Pergamon

*Atmospheric Environment* Vol. 28, No. 20, pp. 3369–3385, 1994

1352–2310/94 $7.00 + 0.00

1352–2310(94)00124-3

# NUMERICAL ADVECTIVE SCHEMES USED IN AIR QUALITY MODELS—SEQUENTIAL AND PARALLEL IMPLEMENTATION

DONALD DABDUB and JOHN H. SEINFELD
Department of Chemical Engineering, California Institute of Technology, Pasadena, CA 91125, U.S.A.

(*First received* 16 *June* 1993 *and in final form* 13 *April* 1994)

**Abstract**—Six algorithms for solving the advection equation are compared to determine their suitability for use in photochemical air quality models. The algorithms tested are the Smolarkiewicz method, the Galerkin finite element method, the numerical method of lines, the accurate space derivative method (ASD), Bott method, and Emde method. Four algorithms for filtering the numerical noise produced when solving the advection equation are also compared. The algorithms are evaluated both on two test problems and in the CIT model. The Galerkin finite element and the ASD methods are implemented in the CIT in parallel computation. Results indicate that the ASD method, coupled with the Forester filter, produces the most accurate results. When the ASD transport solver is implemented in parallel, a speed-up of about 88 is achieved using 256 processors. Furthermore, a new set of optimized Forester filter parameters for grid-based air quality models is determined.

*Key word index*: Air quality models, advection routines, parallel computation.

## 1. INTRODUCTION

Eulerian air quality models (AQMs) are based on the numerical solution of the atmospheric diffusion equation. Splitting methods provide an accurate and economical approach to solve the atmospheric diffusion equation (McRae *et al.*, 1982). The advection equation, one of the component operators in the splitting scheme, accounts for the transport of pollutants under a given wind field.

The two-dimensional advection equation is

$$\frac{\partial C}{\partial t}+\frac{\partial (uC)}{\partial x}+\frac{\partial (vC)}{\partial y}=0 \qquad (1)$$

where $C$ is the concentration, $t$ is time, and $u$, $v$ are the $x$, $y$ components of the wind velocity field. When solving equation (1) numerically, it is well known that numerical diffusion and dispersion degrade the computed solution (Oran and Boris, 1987) as both the amplitude and phase of different Fourier components of the solution tend to be altered by numerical schemes. To overcome these errors, a large number of numerical advection schemes have been developed. Rood (1987) summarizes the development and improvements of many of the methods.

A variety of numerical advection schemes have been tested and compared to determine their suitability for use within air quality models (Chock and Dunker, 1983; Chock, 1985, 1991; Schere, 1983; Sheih and Ludwig, 1985; Tran and Mirabella, 1991). Hov *et al.* (1989) show that, in AQMs, errors in the solution of the transport step are amplified during the chemistry step due to the highly nonlinear nature of the chemistry. Low-order numerical schemes used to solve the advection equation provide nonoscillatory solutions with poor accuracy. High-order numerical schemes, on the other hand, are characterized by computational noise near regions of steep gradients. The oscillatory noise increases in amplitude and propagates into neighboring grid points as the solution time increases, often producing negative values in the distribution being advected. Negative concentrations correspond to physically unrealistic "negative" mass. To overcome this problem different algorithms that "filter" the oscillations have been proposed. A filter is a computational technique, used after each advection solver step, that removes computational noise. A filtered solution is expected to maintain the accuracy of high-order numerical schemes, and present a distribution that is acceptable on physical terms.

This study has the following objectives: (1) to build on the work of Chock and others to test advection routines in both idealized tests and in a three-dimensional grid-based air quality model, the CIT model, (2) to evaluate nonlinear filters that have been proposed both on a test problem and in the three-dimensional CIT model, (3) to evaluate advection routines/filters in a parallel implementation of the CIT model, (4) to provide recommendations on the advection routines/filters of choice for three-dimensional air quality models, for both sequential and parallel environments.

## 2. NUMERICAL ADVECTION ALGORITHMS

Equation (1) is frequently solved in air quality models using splitting methods (Yanenko, 1971), where equation (1) is approximated by the successive solution of

$$\frac{\partial C}{\partial t}+\frac{\partial (uC)}{\partial x}=0 \tag{2}$$

$$\frac{\partial C}{\partial t}+\frac{\partial (vC)}{\partial y}=0. \tag{3}$$

The character of equation (1) and equations (2) and (3) is that material is transported intact, without depletion or diffusion. Numerical methods solving equations (1)–(3) must preserve this property as closely as possible.

A traditional way to evaluate the accuracy of numerical methods of solving equations (2) and (3) is to advect an initial cosine hill distribution described by the following equations (Pepper and Long, 1978):

$$C_{i,j}(0)=\begin{cases} 50(1+\cos\frac{\pi R}{4}) & \text{if } R<4 \\ 0 & \text{for } R\geqslant 4 \end{cases} \tag{4}$$

$$R^2=(x_i-x_0)^2+(y_i-y_0)^2 \tag{5}$$

where $x_0=7$, $y_0=17$. The center of the hill is at grid point (7, 17) with a concentration value of 100. Both $x_i$ and $y_i$ vary from 1 to 33. The hill rotates counterclockwise about the $z$-axis at grid point (17, 17) which is the center of the grid. The angular velocity is set so that it takes $7200\pi$ time units to complete one revolution. Time steps of $30\pi$ units were used in all the computations performed here.

Ideally after an integer number of revolutions the peak of the distribution should be 100 and its center should be located at (7, 17). To evaluate the accuracy of different schemes, the following measures have been traditionally used (Chock and Dunker, 1983; Chock, 1985, 1991; Tran and Mirabella, 1991):

$$\text{Mass conservation ratio}=\sum_{i,j}C_{i,j}(t)\Big/\sum_{i,j}C_{i,j}(0) \tag{6}$$

$$\text{Mass distribution ratio}=\sum_{i,j}C_{i,j}^2(t)\Big/\sum_{i,j}C_{i,j}^2(0) \tag{7}$$

$$\text{Maximum absolute error}=\max(|C_{i,j}(t)-C_{i,j}^e(t)|) \tag{8}$$

where $C_{i,j}$ represents the computed concentration at grid point $i$, $j$, and $C_{i,j}^e$ is the exact concentration at grid point $i$, $j$. The mass conservation ratio and the mass distribution ratio measure the amount of numerical diffusion. Note that a method might present a good mass conservation ratio while maintaining a poor mass distribution. The average absolute error measures an average discrepancy of the mass field from the exact solution. Finally, the maximum absolute error is most sensitive to the displacement and height of the distribution's peak.

Table 1 lists the methods examined in this study. SMOL and GLRK were selected since they are currently implemented in air quality models—the Urban

Table 1. Numerical advection methods considered and concentrations and locations of the peak of a cosine hill after two revolutions

| Method | Characteristics | References | Peak* |
|---|---|---|---|
| Smolarkiewicz method (SMOL) | Iterative scheme based on upstream differences. Positive definite | Smolarkiewicz (1983) | 26 @ (8, 19) |
| Numerical method of lines (NMOL) | Fourth-order directional difference in space. SDRIV2 integration in time. Produces negative concentration | Schiesser (1991) Carver and Hinds (1978) Kahaner *et al.* (1989) | 59 @ (7, 16) |
| Numerical method of lines filtered (FNMOL) | NMOL with Forester filter. Filter parameters: $K=1$, $\mu=0.1$, $m=1$, and $n=2$ | | 44 @ (7, 17) |
| Bott method (BOTT) | Nonlinear renormalization of advective fluxes. Positive definite | Bott (1989a, b) | 73 (7, 17) |
| Emde method (EMDE) | Continuous curvature cubic-spline. Positive definite | Emde (1992) | 80 (7, 17) |
| Galerkin method (GLRK) | Chapeau function Galerkin finite element. Produces negative concentration | MacRae *et al.* (1982) | 90 @ (7, 18) |
| Galerkin method filtered (FGLRK) | GLRK with Forester filter. Filter parameters: $K=1$, $\mu=0.1$, $m=1$, and $n=2$ | | 75 @ (7, 17) |
| Galerkin method filtered (FGLRK2) | GLRK with Forester filter. Filter parameters: $K=3$, $\mu=0.2$, $m=2$, and $n=4$ | | 43 @ (7, 17) |
| Accurate space derivative method (ASD) | Fourier techniques to accurately compute space derivatives. Third-order Taylor expansion to integrate in time. Produces negative concentration | Gazdag (1973) | 98 @ (7, 17) |
| Accurate space derivative method filtered (FASD) | ASD with Forester filter. Filter parameters: $K=1$, $\mu=0.1$, $m=1$, and $n=2$ | | 98 @ (7, 17) |

*Exact value: 100 @ (7, 17).

Airshed Model (UAM) (Morris and Myers, 1990) and the CIT model (Harley *et al.*, 1993), respectively. The numerical method of lines (NMOL) is selected since it has not been compared previously with other methods in a systematic manner and has several potentially desirable attributes. For instance, NMOL converts the PDE into a system of ODEs. NMOL can therefore take advantage of recent progress in the numerical solution of ODEs. The accurate space derivative method (ASD) is selected as an alternative method to those currently used in existing AQMs that provides greater accuracy (Chock, 1991). Other alternative methods included in this study are the Bott solver and then Emde solver. The use of parallel computers opens up the practical uses of the ASD method, since it requires significantly greater computational time on a sequential machine than the other methods in Table 1. Some of the methods have been previously compared by Chock and Dunker (1983) and Chock (1985, 1991). The present study builds on Chock's results and evaluates several of the methods in a full three-dimensional air quality model, under both sequential and parallel implementation.

