Available online at www.sciencedirect.com
SCIENCE DIRECT®
Mathematics and Computers in Simulation 65 (2004) 557–577
MATHEMATICS AND COMPUTERS IN SIMULATION
www.elsevier.com/locate/matcom

Parallel runs of a large air pollution model on a grid of Sun computers

V.N. Alexandrov [a], W. Owczarz [b], P.G. Thomson [c], Z. Zlatev [d,*]

[a] *Computer Science Department, University of Reading, Reading, UK*
[b] *Danish Computing Centre for Research and Education, Technical University of Denmark, DK-2800 Lyngby, Denmark*
[c] *Informatics and Mathematical Modelling, Technical University of Denmark, DK 2800 Lyngby, Denmark*
[d] *National Environmental Research Institute, Frederiksborgvej 399, P. O. Box 358, DK-4000 Roskilde, Denmark*

Received 20 January 2004; received in revised form 20 January 2004; accepted 21 January 2004

Abstract

Large-scale air pollution models can successfully be used in different environmental studies. These models are described mathematically by systems of partial differential equations. Splitting procedures followed by discretization of the spatial derivatives lead to several large systems of ordinary differential equations of order up to 80 millions. These systems have to be handled numerically at up to 250,000 time-steps. Furthermore, many scenarios are often to be run in order to study the dependence of the model results on the variation of some key parameters (as, for example, the emissions). Such huge computational tasks can successfully be treated only if: (i) fast and sufficiently accurate numerical methods are used and (ii) the models can efficiently be run on parallel computers.

The mathematical description of a large-scale air pollution model will be discussed in this paper. The principles used in the selection of numerical methods and in the development of parallel codes will be described. Numerical results, which illustrate the ability of running the fine resolution versions of the model on Sun computers, will be given. Applications of the model in the solution of some environmental tasks will be presented.

Keywords: Air pollution modelling; Partial differential equations; Ordinary differential equations; Numerical methods; Cache utilization; Parallel computations

1. Description of the model

The control of the pollution levels in different highly polluted regions of Europe and North America (as well as in other highly industrialized parts of the world) is an important task for the modern society. Its relevance has been steadily increasing during the last two-three decades. The need to establish reliable control strategies for the air pollution levels will become even more important in the future. Large-scale

* Corresponding author. Tel.: +45-4630-1149; fax: +45-4630-1214.
E-mail addresses: v.n.alexandrov@reading.ac.uk (V.N. Alexandrov), neuwow@unidhp.uni-c.dk (W. Owczarz), pgt@imm.dtu.dk (P.G. Thomson), zz@dmu.dk (Z. Zlatev).

0378-4754/$30.00 © 2004 IMACS. Published by Elsevier B.V. All rights reserved.
doi:10.1016/j.matcom.2004.01.022

air pollution models can successfully be used to design reliable control strategies. Many different tasks have to be solved before starting to run operationally an air pollution model. The following tasks are most important:

- describe in an adequate way all important physical and chemical processes;
- apply fast and sufficiently accurate numerical methods in the different parts of the model;
- ensure that the model runs efficiently on modern high-speed computers (and, first and foremost, on different types of parallel computers);
- use high quality input data (both meteorological data and emission data) in the runs;
- verify the model results by comparing them with reliable measurements taken in different parts of the space domain of the model;
- carry out some sensitivity experiments to check the response of the model to changes of different key parameters; and
- visualize and animate the output results to make them easily understandable also for non-specialists.

In this paper, we shall concentrate our attention on the solution of the first three tasks (however, some visualizations will be used to present results from some real-life runs in the end of the paper). The air pollution model, which is actually used here, is the Danish Eulerian Model (DEM); see [36,38]. However, the principles are rather general, which means that most of the results are also valid for other air pollution models.

1.1. Main physical and chemical processes

Five physical and chemical processes have to be described by mathematical terms in the beginning of the development of an air pollution model. These processes are:

- horizontal transport (advection),
- horizontal diffusion,
- chemical transformations in the atmosphere combined with emissions from different sources,
- deposition of pollutants to the surface, and
- vertical exchange (containing both vertical transport and vertical diffusion).

It is important to describe in an adequate way all these processes. However, this is an extremely difficult task; both because of the lack of knowledge for some of the processes (this is mainly true for some chemical reactions and for some of the mechanisms describing the vertical diffusion) and because a very rigorous description of some of the processes will lead to huge computational tasks which may make the treatment of the model practically impossible. The main principles used in the mathematical description of the main physical and chemical processes as well as the need to keep the balance between the rigorous description of the processes and the necessity to be able to run the model on the available computers are discussed in [36].

1.2. Mathematical formulation of a large air pollution model

The description of the physical and chemical processes by mathematical terms leads to a system of partial differential equations (PDEs) of the following type:

$$\frac{\partial c_s}{\partial t} = -\frac{\partial(uc_s)}{\partial x} - \frac{\partial(vc_s)}{\partial y} - \frac{\partial(wc_s)}{\partial z} + \frac{\partial}{\partial x}\left(K_x\frac{\partial c_s}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y\frac{\partial c_s}{\partial y}\right) + \frac{\partial}{\partial z}\left(K_z\frac{\partial c_s}{\partial z}\right) + E_s - (\kappa_{1s}+\kappa_{2s})c_s + Q_s(c_1, c_2, \ldots, c_q), \ldots, \quad s = 1, 2, \ldots, q, \tag{1.1}$$

where (i) the concentrations of the chemical species are denoted by c_s, (ii) u, v and w are wind velocities, (iii) K_x, K_y and K_z are diffusion coefficients, (iv) the emission sources are described by E_s, (v) κ_{1s} and κ_{2s} are deposition coefficients and (vi) the chemical reactions are denoted by $Q_s(c_1, c_2, \ldots, c_q)$. The CBM IV chemical scheme, which has been proposed in [14], is actually used in the version of DEM (the Danish Eulerian Model; [36,38]) that will be considered in this paper. It should be mentioned here that the CBM IV scheme is also used in other well-known air pollution models.

1.3. Space domain

The space domain of DEM is a 4800 km × 4800 km square, which contains the whole of Europe together with parts of Africa, Asia, the Arctic area and the Atlantic Ocean. Two discretizations of this domain, a coarse one and a fine one, will be used in this paper. The space domain is divided into 96×96 small, 50 km × 50 km, squares when the coarse discretization is applied. The space domain is divided into 480 × 480 small, 10 km × 10 km, squares when the fine discretization is applied. Thus, one of the coarse grid-squares contains 25 small grid-squares.

1.4. Initial and boundary conditions

If initial conditions are available (for example from a previous run of the model), then these are read from the file where they are stored. If initial conditions are not available, then a five day start-up period is used to obtain initial conditions (i.e. the computations are started five days before the desired starting date with some background concentrations and the concentrations found at the end of the fifth day are actually used as starting concentrations).

The choice of lateral boundary conditions is in general very important. However, if the space domain is very large, then the choice of lateral boundary conditions becomes less important; which is stated on p. 2386 in [6]: *"For large domains the importance of the boundary conditions may decline"*. The lateral boundary conditions are represented in the Danish Eulerian Model with typical background concentrations which are varied, both seasonally and diurnally. It is better to use values of the concentrations at the lateral boundaries that are calculated by a hemispheric or global model when such values are available.

For some chemical species, as for example ozone, it is necessary to introduce some exchange with the free troposphere (on the top of the space domain).

The choice of initial and boundary conditions is discussed in [15,36,38–40].

