Computational Challenges in Large-scale Air Pollution Modelling

Tzvetan Ostromsky
Central Laboratory for Parallel Processing
Bulgarian Academy of Sciences
Acad. G. Bonchev Str. 25 A, Sofia, Bulgaria
ceco@copern.bas.bg

Wojciech Owczarz
Danish Computing Centre for Research and Education
Technical University of Denmark - Build. 404
DK-2800 Lyngby, Denmark
Wojciech.Owczarz@uni-c.dk

Zahari Zlatev
National Environmental Research Institute
Frederiksborgvej 399
P. O. Box 358
DK-4000 Roskilde, Denmark
zz@dmu.dk

ABSTRACT

Many difficulties must be overcome when large-scale air pollution models are treated numerically, because the physical and chemical processes in the atmosphere are very fast. This is why it is necessary

1. to use a large space domain in order to be able to study long-range transport of pollutants in the atmosphere,
2. to describe adequately all important processes when an air pollution model is developed and
3. to use fine grids in the discretization of the model.

If all these conditions are taken into account, then the resulting computational tasks are huge and the computer treatment of the model is very difficult even when powerful modern computers are in use. Therefore, the following two major tasks must be solved:

- fast and sufficiently accurate numerical methods are to be selected and
- the selected numerical methods should be efficiently implemented on the available high-speed computers.

The solution of these two important tasks in connection with a particular large-scale air pollution model, the Danish Eulerian Model, will be discussed in this paper. The use of standard parallelization tools allows us to achieve good performance when the model is run on different parallel computers. This will be illustrated in the paper by many numerical examples.

Categories and Subject Descriptors

G.1.0 [**Mathematics of Computing**]: General—*Parallel algorithms*; G.1.7 [**Mathematics of Computing**]: Ordinary Differential Equations—*Initial value problems*; G.1.8 [**Mathematics of Computing**]: Partial Differential Equations—*Method of lines*; J.2 [**Computer Applications**]: Physical Sciences and Engineering—*Earth and atmospheric sciences*

General Terms

Algorithms, Performance

Keywords

Air pollution models, partial differential equations, finite elements, quasi-steady-state-approximation, ordinary differential equations, parallel computations

1. MATHEMATICAL DESCRIPTION OF AN AIR POLLUTION MODEL

Mathematical models are indispensable tools when different air pollution phenomena are to be studied on a large space domain (say, the whole of Europe). All important physical and chemical processes must be taken into account in order to obtain reliable results. This leads to huge computational tasks, which have to be handled on big high-speed computers. A short description of the particular air pollution model used, the Danish Eulerian Model ([54]), will be given in this section. We shall concentrate our attention on the mathematical formulation of this model, but a detailed description of the physical and chemical processes involved in the models as well as many of the validation tests and different air pollution studies that were carried out with the Danish Eulerian Model have been documented in numerous publications (see, for example, [4], [23], [26], [52], [54], [56], [57], [58], [59], [60], [62] or [61]; see also the web-site of the Danish Eulerian Model, [50]).

The Danish Eulerian Model is described by systems of partial differential equations (PDE's):

$$\begin{aligned}\frac{\partial c_s}{\partial t} &= -\frac{\partial(uc_s)}{\partial x} - \frac{\partial(vc_s)}{\partial y} - \frac{\partial(wc_s)}{\partial z} \qquad (1)\\ &+\frac{\partial}{\partial x}\left(K_x\frac{\partial c_s}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y\frac{\partial c_s}{\partial y}\right) + \frac{\partial}{\partial z}\left(K_z\frac{\partial c_s}{\partial z}\right)\\ &+E_s - (\kappa_{1s} + \kappa_{2s})c_s + Q_s(c_1, c_2, \ldots, c_q),\\ &s = 1, 2, \ldots, q.\end{aligned}$$

where

- the concentrations of the chemical species are denoted by $c_s = c_s(x, y, z, t)$,
- $u = u(x,y,z,t)$, $v = v(x,y,z,t)$ and $w = w(x,y,z,t)$ are wind velocities along the coordinate axes,
- K_x, K_y and K_z are diffusion coefficients (K_x and K_y are non-negative constants, while K_z depends both on the spatial and temporal variables; the algorithms for calculating this quantity depend on the meteorological conditions and are normally very complicated),
- the emission sources located in the space domain of the model are described by terms $E_s = E_s(x, y, z, t)$,
- $\kappa_{1s} = \kappa_{1s}(x,y,z,t)$ and $\kappa_{2s} = \kappa_{2s}(x,y,z,t)$ are deposition coefficients related to the dry and wet deposition processes respectively (the wet deposition takes place only when it rains) and
- the non-linear function $Q_s(c_1, c_2, \ldots, c_q)$ is a common term for all chemical reactions in which the species s is involved.

The CBM IV chemical scheme with $q = 35$ chemical species, which has been proposed in [20], is actually used in the version of the Danish Eulerian Model [54] that will be considered in this paper, but experiments with two other chemical schemes, containing $q = 56$ and $q = 168$ chemical species, are carried out at present. It should be mentioned here that the CBM IV scheme is also used in some other well-known air pollution models (which is reflected in some of the papers in [55]).

The Danish Eulerian Model can be used (and has successfully been used in many studies) to calculate concentrations and/or depositions of:

- sulphur compounds,
- nitrogen compounds,
- ammonia-ammonium,
- ozone and
- many hydrocarbons.

2. SPLITTING PROCEDURE

It is difficult to treat the system of PDE's (1) directly. This is the reason for using different kinds of splitting in the field of air pollution modelling. A splitting procedure, based on ideas proposed in [31], [32] and [12], leads to five sub-models, representing the five major physical and chemical processes involved in the long-range transport of air pollutants:

- the horizontal advection,
- the horizontal diffusion,
- the chemistry (together with the emission terms),
- the dry and wet deposition processes and
- the vertical exchange.

