithm, "D" to the direct method, and "WOC" to the weighted orthogonal complement.

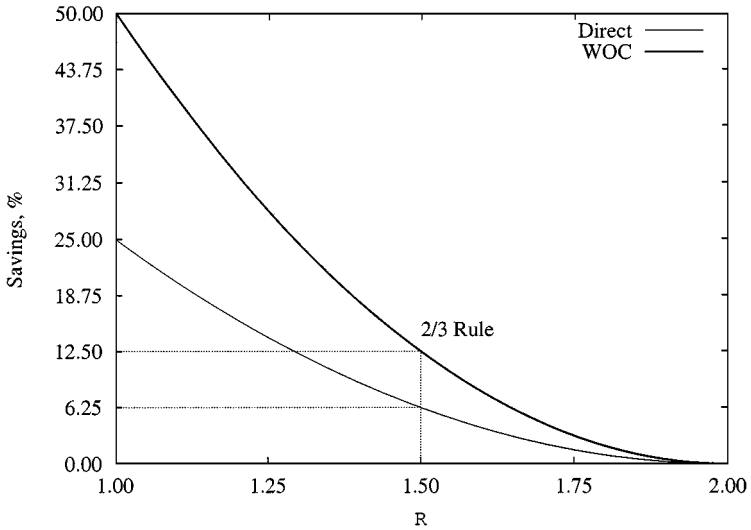

FIG. 2. Theoretical savings curves, relative to the standard algorithm as a function of $R = N_{\text{lat}}/(N_{\text{tr}} + 1)$.

We can estimate the theoretical savings of the direct algorithm relative to the standard algorithm by

$$S_{\text{D}} = 1 - \frac{2\int_0^{N_{\text{tr}}} C_{\text{D}}(m)\,dm}{2\int_0^{N_{\text{tr}}} C_{\text{Std}}(m)\,dm} = \left(1 - \frac{R}{2}\right)^2, \tag{14}$$

where R is the ratio $R = N_{\text{lat}}/(N_{\text{tr}} + 1)$. Specifically, at $R = 1$ (no truncation), the savings is 25%, and at $R = 1.5$ (the 2/3 rule), the savings is 6.25%. The 2s preceding the integrals in (14) are the result of computing both $\tilde{\mathbf{a}}_m^F$ and $\tilde{\mathbf{b}}_m^F$ for every m. Equation (14) is plotted in Fig. 2, as a comparison of algorithm savings versus the standard algorithm.

The direct algorithm as described in [6], and implemented in the results section, contains an additional optimization by noting that $\bar{P}_n^m(\theta_j)$ is zero to absolute machine accuracy in a neighborhood of the poles that increases with increasing m. The effect, not included in (13) and (14) nor Figs. 1 and 2, is to provide a savings of approximately 10%. We will refer to this as the pole optimization. Note that it would be possible to apply the pole optimization to the standard algorithm, but since production codes for spectral global weather or climate models do not generally employ this optimization, not including it with the standard algorithm provides a more meaningful comparison.

3.3. *The Fast Multipole Algorithm*

Jakob-Chien and Alpert [5] proposed an associated Legendre projection which takes advantage of the fast multipole method, resulting in an asymptotically $\mathcal{O}(N_{\text{lat}}^2)$ algorithm. They observe from (4) and (6) that given discrete Fourier functions on a Gauss grid, the projected Fourier function can be computed at any θ by

$$\begin{aligned}\tilde{a}_m(\theta) &= \sum_{n=m}^{N_{\text{tr}}} \sum_{j=1}^{N_{\text{lat}}} \bar{P}_n^m(\theta_j)\bar{P}_n^m(\theta) w_j a_m(\theta_j) \\ &= \sum_{j=1}^{N_{\text{lat}}} w_j a_m(\theta_j) \sum_{n=m}^{N_{\text{tr}}} \bar{P}_n^m(\theta_j)\bar{P}_n^m(\theta),\end{aligned}$$

to which the Christoffel–Darboux formula can be applied to give

$$\frac{\tilde{a}_m(\theta)}{\varepsilon^m_{N_{\rm tr}+1}} = \bar{P}^m_{N_{\rm tr}+1}(\theta)\sum_{j=1}^{N_{\rm lat}} \frac{w_j a_m(\theta_j)\bar{P}^m_{N_{\rm tr}}(\theta_j)}{\sin\theta - \sin\theta_j} - \bar{P}^m_{N_{\rm tr}}(\theta)\sum_{j=1}^{N_{\rm lat}} \frac{w_j a_m(\theta_j)\bar{P}^m_{N_{\rm tr}+1}(\theta_j)}{\sin\theta - \sin\theta_j}, \quad (15)$$

where

$$\varepsilon^m_n = \sqrt{(n^2-m^2)/(4n^2-1)}.$$

For $\theta = \theta_j$ the quotient is evaluated using l'Hôpital's rule. The right-hand side of (15) can be evaluated for $N_{\rm lat}$ different θs for a given m with two applications of the FMM in $\mathcal{O}(N_{\rm lat})$ operations. A complete projection for all m is thus $\mathcal{O}(N_{\rm lat}N_{\rm tr})$.

Application for a Gauss grid is obvious: simply choose $\theta = \theta_j$, where θ_j are Gauss nodes in latitude for $j = 1 \ldots N_{\rm lat}$. For an arbitrary grid, function values on the grid can first be interpolated to a Gauss grid with a single application of the FMM [12] before (15) is computed. Our experiments indicate that this interpolation adds less than 5% overhead to the calculation compared to a Gauss grid computation.

