ELSEVIER

Applied Numerical Mathematics 20 (1996) 191–209

APPLIED NUMERICAL MATHEMATICS

An implicit–explicit approach for atmospheric transport-chemistry problems

J.G. Verwer*, J.G. Blom, W. Hundsdorfer

CWI, P.O.Box 94079, 1090 GB Amsterdam, Netherlands

Abstract

We investigate numerical algorithms for use in air pollution models. The emphasis is on the time integration aspects in connection with advection, vertical turbulent diffusion and stiff chemistry. The time integration scheme considered is a second-order implicit–explicit BDF scheme which handles advection explicitly and vertical turbulent diffusion and chemistry implicitly. The investigation is divided into three parts. In the first part we propose a Gauss–Seidel technique for the implicit solution of the chemistry and vertical turbulent diffusion. In the second part we discuss stability properties of the implicit–explicit BDF scheme, assuming the third-order upwind biased finite difference discretization of the advection operator. In the third part we apply the implicit–explicit scheme to a 3D test model and discuss vectorization and parallelization aspects.

Keywords: Long range transport air pollution models; Time-dependent advection–diffusion reaction; Numerical methods; Vectorization; Parallelization

1. Introduction

Air pollution models take into account many physical processes. From the numerical point of view, important processes are the chemical transformations, advective transport, caused by horizontal wind mainly, and vertical transport caused by turbulent diffusion. This paper is devoted to a numerical study of a 3D model of the form

$$\frac{\partial \rho}{\partial t} + \operatorname{div}(\underline{u}\rho) = \frac{\partial}{\partial \sigma}\left(K\frac{\partial \rho}{\partial \sigma}\right) + r(t,\rho) \tag{1}$$

in spherical coordinates, where $\rho = \rho(\lambda, \phi, \sigma, t)$ is a vector in $\mathbb{R}^m$ of m concentration values, $\underline{u} = ue_\lambda + ve_\phi$ is the horizontal velocity vector with e_λ and e_ϕ the unit vectors on the sphere in

* Corresponding author. E-mail: janv@cwi.nl.

SSDI 0168-9274(95)00126-3

the longitude (λ) and the latitude (ϕ) directions, respectively, and K is a scalar turbulent diffusio coefficient in the vertical direction (σ). The horizontal divergence operator is given by, [18],

$$\mathrm{div}(\underline{u}\rho) = \frac{1}{\alpha \cos\phi}\left(\frac{\partial}{\partial\lambda}(u\rho) + \frac{\partial}{\partial\phi}(v\rho\cos\phi)\right), \tag{2}$$

where α is the radius of the earth. Vertical advection and horizontal diffusion can be added withou essential numerical difficulties. The vector function $r(t,\rho)$ defining the chemical transformations emission and dry deposition, has the special form

$$r(t,\rho) = P(t,\rho) - L(t,\rho)\rho, \tag{3}$$

where $P(t,\rho)$ is the vector of production terms and $L(t,\rho)\rho$ the vector of loss terms with $L(t,\rho$ a diagonal matrix. For many species, the reciprocal of their entry in L is a good approximation o the physical time constant or characteristic reaction time. In virtually all applications, the range o reaction times is so large that we have to face the difficulty of stiffness [8].

Our numerical study considers a scheme which is derived employing the method of lines (MOL) The integration method is based on a second-order implicit–explicit BDF formula which handle advection explicitly, and chemistry and vertical diffusion implicitly. The numerical study focuses o three points. The first is taken up in Section 2 and concerns the solution of chemistry and verti cal diffusion. Normally, for stability reasons, stiffness impedes the use of some form of Newto iteration for computing the implicitly defined solutions. In [13,14] we have shown, for differer box models and using the implicit second-order BDF formula, that for atmospheric chemistry prob lems the simpler Gauss–Seidel iteration can be used with greater or competitive efficiency. In fac the Gauss–Seidel method used is truly explicit and is related to simple, explicit quasi-steady-stat approximation (QSSA) schemes to which it compares very favorably also [14]. An additional ad vantage is the much lower memory demand compared to a Newton method. Here we show that th Gauss–Seidel iteration from [13,14] can be effectively extended to the coupled chemistry diffusio case.

In Section 3, the second point is addressed, namely the stability of the implicit–explicit MO scheme, given that the advection operator is discretized by the mass-conservative, flux-limited finit difference scheme proposed in [7]. This finite difference scheme is based on the third-order upwin biased discretization (the $\kappa = 1/3$ scheme in the terminology of van Leer) and has been found to b very suitable for our application (see also [1,3]). We give results of a Fourier–von Neumann analys which show that the explicit, two-step advection scheme yields stability limits sufficiently large fo the application of the complete implicit–explicit scheme. Because the use of the Gauss–Seidel metho renders the diffusion chemistry computation explicit also, except for tridiagonal matrix inversion the combined implicit–explicit approach results in an efficient, almost explicit, process for models o type (1).

Section 4 deals with the third point, namely the accuracy and efficiency performance of the implicit explicit MOL scheme when applied to a 3D test problem. Efficiency is of utmost importance an since we currently use a Cray C98/4256 (4 vector processors, 256 Mw shared memory), attentio will be paid to vectorization and parallelization of the advection and diffusion chemistry computatio in 3D applications.

2. The Gauss–Seidel process

First consider the 1D diffusion chemistry problem

$$\frac{\partial \rho}{\partial t} = \frac{\partial}{\partial \sigma}\left(K\frac{\partial \rho}{\partial \sigma}\right) + r(t,\rho), \quad t > t_0,\ 0 < \sigma < \sigma_H, \tag{4}$$

supplemented with the initial condition $\rho(\sigma, t_0) = \rho^0(\sigma)$ and the boundary conditions

$$\left(K\frac{\partial \rho}{\partial \sigma}\right)(0,t) = 0, \qquad \left(K\frac{\partial \rho}{\partial \sigma}\right)(\sigma_H,t) = 0. \tag{5}$$

2.1. The discretization

The vertical turbulent diffusion term is discretized on the nonuniform cell-centered grid

$$\Omega_V = \{\sigma_k : \sigma_1 = \tfrac{1}{2}\Delta\sigma_1,\ \sigma_k = \sigma_{k-1} + \tfrac{1}{2}(\Delta\sigma_{k-1} + \Delta\sigma_k),\ 2 \leqslant k \leqslant N_\sigma\}, \tag{6}$$

such that the following ODE system is obtained. For $1 \leqslant k \leqslant N_\sigma$

$$\frac{dc_k}{dt} = f_k(t,c) \doteq d_k(t,c) + r(t,c_k), \quad t > t_0,\ c_k(t_0) = \rho^0(\sigma_k), \tag{7}$$

where $c(t)$ is the complete grid function on Ω_V, $c_k(t) \approx \rho(\sigma_k, t)$ and

$$d_k(t,c) = \frac{8K_k^+\dfrac{c_{k+1}-c_k}{\Delta\sigma_{k+1}+\Delta\sigma_k} - 8K_k^-\dfrac{c_k - c_{k-1}}{\Delta\sigma_k + \Delta\sigma_{k-1}}}{\Delta\sigma_{k+1} + 2\Delta\sigma_k + \Delta\sigma_{k-1}}, \quad 1 \leqslant k \leqslant N_\sigma, \tag{8}$$

with $K_k^\pm = K(t, (\sigma_k + \sigma_{k\pm1})/2)$ and $\Delta\sigma_0 = \Delta\sigma_1$, $\Delta\sigma_{N_\sigma+1} = \Delta\sigma_{N_\sigma}$. Note that K is evaluated halfway between the cell centers, rather than at the cell boundaries, to obtain a consistent discretization of (at least) order one. The boundary conditions are incorporated by putting $K_k^- = 0$ for $k = 1$ and $K_k^+ = 0$ for $k = N_\sigma$. Also note that $c_k(t)$ is a vector in $\mathbb{R}^m$ and that the diffusion operator introduces no coupling between different species. For each of the species, the semi-discrete diffusion operator is equal and represented by the same tridiagonal matrix. The species are coupled through the chemistry term $r(t, c_k)$.