These processes are expressed mathematically by the following five equations:

$$\frac{\partial c_s^{(1)}}{\partial t} = -\frac{\partial(uc_s^{(1)})}{\partial x} - \frac{\partial(vc_s^{(1)})}{\partial y} \qquad (2)$$

$$\frac{\partial c_s^{(2)}}{\partial t} = \frac{\partial}{\partial x}\left(K_x\frac{\partial c_s^{(2)}}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y\frac{\partial c_s^{(2)}}{\partial y}\right) \qquad (3)$$

$$\frac{dc_s^{(3)}}{dt} = E_s + Q_s(c_1^{(3)}, c_2^{(3)}, \ldots, c_q^{(3)}) \qquad (4)$$

$$\frac{dc_s^{(4)}}{dt} = -(\kappa_{1s} + \kappa_{2s})c_s^{(4)} \qquad (5)$$

$$\frac{\partial c_s^{(5)}}{\partial t} = -\frac{\partial(wc_s^{(5)})}{\partial z} + \frac{\partial}{\partial z}\left(K_z\frac{\partial c_s^{(5)}}{\partial z}\right) \qquad (6)$$

$$s = 1, 2, \ldots, q$$

If the model (1) is split into the five sub-models (2) - (6), then the discretization of the spatial derivatives in the right-hand-sides of the sub-models will lead to the solution (successively at each time-step) of five systems ($i = 1, 2, 3, 4, 5$) of ordinary differential equations (ODE's):

$$\frac{dg^{(i)}}{dt} = f^{(i)}(t, g^{(i)}), \quad g^{(i)} \in R^N, \quad f^{(i)} \in R^N, \qquad (7)$$

where $N = N_x \times N_y \times N_z \times N_s$ (N_x, N_y and N_z are the numbers of grid-points along the coordinate axes, while $N_s = q$ is the number of chemical species). The functions $f^{(i)}$, $i = 1, 2, 3, 4, 5$, depend on the particular discretization methods used in the numerical treatment of the different sub-models, while the components of the vector-functions $g^{(i)}$, $i = 1, 2, 3, 4, 5$, are approximations of the concentrations at the grid-points of the space domain.

Much more details about the splitting procedure described by (2) - (7) can be found in [54].

3. NEED FOR EFFICIENT NUMERICAL METHODS AND HIGH-SPEED COMPUTERS

The size of any of the five ODE systems (7) is equal to the product of the number of the grid-points and the number of chemical species. It is clear, therefore, that it grows very quickly when the grid-points and/or the chemical species are increased. This is illustrated in Table 1 for some of the versions of the Danish Eulerian Model.

Table 1: Numbers of equations in different versions of the Danish Eulerian Model with 35 chemical species (the discretization of a horizontal plane of the space domain is shown in the first column, the approximate size of the grid-squares when different grids are used is shown in the second column, the numbers of equations in the 2-D and 3-D versions are given in the third and fourth columns respectively; there are not yet 3-D versions for the fine grids)

Grid	Grid-squares	2-D	3-D
(32×32)	$(150\ km \times 150\ km)$	35840	358400
(96×96)	$(50\ km \times 50\ km)$	322560	3225600
(288×288)	$(16.7\ km \times 16.7\ km)$	2903040	-
(480×480)	$(10\ km \times 10\ km)$	8064000	-

Sometimes it is necessary to perform long simulation processes consisting of several hundreds of runs (see, for example, [4] or [61]). At present these problems are solved by the operational two-dimensional version of the Danish Eulerian Model (see [4], [54] and [61]). In this version the following values of the parameters are used: $N_x = 96$, $N_y = 96$, $N_z = 1$, $N_s = 35$. This leads to the treatment of four ODE systems per time-step; each of them contains 322560 equations (see Table 1). It is more desirable to use

- finer grids,
- 3-D versions of the model and
- more advanced chemical schemes (some tests with chemical schemes containing more species, $N_s = 56$ and with $N_s = 168$, have been extensively tested, but these schemes will not be used in this paper).

The amount of computational work is increased considerably when any of these three requirements is satisfied.

In 1984 A. Jaffe stated in his paper "Ordering the universe: The role of mathematics" ([27]) that,

> *although the computers are becoming faster and faster, they will always be too slow, because the requirements of the scientists are increasing much faster.*

The challenging computational problems, which arise in air pollution modelling show that this statement remains true also in the new millennium.

4. THE NUMERICAL ALGORITHMS SELECTED FOR THE MODEL

The short description of the great numerical difficulties, which arise when large air pollution models are to be treated in different important for the society studies, explains why the search for more efficient numerical methods is continuing. The need of such methods is emphasized in many investigations; see, for example, [39] and [54] and some of the papers in [55]).

It is not necessary to describe in detail all the numerical methods currently used in the Danish Eulerian Model. The major numerical methods used to obtain the results presented in this paper are:

- linear finite element methods in the advection-diffusion part as well as in the vertical exchange part and
- the Quasi-Steady-State-Approximation (QSSA) algorithm in the chemical part.

The implementation of these two numerical methods in the Danish Eulerian Model will be discussed in the next two sub-sections.

4.1 Horizontal advection and diffusion

The finite elements, which are used at present in the Danish Eulerian Model, have been applied in connection with air pollution models in [37] and [38]. More details about this method can be found in [54] and [19]. The systems of ODEs (7), which are obtained from (2) after the application of any finite element semi-discretization in a horizontal grid-plane, are of the following type:

$$P\frac{dg}{dt} = Hg, \quad g \in R^N, \quad P \in R^{N \times N}, \quad H \in R^{N \times N}, \qquad (8)$$

where N_x and N_y are the numbers of grid-points along the horizontal coordinate axes, while $N = N_x \times N_y$ is the total number of grid-points in a horizontal grid-plane. The components of the vector-function g , are as in (7), approximations of the concentrations at the grid-points of the space domain. Matrix P is a constant matrix ($P = I$, where I is the identity matrix, when finite differences are used). Matrix H depends on the wind velocity vectors (consisting of values of u and v at the grid-points). This means that in general matrix H depends both on the spatial variables and on the time variable. The matrices P and H are both banded matrices. Both the structure of these matrices and their elements depend on the particular finite element method which has been selected. If splitting to one-dimensional models is used (as proposed in [32]), then both P and H are tridiagonal matrices.

The next problem is to decide how to handle the ODE system (8). There are two possibilities:

- to select some explicit integration algorithm and

- to apply an implicit integration algorithm.

The use of an explicit integration algorithm leads to the solution (at every time-step; the number of time-steps is normally several thousand) of systems of linear algebraic equations the coefficient matrix of which is P. The fact that P is a constant matrix can efficiently be exploited. This matrix can be factorized in the very beginning of the computations, and the factorization can be used again and again at every time-step. On the other hand, the use of explicit methods leads to restrictions on the time-stepsize when the problem is stiff (see [22], [30] and [54]).

The use of an implicit integration method leads to the solution (again at every time-step) of systems of linear algebraic equations the coefficient matrix of which is of the type: $A = P - \triangle t \beta H$, where β depends on the particular integration method. This means that matrix A is time-dependent and, thus, has to be factorized at every time-step. On the other hand, if an integration method with good stability properties (L-stable or A-stable method; see [22], [30] and [54]) is chosen, then no restrictions on the time-stepsize are imposed for keeping the stability of the computations (however, the time-stepsize should be sufficiently small in order to achieve the required accuracy).