Most of the methods listed in Table 1 are well described in the literature. Therefore, only the particular implementation of NMOL used in this study will be discussed. The numerical method of lines approximates the spatial derivatives of a partial differential equation with an appropriate finite-difference algebraic expression (Schiesser, 1991). The time derivative of the PDE is left unmodified, leading to a system of ordinary differential equations in time. The NMOL implementation used in this study consists of a fourth-order directional finite-difference approximation to discretize the spatial derivatives. Namely, for positive wind velocity

$$\frac{\partial C_i}{\partial x}=\frac{-C_{i-3}+6C_{i-2}-18C_{i-1}+10C_i+3C_{i+1}}{12\Delta x}. \quad (9)$$

Similarly, for negative wind velocity,

$$\frac{\partial C_i}{\partial x}=\frac{-3C_{i-1}-10C_i+18C_{i+1}-6C_{i+2}+C_{i+3}}{12\Delta x} \quad (10)$$

where $C_i$ represents the concentration at grid point $i$ in the one-dimensional sense when solving equations (2) and (3). It is well known that equations (9) and (10) are superior to central finite differences from the accuracy point of view (Carver and Hinds, 1978). The resulting system of ordinary differential equations is solved using the SDRIV2 integrator described by Kahaner *et al.* (1989).

## 3. NONLINEAR FILTERS USED IN THE SOLUTION OF THE ADVECTION EQUATION

As mentioned earlier, the numerical solution of the advection equation presents several difficulties. High-order algorithms produce spurious waves near sharp gradient regions that produce physically unrealistic negative concentrations. To design a filter, one is faced with the question: What is to be done with the "negative" mass that appears in the numerical solution? The simple idea of setting the negative values to zero is not the proper approach, because it is not mass conservative; rather "negative" mass must be redistributed over positive concentrations in the solution. However, there is no specific mathematical or physical guideline on how to perform such a redistribution. To address this problem, various "filtering" techniques have been proposed to remove the negative concentrations and to smooth the positive oscillatory intervals of the distribution.

There are two main approaches to achieve such goals. The first approach consists of selectively introducing nonlinear diffusion to the distribution. The second approach consists of scanning local maxima and local minima on the distribution and adjusting selected distribution values iteratively. The filter step is an important part of the advection computations since the numerical results can be strongly affected by the presence of a filter. Various available filters that can be used in the advection routines in AQMs have not been compared previously. This section compares different filtering techniques that have been proposed in the literature with the goal of determining an optimal filter for use in AQMs.

Let $\mathscr{A}$ be an algorithm (finite difference, finite element, spectral, etc.) to solve equation (2), expressed in the form

$$C_i^{t+\Delta t}=\mathscr{A}(C^t, u) \quad (11)$$

where the subscript $i$ refers to grid block $i$. The redistribution process consists of coupling $\mathscr{A}$ with a filter $\mathscr{F}$ in the following way:

$$\begin{aligned} B_i^t &= C_i^t \\ B_i^{t+\Delta t} &= \mathscr{A}(B^t, u) \\ C_i^{t+\Delta t} &= \mathscr{F}(B^{t+\Delta t}, C_i^t). \end{aligned} \quad (12)$$

$\mathscr{F}$ must satisfy several conditions that arise naturally in air pollution models:

(1) *Mass conservation.* The filter should not add or remove mass from the system,

$$\sum_i C_i^{t+\Delta t}=\sum_i C_i^t. \quad (13)$$

(2) *Positiveness.* The filter should remove all negative values completely,

$$C_i^{t+\Delta t}\geqslant 0 \quad \text{for all} \quad i. \quad (14)$$

(3) *Total variation diminishing.* The filter should guarantee that there are no spurious oscillations near the regions of sharp gradients. The criterion commonly used is the TVD inequality:

$$TV(C^{t+\Delta t})\leqslant TV(C^t) \quad (15)$$

Table 2. Filters for numerical advection methods

| Filter | Characteristics | Reference |
|---|---|---|
| Bartinicki | Scans negative distribution values. Guarantees absence of negative values. Not TVD | Bartinicki (1989) |
| Chapman | Introduces local diffusion. Mass conservative | Chapman (1981) |
| Engquist | Scans local extrema. TVD. Mass conservative | Engquist *et al.* (1989) |
| Forester | Introduces local diffusion. Mass conservative. Requires four problem dependent filter parameters | Forester (1977) |

where

$$TV(C^t)=\sum_i |(C_{i+1}^t - C_i^t)| .$$

(4) *Shape preservation.* The filter should modify the minimum number of grid points to enforce conditions (1)–(3).

An appropriate test case used to study the performance of different filters is a simple one-dimensional advection problem with constant wind velocity. Three different initial conditions, square, triangle, and cosine hill were selected here. For each initial condition Courant numbers of 1, 0.5, 0.1, and 0.01 were used. The Courant number is defined as $(u\,\Delta t)/\Delta x$.

Table 2 summarizes the filters studied. A more detailed description of the filters is given in the Appendix.

## 4. EVALUATION OF THE PERFORMANCE OF ADVECTION ALGORITHMS AND FILTERS ON TEST PROBLEMS

This section evaluates the different advection schemes and nonlinear filters described previously. The evaluation of advection schemes on test problems is kept brief here as this aspect has been presented in detail by Chock and Dunker (1983) and Chock (1985, 1991). As noted above, the evaluation of nonlinear filters for AQMs, on the other hand, has not been studied previously. Finally, the selection of optimal filter parameters is discussed.

### 4.1. *Advection schemes*

When solving the advection equation some algorithms exhibit nonphysical oscillations and/or negative concentrations near steep gradient regions of the solution. To overcome this difficulty the comparison of advection schemes that we will present will utilize the Forester filtering (Forester, 1977) with filter parameters, $K=1$, $\mu=0.1$, $m=1$, and $n=2$. As we will show shortly, this set of Forester filter parameters achieves the best overall accuracy. $K$ is the number of filtering iterations, and $\mu$ is a dimensionless diffusion coefficient. The value of $m$ represents half the wavelength of the lowest frequency noise. The value of $n$ does not represent any physical aspect of the filter. It is simply a numerical parameter that must be large enough to permit continuity of nonzero $C$ values. The emphasis of the comparison in this section is on the advection solver performance. The choice of the Forester filter or a particular set of filter parameters does not give an undue advantage to any one method. If another filter is selected, the relative accuracy of the advection algorithms will remain unmodified.

Figures 1 and 2 show the mass conservation ratio as a function of number of revolutions of the test cosine hill for the different advection solvers studied. The Forester filter slightly improves the mass conservation properties of the schemes; without Forester filtering, most of the schemes still present an acceptable performance from the point of view of conserving mass. However, the SMOL method, after one revolution, is quite mass dissipative. At the other extreme, the NMOL solver is slightly biased to a higher mass conservation ratio. Figures 1 and 2 do show undesirable properties of SMOL. However, the mass conservation ratio does not present sufficient information to discern the methods' relative accuracies.

Figures 3 and 4 show the mass distribution ratio as a function of number of revolutions of the test cosine hill. A low mass distribution ratio indicates a high amount of artificial diffusion present in the numerical scheme. It can be observed that the use of Forester smoothing tends to decrease the mass distribution ratio in all the schemes. ASD seems to be the scheme least affected when filtering is added. SMOL presents the poorest mass distribution ratio of all the schemes studied. Furthermore, Fig. 4 shows that regardless of its acceptable mass conservation ratio, the NMOL scheme has a lower mass distribution ratio than the ASD or Galerkin schemes. Furthermore, it shows that regardless of their almost perfect mass conservation ratio, EMDE and BOTT solvers have a mass distribution ratio worse than ASD and GLRK methods. The mass distribution ratio of the solvers parallels their preservation of peak values as reported in Table 1.

Table 1 presents the resulting distribution of peak magnitude and location after two revolutions for all the schemes studied. The ASD method presents the best peak preservation properties. The computation time used by the GLRK filtered is comparable to that of the SMOL algorithm. The computation time used by NMOL filtered and ASD filtered is about an order of magnitude greater. Results presented in Table 1 agree with those presented by Chock and Dunker (1983) and Chock (1985, 1991).

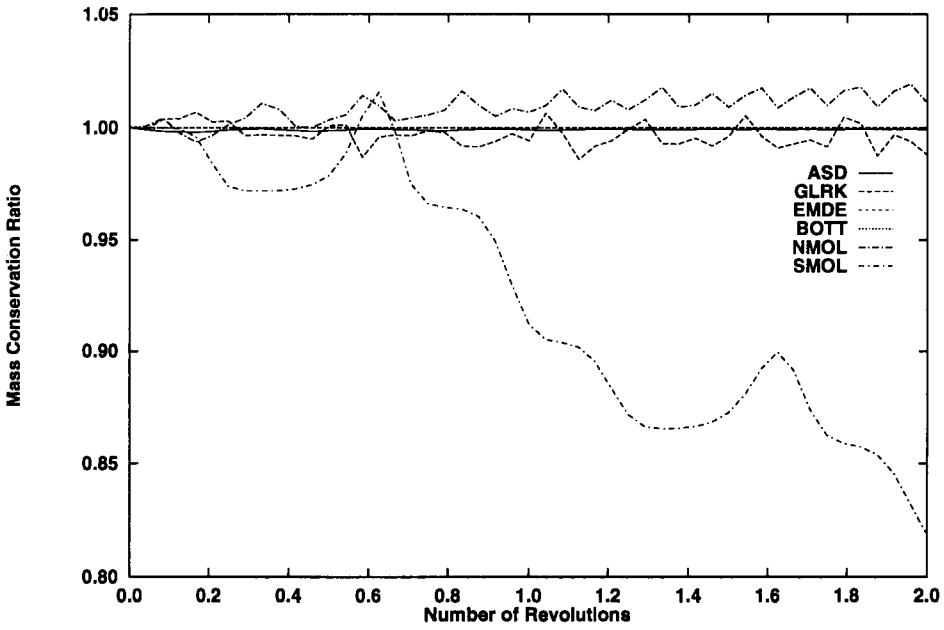


Fig. 1. Mass conservation ratio for unfiltered advection solvers for the rotating cosine hill problem using a time step of $30\pi$.