For the time integration, we consider the two-step BDF formula in variable step form

$$c_k^{n+1} = C_k^n + \gamma\tau f_k(t_{n+1}, c^{n+1}), \quad 1 \leqslant k \leqslant N_\sigma,\ n \geqslant 1, \tag{9}$$

where $c_k^n \approx c_k(t_n)$ and

$$C_k^n = \left((1+q)^2 c_k^n - q^2 c_k^{n-1}\right)/(1+2q), \tag{10}$$

$\tau = t_{n+1} - t_n$, $\gamma = (1+q)/(1+2q)$ and $q = (t_{n+1} - t_n)/(t_n - t_{n-1})$. The initial vector $c_k^0 \doteq c_k(t_0)$, and c_k^1 is assumed to be defined by the first-order implicit Euler rule. This combination yields second-order accurate time stepping which for atmospheric transport applications is sufficient in view of the modest accuracy requirement. Generally, a relative accuracy better than 1% is superfluous.

2.2. The Gauss–Seidel iteration

Suppressing the temporal index $n+1$ and t_{n+1} for notational convenience, we write (9) as

$$c_k = C_k^n + \gamma\tau d_k(c) + \gamma\tau P(c_k) - \gamma\tau L(c_k)c_k, \quad 1 \leqslant k \leqslant N_\sigma. \tag{11}$$

Let $c_k^{(j)}$ be the jth component of c_k and introduce, for $j = 1, \ldots, m$, the following vectors on Ω_V:

$$\boldsymbol{c}^{(j)} = [c_1^{(j)}, \ldots, c_{N_\sigma}^{(j)}]^T, \qquad \boldsymbol{P}^{(j)}(\boldsymbol{c}) = [P^{(j)}(c_1), \ldots, P^{(j)}(c_{N_\sigma})]^T,$$

and, similarly,

$$\boldsymbol{C}^{(j)} = [C_1^{(j)}, \ldots, C_{N_\sigma}^{(j)}]^T.$$

The vector $\boldsymbol{c}$ contains all vectors $\boldsymbol{c}^{(j)}$, $j = 1, \ldots, m$. Then, introducing the diagonal matrices

$$L^{(j)}(\boldsymbol{c}) = \operatorname{diag}(L^{(j)}(c_1), \ldots, L^{(j)}(c_{N_\sigma})), \quad j = 1, \ldots, m, \tag{12}$$

we may write

$$\boldsymbol{c}^{(j)} = \boldsymbol{C}^{(j)} + \gamma\tau\boldsymbol{A}\boldsymbol{c}^{(j)} + \gamma\tau\boldsymbol{P}^{(j)}(\boldsymbol{c}) - \gamma\tau\boldsymbol{L}^{(j)}(\boldsymbol{c})\boldsymbol{c}^{(j)}, \quad j = 1, \ldots, m, \tag{13}$$

where $\boldsymbol{A}$ is the tridiagonal (diffusion) matrix of order N_σ (cf. (8)). Equivalently, we have

$$\boldsymbol{c}^{(j)} = \left(\boldsymbol{I} - \gamma\tau\boldsymbol{A} + \gamma\tau\boldsymbol{L}^{(j)}(\boldsymbol{c})\right)^{-1}\left(\boldsymbol{C}^{(j)} + \gamma\tau\boldsymbol{P}^{(j)}(\boldsymbol{c})\right), \quad j = 1, \ldots, m, \tag{14}$$

since the inverse of the (diagonally dominant) tridiagonal matrix $\boldsymbol{I} - \gamma\tau\boldsymbol{A} + \gamma\tau\boldsymbol{L}^{(j)}(\boldsymbol{c})$ always exists.

The Gauss–Seidel iteration for approximating $\boldsymbol{c}^{(j)}$, $1 \leqslant j \leqslant m$, is carried out on equation (14) and consists of the following calculations. Let $\boldsymbol{c}_{[i]}$ denote the ith iterate for $\boldsymbol{c}$. Then, at step n,

(1) Initial estimation: $i = 0$, $\boldsymbol{c}_{[i]} := \max(0, \boldsymbol{c}^n + q(\boldsymbol{c}^n - \boldsymbol{c}^{n-1}))$.

(2) Compute, in the order $j = 1, \ldots, m$:

(a) $\boldsymbol{L}^{(j)}(\boldsymbol{c}_{[i]}), \boldsymbol{P}^{(j)}(\boldsymbol{c}_{[i]})$.

(b) LU-decompose $\boldsymbol{I} - \gamma\tau\boldsymbol{A} + \gamma\tau\boldsymbol{L}^{(j)}(\boldsymbol{c}_{[i]})$. (15)

(c) Backsolve in (14) for $\boldsymbol{c}_{[i+1]}^{(j)}$.

(d) Update $\boldsymbol{c}_{[i]} := (\boldsymbol{c}_{[i+1]}^{(1)}, \ldots, \boldsymbol{c}_{[i+1]}^{(j)}, \boldsymbol{c}_{[i]}^{(j+1)}, \ldots, \boldsymbol{c}_{[i]}^{(m)})$.

(3) Set $i := i + 1$. If more iterations are required, then go to (2).

Hence the approximations are corrected specieswise and simultaneously over the grid, such that the diffusion term is treated implicitly. This requires the tridiagonal matrix calculations (2)(b), (c) any time a species is corrected. Thus, except for the tridiagonal matrix calculations, the Gauss–Seidel process is truly explicit. No Jacobian matrices for the chemistry system are computed and no additional storage is required. If there were no diffusion, then this process is completely identical to that used for the box models in [13,14]. On the other hand, without chemistry the diffusion is treated implicitly in the usual way. This means that method (15) differs from the well-known classical nonlinear Gauss–Seidel method since this classical method does not distinguish between the diffusion and reaction terms and would be applied directly to (11).

2.3. Numerical illustration

This section presents numerical results for a large, real life test problem obtained with a code based on (9). For the approximate solution of the implicit relations, the Gauss–Seidel (GS) iteration (15) and modified Newton (MN) iteration have been implemented. The stepsize strategy is similar to that in [13,14] and many stiff ODE codes and identical for the GS and MN iterations. To save space we omit details here. We have not implemented an iteration strategy for GS and hence prescribe the number of GS iterations. In our experience, this works well using only a few iterations. A standard LINPACK [5] linear band solver is used for the block-tridiagonal system arising within the MN iteration. We use an analytical Jacobian matrix for the chemistry part. This matrix is sparse which means that Jacobian evaluations are cheap. Unfortunately, for the block-tridiagonal system the fill-in of the matrix factorization is almost complete, ruling out using a sparse system solver as is successfully applied in [15] to box models.

Our test problem of type (4)–(5) is based on the state-of-the-art EMEP MSC-W ozone chemistry (140 reactions between 66 species [10,11]). In [14] we have used the same chemistry in box model tests. The experiments reported here extend these to column model tests in a straightforward way by adding a vertical turbulent diffusion term. Hence $K(t,\sigma)$ depends on the mixing height which depends on the time of day. Photolysis rates undergo a discontinuity at sunset and sunrise. This, and the space-time dependence of $K(t,\sigma)$ causes large local concentration gradients. A nonuniform space grid is used which contains 40 points. This grid covers an air column of height $\sigma_H = 2000$ m. We use the ODE error on this grid in the comparisons. Hence we pay no separate attention to the spatial accuracy, which is approximately 1% in the error norm introduced below in (16). For the grid with 40 points, the dimension of the banded linear system arising in the MN iteration equals $mN_\sigma = 2640$ with a bandwidth equal to $2m+1 = 133$.

The time integration over 112 hours starts at sunrise (04.00 hours) on day one ($t_0 = 14,400$ sec.) and ends at sunset (20.00 hours) on day five ($t = 417,600$ sec.). In all integrations, the 112-hour interval is divided into 56 two-hour intervals, on each of which we restart the integration with the one-step backward Euler formula using a tenfold smaller stepsize than on the previous step. This division into 56 subintegrations was also used in [14] and is in accordance with regular changes in model coefficients and input. Such changes can introduce a discontinuity (as at sunset and sunrise), motivating the many restarts. Note that, if the current procedures were used in an operator splitting scheme, then frequent restarts would also be made.