1.5. Applying splitting procedures

It is difficult to treat the system of PDEs (1.1) directly. This is the reason for using different kinds of splitting. A splitting procedure, which is based on ideas proposed in [23,24], and which leads to five sub-models, has been proposed in [36] and used after that in many studies involving DEM (as, for example, in [38]). Each of the five sub-models obtained by this splitting procedure is representing one of the major physical and chemical processes discussed in Section 1.1; i.e. the horizontal advection, the horizontal diffusion, the chemistry (together with the emission terms), the deposition and the vertical exchange.

In the newest version of DEM, which is used here, the horizontal advection was merged with the horizontal diffusion, while the chemical sub-model was combined with the deposition sub-model. This

means that the number of sub-models is reduced from five to three:

$$\frac{\partial c_s^{(1)}}{\partial t} = -\frac{\partial (w c_s^{(3)})}{\partial z} + \frac{\partial}{\partial z}\left(K_z \frac{\partial c_s^{(3)}}{\partial z}\right) \tag{1.2}$$

$$\frac{\partial c_s^{(2)}}{\partial t} = -\frac{\partial (u c_s^{(2)})}{\partial x} - \frac{\partial (v c_s^{(2)})}{\partial y} + \frac{\partial}{\partial x}\left(K_x \frac{\partial c_s^{(2)}}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y \frac{\partial c_s^{(2)}}{\partial y}\right) \tag{1.3}$$

$$\frac{\mathrm{d}c_s^{(3)}}{\mathrm{d}t} = E_s + Q_s(c_1^{(3)}, c_2^{(3)}, \ldots, c_q^{(3)}) - (\kappa_{1s} + \kappa_{2s}) c_s^{(3)} \tag{1.4}$$

The first of these sub-models, (Section 1.2), describes the vertical exchange. The second sub-model, (1.3), describes the combined horizontal transport (the advection) and the horizontal diffusion. The last sub-model, (1.4), describes the chemical reactions together with emission sources and deposition terms.

The boundary conditions can be treated in a natural way when the splitting procedure described by (1.2)–(1.4) is used. The implementation of the boundary conditions is performed as follows:

- The boundary conditions on the top and the bottom of the space domain are treated in (1.2), where the computations are carried out along the vertical grid-lines.
- The lateral boundary conditions are handled in (1.3), where the the computations are carried out in each of the horizontal grid-planes.
- The computations related to (1.4) are carried out by performing the chemical reactions at each grid-point. It is clear that the computations at any of the grid-points do not depend on the computations at the remaining grid-points. Therefore, no boundary conditions are needed when (1.4) is handled.

The main principles used to treat the sub-models at a given time-step are the same as the principles discussed in [23,24,36]; see also [41]. A thorough discussion of different types of splitting splitting procedure can be found in the recently published book by Hundsdorfer and Verwer [19]. Some convergence results are presented in Farago and Havasi [10].

Splitting allows us to apply different numerical methods in the different sub-models and, thus, to reduce considerably the computational work and to exploit better the properties of each sub-model. These are the main advantages of using splitting. Unfortunately, there are drawbacks also: the splitting procedure is introducing errors, and it is difficult to control these errors. Some attempts to obtain some evaluation of the splitting errors were recently carried out; see [21,9].

1.6. Space discretization

Assume that the space domain is discretized by using a grid with $N_x \times N_y \times N_z$ grid-points, where N_x, N_y and N_z are the numbers of the grid-points along the grid-lines parallel to the Ox, Oy and Oz axes. Assume further that the number of chemical species involved in the model is $q = N_s$. Finally, assume that the spatial derivatives in (1.2) are discretized by some numerical algorithm. Then the system of PDEs (1.2) will be transformed into a system of ODEs (ordinary differential equations):

$$\frac{\mathrm{d}g^{(1)}}{\mathrm{d}t} = f^{(1)}(t, g^{(1)}), \tag{1.5}$$

In a similar way, the system of PDEs (1.3) can be transformed into the following system of ODEs when the spatial derivatives in the right-hand-side of (1.3) are discretized:

$$\frac{\mathrm{d}g^{(2)}}{\mathrm{d}t} = f^{(2)}(t, g^{(2)}), \tag{1.6}$$

There are in fact no spatial derivatives in the right-hand-side of (1.4), because the non-linear functions Q_s can be represented as

$$Q_s(c_1, c_2, \ldots, c_q) = -\sum_{i=1}^{q} \alpha_{si} c_i + \sum_{i=1}^{q}\sum_{j=1}^{q} \beta_{sij} c_i c_j, \quad s = 1, 2, \ldots, q. \tag{1.7}$$

where α_{si} and β_{sij} are coefficients describing the rates of the chemical reactions (for the CBM IV schemes these coefficients are listed in [36]). By using this observation, it is easy to represent (1.4) as a system of ODEs:

$$\frac{\mathrm{d}g^{(3)}}{\mathrm{d}t} = f^{(3)}(t, g^{(3)}), \tag{1.8}$$

The components of functions $g^{(i)}(t) \in R^{N_x \times N_y \times N_z \times N_s}$, $i = 1, 2, 3$, are the approximations of the concentrations (at time t) at all grid-squares and for all species. The components of functions $f^{(i)}(t, g) \in R^{N_x \times N_y \times N_z \times N_s}$, $i = 1, 2, 3$, depend on the numerical method used in the discretization of the spatial derivatives.

A simple linear finite element method is used to discretize the spatial derivatives in (1.2) and (1.3). This method is described in [28,29]. Its implementation in DEM is discussed in [12].

The spatial derivatives can also be discretized by using other numerical methods:

- Pseudospectral discretization (described in detail in [36]).
- Semi-Lagrangian discretization (can be used only to discretize the first-order derivatives, i.e. the advection part should not be combined with the diffusion part when this method is to be applied), see for example [22].
- Methods producing non-negative values of the concentrations. The method proposed in [4] is often used in air pollution modelling. The method from [18]is based on a solid theoretical foundation.

As mentioned above, there are no spatial derivatives in (1.4), which means that the system of ODEs (1.8) is trivially obtained by (1.4).

Much more details about the methods, which can be used in the space discretization, can be found in [36].

1.7. Time integration

It is necessary to couple the three ODE systems (1.5), (1.6) and (1.8). The coupling procedure is connected with the time-integration of these systems. Assume that the values of the concentrations (for all species and at all grid-points) have been found for some $t = t_n$. According to the notation introduced in the previous sub-section, these values can be considered as components of a vector-function $g(t_n) \in R^{N_x \times N_y \times N_z \times N_s}$. The next time-step, time-step $n+1$ (at which the concentrations are found at $t_{n+1} = t_n + \triangle t$, where $\triangle t$ is some increment), can be performed by integrating successively the three systems. The values

of $g(t_n)$ are used as an initial condition in the solution of (1.5). The solution of (1.5) is used as an initial condition of (1.6). Finally, the solution of (1.6) is used as an initial condition of (1.8). The solution of the last system (1.8) is used as an approximation to $g(t_{n+1})$. In this way, everything is prepared to start the calculations in the next time-step, step $n + 2$.

The first ODE system, (1.5), can be solved by using many classical time-integration methods. The so-called θ-method (see, for example, [20]) is currently used in DEM. The choice of numerical method is not very critical in this part, because as it will be shown Section 4, it is normally not very expensive.

Predictor-corrector methods with several different correctors are used in the solution of the ODE system (1.6). The correctors are carefully chosen so that the stability properties of the method are enhanced; see [35]. The reliability of the algorithms used in the advection part was verified by using the well-known rotational test proposed simultaneously in 1968 by [7,25].