Since the ODE system (8) is not very stiff, the use of explicit methods is more desirable. However, the selected methods must have reasonably good stability properties. Moreover, it is necessary to introduce a stability control check. Such a check, based on ideas described in [56] has been implemented. Norms of the wind velocity vectors (the components of which are values of the wind velocities u and v at the grid-ponts) are used in this check, which can be introduced as follows. Assume that

- U and V are some norms of the vectors whose components are values of the wind velocities u and v at the grid points,
- $\triangle x = \triangle y$ is the increment used in the discretization of the horizontal grid-planes,
- $\triangle t$ is the time-stepsize, which has to be used at the current time-step,
- λ^* is a parameter which depends on matrix $P^{-1}H$ and
- h_{imag} is the absolute stability interval on the imaginary axis (see [56]) of the time-integration algorithm used in the solution of (8).

Under these assumptions, it should be expected the calculations at the current time-step to be stable if the following inequality is satisfied:

$$\triangle t < \alpha \, \frac{h_{imag}}{\lambda^*(U+V)} \, \triangle x, \qquad (9)$$

where α is a constant (with $0 < \alpha \leq 1$), which is used as a precaution factor in order to make the application of (9) more reliable.

Note that $\triangle x$ and λ^* are constants when both the spatial grid and the finite element method are already chosen. The values of $\triangle x$ that are used in the different versions of the Danish Eulerian Model are 150 km, 50 km, 16.67 km and 10 km (see Table 1). For the finite elements used in the Danish Eulerian Model, we have $\lambda^* \approx 1.73$. This means that if $U+V$, which is time-dependent, becomes larger, then integration methods with a larger h_{imag} must be selected in an attempt to preserve the stability of the computational process. Thus, we have to vary the integration methods in order both to keep the computations stable and to avoid reductions of the time-stepsize.

The above analysis indicates that we can try to avoid reductions of the time-stepsize by using the stability check (9) in the following way. Three predictor-correctors schemes with different stability properties on the imaginary axis (different values of h_{imag}; $h_{imag} = 1.65$, $h_{imag} = 2.56$ and $h_{imag} = 3.26$) were selected. The more stable predictor-corrector schemes (those with larger values of h_{imag}) are also more expensive. Therefore such schemes should be used only when the norms of the wind velocity vectors are large. If the norms of the wind velocity vectors are not very large, then it is more profitable to apply predictor-correctors schemes which have not very good stability properties (parameter h_{imag} being smaller), but are cheaper.

Strictly speaking, checks of type (9) are valid for constant wind velocity only. Nevertheless, the check based on (9) works very well in the efforts both to ensure stable computations and to avoid reductions of the time-stepsize in the computations with the Danish Eulerian Model (which have been run with meteorological data for 20 different years from 1979 to 1998, i.e. the with many different wind velocity fields).

The diffusion sub-model (3) can be described with a system of the same type as (8) when some discretization based either on finite elements or on finite differences has been applied. Therefore, it is efficient to combine the advection sub-model and the diffusion sub-model when the finite elements are used. Since the advection is the dominating process, the combined advection-diffusion sub-model can be handled with the same procedure for stepsize control as that sketched above.

4.2 Chemistry and deposition

The Quasi-Steady-State-Approximation (QSSA) algorithm, which is currently used in the chemical part of the Danish Eulerian Model, has been proposed in connection with air pollution modelling in [25] for box models (only one grid-point and only chemical reactions). This algorithm has been extensively tested for two more realistic examples:

- the generalized Crowley-Molenkampf rotation test proposed and tested in Hov et al. [26] (the original Crowley-Molenkampf test was simultaneously proposed for testing the accuracy of the numerical algorithms used in the solution of advection equations in Crowley [6] and Molenkampf [33]) and
- the two-dimensional air pollution models studied in Zlatev [54].

It is sufficient to consider the chemical reactions at a single grid-point in order to explain how the QSSA algorithm works. The chemical reactions at the selected grid-point can be described by an ODE system of the following type:

$$\frac{dc_s}{dt} = Q_s(c_1, c_2, \ldots, c_q), \qquad (10)$$

where $c_s \in R$, $s = 1, 2, \ldots, q$, are now values of the concentrations of the chemical species at the chosen grid-point. It is convenient to re-write (10) as follows:

$$\frac{dc_s}{dt} = P_s(c_1, c_2, \ldots, c_q), -L_s(c_1, c_2, \ldots, c_q)c_s, \qquad (11)$$

where the non-negative functions P_s and Q_s are called production and loss terms respectively.

Assume that some approximations c_s^n to the exact solutions $c_s(t_n)$ of (11) at the point t_n have already been found. Then approximations c_s^{n+1} to the exact solutions $c_s(t_{n+1})$ of (11) at the point $t_{n+1} = t_n + \Delta t$ are calculated by using one of the following three formulae when the original QSSA is used:

$$c_s^{n+1} = \frac{P_s}{L_s} \qquad for \quad \Delta t L_s > 10, \qquad (12)$$

$$c_s^{n+1} = \frac{P_s}{L_s} + \left(c_s^n - \frac{P_s}{L_s}\right) e^{-\Delta t L_s} \qquad (13)$$

$$for \quad 0.01 < \Delta t L_s \leq 10,$$

$$c_s^{n+1} = c_s^n - \Delta t(P_s - L_s c_s^n) \qquad (14)$$

$$for \quad \Delta t L_s \leq 0.01.$$

It is assumed in (12)-(14) that non-linear functions P_s and Q_s are calculated for $t = t_n$. This means that the numerical integration method, the QSSA, defined by (12)-(14) is explicit. It is better to incorporate (12) - (14) in an iterative process (see more details about this in [1]).

It is not very efficient to use the original version of QSSA, defined by (12) - (14), in a large-scale air pollution model. Two major computational drawbacks of the original QSSA arise when it is applied in a large air pollution model.

- Three questions, see (12) - (14), have to be asked in order to decide which formula should be used at many grid-points for each of the chemical species when the original QSSA is used in such models. This leads to a very long loop with three "if" statements (in the body of the loop), which deteriorate its performance. Let us reiterate here that in the refined version of the 2-D Danish Eulerian Model the number of grid-points is 230400, while the number of chemical species is 35.
- The value of the exponential function has to be calculate each time when the second formula is selected. The calculation of a value of the exponential function is expensive on many computers.

These drawbacks can be removed if we decide to use always the second formula and if, moreover, we replace the exponential function in (13) with the following approximation:

$$e^{-\Delta t L_s} \approx \frac{1}{1 + \Delta t L_s + 0.5(\Delta t L_s)^2}. \qquad (15)$$

The substitution of the right-hand-side of (15) in (13) leads to the following numerical method:

$$c_s^{n+1} = \frac{c_s^n + (1 + 0.5\Delta t L_s))\Delta t P_s}{1 + \Delta t L_s + 0.5(\Delta t L_s)^2}. \qquad (16)$$

The numerical integration method defined by (16) is used at present as an explicit method (i.e. it is again assumed that non-linear functions P_s and Q_s are calculated for $t = t_n$). However, it is normally used as an implicit numerical method, which can be handled with some iterative method (it has been mentioned that also the original version of QSSA is as a rule used as an implicit method).