Yarvin and Rokhlin [6] improved upon the FMM approach, and their implementation is compared in the current work. In [6], it is observed that Eq. (11) is still the fastest algorithm for large m, and although it is utilized in their fast multipole algorithm, they do not mention at what m the switch is made.

The operation count for the FMM can be approximated by

$$C_{\rm FMM}(m) \approx K N_{\rm lat},$$

where K is a constant and depends in part on some implementation choices such as desired accuracy (we chose for our experiments an accuracy of 10^{-6}). Since the vertical axes in Fig. 1 are scaled by $N^2_{\rm lat}$, $C_{\rm FMM}(m)$, does not have a convenient representation on the plot. As an $\mathcal{O}(N^2_{\rm lat})$ algorithm with significant overhead, the resolution at which its experimental timings break even with the other algorithms is of interest.

3.4. *The Weighted Orthogonal Complement Algorithm*

The weighted orthogonal complement (WOC) projection is developed in [7] for non-truncated expansions. Here we review that algorithm and develop the truncated version. In [7], the $N_{\rm lat} \times N_{\rm lat}$ weighted orthogonal complement matrices $\mathbf{Q}^\ell$ and their inverses $\mathbf{R}^\ell$ are defined for $\ell = 0$ and 1. These in turn define the $N_{\rm lat} \times (N_{\rm lat} - m)$ matrices $\mathbf{Q}^\ell_m$ and $\mathbf{R}^\ell_m$ which are obtained by deleting the first m columns from $\mathbf{Q}^\ell$ and $\mathbf{R}^\ell$, respectively. It is demonstrated that the projection is given by

$$\mathbf{F}_m = \mathbf{Q}^\ell_m \left(\mathbf{R}^\ell_m\right)^T, \qquad \ell = \begin{cases} 0, & m \text{ even}, \\ 1, & m \text{ odd}, \end{cases} \quad (16)$$

which provides an alternative to (11) with the same operation count, but which requires $\mathcal{O}(N^2_{\rm lat})$ storage rather than $\mathcal{O}(N_{\rm lat}N^2_{\rm tr})$. Note that because [7] assumes $N_{\rm tr} = N_{\rm lat} - 1$, $\mathbf{Q}^0_0 = \mathbf{Q}^0$ and $\mathbf{R}^0_0 = \mathbf{R}^0$.

We can use $\mathbf{Q}^\ell$ and $\mathbf{R}^\ell$ to improve the operation count of the projection. Define $N_{\text{lat}} \times m$ matrices $\bar{\mathbf{Q}}_m^\ell$ and $\bar{\mathbf{R}}_m^\ell$ as the first m columns of $\mathbf{Q}^\ell$ and $\mathbf{R}^\ell$ so that

$$\mathbf{Q}^\ell = \left[\bar{\mathbf{Q}}_m^\ell, \mathbf{Q}_m^\ell\right],$$
$$\mathbf{R}^\ell = \left[\bar{\mathbf{R}}_m^\ell, \mathbf{R}_m^\ell\right].$$

Thus $\bar{\mathbf{Q}}_m^\ell$ and $\bar{\mathbf{R}}_m^\ell$ have m columns, or equivalently the size of these matrices increases with increasing m. The sizes of matrices $\mathbf{Q}_m^\ell$ and $\mathbf{R}_m^\ell$ behave in the opposite manner. It follows for the untruncated case that

$$\mathbf{I} = \mathbf{F}_0 = \mathbf{Q}^\ell(\mathbf{R}^\ell)^T = \bar{\mathbf{Q}}_m^\ell\left(\bar{\mathbf{R}}_m^\ell\right)^T + \mathbf{Q}_m^\ell\left(\mathbf{R}_m^\ell\right)^T,$$

for any m. Equivalently, $\mathbf{I} = \mathbf{G}_m + \mathbf{F}_m$, where $\mathbf{G}_m = \bar{\mathbf{Q}}_m^\ell(\bar{\mathbf{R}}_m^\ell)^T$, giving

$$\tilde{\mathbf{a}}_m^F = \left[\mathbf{I} - \bar{\mathbf{Q}}_m^\ell\left(\bar{\mathbf{R}}_m^\ell\right)^T\right]\mathbf{a}_m^F. \tag{17}$$

Thus the projection for a given m can be computed with $N_{\text{lat}} \times m$ mult/adds, which is a significant improvement over (16) for $m < N_{\text{lat}}/2$.

To extend the WOC to truncated projections, we must alter our definitions of the orthogonal complement matrices. Namely, the truncated transform matrices $\mathbf{Q}_m^\ell$ and $\mathbf{R}_m^\ell$ are now obtained by deleting the first m columns from $\mathbf{Q}^\ell$ and $\mathbf{R}^\ell$ *as well as* the last $N_{\text{lat}} - N_{\text{tr}} - 1$ columns, giving them dimension $N_{\text{lat}} \times (N_{\text{tr}} - m + 1)$. Thus $\mathbf{Q}_0^0 \neq \mathbf{Q}^0$ and $\mathbf{R}_0^0 \neq \mathbf{R}^0$ unless $N_{\text{tr}} = N_{\text{lat}} - 1$. These last columns become part of $\bar{\mathbf{Q}}_m^\ell$ and $\bar{\mathbf{R}}_m^\ell$, which now have dimension $N_{\text{lat}} \times (N_{\text{lat}} - N_{\text{tr}} + m - 1)$, and thus the truncation space always resides in these matrices. By changing the matrices in this way, we decrease the number of operations required by (16) for a given m and increase the number of operations required by (17). The WOC operation count is

$$C_{\text{WOC}}(m) = \begin{cases} N_{\text{lat}}(N_{\text{lat}} - N_{\text{tr}} + m - 1), & m \leq N_{\text{tr}} + 1 - N_{\text{lat}}/2, \\ N_{\text{lat}}(N_{\text{tr}} - m + 1), & m > N_{\text{tr}} + 1 - N_{\text{lat}}/2. \end{cases}$$