The solution of (1.8) is much more complicated, because this system is both time-consuming and stiff. Very often the QSSA method is used in this part of the model. The QSSA (quasi-steady-state approximation; see, for example, [16] or [17]) is simple and relatively stable but not very accurate (therefore it has to be run with a small time-stepsize). The QSSA method can be viewed as an attempt to transform dynamically, during the process of integration, the system of ODEs (1.8) into two systems: a system of ODEs and a system of non-linear algebraic equations. These two systems, which have to be treated simultaneously, can be written in the following generic form:

$$\frac{\mathrm{d}g_1}{\mathrm{d}t} = f_1(t, g_1, g_2), \tag{1.9}$$

$$0 = f_2(t, g_1, g_2). \tag{1.10}$$

In this way we arrive at a system of differential-algebraic equations (DAEs). There are special methods for treating such systems as, for example, the code DASSL (see [5]). Problem-solving environments (such as MATLAB or Simulink) can be used in the preparation stage (where a small chemical systems at one grid-point only is used in the tests). More details about the use of such problem solving environments can be found in [30]. A method based on the solution of DAE for air pollution models was recently proposed in [11].

The classical numerical methods for stiff ODE systems (such as the Backward Euler Method, the Trapezoidal Rule and Runge-Kutta algorithms) lead to the solution of non-linear systems of algebraic equations and, therefore, they are more expensive; [20]. On the other hand, these methods can be incorporated with an error control and perhaps with larger time-steps. The extrapolation methods, [8], are also promising. It is easy to calculate an error estimation and to carry out the integration with large time-steps when these algorithms are used. However, it is difficult to implement such methods in an efficient way when all three systems, (1.5), (1.6) and (1.8), are to be treated successively.

Partitioning can also be used [1]. Some convergence problems related to the implementation of partitioning are studied in [37].

The experiments with different integration methods for the chemical sub-model are continuing. The QSSA with some enhancements based on ideas from [31,32]will be used here. The method is described in [1]. There are still very open questions related to the choice of method for the chemical part. The choice of the improved QSSA method was made in order to get well-balanced parallel tasks.

2. Need for high-performance computing in the treatment of large air pollution models

The computers are becoming more and more powerful. Many tasks, which several years ago had to be handled on powerful supercomputers, can be handled at present on PCs or work-stations. However, there are still many tasks that can only be run on parallel computers. This is especially true for the large air pollution models. The size of the computational tasks in some versions of DEM is given in the following two paragraphs in order demonstrate the fact that high-performance computing is needed when large air pollution models are to be treated.

2.1. Size of the computational tasks when 2-D versions are used

Only the two systems of ODEs (1.6) and (1.8) have to be treated in this case. Assume first that the coarse 96×96 grid is used. Then the number of equations in each of the two systems of ODEs (1.6) and (1.8) is equal to the product of the grid points (9216) and the number of chemical species (35), i.e. 322,560 equations have to be treated at each time-step when any of the systems (1.6) and (1.8) is handled. The time-stepsize used in the transport sub-model (1.6) is 900 s. This stepsize is too big for the chemical sub-model; the time-stepsize used in the latter model is 150 s. A typical run of this model covers a period of one year (in fact, as mentioned above), very often a period of extra five days is needed to start up the models. This means that 35,520 time-steps are needed in the transport sub-model, while six times more time-steps, 213,120 time-steps, are needed in the chemical part. If the number of scenarios is not large, then this version of the model can be run on PCs and work-stations. If the number of scenarios is large or if runs over many years have to be performed (which is the case when effects of future climate changes on the air pollution levels is studied), then high-performance computations are preferable (this may be the only way to complete the study when either the number of scenarios is very large or the time period is very long).

Assume now that the fine 480×480 grid is used. Since the number of chemical species remains unchanged (35), the number of equations in each of the systems (1.6) and (1.8) is increased by a factor of 25 (compared with the previous case). This means that 8,064,000 equations are to be treated at each time step when any of the systems (1.6) and (1.8) is handled. The time-stepsize remains 150 s when the chemical part is treated. The time-stepsize has to be reduced from 900 to 150 s in the transport part. This means that a typical run (one year + 5 days to start up the model) will require 213,520 time-steps for each of the systems (1.6) and (1.8). Consider the ratio of the computational work when the fine grid is used and the computational work when the coarse grid is used. For the transport sub-model this ratio is 150, while the ratio is 25 for the chemical-sub-model. It is clear that this version of the model must be treated on powerful parallel architectures.

2.2. Size of the computational tasks when 3-D versions are used

All three sub-models, (1.5)–(1.7), have to be treated in this case. Assume that the number of layers in the vertical direction is n ($n = 10$ is used in this paper). Under this assumption the computational work when both (1.6) and (1.8) is handled by the 3-D versions (either on a coarse grid or on a fine grid) is n times bigger than the computational work for the corresponding 2-D version. The work needed to handle (1.5) is extra, but this part of the total computational work is much smaller than the parts needed to treat (1.6) and (1.8).

The above analysis of the amount of the computational work shows that it is much more preferable to run the 3-D version on high-speed parallel computers when the coarse grid is used. It will, furthermore, be shown that the runs are very heavy when the 3-D version is to be run on a fine grid. In fact, more powerful parallel computers than the computers available at present are needed if meaningful studies with the 3-D version of DEM discretized on a fine grid are to be carried out.

2.3. Exploiting the cache memory of the computer

In the modern computers the time needed for performing arithmetic operations is reduced dramatically (compared with computers which were available 10–15 years ago). However, the reductions of both the time needed to bring the numbers which are participating in the arithmetic operations from the memory to the place in the computer where the arithmetic operation is to be actually performed and the time needed to store the results back in the memory are much smaller. This is why most of the nowadays computers have different caches. It is much more efficient to use data which is in cache than to make references to the memory. Unfortunately, it is very difficult for the user (if at all possible) to control directly the utilization of the cache. Nevertheless, there are some common rules by the use of which the performance can be improved considerably. The rules discussed in [26,27] will be outlined below. These rules have been used in runs on several other computers in [26,27]. It will be shown in Section 4 that these rules are performing rather well also when Sun parallel computers are used.

Consider the 2-D versions of DEM. Assume that the concentrations are stored in an array CONS($N_x \times N_y$, N_s). Each column of this array is representing the concentrations of a given chemical species at all grid-points, while each row is containing the concentrations of all chemical species at a given grid-point. There are seven other arrays of the same dimension.

There are no big problems when the transport sub-model is run (because the computations are carried out by columns). However, even here cache problems may appear, because the arrays are very long. This will be further discussed in Section 4.