The improved, by using (15) and (16), QSSA has been extensively tested in [1]. This method has been used to obtain the results which will be presented in the following part of this paper.

Some more advanced methods for the chemical sub-model have also been implemented in the Danish Eulerian Model and tested in Alexandrov et al. [1], Georgiev and Zlatev [19] and Skelboe and Zlatev [44]. The latter methods will, however, not be used in this paper.

The chemistry sub-model is the most time-consuming part of the model. Therefore, the problem of finding efficient methods for this part of a large air pollution model is still open. The research efforts in this area have been very extensive during the last decade (see, for example, [5], [10], [11], [24], [28], [34], [40], [41], [42], [43], [46], [47], [48]). Some methods, which are discussed in publications about general-purpose numerical algorithms for the solution of ODE systems (see, for example, [7], [8], [9], [22] and [30]) can also be appropriate in the efforts to improve the performance of the chemical module.

Special techniques (as, for example, different implementations of sparse matrix algorithms) can also be very useful in the computer treatment of the chemical part of air pollution models (see [1], [14], [19], [44], [45], [51], and [53]).

The deposition part is treated together with the chemical part. This part of the model does not cause problems (see Zlatev [54]).

4.3 Vertical exchange

The discretization along the vertical grid-lines is carried out by using the same finite elements as in the advection-diffusion part. The application of any finite element method (or any method based on the use of finite differences) in the discretization of the spatial derivatives in (6) leads to the same system of ODEs as in the advection-diffusion part; i.e. the resulting system of ODEs is of the same type as (8).

In fact, many such semi-discretized systems, one per each vertical grid-line, appear. Each of these systems is small (the number of equations being equal to the number of layers). However, the number of these systems (equal to the product $N_x \times N_y \times N_s$) can be very large. If $N_x = 480$, $N_y = 480$ and $N_s = 35$, then 8064000 ODE systems have to be treated at every time-step in the vertical exchange part of the model. Therefore, one must be careful in the choice of numerical methods for this sub-model.

It is important to emphasize here that the vertical diffusion is the dominating process when the sub-model (6) is treated. This implies that the resulting ODE systems of type (8) are now stiff. This is why these systems must be solved with implicit time-integration methods. The simple θ-method (see, for example, [30]) seems to work rather efficiently. However, the use of some time-integration methods of higher order may result in a further improvement of the efficiency.

5. PREPARING THE CODE FOR RUNS ON HIGH-SPEED COMPUTERS

It is very important to exploit in the best possible way the great potential power of the modern supercomputers. This is a very difficult task when large-scale air pollution models are to be run, because

- the codes are very big, containing up to several hundreds subroutines,
- a very large amount of input data (meteorological data and emission data) have to be read and/or interpolated at every time-step and
- a very large amount of output data have to be prepared and stored for future use.

The preparation of efficient versions of the Danish Eulerian model for three types of computer architectures:

- parallel computers with shared memory,
- parallel computers with distributed memory and
- more advanced parallel computers utilizing both shared memory and distributed memory

will be sketched in this section.

5.1 Running the model on shared memory computers

OPEN MP ([49]) commands are used when the code is run on parallel computers with shared memory. The OPEN MP commands are becoming standard commands. Therefore, it is easy to get good results on different shared memory computers when such commands are used.

It is important to identify the parallel tasks and to group them in an appropriate way when necessary. For the different parts of the code this is done in the following way:

- **The horizontal advection and diffusion.** It can easily be seen that, after the splitting procedure, the performance of the horizontal advection can be carried out independently for every chemical compound (and for the 3-D version for every layer). This means that the number of parallel tasks is equal to the number of chemical compounds (and to the product of the chemical compounds and the layers when the 3-D version is used). The same is true for the horizontal diffusion. Moreover, the advection and the diffusion parts can be treated, as already mentioned in the previous section, together. Thus, there are many parallel tasks in this part of the code and, moreover, the parallel tasks are very big.
- **The chemistry and deposition.** These two processes can be carried out in parallel for every grid-point. This means that there are many parallel tasks (the number of parallel tasks is equal to the number of grid-points), but each task is a small task. Therefore, the tasks should be grouped in an appropriate way. This can be done by using chunks. Both the procedure of splitting the data into chunks and the effect of using chunks are discussed in detail in Georgiev and Zlatev [19].
- **The vertical exchange.** The performance of the vertical exchange along each vertical grid-line is a parallel task. The number of these tasks is very large, $N_x \times N_y \times N_s$. If the grid is fine, then the number of these tasks is becoming enormous, see the example given in §4.3. However, the parallel tasks are not very big and have to be grouped. This is done by trying to distribute equally the tasks among the assigned processors.

5.2 Running the model on distributed memory computers

Either the Message Passing Interface (MPI, [21]) or the Parallel Virtual Machine (PVM, [17]) can be used on parallel computers with distributed memory. We started by using PVM (see Bendtsen and Zlatev [3]), but only MPI has been used in the last four-five years (see Georgiev and Zlatev, [18] and [19]).

In the MPI implementation, the space domain of the model is divided into several sub-domains (the number of these sub-domains being equal to the number of the processors assigned to the job). Then each processor works on its own sub-domain.

Two procedures, a pre-processing procedure and a post-processing procedure, are performed in the beginning and in the end of the run.

- **The pre-processing procedure.** In the beginning of the job the input data (the meteorological data and the emission data) are distributed (consistently with the sub-domains) to the assigned processors. In this way, not only is each processor working on its own sub-domain, but it has also access to all meteorological and emission data which are needed in the run.
- **The post-processing procedure.** During the run, each processor prepares output data files. At the end of the job all these files have to be collected on one of the processors and prepared for using them in the future. This is done by the post-processing procedure.

The use of the pre-processing and post-processing procedures is done in order to reduce as much as possible the communications during the actual computations. However, some communications are to be carried out during the computations. The time needed for these communications is very small (normally, several percent).

Much more details about the runs of several versions of the Danish Eulerian Model on parallel computers with distributed memory by using MPI tools can be found in Georgiev and Zlatev, [19].

5.3 Running the code on some more complicated architectures

More complicated computer architectures are becoming available during the last decade. An example for such an architecture is the IBM SMP computer. In fact, some ideas, on which this architecture is built, have been used under the work on the CEDAR project; see [29]. The IBM SMP consists of several nodes. Every node contains several processors.