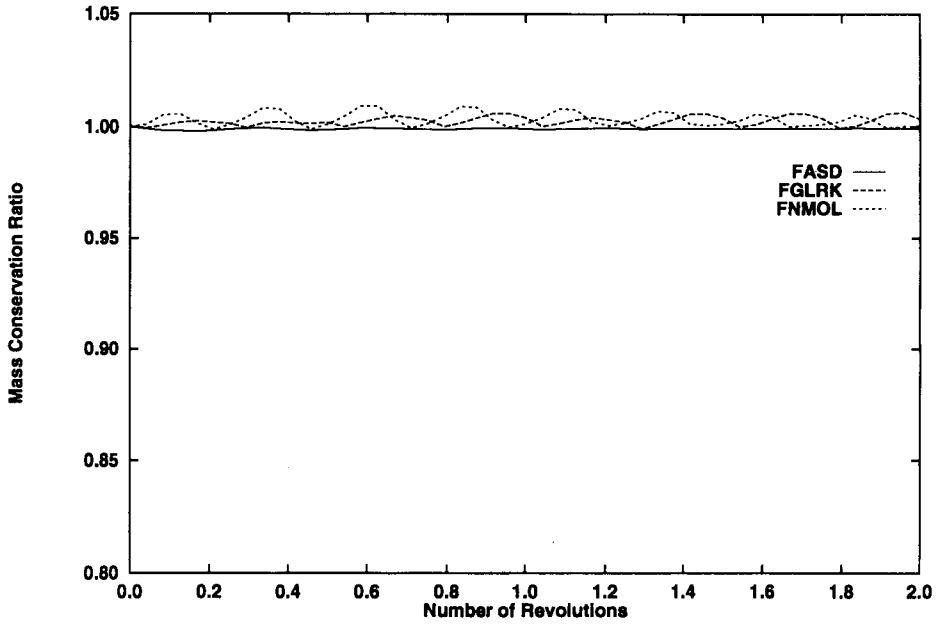


Fig. 2. Mass conservation ratio for Forester filtered advection solvers for the rotating cosine hill problem using a time step of $30\pi$.

A more challenging test problem, as suggested by Odman and Russell (1993), was performed. It consists of using the following velocity field:

$$u_r = 0$$
$$u_\theta = \omega R[1-(R/R_{\max})^2] \quad (16)$$

where $u_r$ and $u_\theta$ represent the radial and angular components of the wind field; $\omega$ is the angular velocity at the peak, adjusted so that the peak completes one full rotation in 240 time steps; $R_{\max}$ is the maximum of $R$ as defined in equation (5). This advective field is characterized by a parabolic angular velocity profile. A parabolic profile is more challenging than the rigid body rotation counter-clockwise rotation since it does not yield constant velocity components along straight

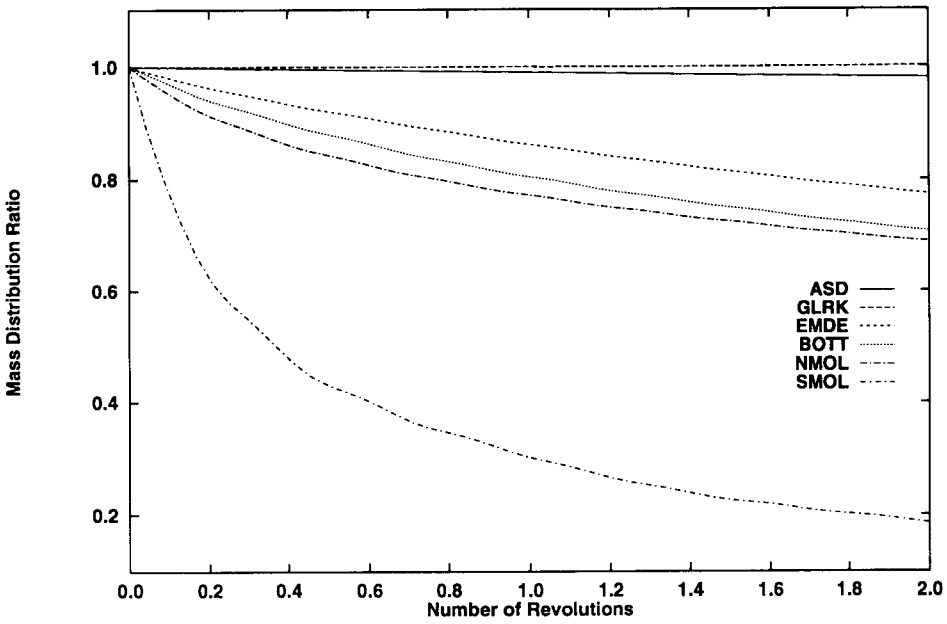


Fig. 3. Mass distribution ratio for unfiltered advection solvers for the rotating cosine hill problem using a time step of $30\pi$.

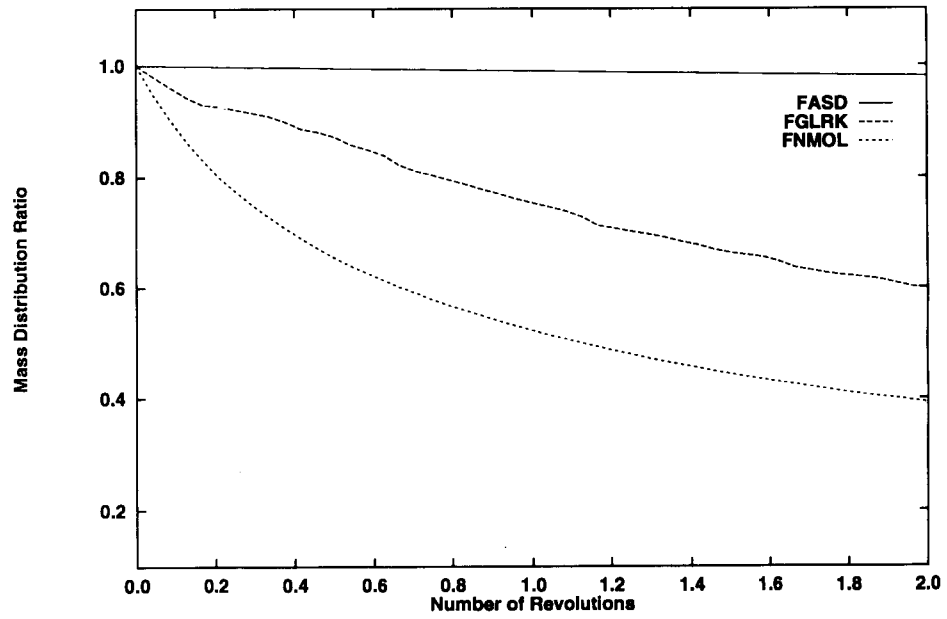


Fig. 4. Mass distribution ratio for Forester filtered advection solvers for the rotating cosine hill problem using a time step of $30\pi$.

lines. Results obtained using the parabolic angular velocity profile present the same relative accuracy for all solvers as that described in Table 1. However, all the solvers present a decrease in accuracy as expected.

### 4.2. *Nonlinear filters*

Figures 5 and 6 show the performance of the different filters for Courant numbers of 1 and 0.1, respectively, as a function of number of integration steps. The error presented equals that of the unfiltered solution. The error norms are defined as

$$|\text{error}|_{L_1} = \sum_i (C_i(t) - C_i^e(t)) \tag{17}$$

$$|\text{error}|_{L_\infty} = \max(|C_i(t) - C_i^e(t)|). \tag{18}$$

The error in the $L_1$ norm is an indication of the

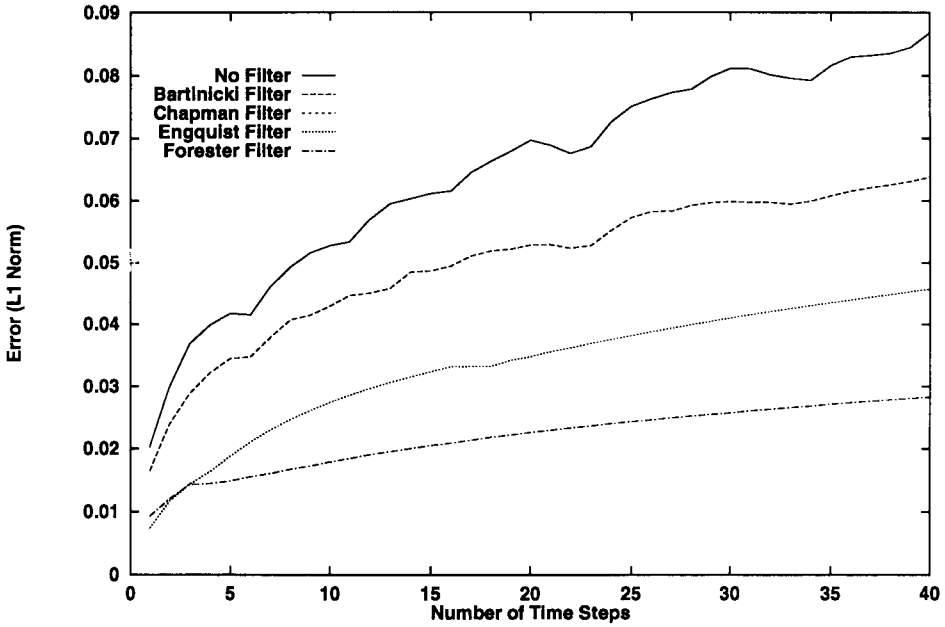


Fig. 5a. $L_1$ error norm for the solution to the one-dimensional advection equation with square pulse initial conditions using a Galerkin solver and various filters (Courant number 1.0).