An estimate of the theoretical savings of the WOC algorithm relative to the standard algorithm is

$$S_{\text{WOC}} = 1 - \frac{2\int_0^{N_{\text{tr}}} C_{\text{WOC}}(m)\,dm}{2\int_0^{N_{\text{tr}}} C_{\text{Std}}(m)\,dm} = 2\left(1 - \frac{R}{2}\right)^2, \tag{18}$$

where R is again the ratio $R = N_{\text{lat}}/(N_{\text{tr}} + 1)$. At $R = 1$ (no truncation), the savings is 50%, and at $R = 1.5$ (the 2/3 rule), the savings is 12.5%. An additional savings is obtained by using the pole optimization described at the end of Section 3.2.

Figure 1 is an illustration of the operation counts for the three $\mathcal{O}(N^3)$ algorithms with and without truncation. The truncated plot is simply a leftward shift of the untruncated plot with the left-most operations eliminated. It also illustrates that there is still no direct algorithm with a lower operation count than the standard approach for $m > N_{\text{tr}} + 1 - N_{\text{lat}}/2$.

Figure 2 is a plot of Eqs. (14) and (18) to illustrate the relative savings of the direct and WOC algorithms with respect to the standard algorithm.

3.5. *The Seminaive Algorithm*

The seminaive algorithm is based upon forward and backward seminaive transforms, which are the standards against which Healy *et al.* [2] compare their transforms. A forward seminaive transform consists of computing $\mathbf{a}_m^C$, which is the discrete cosine transform of $\mathbf{a}_m^F$. Using ideas inspired by Dilts [13], the $\mathbf{a}_m^S$ are computed directly from $\mathbf{a}_m^C$ using dense matrices. The transform matrices converting the $\mathbf{a}_m^S$ back to $\tilde{\mathbf{a}}_m^C$ contain zeroes which can be exploited for a 33.3% savings. Thus the computation of the projected cosine coefficients can be done at a savings of roughly 16.7% compared to the standard algorithm. To this you must add the overhead of the discrete cosine transforms in order to start with $\mathbf{a}_m^F$ and end with $\tilde{\mathbf{a}}_m^F$.

We used the software package SpharmonicKit 2.5 (A.1) to obtain timing results for the seminaive algorithm, which has certain restrictions. Namely, the package requires that $N_{\text{lat}} = 2N_{\text{tr}}$, which we compare to the untruncated algorithms, leaving no suitable option for comparing against 2/3 rule algorithms. SpharmonicKit contains additional transform routines, namely the Driscoll–Healy midpoint algorithm and the hybrid algorithm. These are known to be slower than the seminaive algorithm at the resolutions of interest, so their development was not pursued.

3.6. *The Mohlenkamp Algorithm*

Mohlenkamp [3] has proposed associated Legendre transforms with computational complexities $\mathcal{O}(N^{5/2}\log N)$ and $\mathcal{O}(N^2(\log N)^2)$. These rates are achieved by partitioning latitude space and using local trigonometric expansions. While the $\mathcal{O}(N^2(\log N)^2)$ algorithm is asymptotically faster, the $\mathcal{O}(N^{5/2}\log N)$ algorithm performs better at the resolutions of interest and is the one compared here. Mohlenkamp has made publicly available a library (A.2) called FTSH 1.1 (Fast Transforms for Spherical Harmonics), from which we constructed a projection algorithm for timing comparisons. These routines allow the accuracy to be specified; we chose 10^{-6} to be consistent with the fast multipole projection.

4. SERIAL TIMING RESULTS

Although the discrete harmonic projection has been generalized to arbitrary point distributions, for our experiments we have chosen (when the given algorithm allows it) an unshifted equally spaced distribution which includes the poles, because the majority of global atmospheric models that can take advantage of the projection are formulated on equispaced grids. For this distribution, an equiangular grid is given by $N_{\text{lat}} = N_{\text{lon}}/2 + 1$. Thus for the untruncated projection, $N_{\text{tr}} = \min(N_{\text{lat}}, N_{\text{lon}}/2) - 1 = N_{\text{lon}}/2 - 1 = N_{\text{lat}} - 2$, and for the 2/3 rule projection, $N_{\text{tr}} = [\min(2N_{\text{lat}}, N_{\text{lon}}) - 1]/3 = (N_{\text{lon}} - 1)/3 = 2N_{\text{lat}}/3 - 1$.