Great problems appear in the chemical part, because when the concentration of some species in a given row is modified, some other species in the same row are participating in the computations, which becomes clear from the pseudo Fortran code given below (with $M = N_x \times N_y$ and NSPECIES $= N_s$).

```
DO J = 1, NSPECIES
    DO I = 1, M
        Perform the chemical reactions involving
        species J in grid-point I
    END DO
END DO
```

This code is perfect for some vector machines. However, if cache memory is available, then the computations, as mentioned above, can be rather slow, because in step I, $I = 1, 2, \ldots M$, of the inner loop CONS(I, J) is updated, but the new value of the chemical species J depends on some of the other species K, $K = 1, 2, \ldots, J-1, J+1, \ldots,$ NSPECIES. Thus, when we are performing the Ith step of the second loop, we have to refer to some addresses in row I of array CONS(M, NSPECIES). The same is true for the seven other arrays of the same dimension. It is intuitively clear that it is worthwhile to divide these

arrays into chunks and to carry out the computations by chunks. Assume that we want to use NCHUNKS chunks. If M is a multiple of NCHUNKS, then the size of every chunks is NSIZE $= M$/NCHUNKS, and the code given above can be modified in the following way.

```
DO ICHUNK = 1, NCHUNKS
    Copy chunk ICHUNK from some of the eight large arrays into small two-dimensional arrays with
      leading dimension NSIZE
        DO J = 1, NSPECIES
            DO I = 1, NSIZE
            Perform the chemical reactions involving species J for grid-point I
            END DO
        END DO
    Copy some of the small two-dimensional arrays with leading dimension NSIZE into chunk
      ICHUNK of the corresponding large arrays
END DO
```

Both the operations that are performed in the beginning and in the end of the first loop in the second code are extra. The extra work needed to perform these operations is fully compensated by savings during the inner double loop, which is very time-consuming.

A straight-forward procedure will be to copy the current chunks of all eight arrays in the corresponding small arrays. However, this is not necessary, because some of the arrays are only used as helping arrays in the chemical module. In fact, copies from five arrays are needed in the beginning of the first loop. This means that there is no need to declare the remaining three arrays as large arrays; these arrays can be declared as arrays with dimensions (NSIZE, NSPCIES), which leads to a reduction of the storage needed. The reduction is very considerable for the fine 480×480 grid.

The situation in the end of the first loop is similar; it is necessary to copy back to the appropriate sections of the large arrays only the contents of three small arrays. The number of copies made at the end of the first loop has been reduced from five to three because some information (as, for example, the emissions) is needed in the chemical module (and has to be copied from the large arrays to the small ones), but it is not modified in the chemical module (and, thus, there is no need to copy it back to the large arrays in the end of the first loop).

When the 3-D versions are used, the array CONS$(N_x \times N_y, N_s)$ must be replaced by CONS$(N_x \times N_y, N_s, N_z)$. However, the device described above can be applied, because the computations for each layer can be carried out independently from the computations for the other layers when (1.6) and (1.8) are treated.

It will be shown in Section 4 that the use of chunks leads to considerable savings in computing time in the chemical sub-model.

3. Achieving parallelism

It was explained in the previous section that the discretization of an air pollution model is as a rule resulting in huge computational tasks. This is especially true in the case where the model is discretized

on a fine grid. Therefore it is important to prepare parallel codes which run efficiently on modern parallel computers. The preparation of such a code will be discussed in this section.

3.1. Basic principles used in the preparation of the parallel versions

The preparation of a parallel code is by no means an easy task. Moreover, it may happen that when the code is ready the computing centre exchanges the computer which has been used in the preparation of the code with another (hopefully, more powerful) computer. This is why it is desirable to use only standard tools in the preparation of the code. This will facilitate the transition of the code from one computer to another when this becomes necessary. Only standard OpenMP [33] and MPI [13]tools are used in the parallel versions of DEM.

3.2. Development of OpenMP versions of DEM

The programming for shared memory machines is relatively easy. It is necessary to identify the parallel tasks and to insert in the code appropriate OpenMP directives (which on ordinary sequential machines will be viewed as comments). The parallel tasks in the three sub-models are discussed below.

3.2.1. Parallel tasks in the transport sub-model

This sub-model is mathematically described (after the discretization) by (1.6). It is easy to see that the system of ODEs (1.6) is consisting of $q \times N_z$ independent systems of ODEs, where q is the number of chemical species and N_z is the number of grid-points in the vertical direction. This means that there are $q \times N_z$ parallel tasks. Each parallel task is a system of $N_x \times N_y$ ODEs. In the chemical scheme adopted in DEM there are 35 chemical species, but three of them are linear combinations of other chemical species. N_z is equal to 1 in the 2-D case and to 10 in the 3-D case. Therefore, the actual number of parallel tasks is 32 in the 2-D case and 320 in the 3-D case. The tasks are large and the loading balance in the transport sub-model is perfect. The use of this technique is, thus, very efficient when the number of processors used is a divisor of 32 in the 2-D case and 320 in the 3-D case. Some problems may arise in the 2-D case. If more than 32 processors are available, then it will be necessary to search for parallel tasks on a lower level of the computational process when the 2-D versions are used.

3.2.2. Parallel tasks in the chemical sub-model

This sub-model is mathematically described (after the discretization) by (1.8). It is easy to see that the system of ODEs (1.8) is consisting of $N_x \times N_y \times N_z$ independent systems of ODEs, where N_x, N_y and N_z are the numbers of grid-points along the coordinate axes. The number of parallel tasks is very large (2304000 when the $480 \times 480 \times 10$ grid is used) and the loading balance is perfect. However, the parallel tasks are very small (each parallel task is a system of q ODEs). Therefore, it is necessary to group them in clusters. Moreover, some arrays are handled by rows, which may lead to a large number of cache misses, especially for the fine grid versions. Therefore, chunks are to be used in this part (see the end of the previous version).

3.2.3. Parallel tasks in the vertical exchange sub-model

This sub-model is mathematically described (after the discretization) by (1.5). It is easy to see that the system of ODEs (1.5) is consisting of $N_x \times N_y \times N_s$ independent systems of ODEs. N_x and N_y are the

numbers of grid-points along the coordinate axes O_x and O_y. $N_s = q$ is the number of chemical species. The number of parallel tasks is very large (8,064,000 when the $480 \times 480 \times 10$ grid is used with 35 chemical species) and the loading balance is perfect. However, the parallel tasks are again small (each parallel task is a system of N_z ODEs). Therefore, also in this sub-model it is necessary to group the parallel tasks in an appropriate way. It should also be emphasized that a very long array (its leading dimension being $N_x \times N_y \times N_s$) has to handled by rows. The vertical exchange is not very expensive computationally. Nevertheless, it is desirable to use chunks in the efforts to avoid a large number of cache misses (this is especially true for the fine resolution versions). No chunks are used at present, but there are plans to introduce chunks in the near future.

It is seen from the above discussion that it is very easy to organize the computational process for parallel runs when OpenMP tools are used. Moreover, it is clear that the parallel computations depend on the splitting procedure, but not on the numerical methods that have been selected.

3.3. Development of MPI versions of DEM

The approach used when MPI tools are to be implemented is based in dividing the space domain of the model into p sub-domains, where p is the number of processors which are to be used in the run. Two specific modules are needed in the MPI versions: (i) a pre-processing module and (ii) a post-processing module.

3.3.1. The pre-processing module

The input data is divided into p portions corresponding to the p sub-domains obtained in the division of the space domain. In this way, each processor will work during the whole computational process with its own set of input data.

3.3.2. The post-processing module

Each processor prepares its own set of output data. During the post-processing the p sets of output data corresponding to the p sub-domains are collected and common output files are prepared for future use.

3.3.3. Benefits of using the two modules

Excessive communications during the computational process are avoided when the two modules are used. It should be stressed, however, that not all communications during the computational process are avoided. Some communications along the inner boundaries of the sub-domains are still needed. However, these communications are to be carried only once per step and only a few data are to be communicated. Thus, the actual communications that are to be carried out during the computations are rather cheap when the pre-processing and the post-processing modules are proper implemented.