In the architecture available for us, there were two IBM SMP nodes, each of them containing eight processors. Each node could be considered as a shared memory computer, while message passing is needed across the nodes.

Some runs on this computer will be described here. Again the space domain of the model is divided into sub-domains (one per each node). The pre-processing procedure is used to distribute the data among the nodes, while by the post-processing procedure the data is collected to one of the nodes and prepared for future use. OPEN MP commands are to be used on each node in order to obtain parallel computations within the node (across the processors of the node).

Both 2-D versions and 3-D versions of the Danish Eulerian Model were run of this computer. Moreover, 2-D versions discretized on three grids, the (96×96) grid, the $288 \times 288)$ grid and the (480×480) grid, were tested.

It is important to emphasize that we are using only standard parallelization tools in all these versions of the Danish Eulerian Model; both MPI tools and OPEN MP instructions. Therefore, it should be easy to port this code to other computers of this type.

More details about the organization of the parallel computations on computers of this type can be found in Owczarz and Zlatev, [35] and [36].

6. RESULTS OBTAINED BY USING THE COMPUTERS AT THE DANISH COMPUTING CENTRE

Different versions of the Danish Eulerian Model have been run on three different computers available at the Danish Computing Centre. Some results obtained in these runs are given in the following tables:

- **Table 2.** Results obtained with the 3-D version of the Danish Eulerian Model, which is discretized on a $(96 \times 96 \times 10)$ grid, when a shared memory computer is used. The computer actually used was an SGI Origin 2000.
- **Table 3.** Results obtained with a 2-D version of the Danish Eulerian Model, which is discretized on a fine (480×480) grid, when a distributed memory computer is used. The computer actually used was an IBM SP. It should be emphasized here that the job is so big that it was not possible to run it on less than 8 processors. Therefore, the speed up and the efficiency were calculated by comparing the results obtained when 32 processors are used with the corresponding results obtained when 8 processors are used.
- **Table 4.** Results obtained with a 2-D version of the Danish Eulerian Model, which is discretized on a (96×96) grid, when an IBM SMP computer (two nodes, eight processors per node) is used.

The results show that good speed-up can be achieved on different computers when standard parallelization tools are applied. Much more results can be found in [18], [19], [35] and [36].

Table 2: Computing times (measured in seconds) obtained by using the SGI Origin 2000 computer when the 3-D version of the Danish Eulerian Model is discretized on a $(96 \times 96 \times 10)$ grid.

Processors	Comp. time	Speed-up	Efficiency
1	42907	-	-
32	2215	19.37	61%

7. SCALABILITY OF THE CODE

It is important *to preserve* the efficiency of the code when the size of some of the involved arrays is increased (for example, as a result of refining the grid, increasing of the number of chemical compounds, the transition from the 2-D version to a 3-D version, etc.). This property is often referred to as a scalability of the code. While such a property is highly desirable (the requirements to the air pollution codes are

Table 3: Computing times (measured in seconds) obtained by using the IBM SP computer when the 2-D version of the Danish Eulerian Model is discretized on a (480×480) grid.

Processors	Comp. time	Speed-up	Efficiency
8	54978	-	-
32	15998	3.44	86%

Table 4: Computing times (measured in seconds) obtained by using the IBM SMP computer when the 2-D version of the Danish Eulerian Model is discretized on a (96×96) grid.

Processors	Comp. time	Speed-up	Efficiency
1	5225	-	-
16	424	12.32	72%

permanently increasing), it is by no means clear in advance whether the code has such a property or not when the modern complicated computer architectures are in use.

Some experiments were performed in an attempt to check the scalability of the parallel devices discussed in the previous sections. A (288×288) grid was considered instead of the (96×96) grid considered in the previous sections. Since the space domain remains unchanged (a $4800\,km \times 4800\,km$ area containing the whole of Europe) this corresponds to a transition from cells of size $(50\ km \times 50\ km)$ to cells of size $(16.67\ km \times 16.67\ km)$; see also Table 1. This means that in the refined on a (288×288) grid version of the code the number of grid-points was increased by a factor of 9. The number of chemical species was kept 35.

In the advection part, we had also to decrease the time-stepsize by a factor of 3. Thus, the number of arithmetic operations (or, in other words, the amount of computational work) is increased by a factor of 27 in the advection part.

There was no need to decrease the time-stepsize in the chemical part. This means that the number of arithmetic operations (or, in other words, the amount of computational work) is increased by a factor of 9 in the chemical part.

If a 3-D version of the Danish Eulerian Model is used on a $(288 \times 288 \times 10)$ grid when the computational work in the horizontal advection diffusion part and in the chemical part will be increased with the same factors, 27 and 9 respectively, compared with the 3-D version discretized on a $(96 \times 96 \times 10)$ grid; it should be emphasized, however, that the 3-D version discretized on a $(288 \times 288 \times 10)$ grid is not ready yet. Furthermore, there will be no need to decrease the time-stepsize in the vertical exchange part either when a version discretized on a $(288 \times 288 \times 10)$ grid is prepared. This means that in this part of code the number of arithmetic operations (or, in other words, the amount of computational work) will be increased as in the chemical part (i.e. by a factor of 9).

Table 5: Computing times obtained by using a refined (288×288) grid and a coarse (96×96) grid with the 2-D Danish Eulerian Model. The ratios of the times for the refined and coarse grids are given in the last column. The two codes were run on 16 processors of the IBM SMP computer.

Process	(288×288)	(96×96)	Ratio
Advection	1523	63	24.6
Chemistry	2883	288	10.0
Total	6209	424	14.6

The short analysis presented above indicates that if the code of the 2-D Danish Eulerian Model is scalable, then the computing times should be increased by factors approximately equal to 27 and 9 in the advection part and the chemical part respectively in the transition from a (96×96) grid to the refined (288×288) grid. For the code of the 3-D Danish Eulerian Model the increasing factors for the advection and chemistry part are the same, 27 and 9 respectively, as for the code of the 2-D Danish Eulerian Model. Furthermore the computing time for the vertical exchange part should be increased by a factor of 9 in the transition from a (96×96) grid to the refined (288×288) grid if the code is scalable.

One can also study what is happening in the transition from the the 2-D Danish Eulerian Model discretized on a (96×96) grid to the the 3-D Danish Eulerian Model discretized on a $(96 \times 96 \times 10)$ grid. In this case the computing times for the advection part an for the chemical part is increased by a factor of 10 when the 3-D version is used, while the computing time for the vertical exchange part is relevant only for the 3-D version. Some processes (the processes that are relevant for the surface layer are common for the 2-D version and the 3-D version. Therefore, it is not very clear whether the total computing time will be increased by a factor greater than 10 or not in the transition from the 2-D version to the 3-D version. Some results, compare the results in Table 6 with the results in Table 8 indicate that the increasing factor for the total computing time is less than 10. However, more experiments are needed in order to understand better the situation.