We begin with an attempt to obtain timing results which mirror the theoretical operation count plots of Fig. 1 for a given N_{lat} with no truncation. As Fig. 3 indicates, factors other than operation count have an impact on timings of the standard algorithm, in this case for $N_{\text{lat}} = 65$. This figure represents the results of three nearly identical codes which perform an associated Legendre projection via (11), (16), and (17), respectively. The only optimization included is the symmetry reduction and thus this plot does not represent results for complete algorithms. The codes differ only in the calling arguments to `dgemv`, the double precision basic linear algebra subprogram (BLAS) routine (A.3) which performs a single matrix–vector product. As the first two lines represent algorithms with identical operation

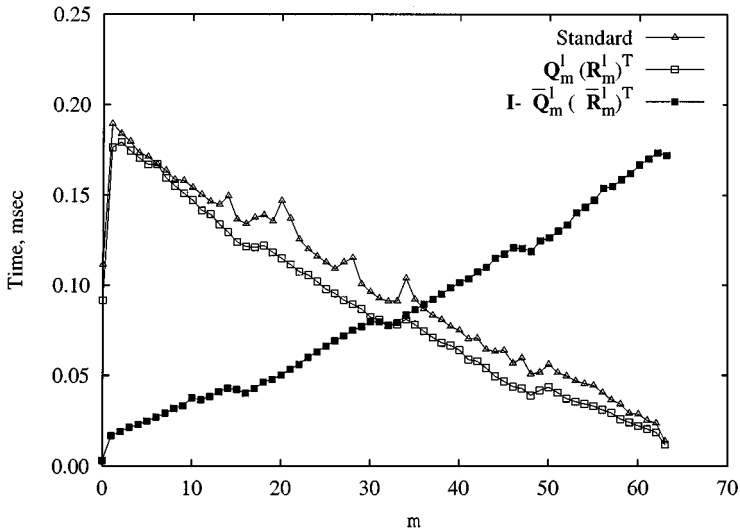

FIG. 3. Standard and weighted orthogonal complement timings as a function of m for the untruncated projection at $N_{\text{lat}} = 65$.

counts, we can conclude that the difference in timings is due to memory and hardware effects. The standard algorithm requires that a new $\mathbf{P}_m$ and $\mathbf{Z}_m^T$ be loaded into cache for every m. The other two routines, however, can reuse the $\mathbf{Q}^\ell$ and $\mathbf{R}^\ell$ once they have been loaded. This strongly suggests that cache reutilization accounts for the improved performance of the WOC algorithms over the standard algorithm. Obviously, this performance gain is resolution and hardware specific; different sizes and levels of cache can have a great effect. In this case, results are on a Sun Ultra 60 processor with 4MB cache and the standard algorithm incurs approximately 15% overhead. Experiments at other resolutions indicate that cache overhead for the standard algorithm at other resolutions are generally worse. Note that the weighted orthogonal complement algorithms are not immune from cache effects, but the performance loss is less pronounced than for the standard algorithm.

It is known [14, 15] that the standard algorithm can be implemented with $\mathcal{O}(N_{\text{lat}}^2)$ storage. Only $\bar{P}_n^0$ and $\bar{P}_n^1$ are stored and all other associated Legendre functions are computed on-the-fly using an orthogonal and consequently very accurate four-term recurrence relation [7]. In fact, this approach can be used to reduce memory bandwidth problems such as those illustrated in Fig. 3. This technique becomes more desirable as processor speeds continue to outpace memory speed. However, the WOC algorithms require fewer operations than an algorithm in which the associated Legendre functions must be computed on-the-fly. For this reason, the compute-on-the-fly algorithm is not considered here.

Both the direct and FMM algorithms also require the loading of new $\mathcal{O}(N_{\text{lat}}^2)$ matrices into cache for every m, and so it is likely that they will exhibit the same type of timing characteristics as the standard algorithm. Figure 4 is a timing plot for these algorithms at the same resolution and on the same scale as Fig. 3. Included in the plot is the complete WOC algorithm, which now includes the pole optimization as well as the crossover switch between Eqs. (16) and (17), and as such it provides the actual performance comparison. Evidently, the direct and FMM algorithms are experiencing poor cache performance as seen by comparing the direct method in Figs. 1 and 4. Nevertheless, the direct method represents a clear savings over the standard approach. At this resolution, the total FMM

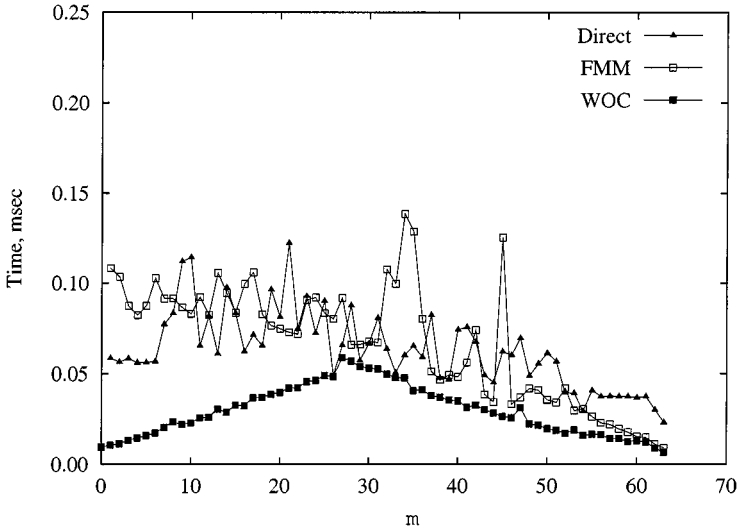

FIG. 4. Direct, fast multipole, and complete weighted orthogonal complement timings as a function of m for the untruncated projection at $N_{\text{lat}} = 65$.

time is roughly equivalent to the direct approach. The WOC curve is relatively smooth, and careful inspection reveals a performance increase compared to the simpler implementation in Fig. 3, which does not include the pole optimization.