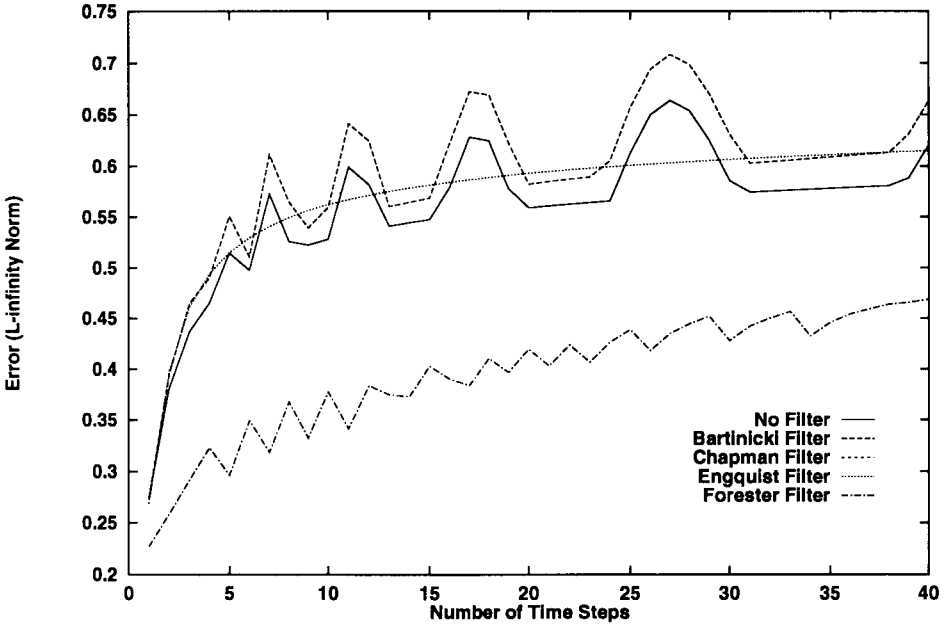


Fig. 5b. $L_\infty$ error norm for the solution to the one-dimensional advection equation with square pulse initial conditions using a Galerkin solver and various filters (Courant number 1.0).

average accuracy of the solution. The error in the $L_\infty$ norm is an indication of the peak conservation of the solution for the initial conditions used in this problem. From the computational point of view, the Courant number determines the upper bound of the number of neighboring points that contain information needed to advance the local solution. Results presented in Figs 5 and 6 correspond to the Galerkin algorithm using a square wave pulse as initial condition. The pulse width is 20 grid points. For reasons of space, and since other initial conditions and other advection solvers yield similar results, Galerkin results are the only ones reported. It was observed that all the filters except for Bartnicki maintain an excellent mass conservation ratio for all Courant numbers studied. Forester filters present the lowest error in the $L_1$ and $L_\infty$

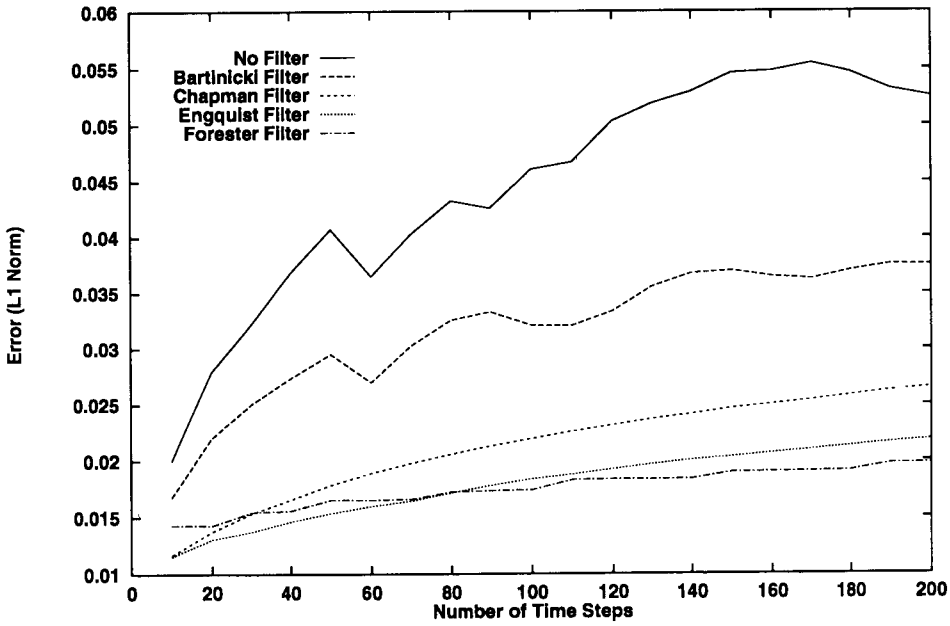


Fig. 6a. $L_1$ error norm for the solution to the one-dimensional advection equation with square pulse initial conditions using a Galerkin solver and various filters (Courant number 0.1).

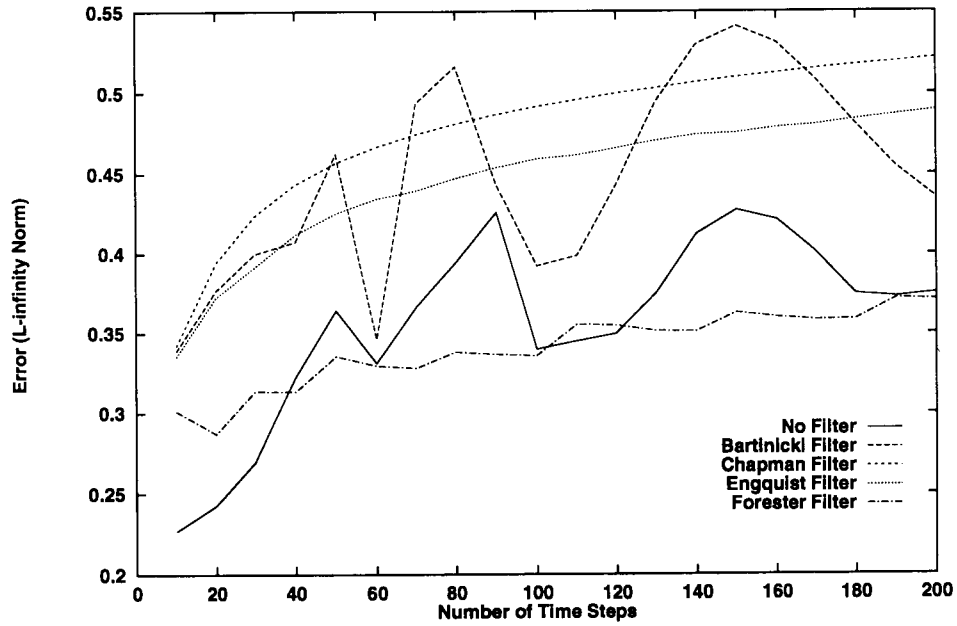


Fig. 6b. $L_\infty$ error norm for the solution to the one-dimensional advection equation with square pulse initial conditions using a Galerkin solver and various filters (Courant number 0.1).

norm sense for a Courant number of 1 as shown in Figs 5a and b. Note that for Courant number of 1 the Chapman filter has no effect on the distribution. For a Courant number of 0.1, the Forester filter can still be considered the best filter to be used based on its error as shown in Figs 6a and b. However, it was observed that for a Courant number of 0.01 the Chapman filter performs somewhat better than the rest of the filters. This Courant number is not a typical condition found in AQMs.

One of the disadvantages of the Forester filter is that it requires four parameters that are dependent on the problem to be solved. These parameters can have a strong effect on the accuracy of the advection scheme being filtered as shown in Table 1 in the FGLRK and FGLRK2 entries. It is observed that using different

values of such parameters can reduce the peak height by more than 40%. Therefore, filter parameters should be carefully selected for a given problem. Forester (1977) attempted to determine by analysis the proper parameter values for the adequate control of computational noise. However, the problem becomes too complex and he concluded that empirical tests should be used. The parameters must be chosen such that each is as small as possible without producing negative concentration values. The values of $K$ and $\mu$ should be small so that artificial diffusion introduced by the filter is minimized. The values of $m$ and $n$ should be the smallest values that permit continuity of nonzero values of the distribution. We have performed an extensive trial-and-error evaluation to determine the optimal set of Forester filter parameters using this test problem. The optimal set of parameters is: $K=1$, $\mu=0.1$, $m=1$, and $n=2$.

Each of the filters described can be applied to any advection solver. Indeed, the advection solver and the filter are treated as two independent modular computational routines in this study. Nevertheless, Odman and Russell (1993) present a tailor-made filter for multi-dimensional finite-element methods. It has been observed in our test runs that a given filter has the same relative effects on all the advection solvers evaluated. In summary, the combination of ASD with the Forester filter shows the best performance for most of the cases studied. At a low Courant number (0.01), which is not as relevant for air quality modeling applications as the other values of the Courant number studied here, the Chapman filter combined with ASD presents the best overall performance.

## 5. EVALUATION OF THE PERFORMANCE OF ADVECTION ALGORITHMS AND FILTERS IN THE CIT MODEL

As shown in Table 1 and Figs 1–4, the test problems indicate that ASD is the most accurate of the methods tested. This section discusses the implementation of the ASD method in the CIT model and comparison of the ASD and GLRK methods. This comparison shows the impact of the advection solver, interacting with chemistry, on peak pollutant concentrations. On the basis of the cosine hill tests, SMOL and NMOL methods have been eliminated from further consideration.

Initial test problems also show that the use of somewhat different Forester filter parameters with the Galerkin solver has dramatic effects on the accuracy of the solution. The ozone concentration predictions of the CIT model with the Galerkin solver using two sets of Forester filters parameters will also be compared. One set corresponds to the filter parameters implemented in the CIT model while the other set corresponds to optimized parameters determined on the test examples. This comparison shows the impact of the filter parameters when transport and chemistry interact within an AQM. As a result of the comparison, a new set of Forester filter parameters is recommended to be used with the CIT model.

### 5.1. *Implementing the ASD method in a three-dimensional AQM*

The horizontal boundary conditions used to solve the advection equations within an air quality model are

$$(uC)\cdot\hat{n}=(u_b(t)C_b(t))\cdot\hat{n}, \quad u\cdot\hat{n}\leqslant 0 \tag{19}$$

$$-\nabla C\cdot\hat{n}=0, \quad u\cdot\hat{n}>0 \tag{20}$$

where $\hat{n}$ is the normal to the boundary, and $u_b(t)$ and $C_b(t)$ are the specified wind speed and concentration at the boundary, respectively. If a function is smooth and periodic its Fourier series does not exhibit the Gibbs phenomenon. Since ASD involves Fourier transforms, it requires periodicity to avoid such Gibbs phenomenon. Therefore, to implement ASD into a three-dimensional AQM, first one must use some computational "artifices" while performing FFTs to meet the periodicity requirement.