Some runs have been performed in an attempt to establish whether the code is scalable or not. Results obtained in the transition from a a (96×96) grid to the refined (288×288) grid for the 2-D Danish Eulerian Model are given in Table 5. The results given in Table 5 indicate that the ratios of the computing times for the refined grid and the coarser grid are close to the expected ratios (see these ratios in Table 5). The conclusion from this experiment is that the code is scalable.

Many other experiments were carried out. The conclusion (that the code is scalable, which was drawn by using the results shown in Table 5) was further supported:

- by using results obtained in some runs with the more precise version in which the space domain is discretized on a (480×480) grid and

- by performing some runs with the 3-D version obtained by using a $(96 \times 96 \times 10)$ grid and comparing the results with the corresponding results calculated by applying the 2-D version obtained by using a (96×96).

8. PORTABILITY OF THE CODE

Some versions of the Danish Eulerian Model have recently been run on several other parallel computers (such as a SUN shared memory computer with up to 16 processors and a CRAY T3E distributed memory computer with up to 64 processors) at EPCC (Edinburgh Parallel Computing Centre). Rather good results have been achieved without any need to make changes in the code. Some results are given in Table 6 -Table 9.

The OPEN MP version of the 3-D Danish Eulerian Model is used on the SUN computer to produce the results given in Table 6. The comparison of the results in Table 6 with the results in Table 2 indicates that the efficiency of the parallel computation is fully preserved in the transition from one shared memory computer to another.

The MPI version of the 2-D Danish Eulerian Model applied on a refined (480×480) grid is used on the CRAY T3E computer to produce the results given in Table 7. As mentioned in Section 6, the code is so large that it cannot be run if only a few processors are used. The results in Table 7 show that the MPI version of the refined 2-D Danish Eulerian Model runs rather efficiently not only on the IBM SP computer, but also on the CRAY T3E computer.

Some runs with the 2-D version of the Danish Eulerian Model applied on a coarser (96×96) grid have also been carried out at EPCC. Results obtained with the OPEN MP code on the SUN computer are given in Table 8. The corresponding results obtained by the MPI code on the CRAY T3E computer are given in Table 9. These results confirm, once again, the statement that the codes (both the OPEN MP codes and the MPI codes) can easily be ported from one computer to another.

Table 6: Computing times (measured in seconds) obtained by using the SUN computer at EPCC when the 3-D version of the Danish Eulerian Model is discretized on a $(96 \times 96 \times 10)$ grid.

Processors	Comp. time	Speed-up	Efficiency
1	37565	-	-
16	3657	10.27	64%

Table 7: Computing times (measured in seconds) obtained by using the CRAY T3E computer at EPCC when the 2-D version of the Danish Eulerian Model is discretized on a (480×480) grid.

Processors	Comp. time	Speed-up	Efficiency
32	18306	-	-
64	9637	1.90	95%

Table 8: Computing times (measured in seconds) obtained by using the SUN computer at EPCC when the 2-D version of the Danish Eulerian Model is discretized on a (96×96) grid.

Processors	Comp. time	Speed-up	Efficiency
1	4356	-	-
16	391	11.14	70%

Table 9: Computing times (measured in seconds) obtained by using the CRAY T3E computer at EPCC when the 2-D version of the Danish Eulerian Model is discretized on a (96×96) grid.

Processors	Comp. time	Speed-up	Efficiency
1	7503	-	-
16	506	14.83	93%

The results could probably be improved by some tuning. Nevertheless, the results show clearly that one should use standard parallelization tools in the attempt to run efficiently the code on different modern supercomputers. This will facilitate the transition from one supercomputer to another.

9. CONCLUDING REMARKS AND PLANS FOR FUTURE WORK

The solution of many problems connected with the choice of numerical algorithms for air pollution models and with parallel runs of such models has allowed us to handle efficiently more problems and bigger problems.

However, the work must be continued, because the requirements of the decision-makers are becoming more and more stringent, which leads to the fact that the computational problems arising in the area of large-scale air pollution modelling are becoming more and more complicated. Twenty years ago, in the beginning of 80s, the number of equations in the semi-discretized problems was about 1000-2000. Now we should like to solve problems that lead to more than 80 million equations per ODE system in the semi-discretized models. These are very challenging tasks which still cannot be solved on the available computers. Therefore it is necessary to attempt to solve the following problems:

- to improve further the numerical methods used in the different modules of the models (trying to apply faster, but sufficiently accurate algorithms) and
- to improve further the performance of the codes when these are run on different high-speed computers by using only standard tools, which will allow us to port easier the codes to new and more powerful architectures when such architectures become available.

While both tasks are in principle simple, the practical solution of these tasks is very difficult. There are many reasons for this. Some of them are listed below.

- The air pollution codes are very long (containing many thousands lines of Fortran statements grouped in many different subroutines).
- The arrays used in the codes are very long and it is not always clear how to rearrange the computations in an optimal way.
- The different modules require the same data (for example, the arrays where the concentrations are stored) ordered in different ways.
- The amount of input and output data is increasing very quickly when the grids are refined. This causes difficulties even when modern big computers are available.
- The output data must be visualized in order to be able to see different trends and relationships between different parameters. The needed for the visialization portions of output data must be found and move to the computers where the graphic tools are available by searching for them in enormous data files.

This list can, of course, be continued. However, this is not necessary. It is much more important to emphasize the fact that the difficulties that are listed above show clearly that there exist a lot of open questions and many unresolved until now tasks in the area of air pollution modelling.

There are a lot of new and interesting problems in air pollution modelling, which were not discussed in this paper. Some examples for such problems are given below:

- The development of versions of the air pollution models with local refinement of the grid in prescribed areas (such a version is discussed in [2]).
- The use of data assimilation in air pollution models; the development of such versions is discussed in some of the papers in [55], see also [15] and [16].
- The treatment of some inverse problems (connected with the question: *How to keep the pollution levels in a given protected area under certain critical levels?*). Some discussion of these possibilities can be found in Dimov and Zlatev [13].
- The solution of some optimization problems (connected to the following question: *How to minimize the expenses of reducing emissions in order to reduce the concentrations and deposition in a given area to certain acceptable levels?*). This important topic is discussed in Dimov and Zlatev [13].

We are planning to study deeply some of the unresolved tasks, and to try to find some efficient solutions of them in the near future.