It is important to emphasize here that the introduction of p sub-domains leads to a reduction of the main arrays by a factor of p. Consider as an illustrations the major arrays used in the chemical sub-model. The dimensions of these arrays are reduced from $(N_x \times N_y, N_s)$ to $(N_x \times N_y/p, N_s)$. It is clear that this is equivalent to the use of p chunks; see Section 2.3. Chunks of length $N_x \times N_y/p$ are still very large. Therefore, the second algorithm given in Section 2.3 has also to be used (in each sub-domain) when the MPI versions are used. However, the reduction of the arrays leads to a reductions of the copies that are to be made in the beginning and in the end of the second algorithm in Section 2.3. Thus, the reduction of the arrays leads to a better utilization of the cache memory.

The automatic reduction of the sizes of the involved arrays, and the resulting from this reduction better utilization of the cache memory, make the MPI versions attractive also when shared memory machines are available. It will be shown in the next section that on Sun computers the MPI versions of DEM are often performing better than the corresponding OpenMP versions.

4. Numerical results

Some results will be presented in this sections to demonstrate (i) the efficiency of the better utilization of the cache memory by using chunks and (ii) the good speed-ups (very often super-linear) that can be achieved when the code is run in parallel. We start by presenting short information about the computers used.

4.1. Description of the grid of Sun computers

Sun computers located at the Danish Centre for Scientific Computing (the Danish Technical University in Lyngby) were used in the runs. The computers and the their characteristics are shown in Table 1. All these computers were connected with a 1 Gbit/s switch.

The computers are united in a grid (consisting of 216 processors) so that a job sent without a special demand will be assigned on the computer on which there are sufficiently many free processors. The different computers have processors of different power (therefore, it is in principle possible to use the grid as a heterogeneous architecture, but this option is not available yet).

We are in general allowed to use no more than 16 processors, but several runs on more that 16 processors were performed with a special permission from the Danish Centre for Scientific Computing. In the runs in this section we used only “Newton” (i.e. we had always a requirement specifying the particular computer on which the job must be run)

More details about the high speed computers that are available at the Technical University of Denmark can be found in [34].

4.2. Running the MPI versions of DEM

Four MPI versions of DEM have been tested: (i) the 2-D model on a coarse grid, (ii) the 3-D version on a coarse grid, (iii) the 2-D version on a fine grid and (iv) the 3-D version on a fine grid.

Table 1
The computers available at the Sun grid

Computer	Type	Power	RAM	Processors
Bohr	Sun Fire 6800	UltraSparc-III 750 MHz	48 GB	24
Erlang	Sun Fire 6800	UltraSparc-III 750 MHz	48 GB	24
Hald	Sun Fire 12k	UltraSparc-III 750 MHz	144 GB	48
Euler	Sun Fire 6800	UltraSparc-III 750 MHz	24 GB	24
Hilbert	Sun Fire 6800	UltraSparc-III 750 MHz	36 GB	24
Newton	Sun Fire 15k	UltraSparc-IIIcu 900 MHz	404 GB	72

The problems were run with three different sizes NSIZE of chunks: (a) the minimal size of the chunks, NSIZE = 1 for all cases, (b) a medium size of the chunks, NSIZE = 24 for all cases and (c) the maximal size of the chunks, which is NSIZE = 1152 for the coarse grid when eight processors are used and NSIZE = 28800 for the fine grid (again when eight processors are used).

Finally, in most of the cases both 1 processor and eight processors were used. Some of the jobs were also run on more than eight processors.

All runs of the versions discretized on the coarse grid were run for the typical period of one year (in which case it is possible to study seasonal variations). The 2-D version of DEM discretized on the fine grid was run over a period of one month. Finally, the 3-D version of DEM discretized on the fine grid was run over a time period of 42 h. This is a rather short period, but it is still meaningful to a certain degree because several changes from day to night and from night to day occur in this period, which is important for the test of the photo-chemical reactions.

The computing times in all tables are given in seconds. The abbreviations used in the tables can be explained as follows:

- ADV stands for the horizontal transport + diffusion process,
- CHEM stands for the process uniting the chemical reactions, the treatment of the emissions and the deposition part,
- COMM stands for the part needed to perform communications along the inner boundaries,
- VERT stands for the vertical exchange processes
- TOTAL stands for the total computing time (including the sum of the times given in the same column above the last item + the computing times needed for performing input-output operations, pre-processing, post-processing, etc.)

The percentages of the computing times for the different processes related to the total computing times are given in the columns under “Part”. The “Speed-up” is the ratio of the computing time on one processor and the computing time on p processors (where p is the number of processors that are used in the run under considerations; as mentioned above, eight processors were as a rule used in our experiments).

4.2.1. Running the 2-D MPI version discretized on the coarse grid

Results from the six runs with this code are shown in Table 2 (runs on one processor performed by using three values of NSIZE) and Table 3 (runs on eight processors performed again with three values of NSIZE).

Table 2
Running DEM discretized on a 96 × 96 × 1 grid on one processor

Process	NSIZE = 1		NSIZE = 24		NSIZE = 1152	
	Time	Part	Time	Part	Time	Part
ADV	17617	28.2%	16035	32.6%	16742	26.8%
CHEM	37353	59.8%	26671	54.2%	38828	62.1%
COMM	2	0.0%	2	0.0%	2	0.0%
TOTAL	62443	100.0%	49239	100.0%	62510	100.0%

Table 3
Running DEM discretized on a 96 × 96 × 1 grid on eight processors

Process	NSIZE = 1			NSIZE = 24			NSIZE = 1152		
	Time	Part	Speed-up	Time	Part	Speed-up	Time	Part	Speed-up
ADV	851	11.1%	20.7	893	13.2%	18.0	860	11.4%	19.5
CHEM	4186	54.4%	8.9	2936	43.4%	6.8	4362	57.6%	8.9
COMM	791	10.4%	–	1110	16.4%	–	452	6.0%	–
TOTAL	7625	100.0%	8.2	6766	100.0%	7.3	7577	100.0%	8.2

Table 4
Running DEM discretized on a 96 × 96 × 10 grid on one processor

Process	NSIZE = 1		NSIZE = 24		NSIZE = 1152	
	Time	Part	Time	Part	Time	Part
ADV	169776	31.5%	159450	37.8%	169865	30.9%
CHEM	337791	62.7%	233471	55.3%	348769	63.4%
VERT	23221	4.3%	21473	5.1%	23014	4.2%
COMM	2	0.0%	2	0.0%	2	0.0%
TOTAL	538953	100.0%	421763	100.0%	549835	100.0%

4.2.2. Running the 3-D MPI version discretized on the coarse grid

Results from the six runs with this code are shown in Table 4 (runs on one processor performed by using three values of NSIZE) and Table 5 (runs on eight processors performed again with three values of NSIZE).

4.2.3. Running the 2-D MPI version discretized on the fine grid

Results from the six runs with this code are shown in Table 6 (runs on one processor performed by using three values of NSIZE) and Table 7 (runs on eight processors performed again with three values of NSIZE).

4.2.4. Running the 3-D MPI version discretized on the fine grid

Results from the six runs with this code are shown in Table 8 (runs on one processor performed by using three values of NSIZE) and Table 9 (runs on eight processors performed again with three values of NSIZE).