Given the results in Figs. 3 and 4, we can predict that the full projection algorithm timings should result in a greater savings for the WOC algorithm relative to the others than is predicted by the operation counts in Section 3. This prediction is realized by the results plotted in Figs. 5 and 6, which are full projection timings without truncation and with the $2/3$ rule, respectively. The sequence of resolutions studied are as in [5] (except for the seminaive algorithm, which requires that N_{lat} be a power of two), and the maximum

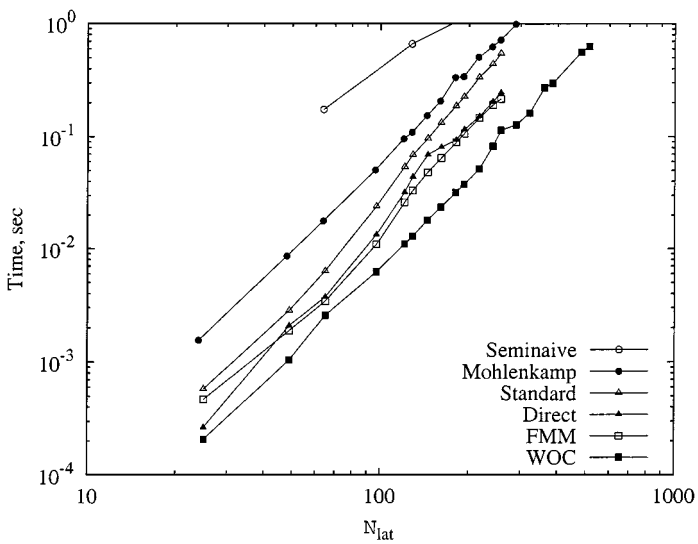

FIG. 5. Experimental, untruncated projection timings vs N_{lat}.

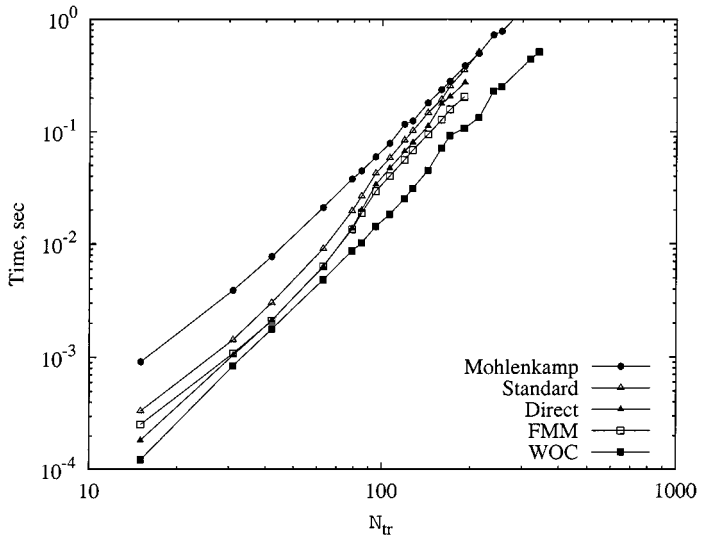

FIG. 6. Experimental, 2/3 rule projection timings vs N_{tr}.

resolution for a given algorithm is determined by the largest problem that can be stored in the 128MB of memory on the Sun Ultra 60 workstation on which the experiments were performed. Because of its $\mathcal{O}(N_{lat}^2)$ storage requirement, the WOC algorithm could be timed at higher resolutions than the other algorithms.

Predictably, the standard and WOC algorithm timings bound the direct method timings, but more interestingly, these two $\mathcal{O}(N_{lat}^3)$ algorithms also bound the FMM, which indicates that the FMM has not yet reached an asymptote of $\mathcal{O}(N_{lat}^2)$.

The FMM performance can be characterized as follows: it is always faster than the standard algorithm; it is slower than the WOC algorithm at all of the resolutions studied; and it performs comparably to the direct method, although it does break even and surpass the direct method at $N_{tr} = 63, 79$ for the untruncated and 2/3 rule projections, respectively.

The Mohlenkamp projection is slower than the standard algorithm at almost all the resolutions studied, although it may exhibit a slope slightly less than the $\mathcal{O}(N^3)$ curves. For the 2/3 rule, it intersects the standard curve at approximately $N_{tr} = 200$, which is consistent with Mohlenkamp's claim of break-even timings at $N_{tr} = 128$ [3]. The seminaive projection, which is timed only for the untruncated case, exhibits significant overhead because of the required discrete cosine transforms and the fact that the required latitude points are double the number of the other algorithms.

Figures 5 and 6 are log–log plots to highlight the slopes of the lines, but this format makes the percent savings of the various algorithms relative to the standard algorithm difficult to discern. These savings are plotted in Figs. 7 and 8 for the untruncated and 2/3 rule cases, respectively. The seminaive and Mohlenkamp algorithms are not shown because they did not result in savings for the resolutions studied.