One of the major tools that will be used in the efforts to resolve some of the challenging tasks that are listed above is an object-oriented approach in the programming work related to large-scale air pollution models. An object-oriented version of the Danish Eulerian Model is under development (see Antonov [2]). The hope is that the flexibility of the object-oriented approach, which is applied in this new version, will facilitate the solution of many problems which are, at present, causing great difficulties

10. ACKNOWLEDGMENTS

This research was partly supported by the NATO Scientific Programme under the projects ENVIR.CGR 930449 and OUTS.CGR.960312, by the EU ESPRIT Programme under projects WEPTEL and EUROAIR and by NMR (Nordic Council of Ministers) under a common project for performing sensitivity studies with large-scale air pollution models in which scientific groups from Denmark, Finland, Norway and Sweden are participating. Furthermore, a grant from the Danish Natural Sciences Research Council gave us access to all Danish supercomputers.

The first author spent some time at the Edinburgh Parallel Computing Centre (EPCC), under the TRACS (Training and Research on Advanced Computing Systems) programme, where he tested our codes on the EPCC computers. His stay at EPCC has been supported by the European Commission (grant No. HPRI-1999-CT-00026). Finally, the first author has also been supported by the Bulgarian Scientific Council (grant No. I-901/99).

11. REFERENCES

[1] V. Alexandrov, A. Sameh, Y. Siddique and Z. Zlatev, Numerical integration of chemical ODE problems arising in air pollution models, *Environmental Modelling and Assessment*, 2:365–377, 1997.

[2] A. Antonov, *Object-oriented Framework for Large Scale Air Pollution Models*, PhD Thesis, Institute for Mathematical Modelling, Technical University of Denmark, DK-2800 Lyngby, Denmark, 2001.

[3] C. Bendtsen and Z. Zlatev, Running air pollution models on message passing machines, In *Parallel Virtual Machine and Message Passing Interface* (M. Bubak, J. Dongarra and J. Wasniewski, eds.), pages 417-426. Springer, Berlin, 1997.

[4] A. Bastrup-Birk, J. Brandt, I. Uria and Z. Zlatev, Studying cumulative ozone exposures in Europe during a seven-year period, *Journal of Geophysical Research*, 102:23917–23935, 1997.

[5] D. P. Chock, S. L. Winkler and P. Sun, Comparison of stiff chemistry solvers for air quality models, *Environ. Sci. Technol.*, 28:1882-1892, 1986.

[6] W. P. Crowley, Numerical advection experiments, *Monthly Weather Review*, 96:1–11, 1968.

[7] P. Deuflhard, Order and stepsize control in extrapolation methods, **Numer. Math**,. 41:399-422, 1983.

[8] P. Deuflhard, Recent progress in extrapolation methods for ordinary differential equations, *SIAM Rev.*, 27:505-535, 1985.

[9] P. Deuflhard, E. Hairer and M. Zugch, One step and extrapolation methods for differential-algebraic equations, *Numer. Math.*, 51:501-516, 1987.

[10] P. Deuflhard and U. Nowak, Efficient numerical simulation and identification of large reaction systems, *Ber. Bensendes Phys. Chem.*, 90:940-946, 1986.

[11] P. Deuflhard, U. Nowak and M. Wulkow, Recent development in chemical computing, *Computers Chem. Engng.*, 14:1249-1258, 1990.

[12] I. Dimov, I. Farago and Z. Zlatev, Commutativity of the operators in splitting methods for air pollution models. Report 04/99, Central Laboratory for Parallel Processing, Bulgarian Academy of Sciences, Acad. G. Bonchev, Bl. 25A, 1113 Sofia, Bulgaria, 1999.

[13] I. Dimov and Z. Zlatev, Optimization problems in air pollution modelling, In *Handbook on Applied Optimization* (P. M. Pardalos and G. C. Resende, eds.), Oxford University Press, London-Oxford, to appear.

[14] I. S. Duff, A. M. Erisman and J. K. Reid, Direct methods for sparse matrices, Oxford University Press, London-Oxford, 1986.

[15] H. Elbern, H. Schmidt and A. Ebel, Variational data assimilation for tropospheric chemistry modelling. *Journal of Geophysical Research*, 102:15967-15985, 1997.

[16] H. Elbern, H. Schmidt and A. Ebel, Implementation of a parallel 4D-variational chemistry data-assimilation scheme. *Environmental Management and Health*, 10:236-244, 1999.

[17] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek and V. Sunderam, *PVM: Parallel Virtual Machine, A User's Guide and Tutorial for Networking Parallel Computing.* MIT Press, Cambridge, Massachusetts, 1994.

[18] K. Georgiev and Z. Zlatev, Running an advection-chemistry code on message passing computers, In *Recent Advances in Parallel Virtual Machine and Message Passing Interface* (V. Alexandrov and J. Dongarra, eds.), pages 354-363. Springer, Berlin, 1998.

[19] K. Georgiev and Z. Zlatev, Parallel Sparse Matrix Algorithms for Air Pollution Models, *Parallel and Distributed Computing Practices*, to appear.

[20] M. W. Gery, G. Z. Whitten, J. P. Killus and M. C. Dodge, A photochemical kinetics mechanism for urban and regional computer modeling, *Journal of Geophysical Research*, 94:12925–12956, 1989.

[21] W. Gropp, E. Lusk and A. Skjellum, *Using MPI: Portable programming with the message passing interface*, MIT Press, Cambridge, Massachusetts, 1994.

[22] E. Hairer and G. Wanner, *Solving ordinary differential equations, II: Stiff and differential-algebraic problems*, Springer, Berlin, 1991.

[23] R. M. Harrison, Z. Zlatev and C. J. Ottley, A comparison of the predictions of an Eulerian atmospheric transport chemistry model with measurements over the North Sea, *Atmospheric Environment*, 28:497-516, 1994.

[24] O. Hertel, R. Berkowicz, J. Christensen and Ø. Hov, Test of two numerical schemes for use in atmospheric transport-chemistry models, *Atmospheric Environment*, 27A:2591-2611, 1993.

[25] E. Hesstvedt, Ø. Hov and I. A. Isaksen, Quasi-steady-state approximations in air pollution modelling: comparison of two numerical schemes for oxidant prediction, *International Journal of Chemical Kinetics*, 10:971–994, 1978.

[26] Ø. Hov, Z. Zlatev, R. Berkowicz, A. Eliassen and L. P. Prahm, Comparison of numerical techniques for use in air pollution models with non-linear chemical reactions, *Atmospheric Environment*, 23:967–983, 1988.

[27] A. Jaffe, Ordering of the universe: The role of mathematics, *SIAM Review*, 26:473-500, 1984.

[28] L. O. Jay, A. Sandu, F. A. Potra and G. R. Carmichael, Improved QSSA methods for atmospheric chemistry integration, *SIAM J. Sci. Comput.* 18:182-202, 1997.

[29] D. J. Kuck, E. S. Davidson, D. H. Lawrie and A. H. Sameh, Parallel supercomputing today and the CEDAR approach, *Science*, 231:967-974, 1986.