We have carried out two different experiments. The first serves to provide insight in the accuracy efficiency performance of GS iteration when varying RTol (the relative tolerance parameter for the variable time stepsize selection) and the number of GS iterations. The second serves to compare GS iteration with MN iteration. Efficiency is measured by CPU time and accuracy by the number of correct digits

$$\mathrm{SDA} = -\log\left(\frac{1}{66}\sum_{j=1}^{66}\left[\sum_{n=8}^{56}\sum_{k=1}^{40}(\mathrm{sol}_{kj}^n - \mathrm{app}_{kj}^n)^2\right]^{1/2} \Big/ \left[\sum_{n=8}^{56}\sum_{k=1}^{40}(\mathrm{sol}_{kj}^n)^2\right]^{1/2}\right) \tag{16}$$

where sol_{kj}^n denotes a highly accurate approximation to the ODE solution on the grid and app_{kj}^n the numerical solution. The times t_n are restricted to $t_n = 14400 + 7200n$ with $8 \leqslant n \leqslant 56$, j runs over

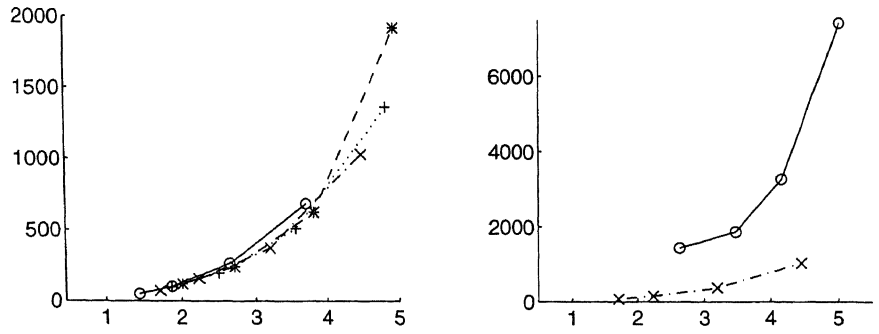

Fig. 1. 1D experiment, CPU sec. versus SDA: (a) GS iteration, (b) GS iteration (×) versus MN iteration (○).

all species, and k runs over the grid. Hence we sample at the end of each 2 hour interval, but for the first time at sunset on the first day because we start at sunrise with an arbitrary initial condition. Thus we assume that we have fully eliminated the initial transients at the first sunset. In (16) we first compute an l_2 error in time and over the grid for each species and then average over all species.

Fig. 1(a) gives results of the first experiment which comprises 16 integrations. Each of the four lines corresponds to a prescribed number of GS iterations (○-2, ×-4, +-6, ∗-8) and connects results for four values of RTol (10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}). This enables us to compare the use of different values of RTol and a fixed number of GS iterations with the use of a different number of GS iterations and a fixed value of RTol. First we notice that decreasing RTol by the chosen factor of 10 also reduces the error by this same factor, approximately. This indicates that the variable stepsize strategy works well. Because the four lines almost coincide, we conclude a good strategy is to keep the number of GS iterations low and to take RTol small, rather than using a large number of iterations and a crude tolerance. Then we do not need a GS iteration strategy and we do not risk performing a number of GS iterations larger than the amount required to reach the accuracy of the implicit second-order BDF method.

Fig. 1(b) gives results of the second experiment. Accuracy is plotted against efficiency for four integrations using RTol = 10^{-1}, 10^{-2}, 10^{-3} and 10^{-4}, both for GS iteration and MN iteration. Here we have fixed the number of GS iterations to 4. In line with the results of the first experiment, we see that MN iteration always results in smaller errors (close to the error of the BDF formula), but clearly at the expense of much higher costs. GS iteration appears to be four to five times more efficient. This strongly favors GS iteration, certainly so for 3D problems where 1D calculations of the type considered here need to be carried out at thousands of points in a horizontal grid.

3. The implicit–explicit two-step BDF scheme

In this section, we present the implicit–explicit two-step BDF scheme for (1) and discuss its stability. We assume that the advection operator (2) is discretized on a cell-centered uniform grid

$$\Omega_H = \{(\lambda_i, \phi_j) : \lambda_i = (i - \tfrac{1}{2})\Delta\lambda,\ \phi_j = (j - \tfrac{1}{2})\Delta\phi\},$$

by means of the mass-conservative, flux-limited, finite difference scheme proposed in [7]. This scheme is based on the third-order upwind biased method which is equivalent to the $\kappa = 1/3$ scheme of van Leer and derived with the aim of producing positive and monotone solutions and little artificial diffusion. It has been found to be very suitable for our application (see [1,3,7]). For the sake of brevity, we refer to [7] for the actual formulas and their derivation.

3.1. The implicit–explicit scheme

Let $c(t)$ denote the semi-discrete grid function on the cell-centered 3D grid $\Omega_H \times \Omega_V$, where Ω_V is defined by (6), with components $c_{ijk}(t)$, now approximating ρ at the grid point $(\lambda_i, \phi_j, \sigma_k)$. Let

$$\frac{dc}{dt} = g(t,c) + f(t,c) \tag{17}$$

denote the associated semi-discrete 3D problem. Components f_{ijk} are defined by (7) and components g_{ijk} represent the numerical advection scheme (boundary conditions for the advection operator are omitted here). Recall that each component c_{ijk} itself is a vector in $\mathbb{R}^m$. Also recall that $f(t,c)$ is only coupled in Ω_V and $g(t,c)$ only in Ω_H. Further, since in the advection operator species are not coupled to one another, we might consider the semi-discrete advection system for each species separately.

Assuming constant stepsizes, for simplicity of presentation, the two-step implicit BDF formula applied to (17) is

$$c^{n+1} = \tfrac{4}{3}c^n - \tfrac{1}{3}c^{n-1} + \tfrac{2}{3}\tau g(t_{n+1}, c^{n+1}) + \tfrac{2}{3}\tau f(t_{n+1}, c^{n+1}), \quad n \geqslant 1. \tag{18}$$

However, we do not wish to integrate (17) implicitly, and replace (18) by the implicit–explicit scheme

$$c^{n+1} = \tfrac{4}{3}c^n - \tfrac{1}{3}c^{n-1} + \tfrac{2}{3}\tau g(t_{n+1}, 2c^n - c^{n-1}) + \tfrac{2}{3}\tau f(t_{n+1}, c^{n+1}). \tag{19}$$

Likewise, for $n = 0$, the implicit Euler rule is replaced by the first-order implicit–explicit Euler rule

$$c^1 = c^0 + \tau g(t_0, c^0) + \tau f(t_1, c^1). \tag{20}$$

Integration schemes of this type are well known, [2,12]. For our application (19) is particularly attractive as the method is only 1D implicit. One step with (19) amounts to computing advection explicitly at all horizontal grids Ω_H, and vertical diffusion and chemistry implicitly and coupled along all vertical grids Ω_V perpendicular to Ω_H, as discussed in Section 2.

The implicit–explicit approach may be viewed as splitting within a method. The additional error introduced by "internally splitting" advection from diffusion and chemistry preferably should be of the same size as the error of the original BDF formula. Substitution of a sufficiently differentiable solution $c(t)$ into (19) yields the local truncation error

$$\frac{2}{9}\tau^3 \frac{d^3}{dt^3}c(t_n) + \frac{2}{3}\tau^3 g'(t_n, c(t_n)) \frac{d^2}{dt^2}c(t_n) + O(\tau^4). \tag{21}$$

Hence the extrapolation in (19) adds the product of the Jacobian g' with the second derivative of c to the original local truncation error, but the local error remains $O(\tau^3)$. Thus, to retain the level of local accuracy, the size of this product should be comparable to the size of the third derivative of the

solution. Alternatively, we can examine the local temporal accuracy of the implicit–explicit formula for the PDE itself, by directly applying the integration formula to (1). The local truncation error expression then becomes

$$\begin{aligned}&\tfrac{2}{9}\tau^3\rho_{ttt}(t_n) + \tfrac{2}{3}\tau^3\,\mathrm{div}(\underline{u}[\,\rho(t_{n+1}) - 2\rho(t_n) + \rho(t_{n-1})\,]) + O(\tau^4)\\ &\quad = \tfrac{2}{9}\tau^3\rho_{ttt}(t_n) + \tfrac{2}{3}\tau^3\,\mathrm{div}(\underline{u}\rho_{tt}(t_n)) + O(\tau^4).\end{aligned} \tag{22}$$

We see that, if the third temporal derivative is of the same size as the divergence of the velocity times the second temporal derivative, then no reduction in local accuracy will result. This observation is of relevance only when the spatial error is negligibly small. Of course, if the spatial error dominates, as is surely the case for our experiments, then "internal splitting" will never harm accuracy.