There have been different approaches developed to meet the need for periodicity. Roach (1978) describes a technique termed "reduction to periodicity". It consists of splitting the value of the concentrations, $C(x)$ into a periodic function, $F(x)$ and a polynomial of a given degree, $P(x)$, $C(x)=P(x)+F(x)$. The coefficients of the polynomial $P(x)$ are chosen so that the residual $F(x)$ has periodic derivatives at the boundaries. To compute the spatial derivatives of $C(x)$ the fast Fourier transform (FFT) is applied to $F(x)$ only. The derivatives of $P(x)$ are obtained analytically. Chock (1991) describes an alternative technique called "periodicity recovery". It consists of extending the domain of the solution and using a polynomial or spline function fitted to assure periodicity. Wengle and Seinfeld (1978) proposed to expand $C(x)$ into Chebyshev polynomials. Finally, Gazdag (1973) used mirror techniques that consist of doubling the domain with the mirror image of the data to be transformed. The ASD transport solver was implemented into the CIT photochemical model using periodicity recovery.

Currently, the chemistry solver for AQMs is the most computationally intensive part of the numerical solution. For instance, the CIT model with the Galerkin advection routine spends about 90% of its CPU time on the chemistry solver and approximately 5% on the horizontal transport solver. Implementing ASD into the CIT model, as executed on a sequential computer, causes the full model to run four times slower than when using the Galerkin advection solver. The reason that ASD requires greater CPU time than Galerkin is that ASD performs three FFTs and three inverse FFTs computations per time step. On the other hand, the Galerkin method only solves a tridiagonal system per time step. Table 3 presents detailed timing data for a 24-h run of the CIT model on an IBM RISC 580 (sequential architecture) for different modeling cases.

Table 3. Performance and CPU usage distribution for a 24-hour simulation of the South Coast Air Basin under different numerical schemes

| Transport | Chemistry | Total time (s) | Chemistry integration (s) | Other comput. (s) | Relative speed | Chemistry comput. (s) % | Other comput. % |
|---|---|---|---|---|---|---|---|
| Galerkin | On | 4419.08 | 3967.66 | 451.42 | 10.52 | 89.78 | 10.22 |
| Galerkin | Off | 419.94 | 0.00 | 419.94 | 1.00 | 0.00 | 100.00 |
| ASD | On | 17507.12 | 3975.30 | 13531.82 | 41.69 | 22.71 | 77.29 |
| ASD | Off | 13327.46 | 0.00 | 13327.46 | 31.74 | 0.00 | 100.00 |

### 5.2. *Evaluation of the Galerkin and ASD methods in the CIT model—Sequential implementation*

The simulations reported here are for the same conditions as those reported in Harley *et al.* (1993) for 27 August 1987 in the South Coast Air Basin. The reader is referred to Harley *et al.* (1993) for all details of the simulation.

Figure 7 shows the predicted ground-level ozone concentrations in the South Coast Air Basin at 14:00 hours on 27 August 1987. The computations were performed using the Galerkin solver and Forester filter with the parameters originally used in the model, $K=3$, $\mu=0.2$, $m=2$, and $n=4$. Previous tests (FGLRK and FGLRK2 entries in Table 1) indicate that these parameters introduce excessive artificial diffusion in by the filtering step. Figure 8 shows the predicted ground-level ozone concentrations for the same conditions as in Fig. 7 with the only change being that the filter parameters are selected as $K=1$, $\mu=0.1$, $m=1$, and $n=2$, those found in the test example described above to produce the best concentration peak results. It is observed that with the new filter parameters an ozone peak appears in the north west region of the modeled area, the ozone peak in the eastern part of the region expands in size, and there is a small decrease in the ozone peak of the south central region of the domain. Figure 9 shows CIT model predictions using the ASD method as the advection solver with the new filter parameters. Ozone maxima are predicted in the same location as with the Galerkin method with the corrected filter parameters. However, when the ASD method is used the ozone maxima in the eastern portion of the South Coast Air Basin are enhanced by about 20 ppb, attributable entirely to the better properties of the ASD method relative to the Galerkin method. These results indicate that the choice of advection routine in a photochemical air quality model can have a measurable effect of predicted levels of ozone and other species.

## 6. PARALLEL IMPLEMENTATION OF ADVECTION SCHEMES

The use of the ASD method requires a significantly greater amount of computer time than the Galerkin or Smolarkiewicz methods currently implemented in AQMs. This is a major practical disadvantage if sequential computers are to be used. To overcome this problem ASD can be implemented on a parallel computer. The basic idea behind the parallel implementation of the advection solver is to perform the transport computations of all rows or columns simultaneously. This section discusses three approaches to implement the transport computations in parallel: Extended Arrays (EA), Designated Transport Nodes (DTN) and Dynamically Balanced (DYB).

The solution of the advection equation in parallel is more challenging than the integration of the chemistry portion of an AQM. Transport calculations inherently require nonlocal data (i.e. concentration values, wind, etc.) to be able to predict the grid concentration values after each transport step. Furthermore, AQMs typically spend less than 10% of their CPU time solving the advection operator. Due to the challenges presented by parallelizing transport and to the relatively small computational time required by most advection solvers, one might conclude at first thought that perhaps it is not effective to perform transport computations in parallel. This is not the case. The speed-up predicted by Amdahl's law becomes increasingly sensitive to the fraction of the code that is already parallelized. In the case of air quality models, in which about 90% of the computations are parallelized, the speed-up gained by implementing transport computations in parallel is predicted to become significant.

The numerical solution of the advection equation requires neighbor grid cell data that are not always available in local nodes. The number of neighboring grid cells required is dependent on the underlying transport algorithm used. For instance, a second-order finite-element method needs two neighboring grid cells in each direction to predict the local grid cell concentration. A spectrally accurate method, like ASD, needs all the data of the entire row or column to be able to predict any local grid cell concentration. The reason for such data dependency is that spectral methods involve global operations like the transformation of data to the Fourier domain. Finite element algorithms and spectral methods are perfect examples to show the wide differences in data flow structures imposed by two transport solvers. The optimum technique to implement is dependent on the

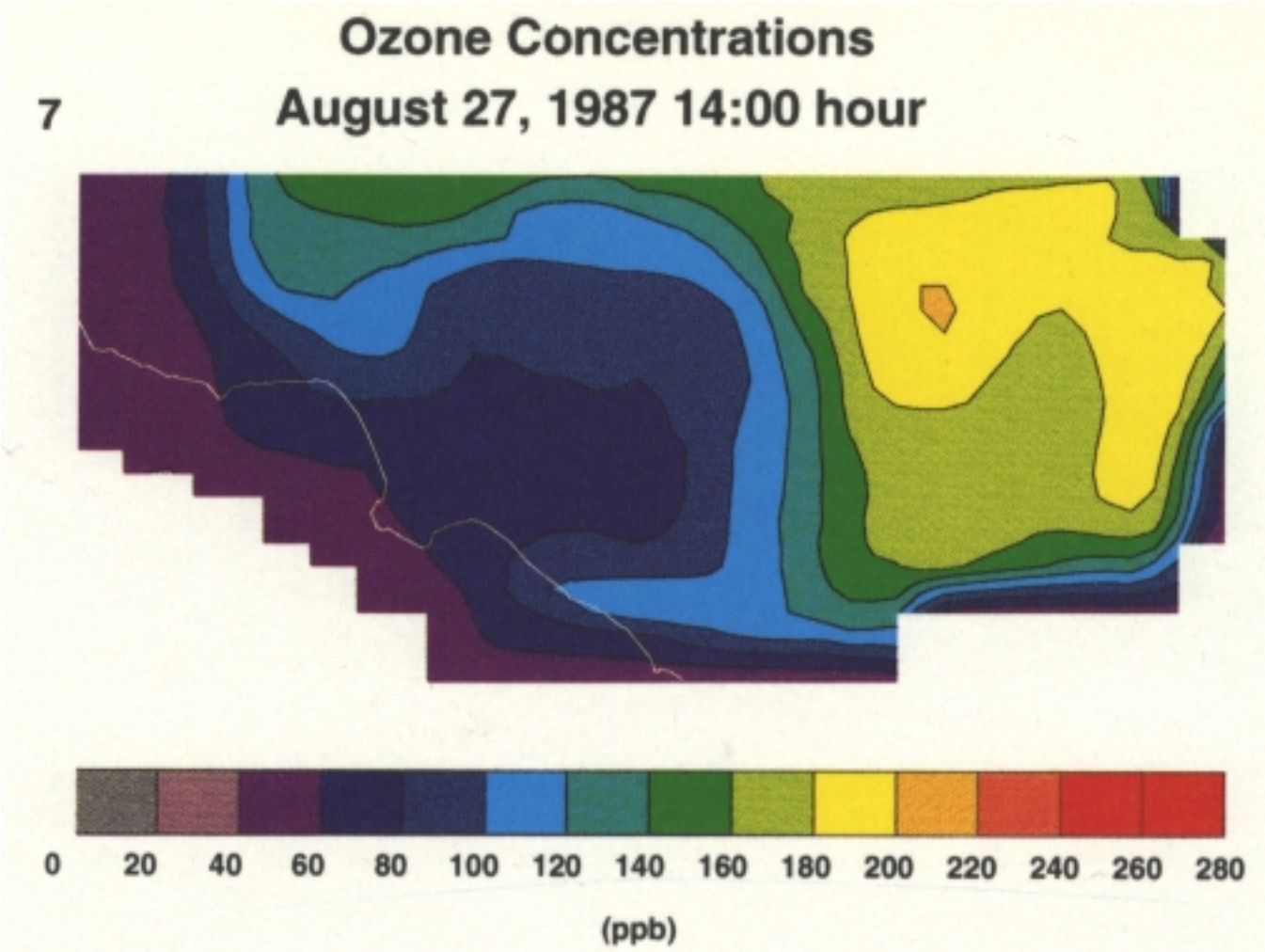


Fig. 7. Ozone concentrations at 14:00 h (27 August 1987) predicted by the CIT model using the Galerkin advection solver and filter parameters: $K=3$, $\mu=0.2$, $m=2$, and $n=4$.