Recall that the theoretical savings resulting from operation counts for the direct method is 12.5% and 6.25% for untruncated and 2/3 rule, respectively. For the WOC algorithm, the operation count savings are 50% and 25%, respectively. The results of Figs. 7 and 8 are always superior to these estimates and often far superior. These additional savings are due to the pole optimization, and in the case of the WOC, cache reutilization. Note that the minimum savings for the WOC algorithm occur at or near the resolution plotted in

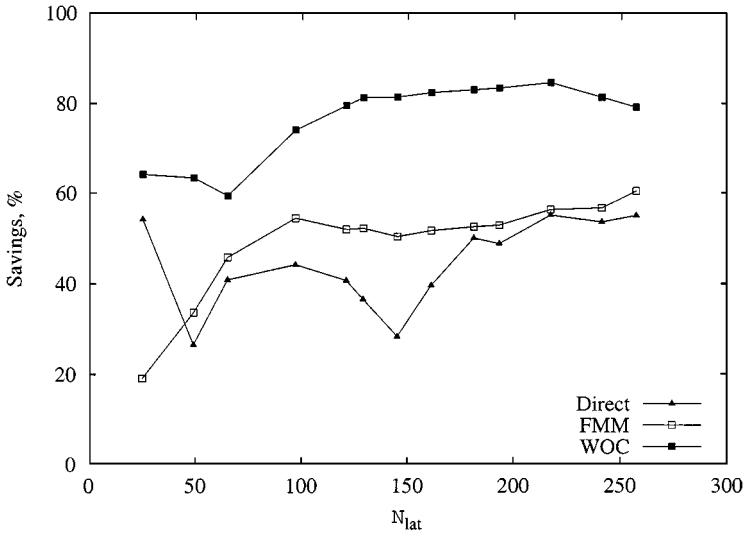

FIG. 7. Experimental, untruncated projection savings relative to the standard algorithm vs N_{lat}.

Fig. 3, indicating that the cache overhead for the standard algorithm plotted in this figure is worse for higher resolutions (or equivalently, the cache efficiency for the WOC algorithm is better). The percent savings for the WOC algorithm is often higher than 80% and 60% for the untruncated and 2/3 rule projections, respectively.

5. A DISCUSSION OF PARALLEL IMPLEMENTATIONS

Global weather and climate codes are large enough that distributed memory parallel implementations are necessary, and if a projection algorithm is to be used in such a code,

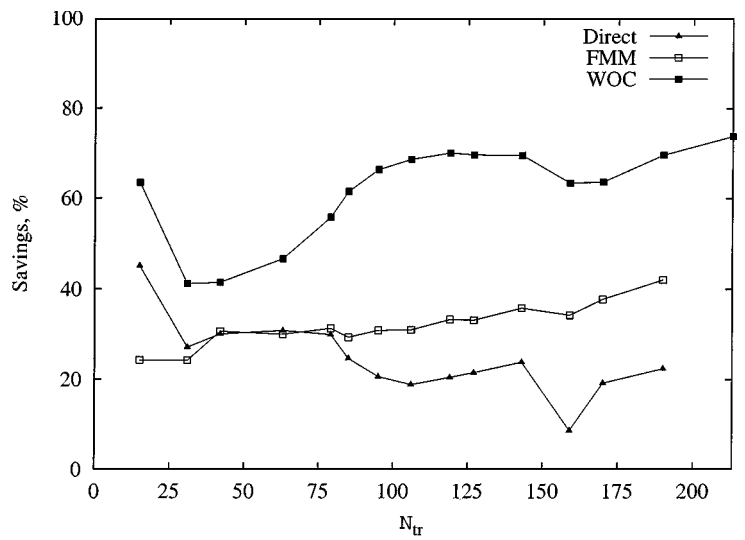

FIG. 8. Experimental, 2/3 rule projection savings relative to the standard algorithm vs N_{tr}.

a strategy for parallel implementation must be devised. In current spectral models, communication is isolated and minimized by utilizing 1-D decompositions in the latitude and longitude directions and using a parallel transpose algorithm to transfer between the two representations, depending on which decomposition is needed at which point in the execution of a single time step. In practice, the discrete Fourier functions $\mathbf{a}_m^F$ and $\mathbf{b}_m^F$ are transposed, so these functions are decomposed two ways, namely, with respect to latitude coordinates θ_j and with respect to zonal wave number m. Since the projection operates on $\mathbf{a}_m^F$ and $\mathbf{b}_m^F$ and all the algorithms can be logically decomposed with respect to m, parallel implementation of the projection algorithm is straightforward: decompose with respect to m; communication will occur elsewhere in the model.

A well-designed model using the associated Legendre projection (see, for example, [10]) will not add any additional transposes compared to the standard spectral transform method. Therefore, the key issue in parallelizing a projection algorithm is domain decomposition with respect to m and the resulting load balance. Load balancing for triangular work profiles such as the standard algorithm is accomplished in the straightforward manner of matching m with large operation counts with corresponding m with low operation counts. For the standard algorithm, wave number m would be matched with wave number $N_{\text{tr}} - m$, and then the matched wave numbers could be distributed equally across processors. Similarly, for the untruncated WOC algorithm, m could be matched with $N_{\text{lat}}/2 + m$ (for $m < N_{\text{lat}}/2$) to achieve a balanced workload.

A load balancing problem occurs when we attempt to decompose the direct or WOC algorithms for a truncated projection. As an inspection of the right plot of Fig. 1 reveals, there is no trivial matching of wave numbers which will balance the direct or WOC workloads while simultaneously allowing an *equal* decomposition with respect to m. This equal decomposition is critical because there are other pieces of the model which will require a decomposition with respect to m: spatial differentiation, right-hand side calculations, time stepping, elliptic solvers, and so forth. These other pieces can be balanced with an equal number of m per processor.

Since the WOC projection algorithm is the most efficient in terms of operation count, cache utilization, and overall performance, we will restrict our investigation of parallel implementations to this algorithm. We seek first a timing profile of the projection as a function of m for a typical resolution ($N_{\text{lat}} = 65$) using the 2/3 rule ($N_{\text{tr}} = 42$). This profile is given in Fig. 9.