3.2. Linear stability

Treating advection explicitly obviously has a large impact on stability. To examine this, we consider the linear, constant coefficient system

$$\rho_t + u\rho_\lambda = K\rho_{\sigma\sigma} + M\rho, \tag{23}$$

where K is a scalar and M is a matrix representing the chemistry. Note that, for the purpose of linear stability analysis, it suffices to consider only 1D advection. Conclusions for 2D (and 3D) advection can be drawn immediately from the 1D analysis. Assuming that M is similar to its diagonal eigenvalue matrix, it is sufficient to study the componentwise equations arising in the eigensystem expansion. Using the same notation, we thus proceed with the scalar equation

$$\rho_t + u\rho_\lambda = K\rho_{\sigma\sigma} + \mu\rho. \tag{24}$$

We apply the Fourier method of von Neumann and thus assume that the κ-scheme is applied without limiting and that Ω_V is uniform. The semi-discrete scheme can then be written as

$$\begin{aligned}&\frac{\mathrm{d}c_{i,k}}{\mathrm{d}t} + \frac{u}{12}\frac{c_{i-2,k} - 8c_{i-1,k} + 8c_{i+1,k} - c_{i+2,k}}{\Delta\lambda}\\ &+ \mathrm{sign}(u)\,\frac{u}{12}\,(\Delta\lambda)^3\frac{c_{i-2,k} - 4c_{i-1,k} + 6c_{i,k} - 4c_{i+1,k} + c_{i+2,k}}{(\Delta\lambda)^4}\\ &\quad = K\frac{c_{i,k-1} - 2c_{i,k} + c_{i,k+1}}{(\Delta\sigma)^2} + \mu c_{i,k}.\end{aligned} \tag{25}$$

Fourier analysis leads to the characteristic equation

$$(1 - \tfrac{2}{3}\tau(A_\sigma + \mu))\,\alpha^2 - (\tfrac{4}{3} + \tfrac{4}{3}\tau A_\lambda)\,\alpha + (\tfrac{1}{3} + \tfrac{2}{3}\tau A_\lambda) = 0, \tag{26}$$

where A_λ and A_σ are the advection and diffusion eigenvalues,

$$A_\lambda = -\frac{1}{3}\frac{u}{\Delta\lambda}\left(\sqrt{-1}\sin\theta_\lambda(4 - \cos\theta_\lambda) + \mathrm{sign}(u)(1 - \cos\theta_\lambda)^2\right), \tag{27}$$

$$A_\sigma = \frac{2K}{\Delta\sigma^2}(\cos\theta_\sigma - 1), \tag{28}$$

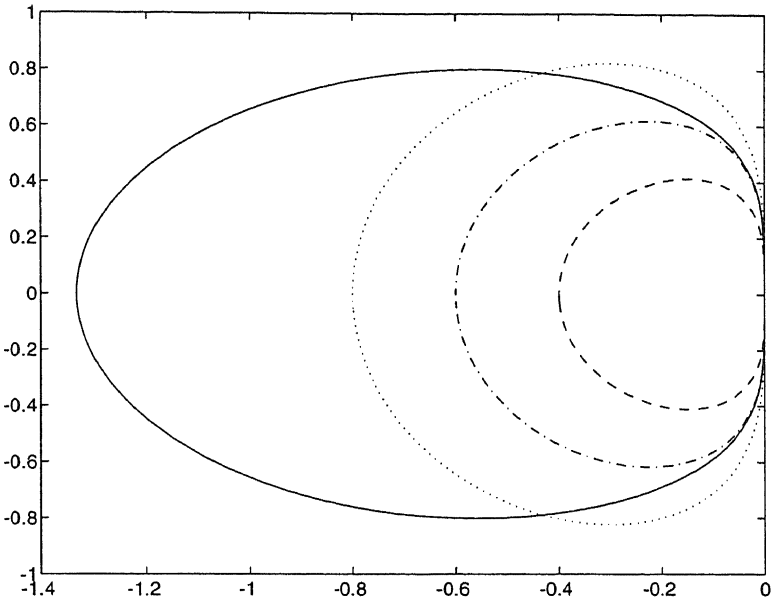

Fig. 2. The stability region S of the explicit two-step scheme (solid line). The dashed, dashdotted and dotted line are the curves for τA_λ for the CFL numbers $\nu = 0.3, 0.45, 0.6$, respectively.

with $\theta_\lambda = \omega_\lambda \Delta\lambda$, $\theta_\sigma = \omega_\sigma \Delta\sigma$ and $|\theta_\lambda|, |\theta_\sigma| \leqslant \pi$. When the term $\operatorname{sign}(u)(1 - \cos\theta_\lambda)^2$ is removed from (27), the advection eigenvalue A_λ for the classical fourth-order central difference scheme results. Then A_λ is purely imaginary. In the upwind case (with the term $\operatorname{sign}(u)(1 - \cos\theta_\lambda)^2$ in place in (27)), A_λ is complex with a negative real part. We first determine the stability region S of the explicit two-step scheme contained in (19),

$$c^{n+1} = \tfrac{4}{3}c^n - \tfrac{1}{3}c^{n-1} + \tfrac{2}{3}\tau g(t_{n+1}, 2c^n - c^{n-1}). \tag{29}$$

This is easily done with the root locus curve computation [6]. If $\tau A_\lambda \in S$ for all θ_λ, then we have von Neumann stability for the explicit advection scheme. In Fig. 2, we have plotted the boundary of S and the curve τA_λ, as a function of θ_λ, for three trial-and-error values of the Courant, Friedrichs and Lewy (CFL) number

$$\nu_\lambda = \frac{\tau|u|}{\Delta\lambda}, \tag{30}$$

namely, for $\nu_\lambda = 0.3, 0.45, 0.6$. The figure shows that the explicit two-step scheme is stable and that the maximal CFL number is 0.45, approximately. In higher space dimension the derivation goes entirely similarly. For example, for

$$\rho_t + u\rho_\lambda + v\rho_\phi = 0, \tag{31}$$

we must have $\tau(A_\lambda + A_\phi) \in S$ for all frequencies. Here A_ϕ is the "advection eigenvalue" defined similarly to A_λ. A safe upperbound for the stepsize τ of the explicit advection scheme then is obtained from

$$\tau\left(\frac{|u|}{\Delta\lambda} + \frac{|v|}{\Delta\phi}\right) \leqslant 0.45. \tag{32}$$

For central differences, the explicit advection scheme will be unstable for all $\tau > 0$, because the eigenvalues then lie on the imaginary axis which is not intersected by S. In this case, one should start from a different implicit method for developing an implicit–explicit variant [2,12].

The CFL condition is comparable to those found in [7] for a number of explicit Runge–Kutta methods. Also the results reported in [3], where the global spherical advection problem is discussed, show that (29) combined with the κ-scheme works well. For (19), the CFL condition seems certainly acceptable, since, in practice, stepsizes taken by this method will be determined mainly by the stiff chemistry and vertical diffusion and hence will generally be smaller than the largest permissible advection stepsize. Of course, the explicit advection approach should not reduce the excellent stability of the implicit BDF method for the stiff chemistry and vertical diffusion computation. The question is thus, will (19) be stable for any stepsize τ_a for which the explicit advection scheme is stable? This implies that the root condition for (26) must be satisfied for all possible values of $\tau_a A_\lambda$ and $1 - \frac{2}{3}\zeta$ where $\zeta = \tau_a(A_\sigma + \mu)$. For $\zeta < 0$ this holds and is a consequence of the theorem below. We have $\zeta < 0$ if the chemistry eigenvalue μ is negative. The theorem does not hold for arbitrary complex ζ with $\mathrm{Re}(\zeta) \leqslant 0$. However this seems to be a redundant observation for atmospheric chemical kinetics problems since these seem to give rise to negative μ.