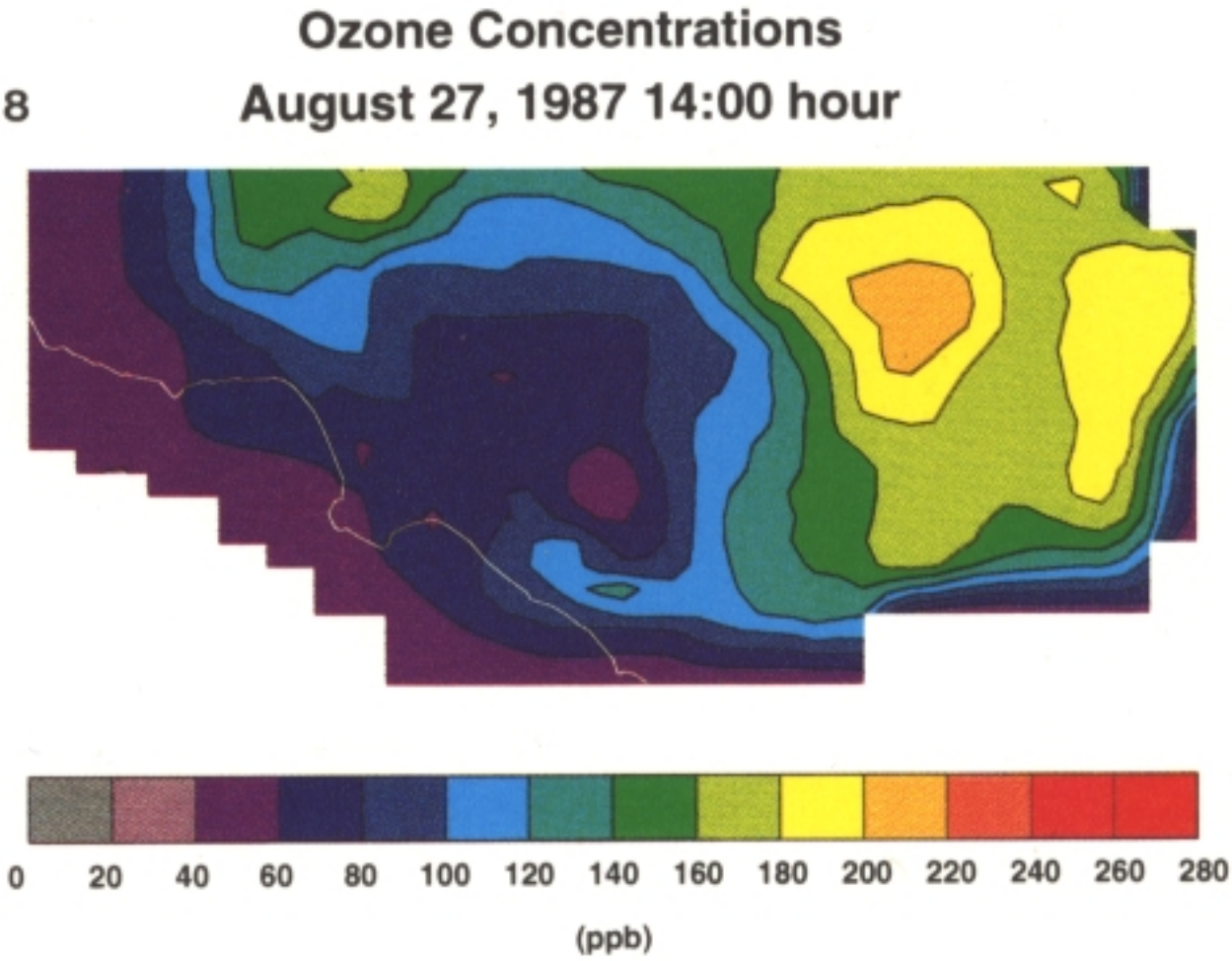


Fig. 8. Ozone concentrations at 14:00 h (27 August 1987) predicted by the CIT model using the Galerkin advection solver and filter parameters: $K=1$, $\mu=0.1$, $m=1$, and $n=2$.

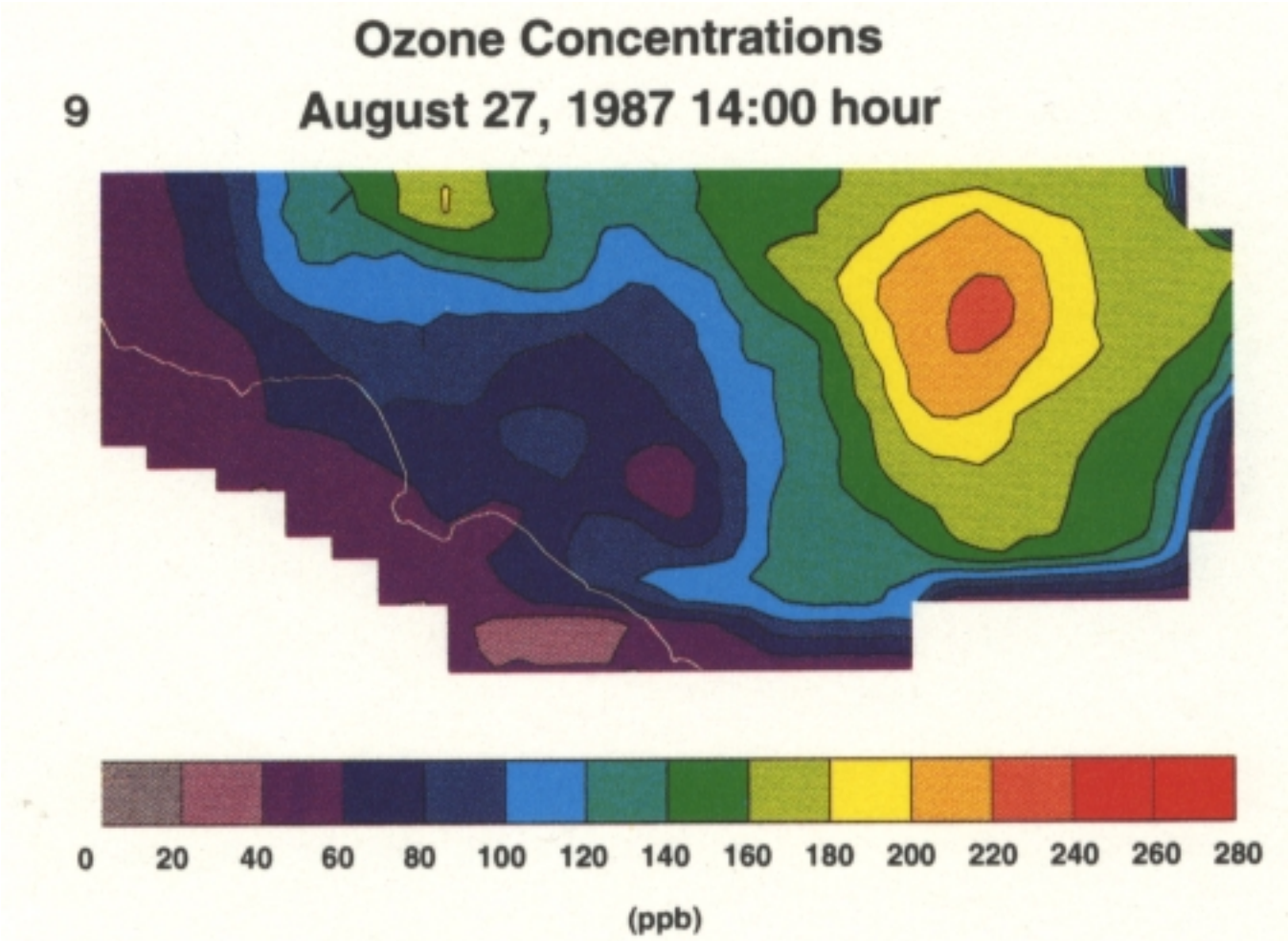


Fig. 9. Ozone concentrations at 14:00 h (27 August 1987) predicted by the CIT model using the ASD advection solver and filter parameters: $K=1$, $\mu=0.1$, $m=1$, and $n=2$.

numerical method used to carry the transport step, the degree of modularity and portability desired, and the number of nodes available.

The following are desirable characteristics for the ideal implementation of transport solvers in a parallel environment:

(1) *Minimum communication.* The message passing time spent on interprocessor communications is expensive relative to computational time. It is desirable to pass a single long message rather than an equivalent message size using various send/receive operations, the reason being that every time a message is passed there is initialization overhead involved in the process. The ideal implementation would minimize the data flow among nodes.

(2) *Portability.* The parallel implementation should contain only simple synchronous or asynchronous sends and receives. It should avoid global message operations such as broadcast, global sums, etc. The reason for such constraints is that all the message passing protocols (e.g. PVM, P4, NX, EXPRESS) support such operations allowing the code to be easily ported among different parallel compilers.

(3) *Modularity.* The code should be modular in the sense that the transport solver of the model is a single routine that can be easily replaced by a different algorithm. In addition, the parallel code should be written in such a way that the data flow between nodes, for any number of nodes, remains unmodified for any advection solver. Modularity should be maintained while keeping the internode communication to a minimum as described in equation (1).

(4) *Load balance.* To perform a horizontal transport step it is necessary for a particular node to receive nonlocal data from one or more other neighboring nodes. At times, some of the neighbor nodes are still performing other computations and cannot send the data at the exact time of request. As a result, the node requesting the data stays idle until the neighbor node is able to perform the send operation. The ideal parallel implementation would minimize such idle time spend between sends and receives of data among all nodes.

Extended Arrays, EA (Fox *et al.*, 1988), is an implementation that minimizes internode communication while optimally maintaining the CPU load balance. The name implies that local concentration arrays are extended at the boundaries to make room for the data needed. Each node sends the boundary concentration data to all its neighbors. The number of grid cells sent as boundaries is the minimum one imposed by the advection solver used. Each node will, of course, receive data from its neighbors. After the data are received all the nodes perform the transport step locally in small domains. The use of EA in AQMs is recommended when the algorithm used requires a few grid cells located in neighbor nodes and speed is the primary concern. The EA approach to implementing the transport is not modular. When using EA techniques, changing the algorithm requires extensive modifications to the parallel implementation.