Our goal now is to apply various decomposition strategies, with respect to m, to the data presented in Fig. 9 in order to obtain an estimate of how load imbalance will affect scaling. The first strategy will be the "natural" decomposition, where contiguous chunks of size N_{tr}/p are assigned to p processors numbered $0 \ldots p-1$. The second strategy will be the "round robin" decomposition, in which each consecutive m is assigned to a new processor, starting with 0. When the last processor number is reached, the processor assignment is reset to 0. The last strategy to be investigated will be "reversed round robin" in which processor assignment alternates between $[0 \ldots p-1]$ and $[p-1 \ldots 0]$.

These three strategies are applied to the data of Fig. 9, and the total computation time per processor, T_k, where $k = 0 \ldots p-1$, is accumulated for each. The speedup efficiency can be computed from

$$E = T_{\text{tot}}/p \max_k T_k,$$

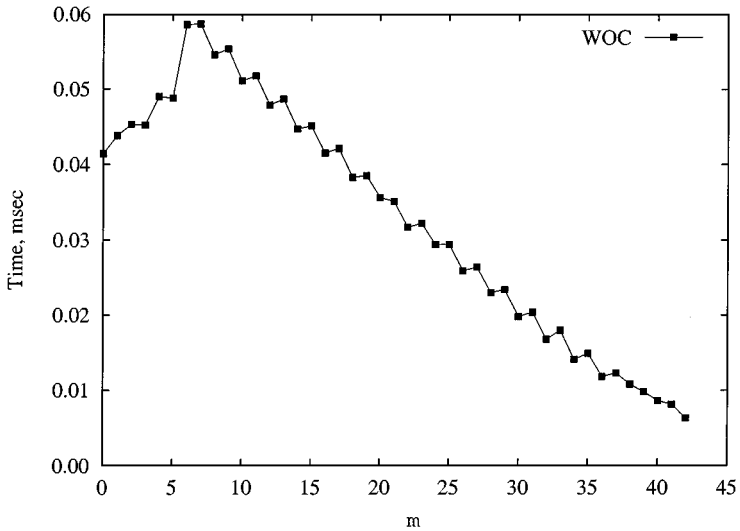

FIG. 9. Weighted orthogonal complement projection timings as a function of m for $N_{\text{lat}} = 65$ using the $2/3$ rule.

where T_{tot} is the total execution time $T_{\text{tot}} = \sum_{k=0}^{p-1} T_k$. Distributions are considered for up to $N_{\text{tr}}/2 = 21$ processors. The resulting efficiencies are plotted in Fig. 10. Clearly, the natural decomposition, which drops immediately to 70% and declines from there, is too inefficient to consider. The round robin strategy steadily declines to 80% efficiency for the maximum processor count, but the reversed round robin decomposition hovers at or above 90% efficiency for all processor counts, making it the preferred strategy of those studied.

This 10% loss from load imbalance may be acceptable in many cases, especially since the serial WOC algorithm provides around 60% savings for most of the resolutions studied. If not, a more complicated decomposition would be required, in which unequal numbers

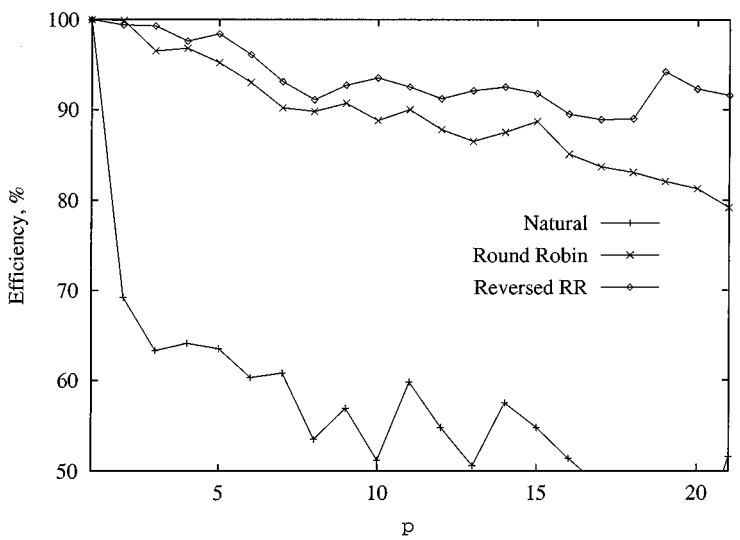

FIG. 10. Estimated speedup efficiencies, resulting from load imbalance only, versus number of processors p for various decomposition strategies applied to the $2/3$ rule weighted orthogonal complement projection.

of m are stored on each processor. Load balancing should then be achieved in terms of the entire time step algorithm for the model, which is beyond the scope of this paper.

6. CONCLUSIONS

The associated Legendre projection is a key component in the development of faster spectral climate and weather models. Here we have reviewed six different algorithms: standard; direct, fast multipole, weighted orthogonal complement, seminaive, and Mohlenkamp.

The standard algorithm provides the baseline comparison because it represents the current behavior of associated Legendre transforms in modern atmospheric codes. The direct method is a logical improvement on this approach, although the weighted orthogonal complement projection has both a better operation count and a better memory requirement. The fast multipole method, as a theoretically $\mathcal{O}(N_{\text{lat}}^2)$ algorithm, holds the promise of a truly fast algorithm, although its large overhead has prevented this from being realized at resolutions of today and the near future. The Mohlenkamp algorithm, while potentially competitive at higher resolutions, did not provide savings at the resolutions of interest.