Theorem 1. *Let the complex number $z \in S$. The roots of*

$$(1 - \tfrac{2}{3}\zeta)\,\alpha^2 - (\tfrac{4}{3} + \tfrac{4}{3}z)\,\alpha + (\tfrac{1}{3} + \tfrac{2}{3}z) = 0,$$

then lie in the unit disk for any $\zeta < 0$.

Proof. Let α_1 and α_2 be the roots for $\zeta = 0$. As $z \in S$, we have $|\alpha_j| \leqslant 1$, $j = 1, 2$. Now write the characteristic equation as

$$\left(1 - \frac{\alpha_1}{\alpha}\right)\left(1 - \frac{\alpha_2}{\alpha}\right) = \frac{2}{3}\zeta \tag{33}$$

and consider the stability domain for ζ, for any fixed $z \in S$. On the boundary of this domain we have a root $|\alpha| = 1$, which implies that $|\alpha_j/\alpha| \leqslant 1$. It then follows from (33) that this boundary cannot intersect the negative axis and hence the entire negative ζ-axis must belong to the stability domain since we have stability for $\zeta \to -\infty$. □

3.3. The exact CFL condition

For multistep implicit–explicit methods, necessary and sufficient conditions for von Neumann stability can be difficult to determine. A semi-analytical approach which gives sufficient conditions for a variety of methods has been proposed in [16] for schemes based on second or fourth-order central differences and in [17] for schemes using the third-order upwind discretization for advection. Our approach above is standard and also semi-analytical. The approximation 0.45 for the maximal CFL number was found by plotting the boundary of the stability region S and three curves of τA_λ. By means of Theorem 1, we then established that the critical stepsize for von Neumann stability in the implicit–explicit case is determined by the critical stepsize for the explicit case and hence by the maximal CFL number.

Although the approximation 0.45 is quite accurate, we will now present the derivation of the true maximal CFL number, denoted below by $\nu_{\max}$. This derivation involves complicated algebraic expressions so that the final step of the proof is solved with help of the computer algebra package MAPLE.

Theorem 2. *To ten decimal digits accuracy, the maximal CFL number is given by*

$$\nu_{\max} = 0.4617485908. \tag{34}$$

Proof. Set $\nu = \nu_\lambda$, $\theta = \theta_\lambda$ and $A = A_\lambda$. By definition

$$\nu_{\max} = \max[\nu: |\alpha_1|, |\alpha_2| \leqslant 1 \text{ for } |\theta| \leqslant \pi], \tag{35}$$

where α_1, α_2 are the zeros of the characteristic polynomial

$$p(\alpha) = a_2\alpha^2 + a_1\alpha + a_0 \tag{36}$$

with coefficients

$$a_2 = 1, \qquad a_1 = -\left(\tfrac{4}{3} + \tfrac{4}{3}\tau A\right), \qquad a_0 = \tfrac{1}{3} + \tfrac{2}{3}\tau A. \tag{37}$$

Let

$$p^*(\alpha) = \bar{a}_2 + \bar{a}_1\alpha + \bar{a}_0\alpha^2, \quad p_1(\alpha) = \bar{a}_2 a_1 - \bar{a}_1 a_0 + (\bar{a}_2 a_2 - \bar{a}_0 a_0)\alpha. \tag{38}$$

According to [9, Theorem 6.1], the zeroes α_1, α_2 lie in the unit disk if and only if

$$\text{(i) } |p^*(0)| > |p(0)|, \quad \text{and} \quad \text{(ii) } |\alpha_0| \leqslant 1, \tag{39}$$

where α_0 is the zero of p_1.

Condition (i) is equivalent to

$$|a_0|^2 = \tfrac{1}{9} - \tfrac{12}{81}(q-1)^2\nu + \tfrac{4}{81}[(q-1)^4 + (1-q^2)(4-q)^2]\nu^2 < 1, \tag{40}$$

where $q = \cos(\theta)$. Because $|a_0|^2$ is an increasing parabola in ν, with value $1/9$ at $\nu = 0$, it follows that (40) holds for $0 \leqslant \nu \leqslant \nu_0$ if this inequality holds for ν_0. Let us substitute the trial value $\nu_0 = 1$. Then (40) holds if and only if

$$2q^3 - 6q^2 - 3q - 2 < 0. \tag{41}$$

It is readily shown that this is the case for all $|q| \leqslant 1$.

Condition (ii) is equivalent to

$$|a_1 - \bar{a}_1 a_0| \leqslant |1 - \bar{a}_0 a_0| = 1 - |a_0|^2. \tag{42}$$

Write $\tau A = \mathrm{Re} + \mathrm{i}\,\mathrm{Im}$. Then (42) is equivalent to

$$3\,\mathrm{Re}^4 + 3\,\mathrm{Im}^4 - 2\,\mathrm{Re}^3 + 6\,\mathrm{Re}^2\,\mathrm{Im}^2 - 2\,\mathrm{Im}^2\,\mathrm{Re} - 5\,\mathrm{Re}^2 + 4\,\mathrm{Re} \leqslant 0. \tag{43}$$

From the expressions

$$\mathrm{Re} = -\tfrac{1}{3}\nu(q-1)^2, \qquad \mathrm{Im}^2 = \tfrac{1}{9}\nu^2(1-q^2)(4-q)^2, \tag{44}$$

it follows that (43) holds for $q = 1$, so that in the remainder we may assume $-1 \leqslant q < 1$. Substitution of the expressions for Re and Im into (43) yields, after a lengthy computation, the equivalent condition

$$Q(\nu, q) = Q_0(q) + Q_1(q)\frac{\nu}{3} + Q_2(q)\left(\frac{\nu}{3}\right)^2 + Q_3(q)\left(\frac{\nu}{3}\right)^3 \leqslant 0, \tag{45}$$

where

$$\begin{aligned} Q_0(q) &= -4, \\ Q_1(q) &= -5(q-1)^2, \\ Q_2(q) &= (q-1)(8q^2 - 10q - 34), \\ Q_3(q) &= 48q^4 - 120q^3 - 333q^2 + 510q + 867. \end{aligned} \tag{46}$$

Because $Q_3(q) > 0$, $Q(\nu, q) \to \pm\infty$ for $\nu \to \pm\infty$. Further, $Q_0(q) = -4$ and $Q_1(q) < 0$, so that $Q(\nu, q)$ has a minimum for $\nu > 0$ and a maximum for $\nu < 0$. Hence, the cubic polynomial $Q(\nu, q)$ has one positive zero and the value of this zero, minimized with respect to q over the interval $-1 \leqslant q < 1$, is just the maximal CFL number (35), provided this zero is less than or equal to one (cf. condition (i) of (39)).

For computing the positive zero of $Q(\nu, q)$ and for the minimization with respect to q, the computer algebra package MAPLE was used. The resulting closed expression for $\nu_{\max}$ is very long and complicated; for the sake of brevity, we give the number $\nu_{\max}$ to 10 decimal digits accuracy. □

4. The performance for 3D applications

In this section, we present results of the implicit–explicit MOL scheme for a 3D test problem. We also discuss the speedup obtained by vectorization and parallelization (on a Cray C98/4256).

4.1. The 3D test problem

The 3D test problem (1)–(3) is based on an extension of the 1D column model to a 3D model on the sphere. The cell-centered horizontal grid covers an area of 7.5° square, arbitrarily chosen near the equator. This corresponds to an 850 km square, approximately. We take a uniform longitude–latitude grid in the horizontal directions,

$$\begin{aligned} \Omega_H = \{(\lambda_i, \phi_j) &: \lambda_0 = -97.5°,\ \phi_0 = -7.5°, \\ &\lambda_i = (i - \tfrac{1}{2})\Delta,\ \phi_j = (j - \tfrac{1}{2})\Delta;\ i, j = 1, \ldots, N\} \end{aligned} \tag{47}$$

with cell width $\Delta = 7.5°/N$. In the vertical direction, the domain definition and the discretization are the same as in the column model described in Section 2.5. Note that the emission input, deposition, etc. are also defined in the same way as in this column model. Thus we take the "rural case" emission input on the whole domain, except for a square inside (see Fig. 3) where we switch to the "urban case" emission input (cf. [14]) which is approximately a factor 10 larger. This serves to create significantly larger (also by an order of magnitude of 10) concentrations at the urban area,

Fig. 3. The horizo
domain "rural" emi
characteristic.

which then dow
horizontal transp
beginning of Se
imposed with su

$u(\lambda, \phi) =$

$v(\lambda, \phi) =$

where $\beta = -3\pi$
3.6 m/sec.