Table 5
Running DEM discretized on a 96 × 96 × 10 grid on eight processors

Process	NSIZE = 1			NSIZE = 24			NSIZE = 1152		
	Time	Part	Speed-up	Time	Part	Speed-up	Time	Part	Speed-up
ADV	18968	27.4%	9.0	18498	33.3%	8.6	18641	26.3%	9.1
CHEM	41334	59.6%	8.2	29189	52.3%	8.0	43291	61.3%	8.1
VERT	1213	1.7%	19.1	1200	2.2%	17.9	1240	1.8%	18.6
COMM	911	1.3%	–	878	1.6%	–	973	1.4%	–
TOTAL	69325	100.0%	7.8	55723	100.0%	7.6	70653	100.0%	7.8

Table 6
Running DEM discretized on a 480 × 480 × 1 grid on one processor

Process	NSIZE = 1		NSIZE = 24		NSIZE = 28800	
	Time	Part	Time	Part	Time	Part
ADV	485062	63.9%	484923	70.3%	491704	41.7%
CHEM	224804	29.1%	143923	20.9%	611502	51.8%
COMM	1	0.0%	1	0.0%	2	0.0%
TOTAL	771261	100.0%	690027	100.0%	1179518	100.0%

Table 7
Running DEM discretized on a 480 × 480 × 1 grid on eight processors

Process	NSIZE = 1			NSIZE = 24			NSIZE = 28800		
	Time	Part	Speed-up	Time	Part	Speed-up	Time	Part	Speed-up
ADV	34499	45.5%	14.1	34567	48.9%	14.0	33589	26.8%	14.6
CHEM	27159	35.8%	8.3	18816	26.6%	7.6	69168	55.2%	8.4
COMM	5937	7.8%	–	8128	11.5%	–	14474	11.6%	–
TOTAL	75854	100.0%	10.2	70856	100.0%	9.7	125246	100.0%	9.4

Table 8
Running DEM discretized on a 480 × 480 × 10 grid on one processor

Process	NSIZE = 1		NSIZE = 24		NSIZE = 28800	
	Time	Part	Time	Part	Time	Part
ADV	261631	67.0%	271419	72.9%	268337	49.8%
CHEM	86317	22.1%	56797	15.3%	228216	42.3%
VERT	40721	10.4%	42320	11.4%	41223	7.6%
COMM	1	0.0%	1	0.0%	1	0.0%
TOTAL	390209	100.0%	372173	100.0%	539319	100.0%

Table 9
Running DEM discretized on a 480 × 480 × 10 grid on eight processors

Process	NSIZE = 1			NSIZE = 24			NSIZE = 28800		
	Time	Part	Speed-up	Time	Part	Speed-up	Time	Part	Speed-up
ADV	13606	46.2%	19.2	13515	52.7%	20.1	13374	28.9%	20.1
CHEM	10398	35.3%	8.3	6681	26.0%	8.5	25888	56.0%	8.8
VERT	2830	9.6%	14.4	2802	10.9%	15.1	2709	5.9%	15.2
COMM	2316	7.9%	–	2340	9.1%	–	3925	8.5%	–
TOTAL	29449	100.0%	13.3	25654	100.0%	14.5	46210	100.0%	11.7

Table 10
Running DEM discretized on a 96 × 96 × 10 grid on 16 processors

Process	Time	Part	Speed-up-8	Speed-up-1
ADV	8044	27.4%	2.3	19.8
CHEM	14261	48.5%	2.1	16.4
VERT	388	1.3%	3.1	55.3
COMM	4203	14.3%	–	–
TOTAL	29389	100.0%	1.9	14.6

The Speed-up-8 factors are calculated as ratios of the computing times obtained when eight processors are used (which are given in Table 5) and the computing times when 16 processors are used. The Speed-up-1 factors are calculated as ratios of the computing times obtained when 1 processor is used (which are given in Table 4) and the computing times when 16 processors are used.

4.2.5. Major conclusions from the runs

It is seen that the exploitation of the cache memory is always giving good results (compare the results for NSIZE = 24 with the results for NSIZE = 1 and NSIZE = 1152 (28, 800)). The speed-ups for the physical processes are super-linear (greater for ADV and VERT than for CHEM, which should be expected, because chunks are used in the chemical parts). The speed-ups for the total computing time are lower, but anyway at least close to linear.

4.3. Scaling results for the MPI versions

It has been shown in the previous section that the computing times are reduced by a factor close to 8 (and in many cases by a factor greater than 8) when the number of the processors used is increased from 1 to 8. It is desirable that the same tendency holds when the number of processors is greater than 8 (i.e. it is desirable that increasing the number of processors used by a factor of k will results in decreasing the computing times by a factor approximately equal to k). It is often said that the parallel algorithm scales well when such a trend can be obtained.

Several runs were performed on 16 processors and the results were compared with those obtained on eight processors. Some results, which are obtained when the 3-D version of DEM are run, are given in Table 10 for the coarse grid version. Super-linear speed-ups were registered for the main physical processes, while nearly linear speed-ups were found for the total computing times.

With a special permission from the Danish Centre for Scientific Computing, several runs were performed by using up to 60processors. The 3-D refined version, where high efficiency is most desirable, was used in this runs. The results are given in Table 11. Comparing the results in Tables 10 and 11, it is seen that

Table 11
Running DEM discretized on a 480 × 480 × 10 on different numbers of processors

Processors	Time	Speed-up
1	372173	–
15	12928	28.79
30	7165	51.94
60	4081	91.20

Table 12
Running DEM discretized on a 480 × 480 × 1 grid on eight processors by using the MPI version and the OpenMP version

Process	MPI version	OpenMP version
ADV	822291	1663812
CHEM	393158	596920
COMM	255785	–
TOTAL	1782752	2614983

The time period for these two runs was one year.

the fact that very long arrays are split into many much shorter arrays is leading to higher efficiency when the problem discretized on a 480 × 480 × 10 grid is treated. Indeed, the super-linear speed-up, which was observed for this problem in the transition from one to eight processors (see Table 4.9), is also seen in Table 11 for up to 60 processors.

The results presented in Tables 10 and 11 indicate that the parallel algorithms applied in DEM scale very well.

4.4. Comparing MPI versions with OpenMP versions

The Sun computers, which were used to calculate the results in Section 4.2 are shared memory machines. Therefore, one should expect the OpenMP versions of the code to be more efficient than the MPI versions. In fact, the MPI versions are more efficient. In the previous section it was explained why this should be expected (the arrays used in connections with the sub-domains are much smaller, which leads to a better utilization of the cache memory of the computer). Some results are given in Table 12 in order to illustrate the fact that the leading dimension of of arrays is reduced when the MPI versions are used results also in reduction of the computing times.

The question: *is it possible to increase the efficiency of the OpenMP version?* is interesting. Some preliminary results obtained by Mohammed Abdi (a student from the Computer Science Department of the University of Reading who visited the National Environmental Research institute of Denmark in connection with his MSc thesis) indicate that this could be done. The data was divided into sub-domains (the number of sub-domains being equal to the number of processors). Then loops over the sub-domains are carried out in parallel by using OpenMP directives (mainly the directive *parallel do*). However, in this way we are trying to use, in a manual way, the same approach as in MPI. It is clear that this means that one of the major advantages of the OpenMP technique (easy programming by inserting only directives at appropriate places) is lost. Nevertheless, some savings can be achieved because (i) no MPI subroutines are called and (ii) the communications can be performed (perhaps in parallel) by using simple FORTRAN loops. The experiments in this direction are continued.

4.5. Plans for further improvements of the performance

The improvement of the fine resolution versions of DEM, especially the 3-D fine resolution version, is an important task which must be resolved in the near future. It is necessary both to improve the performance of the different versions of the model and to have access to more processors (and/or to more powerful computers) in order to be able to run operationally fine resolution versions of DEM.