To overcome modularity problems presented by the EA approach one can designate a particular node that receives not only the boundary of grid cells located in the neighbor nodes, but all the grid cells of a particular row or column. After receiving the data, the designated transport node performs the transport computations and then sends back the new grid cell concentrations to the appropriate nodes. All columns or rows are being processed simultaneously at a given horizontal transport step. This approach is named Designated Transport Node, DTN. The results presented by Dabdub and Seinfeld (1994) were based on a DTN approach to implementing the transport in parallel. The advantages of DTN are that it is easy to code and is fully modular. DTN is easy to code since the programmer is not concerned about nodes with special cases. When implementing EA for a variable number of processors the programmer must be careful about corner nodes that have no neighbors. DTN is fully modular, like the sequential case, because the node performing the transport step has all the grid cell concentrations in case they are needed by a specific transport solver. On the other hand, DTN has two main disadvantages: it is slower than EA and it is not well balanced. To gain modularity DTN requires a greater number of internode communication than EA. The increase in communication traffic decreases performance results. In addition, if the number of rows or columns is smaller than the number of nodes available, DTN presents load imbalancing that affects performance further. Typical urban AQMs have less than 100 rows or columns. If there are 512 nodes available DTN will leave most of the nodes idle when performing the transport computations.

The Dynamical Balance, DYB, approach is an attempt to maintain the modularity of DTN while increasing its performance by decreasing the idle node time. Instead of having a predetermined node to perform the transport computations for all the species and layers of a particular row or column, DYB assigns the first $N$ ground rows or columns to be performed in the first $N$ nodes, where $N$ is the number of available nodes. The next $N$ rows or columns continue to be evenly distributed among the available $N$ nodes. This distribution continues until all rows or columns have been assigned a transport node. Figure 10 illustrates the X-transport distribution of a computational domain with three layers containing five rows each among three nodes. The number assigned to each row corresponds to the node number in which the transport computations will be performed for the row. Note that, to maintain the load as balanced as possible, the distribution of rows or columns in layer $m$ starts with the node number having fewer rows or columns assigned in layer $(m-1)$. As it can be seen, the DYB approach provides a well balanced load while maintaining the modularity of the code. Nevertheless, the programming required to implement DYB for the

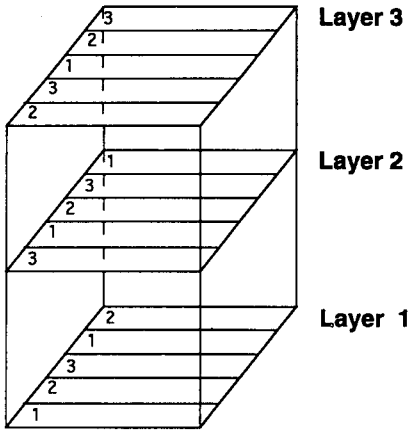


Fig. 10. Example of row distribution to perform X-transport computations using a DYB approach to implement transport. The example is based on a three-layer domain containing five rows. The numbers shown in each row denote the processor number where the transport computations for such row are to be performed.

general case (any number of rows, columns, layers and nodes) requires more effort than EA or DTN. The main advantage of DYB is that it reduces the overall idle time of the nodes, especially in the massively parallel regime.

Figure 11 shows the CPU time vs the number of processors for a DTN and DYB approach to the parallel implementation of the CIT model using the Galerkin solver. The model was run on the Intel Touchstone Delta using NX as the message passing protocol. Figure 11 shows that similar times are required using the two different approaches to implement the algorithm in parallel. It is observed that the CPU time starts to flatten in the high number of processors regime. The reason for such behavior is that the workload required by the Galerkin solver is so light that it does not benefit from having extra nodes to perform it. Indeed, it is observed that the case for 256 nodes is slightly slower than the 128 nodes since at that point the nodes start to interfere with each other to efficiently carry out the computations. When using the Galerkin algorithm, the nodes spend a small fraction of their time performing the transport step. Therefore, the potential idle times induced by DTN are not significant. In general, when using any transport solver that it is not computationally intensive, the simple and easier to code DTN approach should be followed. The best speed-up observed for the Galerkin algorithm is 30.01 when using the DYB approach and 128 nodes.

Figure 12 shows the CPU time versus the number of processors for a DTN and DYB approach to the parallel implementation of the CIT model using the ASD solver. It is observed that the CPU also flattens in the high number of processors regime for the DTN approach. In this case, the flat region does not occur because the workload is light, but because the number of processors at that point exceeds the number of rows or columns in the computational grid. As a result, increasing the number of nodes for the DTN approach

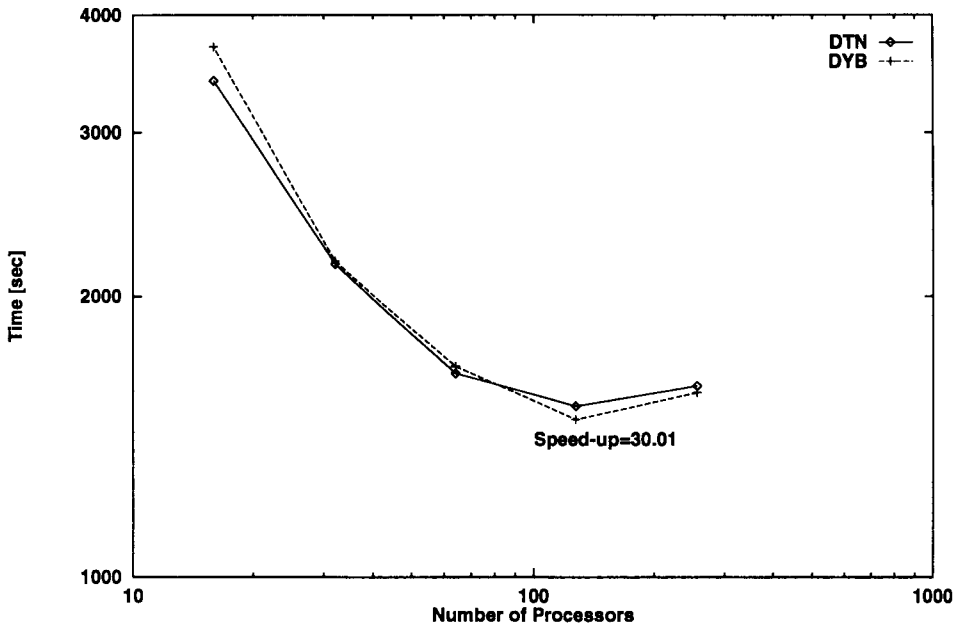


Fig. 11. CPU time for a 24-h simulation of the South Coast Air Basin vs number of nodes for the Galerkin transport implementation of the CIT model using DTN and DYB implementation strategies.

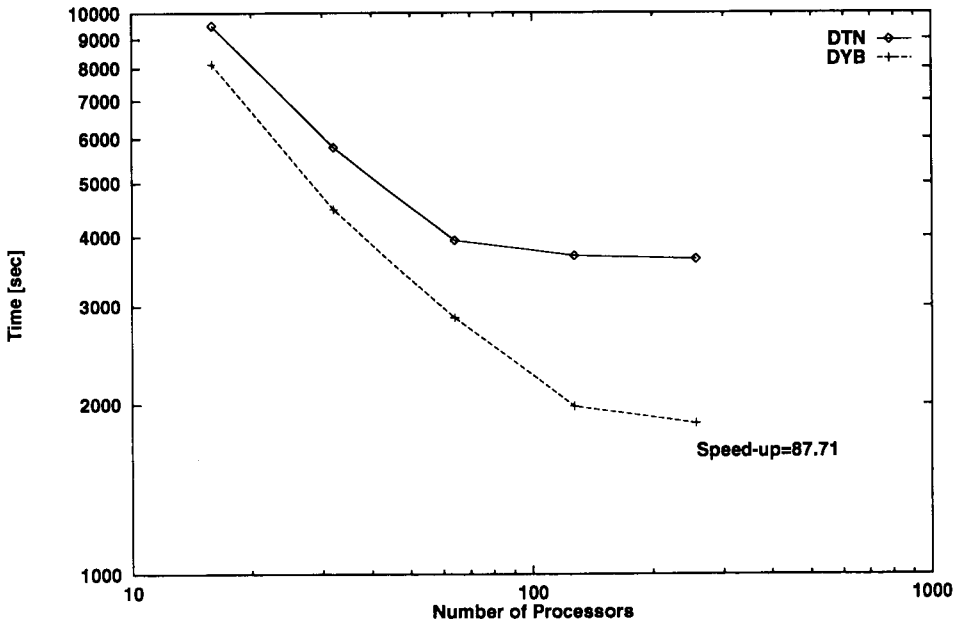


Fig. 12. CPU time for a 24-h simulation of the South Coast Air Basin vs number of nodes for the ASD transport implementation of the CIT model using DTN and DYN implementation strategies.

only increases their idle time. For the ASD method the DYB approach is clearly superior, especially in the massively parallel regime. The best speed-up factor observed for the ASD algorithm is 87.71 when using the DYB approach and 256 nodes. In general, when using a computationally intensive transport solver and performing the computations using a high number of nodes the DYB approach should be followed.

## 7. CONCLUSIONS

Our results confirm those of Chock that the ASD method stands as the best advection scheme tested based on its mass distribution ratio, mass conservation ratio, average absolute error, and peak preserving properties. Its high accuracy, however, is achieved at the price of a greater computation time that might be unacceptable on a sequential machine. However, with the use of parallel computers the computational time is substantially reduced. Indeed, it is well known that using a more CPU-intensive algorithm for a given amount of interprocess communication results in greater speed-ups on a parallel environment. A typical 24-h simulation using a Galerkin solver on the CIT model takes about 25 min using 256 nodes on the Touchstone Delta. The same run, using the more accurate ASD solver takes about 30 min using 256 nodes on the Touchstone Delta.