We take N
solution, we tal
tests, different
models of regic
horizontal dom

Let us elabc
consider the co
in (2) so that
yields

$\tau \leqslant 0.225$

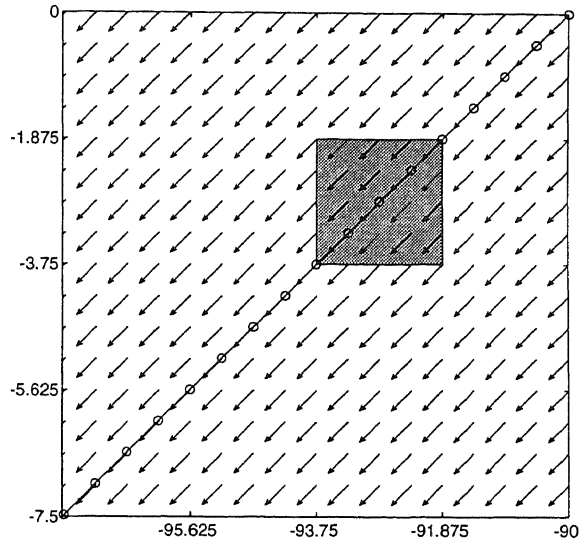

Fig. 3. The horizontal domain with the windfield. In the gray square we impose "urban" emissions, in the rest of the domain "rural" emissions. The diagonal line gives the points where the solution is compared to the 1D solution along the characteristic.

which then downwind gradually must be reduced to the lower rural levels. The discretization of the horizontal transport term (2) is based on the flux-limited finite difference scheme referenced at the beginning of Section 3. We use zero Neumann boundary conditions. A divergence-free windfield is imposed with such a direction and strength that the area is crossed diagonally in four days

$$u(\lambda,\phi) = \frac{\pi\alpha}{24}\frac{\sqrt{2}}{345600}\cdot(\cos\beta\cos\phi + \sin\beta\sin\phi\cos\lambda), \tag{48}$$

$$v(\lambda,\phi) = -\frac{\pi\alpha}{24}\frac{\sqrt{2}}{345600}\cdot\sin\beta\sin\lambda, \tag{49}$$

where $\beta = -3\pi/4$ is the angle with the equator. This corresponds to a wind speed of approximately 3.6 m/sec.

We take $N = 32, 64, 128$ and we start our computations at $t_0 = 14400 + 7200 \cdot 8$. As initial solution, we take at all grid points the vertical 1D solution of the "rural case" at that time. In our tests, different resolutions were chosen to show the performance of the numerical algorithms in 3D models of regional to urban scale. Notice that the chosen values for N correspond to grid sizes in the horizontal domain of approximately 26.6, 13.3 and 6.6 km.

Let us elaborate on the CFL restriction (32) for this test problem. For this purpose, we first consider the corner point $(-90°, 0°)$, where $u = v = -\pi\alpha/(24 \cdot 345600)$. Because $\phi = 0°$, $\cos\phi = 1$ in (2) so that (32) becomes $\tau|u|/(\alpha\Delta) \leqslant 0.225$. Inserting $\Delta = 7.5°/N = 2\pi/(48N)$ radians, this yields

$$\tau \leqslant 0.225 \cdot \frac{24 \cdot 345600}{\pi\alpha}\frac{\alpha\pi}{24N} \approx \begin{cases} 2430.0 \text{ sec.}, & \text{for } N = 32, \\ 1215.0 \text{ sec.}, & \text{for } N = 64, \\ 607.5 \text{ sec.}, & \text{for } N = 128. \end{cases} \tag{50}$$

Because the angles ϕ, λ do not vary much over the spatial domain, these inequalities are approximately true everywhere. The linear stability analysis of Section 3.2 predicts that if (50) is satisfied, then the implicit–explicit BDF scheme will be stable. In the actual application, variable stepsizes are used. Consequently, if violation of (50) would result in instability, then the local error control must detect its onset and reduce the stepsize to a level which ensures stability.

4.2. Vectorization and parallelization

For $N = 128$, the dimension of the complete semi-discrete ODE system is $129 \cdot 129 \cdot 40 \cdot 66 \approx 44 \cdot 10^6$. Obviously, speed is then of utmost importance and because we use a computer with four vector processors (Cray C98/4256), a natural question is how to obtain good parallel vector speed. The integration scheme (19) has been implemented in a modular way. Separate routines perform the flux computations and the explicit advection part. A straightforward implementation of these is automatically optimized by the compiler both with respect to parallelization and to vectorization, resulting for these parts in a good performance. The subroutines are analogous to the ones used in [3]. The core of the implicit solver for the chemical transformations and the vertical turbulent diffusion is the Gauss–Seidel method (15). This method might be applied for all horizontal grid points but the computations are independent. Therefore we implemented loops over all horizontal grid points inside the items 2(a)–2(d) of (15). With this implementation the Gauss–Seidel process vectorizes very well. To obtain a good performance on a shared memory system with only a few processors, item 2(a) can be parallelized over the vertical grid, while in items 2(b) and 2(c) the loop over the horizontal grid may be distributed over the processors.

4.3. Test results

In Fig. 4, the horizontal distribution of the O_3 concentration in the first vertical layer is plotted. It shows that the transitions between the rural and the urban area are very steep, while downwind a strong dependence of the ozone level on the higher urban emission exists. Plots on the coarse grid ($N = 32$) show a comparable behavior. To assess accuracy, we compare the numerical solution with the true 1D solution obtained along the characteristic from $(-90°, 0°)$ to $(-97.5°, -7.5°)$. The error is measured by

$$\mathrm{ERR}_j^n = \left[\sum_{k=1}^{40} (\mathrm{sol}_{kj}^n - \mathrm{app}_{kj}^n)^2\right]^{1/2} \Big/ \left[\max\left(1, \sum_{k=1}^{40} (\mathrm{sol}_{kj}^n)^2\right)\right]^{1/2}, \tag{51}$$

$$\mathrm{SDA}^n = -\log_{10}\left(\frac{1}{66}\sum_{j=1}^{66} \mathrm{ERR}_j^n\right), \tag{52}$$

$$\mathrm{SDA} = -\log_{10}\left(\frac{1}{N \cdot 66}\sum_{n=1}^{N}\sum_{j=1}^{66} \mathrm{ERR}_j^n\right), \tag{53}$$

computed at all grid points through which the characteristic travels. Here, sol_{kj}^n denotes a highly accurate approximation to the semi-discrete 1D solution along the characteristic at time