[30] J. D. Lambert, *Numerical methods for ordinary differential equations.* Wiley, New York, 1991.

[31] G. I. Marchuk, *Mathematical Modeling for the Problem of the Environment*, Studies in Mathematics and Applications, No. 16, North-Holland, Amsterdam, 1985.

[32] G. J. McRae, W. R. Goodin and J. H. Seinfeld, Numerical solution of the atmospheric diffusion equations for chemically reacting flows, *Journal of Computational Physics*, 45:1–42, 1984.

[33] C. R. Molenkampf, Accuracy of finite-difference methods applied to the advection equation, *Journal of Applied Meteorology*, 7:160–167, 1968.

[34] M. T. Odman, N. Kumar and A. G. Russell, A comparison of fast chemical kinetic solvers for air quality modeling, *Atmospheric Environment*, 26A:1783-1789, 1992.

[35] W. Owczarz and Z. Zlatev, Running a large air pollution model on an IBM SMP computer, *International Journal on Computer Research*, to appear.

[36] W. Owczarz and Z. Zlatev, Parallel matrix computations in air pollution modelling, Internal report, National Environmental Research Institute, DK-4000 Roskilde, Denmark, 2000.

[37] D. W. Pepper and A. J. Baker, A simple one-dimensional finite element algorithm with multidimensional capabilities, *Numerical Heath Transfer*, 3:81-95, 1979.

[38] D. W. Pepper, C. D. Kern and P. E. Long, Jr., Modelling the dispersion of atmospheric pollution using cubic splines and chapeau functions, *Atmospheric Environment*, 13:223–237, 1979.

[39] L. K. Peters, C. M. Berkowitz, G. R. Carmichael, R. C. Easter, Fairweather, G. Ghan, J. Hales, L. Leung, W. Pennell, F. Potra, R. D. Saylor and T. Tsang, The current state and future direction of Eulerian models in simulating the tropospherical chemistry and transport of trace species: A review, *Atmospheric Environment*, 29:189–221, 1995.

[40] A. Sandu, F. A. Potra, G. R. Carmichael and V. Damian, Efficient implementation of fully implicit methods for atmospheric chemical kinetics, *J. Comp. Phys.*, 129:101-110, 1996.

[41] A. Sandu, J. Verwer, J. Bloom, E. Spee and G. Carmichael, Benchmarking stiff ODE systems for atmospheric chemistry problems: II. Rosenbrock solvers, *Atmospheric Environment*, 31:3459–3472, 1997.

[42] A. Sandu, J. G. Verwer, M. van Loon, G. R. Carmichael, F. A. Potra, D. Dabdub and J. H. Seinfeld, Benchmarking stiff ODE systems for atmospheric chemistry problems: I. Implicit versus Explicit, *Atmospheric Environment*, 31:3151-3166, 1997.

[43] D. S. Shieh, Y. Chang and G. R. Carmichael, The evaluation of numerical techniques for solution of stiff ordinary differential equations arising from chemical kinetic problems, *Environ. Software*, 3:28-38, 1988.

[44] S. Skelboe and Z. Zlatev, Exploiting the natural partitioning in the numerical solution of ODE systems arising in atmospheric chemistry, In *Numerical Analysis and Its Applications* (L. Vulkov, J. Wasniewski, P. Yalamov, eds.), pages 458-465. Springer, Berlin, 1997.

[45] J. Swart and J. Blom, Experience with sparse matrix solvers in parallel ODE software, *Comp. Math. Appl.*, 31:43–55, 1996.

[46] 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*, 30:49-58, 1996.

[47] J. G. Verwer, M. van Loon, An evaluation of explicit pseudo-steady state approximation for stiff ODE systems from chemical kinetics, *J. Comp. Phys.* 113:347-352, 1996.

[48] J. G. Verwer and D. Simpson, Explicit methods for stiff ODE's from atmospheric chemistry, *Appl. Numer. Math.* 18:413-430, 1995.

[49] WEB-site for OPEN MP tools, *http://www.openmp.org*, 1999.

[50] WEB-site for the Danish Eurlerian Model (National Environmental Research Institute, DK-4000 Roskilde, Denmark), *http://www.dmu.dk/AtmosphericEnvironment/DEM*, 1999.

[51] R. A. Willoughby, Sparse matrix algorithms and their relation to problem classes and computer architecture, In *Large Sparse Sets of Linear Equations* (J. K. Reid, ed.), pages 255-277. Academic Press, New York, 1970.

[52] Z. Zlatev, Application of predictor-corrector schemes with several correctors in solving air pollution problems, *BIT*, 24:700–715, 1984.

[53] Z. Zlatev, *Computational Methods for General Sparse Matrices*, Kluwer Academic Publishers, Dordrecht, 1991.

[54] Z. Zlatev, *Computer Treatment of Large Air Pollution Models*, Kluwer Academic Publishers, Dordrecht, 1995.

[55] Z. Zlatev, J. Brandt, P. J. H. Builtjes, G. Carmichael, I. Dimov, J. Dongarra, H. van Dop, K. Georgiev, H. Hass and R. San Jose, eds., *Large Scale Computations in Air Pollution Modelling*, Kluwer Academic Publishers, Dordrecht, 1999.

[56] Z. Zlatev, J. Christensen and A. Eliassen, Studying high ozone concentrations by using the Danish Eulerian Model, *Atmospheric Environment*, 27A:845-865, 1993.

[57] Z. Zlatev, J. Christensen and Ø. Hov, An Eulerian air pollution model for Europe with nonlinear chemistry, *Journal of Atmospheric Chemistry*, 15:1-37, 1992.

[58] Z. Zlatev, I. Dimov and K. Georgiev, Studying long-range transport of air pollutants, *Computational Science and Engineering*, 1(3):45-52, 1994.

[59] Z. Zlatev, I. Dimov and K. Georgiev, Three-dimensional version of the Danish Eulerian Model, *Zeitschrift für Angewandte Mathematik und Mechanik*, 76:473–476, 1996.

[60] Z. Zlatev, I. Dimov, Tz. Ostromsky, G. Geernaert, I. Tzvetanov and A. Bastrup-Birk, Calculating Losses of Crops in Denmark Caused by High Ozone Levels, *Environmental Modeling and Assessment*, to appear.

[61] Z. Zlatev, J. Fenger and L. Mortensen, Relationships between emission sources and excess ozone concentrations, *Computers and Mathematics with Applications*, 32(11):101–123. 1996.

[62] Z. Zlatev, G. Geernaert and H. Skov, A study of ozone critical levels in Denmark over a ten-year period, *EUROSAP Newsletter*, 36: 1-9, 1999.