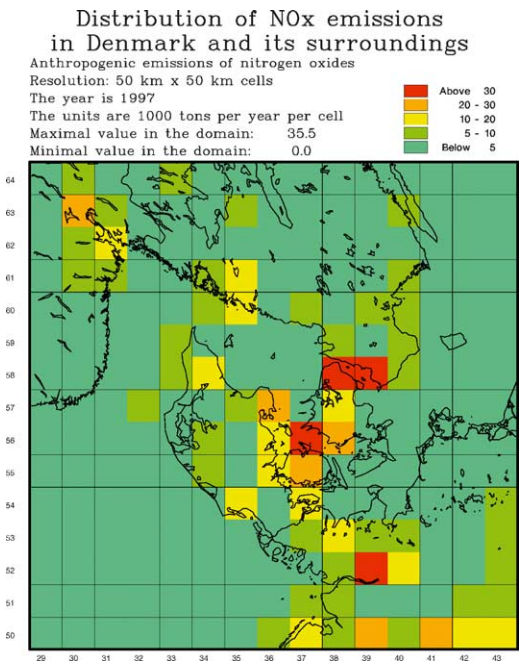

Fig. 1. Danish NO_x emissions in 1997.

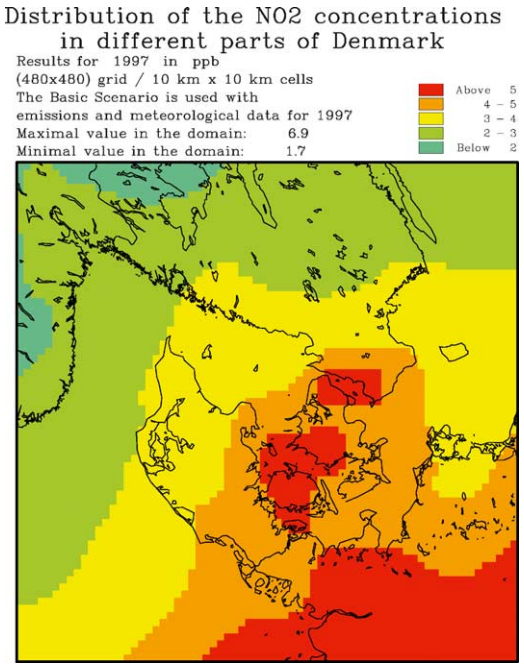

Fig. 3. NO_2 pollution in Denmark—fine resolution.

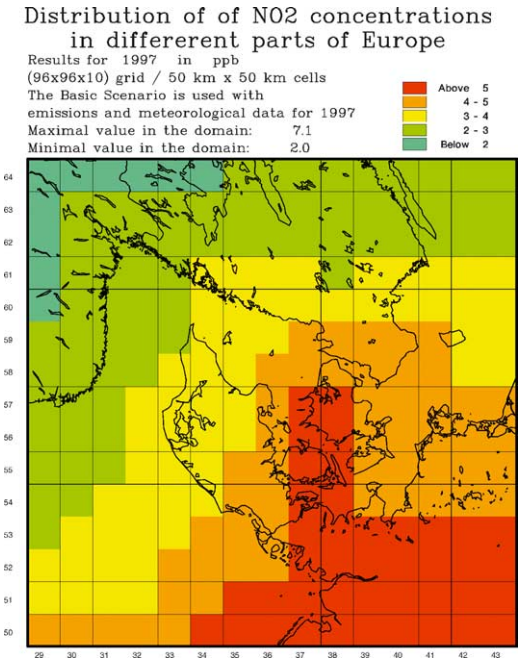

Fig. 2. NO_2 pollution in Europe—fine resolution.

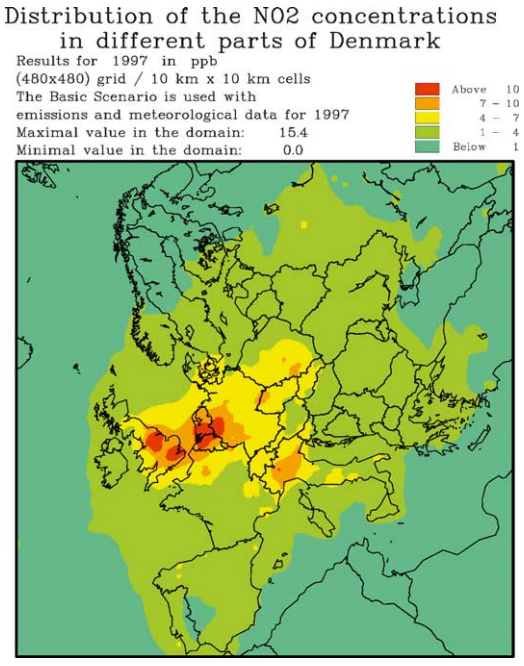

Fig. 4. NO_2 pollution in Europe—fine resolution.

5. Some practical applications of DEM

Some results obtained by running DEM with meteorological and emission data for 1997 will be presented in this section. These results will demonstrate the usefulness of using fine resolution version of DEM.

The comparison of the concentration levels that are calculated by the model with the input levels of the emissions used is important. For species like SO_2, NO_2 and NH_3 the calculated by the model pollution levels should reflect the pattern of the emissions used.

We choose to make some comparisons for NO_2concentrations in an area containing Denmark. The pattern of the corresponding NO_x emissions is seen in Fig. 1. It is seen that the largest emissions are in the regions of the three largest Danish cities (Copenhagen, Århus and Odense). This is not a surprise, because the traffic in cities is one of the major sources for the NO_x emissions.

The calculated by the coarse resolution version of the model pattern for the NO_2 concentrations is shown in Fig. 2. It is immediately seen that concentrations are smoothed very much when the coarse grid is used (and the pattern calculated by the model is not very similar to the input pattern of the related emissions).

The use of the fine resolution version of DEM calculates a pattern of the NO_2 concentrations which is clearly closer to the pattern of the NO_x emissions. This can be seen by comparing the highest concentration levels in Fig. 3 with the highest emission levels in Fig. 1.

The distribution of the NO_2 concentrations in the whole model space domain are shown in Fig. 4. (note that the scale used in Fig. 4 is different from the scale used in Figs. 2 and 3). It is seen that Denmark is located between highly polluted regions in Central and Western Europe and regions in Scandinavia, which are not very polluted.

It should be mentioned here that the results in Figs. 2 and 3 are obtained by zooming in Fig. 4 to the region containing Denmark (and, as already mentioned, by changing the scale). Zooming might be used to get more details for the distribution of the NO_2 concentrations (or the concentrations of any other of the studied by the model chemical species) in any sub-domain of the space domain of DEM.

The results presented in Fig. 3 indicate that the fine resolution version is producing results which are qualitatively better than the results produced by the coarse resolution version. Quantitative validation of the models results can be obtained by comparing concentrations calculated by the model with measurements. Such comparisons were carried out in [2,3,15,42,43] for the coarse resolution version. It is still very hard to carry out such extensive studies by using the fine resolution versions, but the results presented in this paper indicate that this will become possible in the near future.

Acknowledgements

A grant (CPU-1101-17) from the Danish Centre for Scientific Computing (DCSC) gave us access to the Sun computers at the Technical University of Denmark. The members of the staff of DCSC helped us to resolve some difficult problems related to the efficient exploitation of the grid of Sun computers. We improved the presentation of the results by following the constructive remarks of an unknown referee. We should like to thank the referee very much.