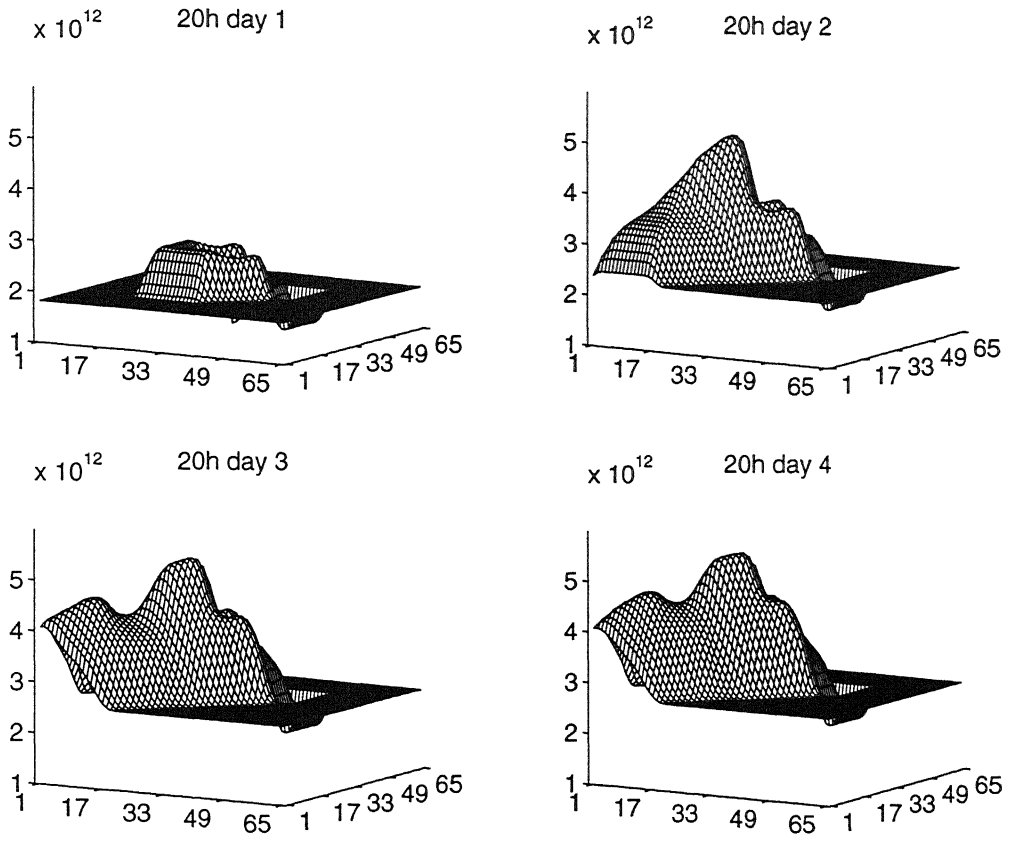

Fig. 4. The ozone concentration at sunset of each day on the horizontal domain ($N = 64$) in the first vertical layer.

$$T_n = t_0 + n\,(4 \cdot 24 \cdot 3600)/N \quad \text{for } n = 1, \ldots, N,$$

which represents the true solution of the 3D problem at the vertical column at the grid point $(N - n, N - n)$ at that time. Likewise, app_{kj}^n denotes the numerical vertical column solution at this grid point at that time. (See Fig. 3, where we indicated 17 of those points with a ○.) Because we use variable stepsizes in time, linear interpolation in time is used to get an approximation at T_n.

We used four GS iterations and made runs for two different time tolerances, viz., RTol = 0.1 and RTol = 0.01. As in the 1D case, after every two hours, the integration is restarted with the backward Euler formula and a tenfold smaller stepsize. At the $65 \times 65 \times 40$ grid, for RTol = 0.1 the maximum and average stepsize taken are, respectively, 2400 sec. and 307 sec. For RTol = 0.01 these values are 1118 sec. and 144. This amounts to a total of 1125 integration steps for RTol = 0.1 and 2407 for RTol = 0.01.

Table 1 shows that on the coarse grid, $N = 32$, and to a lesser extent also for $N = 64$, RTol = 0.01 does not result in a more accurate solution. This obviously indicates that the spatial errors dominate. To illustrate this we have plotted in Fig. 5 the SDA^n values in all grid points through which the characteristic travels. Note that in the first quarter no spatial errors are present since in the right upper

Table 1
Performance on 1 CPU

	129 × 129 × 40			65 × 65 × 40			33 × 33 × 40		
RTol	SDA	CPUs	Mflop	SDA	CPUs	Mflop	SDA	CPUs	Mflop
0.01	1.71	4.45e4	570	1.54	1.20e4	550	1.24	3.31e3	500
0.1	1.39	2.43e4	570	1.44	6.06e3	550	1.24	1.64e3	500

Table 2
Performance on 1 CPU for GS process and flux computations

		129 × 129 × 40		65 × 65 × 40		33 × 33 × 40	
		CPUs	Mflop	CPUs	Mflop	CPUs	Mflop
RTol=0.01	GS	2.49e4	610	6.50e3	600	1.67e3	590
	Flux_λ	7.00e3	510	1.91e3	500	6.53e2	360
	Flux_ϕ	6.58e3	540	1.78e3	530	4.80e2	490
RTol=0.1	GS	1.36e4	610	3.29e3	590	8.27e2	590
	Flux_λ	3.83e3	510	9.66e2	500	3.24e2	360
	Flux_ϕ	3.60e3	540	8.99e2	530	2.38e2	490

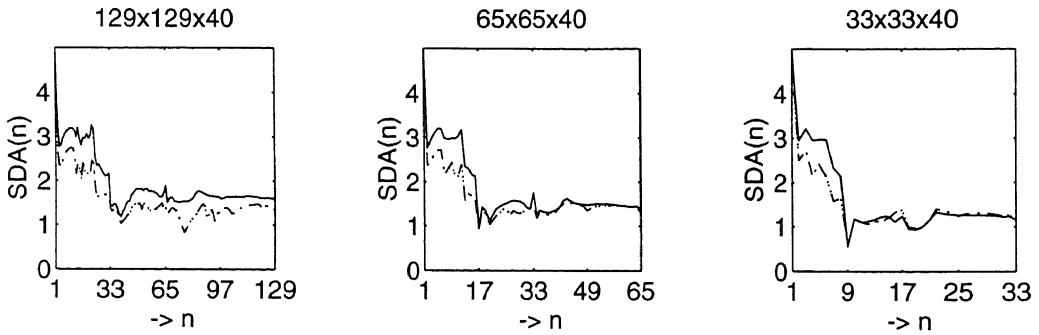

Fig. 5. The errors at the grid points through which the characteristic travels, shown for the three different space grids and the two time tolerances RTol = 0.1 (dashdotted) and RTol = 0.01 (solid).

square of the domain as the solution is constant over the horizontal grid. It is clear that the steep gradients in the rural–urban transition result in dips in the accuracy. On the coarse grids, after the first transition, the error is no longer influenced by the time tolerance and the spatial errors start to dominate. On the finest grid, $N = 128$, the accuracy for RTol = 0.1 is even lower than on the grid with $N = 64$. In Fig. 5, it can be seen that the drop in accuracy occurs after the urban area. The reason is that we did not impose the CFL restriction on the time stepsize assuming that the local error control would detect instabilities in a timely way and reduce the stepsize to a stable level. This obviously did not happen during the nights when two time steps were taken of approximately 900 sec. resulting in a drop of accuracy caused by instabilities in the area just SW of the urban square. A rerun starting at 20 h on day 2 with the CFL restriction imposed confirmed this. Note also that, for the stricter time tolerance RTol = 0.01, this accuracy drop caused by instabilities does not occur.

Table 3
ATExpert data of the parallel fraction (Par. Fr.) and the predictions of the performance on a dedicated machine for the whole program, the GS process and the flux computations. The number of processors is 4, 8 and 16, and the bracketed numbers give the speedup predicted by Amdahl's law

	Par. Fr.	4	8	16
Overall	97%	3.4 (3.7)	5.4 (6.6)	7.7 (11.0)
GS	100%	3.4 (4.0)	5.4 (8.0)	7.8 (16.0)
Flux_λ	96%	3.5 (3.6)	6.3 (6.3)	8.9 (10.0)
Flux_ϕ	96%	3.5 (3.6)	6.1 (6.3)	8.7 (10.0)

The space discretization scheme in the advection part will be first-order at the rapid transition points due to the limiting procedure. Fig. 5 confirms this. Beyond the first transition, the errors plotted in Fig. 5 are close to 2% ($N = 128$), 4% ($N = 64$), and 6% ($N = 33$). These errors may seem rather large but are believed to be quite acceptable for modeling purposes.

The vectorization of the code is very satisfying. To interpret the figures in Table 1 and 2 bear in mind that one CPU of a C90 has a clock period of 4.2 ns and a double vector pipe. This gives a theoretical peak performance on 1 processor of 476 Mflop/s and 952 when chaining an add and a multiply. As was already shown in [3], the explicit part of the solver, which consists mainly of flux computations, has a performance of approximately 0.5 Gflop/s. The implicit part of the solver is dominated by the Gauss–Seidel process which reaches even 0.6 Gflop/s. To measure the Megaflop rate and the CPU time of a routine we used the Cray utility Perftrace [4], that gives the hardware performance by program unit.