The second-best algorithm tested was the finite-element Galerkin scheme. If computation time is a constraint, as when running AQMs on sequential machines, the Galerkin algorithm is recommended. The Smolarkiewicz advection solver was found to produce poor results for the tests performed. Therefore, the use of this method in the UAM model should be reconsidered.

For Courant numbers relevant for air pollution modeling (greater than 0.1) the Forester filter performs the best for all the cases studied. The Forester filter parameters are found to have a significant impact on the results of the advection solver. In particular, we have been able to optimize the choice of the filter parameters used in the CIT model as $K=1$, $\mu=0.1$, $m=1$, and $n=2$. Implementing these filter parameters into the CIT model leads to ozone peaks that are not present with the previous filter parameters. The ASD method confirms the validity of such peaks, and produces a greater concentration at the peaks as expected.

When implementing the advection solver in a parallel environment the EA approach should be used if speed is the greatest concern. DTN offers a fully modular approach that is easy to code, but it sacrifices some performance. DTN is recommended when the number of nodes available is smaller than the number of rows or columns in the computational region of the AQM and a fast transport solver is being used such as Galerkin. Finally, the DYB approach is recommended when computing in the massively parallel regime while using intensive transport solvers such as ASD.

*Acknowledgments*—This work was supported in part by a grant from the IBM Environmental Research Program. Any

opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the IBM corporation. The information in this document has been funded wholly or in part by the United States Environmental Protection Agency High Performance Computing and Communication Program under cooperative agreement CR 822051-01-0. This research was performed in part using the Intel Delta System operated by Caltech on behalf of the Concurrent Supercomputing Consortium.

## REFERENCES

Bartnicki J. (1989) A simple filtering procedure for removing negative values from numerical solutions of the advection equation. *Envir. Software* **4** (4), 187–201.

Bott A. (1989a) A positive definite advection scheme obtained by nonlinear renormalization of the advective fluxes. *Mon. Wea. Rev.* **117,** 1006–1015.

Bott A. (1989b) Reply. *Mon. Wea. Rev.* **117,** 2633–2636.

Canosa J. and Gazdag J. (1976) The Korteweg-de Vries-Burger equation. *J. comput. Phys.* **23,** 393–403.

Carver M. B. and Hinds H. W. (1978) The method of lines and the advection equation. *Simulation* **31,** 59–69.

Chapman M. (1981) FRAM—Nonlinear damping algorithms for the continuity equation. *J. comput. Phys.* **44,** 84–103.

Chock D. P. (1985) A comparison of numerical methods for solving the advection equation—II. *Atmospheric Environment* **19,** 571–586.

Chock D. P. (1991) A comparison of numerical methods for solving the advection equation—III. *Atmospheric Environment* **25A,** 853–871.

Chock D. P. and Dunker A. M. (1983) A comparison of numerical methods for solving the advection equation. *Atmospheric Environment* **17,** 11–24.

Dabdub D. and Seinfeld J. H. (1994) Air quality modeling on massively parallel computers. *Atmospheric Environment* **28,** 1679–1687.

Emde D. V. D. (1992) Solving conservation laws with parabolic and cubic splines. *Mon. Wea. Rev.* **120,** 82–92.

Engquist B., Lötstedt P. and Sjögreen B. (1989) Nonlinear filters for efficient shock computation. *Math. Comput.* **52,** 509–537.

Forester C. K. (1977) Higher order monotonic convective difference schemes. *J. comput. Phys.* **23,** 1–22.

Fox G., Johnson M., Lyzenga G., Otto S., Salmon J. and Walker D. (1988) *Solving Problems on Concurrent Processors, Vol. I.* Prentice-Hall, Englewood Cliffs, NJ.

Gazdag J. (1973) Numerical convective schemes based on accurate computation of space derivatives. *J. comput. Phys.* **13,** 100–113.

Harley R. A., Russell A. G., McRae G. J., Cass G. R. and Seinfeld J. H. (1993) Photochemical modeling of the southern California air quality study. *Envir. Sci. Technol.* **27,** 378–388.

Kahaner D., Moler C. and Nash S. (1989) *Numerical Methods and Software.* Prentice-Hall, Englewood Cliffs, NJ.

Hov Ø., Zlatev Z., Berkowicz R., Eliassen A. and Prahm L. P. (1989) Comparison of numerical techniques for use in air pollution models with nonlinear chemical reactions. *Atmospheric Environment* **23,** 967–983.

McRae G. J., Goodin W. R. and Seinfeld J. H. (1982) Numerical solution of the atmospheric diffusion equation for chemically reacting flows. *J. comput. Phys.* **45,** 1–42.

Morris R. E., Myers T. C. (1990) User's guide for the Urban Airshed Model—Vol. I: User's manual for UAM (CB-IV). U.S. Environmental Protection Agency (EPA-450/4-90-007a).

Odman M. T. and Russell A. G. (1993) A nonlinear filtering algorithm for multi-dimensional finite element pollutant advection schemes. *Atmospheric Environment* **27A,** 793–799.

Oran E. S. and Boris J. P. (1987) *Numerical Simulation of Reactive Flow.* Elsevier, New York.

Pepper D. W. and Long P. E. (1978) A comparison of results using second-order moments with and without width correction to solve the advection equation. *J. appl. Met.* **17,** 228–233.

Roache P. J. (1978) A pseudo-spectral FFT technique for non-periodic problems. *J. comput. Phys.* **27,** 204–220.

Rood R. B. (1987) Numerical advection algorithms and their role in atmospheric transport and chemistry models. *Rev. Geophys.* **25,** 71–100.

Schere K. L. (1983) An evaluation of several numerical advection schemes. *Atmospheric Environment* **17,** 1897–1907.

Schiesser W. E. (1991) *The Numerical Method of Lines.* Academic Press, San Diego, CA.

Sheih C. M. and Ludwig F. L. (1985) A comparison of numerical pseudodiffusion and atmospheric diffusion. *Atmospheric Environment* **19,** 1065–1068.

Smolarkiewicz P. K. (1983) A simple positive definite advection scheme with small implicit diffusion. *Mon. Wea. Rev.* **111,** 479–486.

Tran K. T. and Mirabella V. A. (1991) A comparison of advection schemes in existing photochemical grid models. Air and Waste Management Assoc., No 91-66.2, 84th Annual meeting AWMA, Vancouver, BC, 16–21 June.

Wengle H. and Seinfeld J. H. (1978) Pseudospectral solution of atmospheric diffusion problems. *J. comput. Phys.* **26,** 87–106.

Yanenko N. N. (1971) *The Method of Fractional Steps.* Springer, New York.

## APPENDIX

### *Description of filters*

*Bartnicki filter.* The Bartnicki filter (Bartnicki, 1989) uses the simplest redistribution strategy. It consists of adding all the negative mass and subtracting equal amounts from all positive values present in the distribution. The total amount of the mass subtracted from the positive values equals the total initial negative mass. The filter does not modify zero values of the distribution. Note that if all distribution values are nonnegative the filter does not alter the distribution. It has been found that two iterations are sufficient to remove all the negative mass in the cases studied. While, this strategy guarantees the absence of negative values present after the filtering step, the filter does not sufficiently smooth the solution's positive oscillations.

*Chapman filter.* FRAM (Filtering Remedy and Methodology) was presented by Chapman (1981). The idea behind the filter is to introduce a strong nonlinear dissipation to the advection equation to dampen spurious oscillations. The FRAM algorithm can be outlined as follows:

(1) Calculate a provisional advanced time solution $\tilde{C}_i^{t+\Delta t}$ using a higher-order scheme.

(2) Calculate local bounds on the advanced time solution. In one dimension and constant wind velocity the bounds are

$$C_{i\,\min}^t = \min(C_{i-1}^t, C_i^t, C_{i+1}^t) \tag{A1}$$

$$C_{i\,\max}^t = \max(C_{i-1}^t, C_i^t, C_{i+1}^t). \tag{A2}$$

(3) Introduce local diffusion where the provisional solution is not within the bounds. Conserving $C$, the local diffusion is introduced as

$$C_i^{t+\Delta t} = \tilde{C}_i^{t+\Delta t} + (\alpha_i^t + \alpha_{i+1}^t)(C_{i+1}^t - C_i^t) + (\alpha_i^t + \alpha_{i-1}^t)(C_{i-1}^t - C_i^t) \tag{A3}$$

where $\alpha_i^t$ can be thought of as a diffusion coefficient that is computed using the provisional concentrations bounds computed in equations (A2) and (A3).

*Engquist filter.* Engquist *et al.* (1988) have proposed a series of nonlinear filters of differing complexity. The filters do not introduce any diffusion to the advection equation. The basic algorithm behind all the filters proposed is

(1) Scan $C_i^{t+\Delta t}$ and correct the $i$-values that are local maxima or local minima. Corrections are made by decreasing the maxima and by increasing the minima.

(2) If point $i$ is to be corrected, the same correction must be subtracted from point $i-1$ or $i+1$, whichever has the greatest distance to $C_i^{t+\Delta t}$.

(3) No value may be corrected so that it is a new extremum.

The filter implemented in this study is algorithm 2.4 described by Engquist *et al.* (1988). It makes the smallest correction to the distribution while still being TVD.

*Forester filter.* To dampen the spurious oscillations the Forester filter (Forester, 1977) introduces a local diffusion to ensure that local extrema are separated by $2n$ mesh intervals. The filter uses $n$ as a parameter, as well as $m$, $K$, and $\mu$. $K$ is the number of filtering iterations and $\mu$ is a dimensionless diffusion coefficient. The parameters $m$ and $n$ determine the noise wavelength to be filtered. These free parameters are problem dependent and they might significantly affect the performance of the filter. The Forester filter is described by the following iterative scheme:

$$C_i^{k+1} = C_i^k + \frac{\mu}{2}[(C_{i+1} - C_i)(\psi_i + \psi_{i+1}) - (C_i - C_{i-1})(\psi_i + \psi_{i-1})]^k \tag{A4}$$

where $C_i^{k+1}$ is the value of $C_i$ after $k$ iterations of the filter. The values of $\psi_i$ are either 0 or 1 and determine the points at which smoothing occurs. They are computed from the filter parameters $m$ and $n$.