References

[1] V. Alexandrov, A. Sameh, Y. Siddique, Z. Zlatev, Numerical integration of chemical ODE problems arising in air pollution models, Environ. Modell. Assess. 2 (1997) 365–377.
[2] C. Ambelas Skjøth, A. Bastrup-Birk, J. Brandt, Z. Zlatev, Studying variations of pollution levels in a given region of Europe during a long time-period, Syst. Anal. Modell. Simul. 37 (2000) 297–311.
[3] A. Bastrup-Birk, J. Brandt, I. Uria, Z. Zlatev, Studying cumulative ozone exposures in Europe during a seven-year period, J. Geophys. Res. 102 (1997) 23917–23935.
[4] A. Bott, A positive definite advection scheme obtained by non-linear renormalization of the advective fluxes, Monthly Weather Rev. 117 (1989) 1006–1015.
[5] K. Brenan, S. Campbell, L. Petzold, Numerical Solution of Initial Value Problems in Differential-Algebraic Equations, SIAM, Philadelphia, 1996.
[6] R.A. Brost, The sensitivity to input parameters of atmospheric concentrations simulated by a regional chemical model, J. Geophys. Res. 93 (1988) 2371–2387.
[7] W.P. Crowley, Numerical advection experiments, Monthly Weather Rev. 96 (1968) 1–11.
[8] P. Deuflhard, Recent progress in extrapolation methods for ordinary differential equations, SIAM Rev. 27 (1985) 505–535.
[9] I. Dimov, I. Farago, A. Havasi, Z. Zlatev, L-Commutativity of the operators in splitting methods for air pollution models, Annal. Univ. Sci. Budapest 44 (2001) 129–150.
[10] I. Farago, Á. Havasi, On the convergence and local splitting error of different splitting schemes, internal report, Eötvös Loránd University, Budapest.
[11] R. Djouad, B. Sportisse, Solving reduced chemical models in air pollution modelling, Appl. Numer. Math. 40 (2003) 49–61.
[12] K. Georgiev, Z. Zlatev, Parallel Sparse Matrix Algorithms for Air Pollution Models, Parallel Distrib. Comput. Pract. 2 (1999), 429–442.
[13] W. Gropp, E. Lusk, A. Skjellum, Using MPI: Portable Programming with the Message Passing Interface, MIT Press, Cambridge, MA, 1994.
[14] M.W. Gery, G.Z. Whitten, J.P. Killus, M.C. Dodge, A photochemical kinetics mechanism for urban and regional computer modeling, J. Geophys. Res. 94 (1989) 12925–12956.
[15] A. Havasi, Z. Zlatev, Trends of Hungarian air pollution levels on a long time-scale, Atmos. Environ. 36 (2002) 4145–4156.
[16] E. Hesstvedt, I.A. Isaksen, Quasi-steady-state approximations in air pollution modelling: comparison of two numerical schemes for oxidant prediction, Int. J. Chem. Kinet. 10 (1978) 971–994.
[17] Z. Zlatev, R. Berkowicz, A. Eliassen, L.P. Prahm, Comparison of numerical techniques for use in air pollution models with non-linear chemical reactions, Atmos. Environ. 23 (1988) 967–983.
[18] W. Hunsdorfer, B. Koren, M. van Loon, J.G. Verwer, A positive finite difference advection scheme, J. Comput. Phys. 117 (1995) 35–46.
[19] W. Hunsdorfer, J.G. Verwer, Numerical Solution of Time-Dependent Advection–Diffusion-Reaction Equations, Springer, Berlin, 2003.
[20] J.D. Lambert, Numerical Methods for Ordinary Differential Equations, Wiley, New York (1991).
[21] D. Lancer, J.G. Verwer, Analysis of operators splitting for advection–diffusion-reaction problems in air pollution modelling, J. Comput. Appl. Math. 111 (1999) 201–216.
[22] M. van Loon, Testing interpolation and filtering techniques in connection with a semi-Lagrangian method, Atmos. Environ. 27A (1993) 2351–2364.
[23] G.I. Marchuk, Mathematical Modeling for the Problem of the Environment, Studies in Mathematics and Applications, No. 16, North-Holland, Amsterdam, 1985.
[24] G.J. McRae, W.R. Goodin, J.H. Seinfeld, Numerical solution of the atmospheric diffusion equations for chemically reacting flows, J. Comput. Phys. 45 (1984) 1–42.
[25] C.R. Molenkampf, Accuracy of finite-difference methods applied to the advection equation, J. Appl. Meteorol. 7 (1968) 160–167.
[26] W. Owczarz, Z. Zlatev, Running a large air pollution model on an IBM SMP computer, Int. J. Comput. Res. 10 (4) (2001) 321–330.
[27] W. Owczarz, Z. Zlatev, Parallel matrix computations in air pollution modelling, Parallel Comput. 28 (2002) 355–368.
[28] D.W. Pepper, A.J. Baker, A simple one-dimensional finite element algorithm with multidimensional capabilities, Numer. Heath Transfer 3 (1979) 81–95.

[29] D.W. Pepper, C.D. Kern, P.E. Long Jr., Modelling the dispersion of atmospheric pollution using cubic splines and chapeau functions, Atmos. Environ. 13 (1979) 223–237.
[30] L.F. Shampine, M.W. Reichelt, J.A. Kierzenka, Solving Index-1 DAEs in MATLAB and Simulink, SIAM Rev. 41 (1999) 538–552.
[31] 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 (1996) 347–352.
[32] J.G. Verwer, D. Simpson, Explicit methods for stiff ODE's from atmospheric chemistry, Appl. Numer. Math. 18 (1995) 413–430.
[33] WEB-site for OPEN MP tools, http://www.openmp.org, 1999.
[34] WEB-site of the Danish Centre for Scientific Computing at the Technical University of Denmark, Sun High Performance Computing Systems, http://www.hpc.dtu.dk, 2002.
[35] Z. Zlatev, Application of predictor-corrector schemes with several correctors in solving air pollution problems, BIT 24 (1984) 700–715.
[36] Z. Zlatev, Computer Treatment of Large Air Pollution Models, Kluwer Academic Publishers, Dordrecht, Boston, London, 1995.
[37] Z. Zlatev, Partitioning ODE systems with an application to air pollution models, Comput. Math. Appl. 42 (2001) 817-832.
[38] Z. Zlatev, Massive data set issues in air pollution modelling, in: J. Abello, P.M. Pardalos, M.G.C. Resende (Eds.), Handbook on Massive Data Sets, Kluwer Academic Publishers, Dordrecht, Boston, London, 2002, pp. 1169–1220.
[39] Z. Zlatev, J. Christensen, A. Eliassen, Studying high ozone concentrations by using the Danish Eulerian Model, Atmos. Environ. 27A (1993) 845–865.
[40] Z. Zlatev, J. Christensen, An Eulerian air pollution model for Europe with nonlinear chemistry, J. Atmos. Chem. 15 (1992) 1–37.
[41] Z. Zlatev, I. Dimov, K. Georgiev, Studying long-range transport of air pollutants, Comput. Sci. Eng. 1 (3) (1994) 45–52.
[42] Z. Zlatev, I. Dimov, Tz. Ostromsky, G. Geernaert, I. Tzvetanov, A. Bastrup-Birk, Calculating losses of crops in Denmark caused by high ozone levels, Environ. Modell. Assess. 6 (2001) 35–55.
[43] Z. Zlatev, J. Fenger, L. Mortensen, Relationships between emission sources and excess ozone concentrations, Comput. Math. Appl. 32 (11) (1996) 101–123.