Parallelization is done using the Cray Autotasking system, which automatically distributes loop iterations to multiple processors, optionally guided by user directives. For Autotasking, the Cray tool ATExpert [4] can be used to predict speedups for a number of processors on a dedicated system from data collected from a run on a nondedicated system. Table 3 shows the information obtained by ATExpert on the $65 \times 65 \times 40$ grid. The parallel fraction gives an indication of the optimal speedup according to Amdahl's law $S = 1/(f_S + f_P/N)$, where f_S and f_P are the sequential and parallel fraction, respectively, and N the number of processors. For example, a parallel fraction of 97% gives a speedup of 3.7 on a 4-processor machine and 11.0 on 16 processors. The figures in the table indicat that the actual speedup would be much lower for the GS process, especially for 16 processors, wher the optimal speedup of 16 is predicted to reduce to an actual speedup of only 7.8. There are two main reasons for this. The first is load imbalance. For example, the computation of the production and loss terms in the Gauss–Seidel process is parallelized over the vertical grid (40 grid points) and gives a satisfactory speedup for 8 processors, but of course not for 16 processors. The second is the system time needed to invoke and terminate a parallel region (3600 clock periods on a C90). If the parallel sections do not contain enough work, most of the work will be done by the master task and a few slaves. On a 65×65 grid, the actual speedup does not approach the optimal because the amount of work needed to do a part of the decomposition or the backsolve is of the order of only a few operations per loop iteration and because the loop over the horizontal grid needs to be parallelized as well as vectorized, resulting in a relatively small number of points per processor. On the 129×129, horizontal grid this overhead is less important and indeed inspection of a few loops showed that the speedup is significantly higher than for the coarser grid.

5. Final remarks

Air pollution codes usually employ operator splitting. Following the method of lines, in this paper we discussed numerical integration based on the variable stepsize, second-order BDF formula. This choice of integration formula is natural in view of its excellent performance for stiff ODE problems from chemical kinetics in the low accuracy range. However, since we deal with a huge system of ODEs obtained after the spatial discretization of the advection and diffusion terms, it would be very cumbersome to apply the BDF formula in its fully implicit form. We therefore have modified it to an implicit–explicit form which treats advection explicitly and vertical turbulent diffusion and chemistry implicitly. As outlined in Section 3, this implicit–explicit modification is appropriate for our application as regards local accuracy and stability.

The implicit–explicit modification means that the main processes of advection, chemical transformation and vertical turbulent diffusion are treated according to their general physical time constants. Advection is rather slow and can thus be treated explicitly. Certain chemical species and associated vertical turbulent diffusions have small time constants of the same magnitude. The error introduced in treating these two latter processes decoupled, as in operator splitting methods, is therefore difficult to estimate and can be avoided by solving them coupled and by an implicit approach.

However, the remaining 1D solution of the vertical turbulent diffusion and chemistry is a huge task, since this has to be done at every point from the horizontal grid. For this task we have developed the Gauss–Seidel technique (15), which for the 1D example problem from Section 2.4 has been shown to be 4 to 5 times more efficient than the usual modified Newton iteration supplied with a linear banded solver. A second advantage of the Gauss–Seidel technique, compared to modified Newton, is its low memory requirement. No Jacobian matrices need to be stored which makes it possible to exploit grid vectorization in core memory for the very large problem sizes shown here. In view of the CPU times required, it is obvious that good vectorization and parallelization is a practical necessity. The flop rates of 0.5 Gflop/s, about half the peak performance, that we measured for the 3D test problem, illustrate that the vectorization of our implementation on the C90 is very satisfying. The algorithms used are naturally parallel. Without much effort, we have made the parallel fraction of the total program about 97%. On the 65×65 horizontal grid the actual speedup obtained by parallelization on 4 processors is 3.4 which is quite acceptable, but for a larger number of processors, say 16, the amount of work in some of the parallelized loops is too small to reach the optimal speedup of 11.0.

Our 3D test problem is realistic as regards the chemistry scheme. We used the EMEP MSC-W ozone chemistry which is state-of-the-art in the field of regional air pollution modeling. Atmospheric and meteorological conditions, like the vertical turbulent diffusion, the windfield, humidity and temperature, have been prescribed in analytic form and hence we have not simulated a genuine atmospheric pollution problem in all respects. However, numerically the example problem provides a challenging test. The tenfold higher urban emissions in part of the computational modeling domain give rise to sharp transition regions which are difficult for advection schemes. This, combined with the realistic chemistry and the vertical turbulent diffusion modeling, makes it a challenging 3D test problem well suited for benchmarking numerical codes, both with respect to the discretization aspects of accuracy and stability and the high performance computing aspects of vectorization and parallelization.

Acknowledgements

We gratefully acknowledge support from the RIVM (projects EUSMOG and CIRK) and from Cray Research Inc. under grant CRG 94.01 via the Stichting Nationale Computer Faciliteiten (National Computing Facilities Foundation, NCF). Discussions with David Simpson (Norwegian Meteorological Institute) on various aspects of the 1D test model are gratefully acknowledged. We thank Christoph Kessler for his comments on the final draft of the preprint.

References

[1] D.J. Allen, A.R. Douglass, R.B. Rood and P.D. Guthrie, Application of a monotonic upstream-biased transport scheme to three-dimensional constituent transport calculations, *Monthly Weather Rev.* 119 (1981) 2456–2464.

[2] U.M. Ascher, S.J. Ruuth and B. Wetton, Implicit–explicit methods for time-dependent PDEs, *SIAM J. Numer. Anal.* 32 (1995) 797–823.

[3] J.G. Blom, W. Hundsdorfer and J.G. Verwer, Vectorization aspects of a spherical advection scheme on a reduced grid, Report NM-R9418, CWI, Amsterdam (1994).

[4] Cray Research, Inc., UNICOS performance utilities reference manual, SR-2040 6.0 edition (1991).

[5] J.J. Dongarra, C.B. Moler, J.R. Bunch and G.W. Stewart, *LINPACK Users' Guide* (SIAM, Philadelphia, PA, 1979).

[6] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations* II (Springer-Verlag, New York, 1991).

[7] W. Hundsdorfer, B. Koren, M. van Loon and J.G. Verwer, A positive finite-difference advection scheme, *J. Comput. Phys.* 117 (1995) 35–46.

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

[9] J.J.H. Miller, On the location of zeros of certain classes of polynomials with applications to numerical analysis, *J. Inst. Math. Appl.* 8 (1971) 397–406,

[10] D. Simpson, Photochemical model calculations over Europe for two extended summer periods: 1985 and 1989: model results and comparisons with observations, *Atmospheric Environment* 27A (1993) 921–943.

[11] D. Simpson, Y. Andersson-Skold and M.E. Jenkin, Updating the chemical scheme for the EMEP MSC-W model: current status, Report EMEP MSC-W Note 2/93, The Norwegian Meteorological Institute, Oslo (1993).

[12] J.M. Varah, Stability restrictions on second order, three-level finite-difference schemes for parabolic equations, *SIAM J. Numer. Anal.* 17 (1980) 300–309.

[13] J.G. Verwer, Gauss–Seidel iteration for stiff ODEs from chemical kinetics, *SIAM J. Sci. Comput.* 15 (1994) 1243–1250.

[14] J.G. Verwer and D. Simpson, Explicit methods for stiff ODEs from atmospheric chemistry, *Appl. Numer. Math.* 18 (1995) 413–430.

[15] J.G. Verwer, J.G. Blom, M. van Loon and E.J. Spee, A comparison of stiff ODE solvers for atmospheric chemistry problems, *Atmospheric Environment* 30 (1996) 49–58.

[16] P. Wesseling, A method to obtain von Neumann stability conditions for the convection–diffusion equation, in: M.J. Baines and K.W. Morton, eds., *Proceedings ICFD Conference on Numerical Methods in Fluid Dynamics*, Oxford (Oxford University Press, Oxford, 1995).

[17] P. Wesseling. Von Neumann stability conditions for the convection–diffusion equation, Report 95-25, Faculty of Technical Mathematics and Informatics, Delft University of Technology (1995).

[18] D.L. Williamson, Review of numerical approaches for modeling global transport, in: H. van Dop and G. Kallos, eds., *Air Pollution Modeling and Its Application* IX (Plenum Press, New York, 1992) 377–394.