The seminaive projection performed poorly compared to the other algorithms, but we note a potential usefulness of the approach. In the model proposed by Cheong [10], spectral calculations such as elliptic solvers, differentiation and time-stepping are done in double Fourier space, which is strongly analogous to $\mathbf{a}_m^C$, the cosine transform of the discrete Fourier functions. To apply the other algorithms to this model, sine and cosine transforms are required to first transform the double Fourier coefficients to $\mathbf{a}_m^F$. Thus, if the seminaive algorithm could be modified to (1) act upon double Fourier coefficients with (2) $N_{\text{tr}} > N_{\text{lat}}/2$, it could actually *eliminate* sine and cosine transforms from the double Fourier model which are necessary with the other algorithms. This will be the topic of a future study.

For resolutions studied here ($N_{\text{lat}} < 300$), the weighted orthogonal complement is the most efficient algorithm. This is due in part to its relatively low operation count, but the algorithm's reutilization of matrices $\mathbf{Q}^\ell$ and $\mathbf{R}^\ell$ also permits a cache efficiency which results in an overall savings, compared to the standard algorithm, from 60–80% for the untruncated projection and from 40–70% for the $2/3$ rule projection. For the fast multipole method, these savings are 20–40% and 20–60%, respectively.

As for parallel implementation on a distributed memory architecture, the only problem presented by the weighted orthogonal complement projection is that the nonstandard work profile (as a function of m) contributes to a load imbalance. In the context of a full model, a decomposition with respect to m should distribute an equal number of zonal wave numbers to each processor. This prevents a perfect balance of the projection work load. However, experiments indicate that a reversed round robin decomposition results in only a 10% or less loss in speedup efficiency as a result of imbalance. Nevertheless, the overall savings of the weighted orthogonal complement algorithm still makes it the most attractive alternative.

APPENDIX: INTERNET RESOURCES

The seminaive transforms used here (in addition to the Driscoll–Healy midpoint and hybrid algorithms) are a part of SpharmonicKit 2.5, available at

http://www.cs.dartmouth.edu/~geelong/sphere (A.1)

The Mohlenkamp transforms are a part of the Fast Transforms for Spherical Harmonics (FTSH 1.1) library available at

http://amath.colorado.edu/appm/faculty/mjm/libftsh.html (A.2)

The Basic Linear Algebra Subprograms (BLAS) can be found at

http://www.netlib.org/blas (A.3)

ACKNOWLEDGMENTS

The authors thank Norman Yarvin of Yale University for providing code for the direct and fast multipole projections, Martin Mohlenkamp of the University of Colorado for allowing us to test his code before he made it public, Peter Kostelec of Dartmouth University for aiding us in implementing SpharmonicKit routines, and finally Rich Loft of the NCAR Scientific Computing Division for many valuable discussions.

REFERENCES

1. J. Driscoll and D. Healy, Computing Fourier transforms and convolutions on the 2-sphere, *Adv. Appl. Math.* **15**, 202 (1994).
2. D. M. Healy, Jr., D. N. Rockmore, and S. B. Moore, An FFT for the 2-sphere and applications, in *Proceedings ICASSP 96, Atlanta, May 1996*, pp. 1323–1326.
3. M. J. Mohlenkamp, A fast transform for spherical harmonics, *J. Fourier Anal. Appl.* **5**(2/3), 159 (1999).
4. W. F. Spotz, M. A. Taylor, and P. N. Swarztrauber, Fast shallow water equation solvers in latitude–longitude coordinates, *J. Comput. Phys.* **145**, 432 (1998).
5. R. Jakob-Chien and B. K. Alpert, A fast spherical filter with uniform resolution, *J. Comput. Phys.* **136**, 580 (1997).
6. N. Yarvin and V. Rokhlin, A generalized one-dimensional fast multipole method with applications to filtering of spherical harmonics, *J. Comput. Phys.* **147**, 594 (1998).
7. P. N. Swarztrauber and W. F. Spotz, Generalized discrete spherical harmonic transforms, *J. Comput. Phys.* **159**, 213 (2000).
8. B. Machenhauer and R. Daley, *A Baroclinic Primitive Equation Model with Spectral Representation in Three Dimensions*, Technical Report 4 (Institute for Theoretical Meteorology, Copenhagen University, 1972).
9. P. N. Swarztrauber, On the spectral approximation of discrete scalar and vector functions on the sphere, *SIAM J. Numer. Anal.* **16**(6), 934 (1979).
10. H.-B. Cheong, Double Fourier series on a sphere: Applications to elliptic and vorticity equations, *J. Comput. Phys.* **157**, 327 (2000).
11. H. Ritchie, Application of the semi-Lagrangian method to a spectral model of the shallow water equations, *Mon. Weather Rev.* **116**, 1587 (1988).
12. J. P. Boyd, Multipole expansions and pseudospectral cardinal functions: A new generalization of the fast Fourier transform, *J. Comput. Phys.* **103**, 184 (1992).
13. G. A. Dilts, Computation of spherical harmonic expansion coefficients via FFT's, *J. Comput. Phys.* **57**, 439 (1985).
14. P. N. Swarztrauber, The vector harmonic transform method for solving partial differential equations in spherical geometry, *Mon. Weather Rev.* **121**(12), 3415 (1993).
15. J. C. Adams and P. N. Swarztrauber, Spherepack 3.0: A model development facility, *Mon. Weather. Rev.* **127**, 1872 (1999).