M a t h e m a t i c a
B a l k a n i c a

New Series Vol. 20, 2006, Fasc. 1

Computer Treatment of Partial Differential Equations Arising in Environmental Modelling

Zahari Zlatev

Presented at Conference on "PDE Methods in Appl. Math. and Image Proc."

Large-scale air pollution models can successfully be used in different studies. These models are described mathematically by systems of partial differential equations (PDEs). 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.

1. PDEs arising in environmental modelling

Many air pollution models are described mathematically with a system of partial differential equations (PDEs) of the following type:

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

The number q of equations in (1) is equal to the number of chemical species. The other quantities in (1) are (i) the concentrations c_s of the chemical species (ii) the wind velocities u, v and w, (iii) the diffusion coefficients K_x, K_y and K_z, (iv) the emission sources E_s, (v) the deposition coefficients κ_{1s} and κ_{2s} and (vi) the chemical reactions $Q_s(c_1, c_2, \ldots, c_q)$. A particular model, the Unified Danish Eulerian Model (UNI-DEM), will consistently be used in this paper in order to facilitate the exposition of the results. However, the main ideas are applicable to many environmental models (and also for many models arising in other fields of science and engineering).

The space domain of UNI-DEM is a 4800 $km \times 4800$ km square containing Europe and parts of Africa, Asia, the Arctic area and the Atlantic Ocean. Initial and boundary conditions are read from a file (being either prepared during another run of UNI-DEM or by running a hemi-spherical model). Initial and boundary conditions are discussed in [7], [21], [51], [53], [54] and [55].

Only basic ideas used when the system of PDEs (1) is handled on computers will be discussed in this paper (full description of the methods used in the numerical treatment of such models can be found in [51], [53]). Applications are discussed in [3], [4], [21], [57] and [58].

2. Need for splitting procedures

It is difficult to treat the system of PDEs (1) directly. This is the reason for using different kinds of splitting. A splitting procedure, which is based on ideas proposed in [31] and [32] and which leads to five sub-models, has been applied in [51] and used in many studies (as, for example, in [53]). Each of the five sub-models obtained by this splitting procedure is representing one of the major physical and chemical processes described mathematically in (1); i.e. (i) the horizontal advection, (ii) the horizontal diffusion, (iii) the chemistry (together with the emission terms), (iv) the deposition and (v) the vertical exchange.

In the newest version of UNI-DEM the number of sub-models is reduced from five to three by merging the horizontal advection with the horizontal diffusion and the chemical sub-model with the deposition sub-model:

$$\frac{\partial c_s^{(1)}}{\partial t} = -\frac{\partial (w c_s^{(3)})}{\partial z} + \frac{\partial}{\partial z}\left(K_z \frac{\partial c_s^{(3)}}{\partial z}\right) \tag{2}$$

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

$$(4) \quad \frac{dc_s^{(3)}}{dt} = E_s + Q_s(c_1^{(3)}, c_2^{(3)}, \ldots, c_q^{(3)}) - (\kappa_{1s} + \kappa_{2s})c_s^{(3)}$$

The first of these sub-models, (2), describes the vertical exchange. The second sub-model, (3), combines the horizontal transport (the advection) and the horizontal diffusion. The last sub-model, (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 (2) - (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 (2), where the computations are carried out along the vertical grid-lines.
- The lateral boundary conditions are handled in (3), where the the computations are carried out in each of the horizontal grid-planes.
- The computations related to (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 (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 [31], [32] and [51]; see also [56]. A thorough discussion of different types of splitting splitting procedure can be found in the book ([25]), which was recently published by Hundsdorfer and Verwer. Some convergence results are presented in Farago and Havasi ([15]).

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. Attempts to evaluate the splitting errors were recently carried out; see [28], [10] and [11].

3. 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

(i) the number of chemical species involved in the model is $q = N_s$ and (ii) the spatial derivatives in (2) are discretized by some numerical algorithm. Then the system of PDEs (2) is transformed into a system of ODEs (ordinary differential equations):

$$\frac{dg^{(1)}}{dt} = f^{(1)}(t, g^{(1)}), \tag{5}$$

The system of PDEs (3) is transformed into the following system of ODEs when the spatial derivatives in the right-hand-side of (3) are discretized:

$$\frac{dg^{(2)}}{dt} = f^{(2)}(t, g^{(2)}), \tag{6}$$

There are in fact no spatial derivatives in the right-hand-side of (4), because the non-linear functions Q_s can be represented as

$$Q_s(c_1, c_2, \ldots, c_q) = -\sum_{i=1}^{q} \alpha_{si} c_i + \sum_{i=1}^{q} \sum_{j=1}^{q} \beta_{sij} c_i c_j, \qquad s = 1, 2, ..., q. \tag{7}$$

where α_{si} and β_{sij} are coefficients describing the rates of the chemical reactions (for the CBM IV scheme, [20], these coefficients are listed in [51]). By using this observation, it is easy to represent (4) as a system of ODEs:

$$\frac{dg^{(3)}}{dt} = f^{(3)}(t, g^{(3)}), \tag{8}$$

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 $f^{(i)}(t, g) \in R^{N_x \times N_y \times N_z \times N_s}$, $i = 1, 2, 3$, depend on the numerical method used to discretize the spatial derivatives.

A **finite element method** is used to discretize the spatial derivatives in (2) and (3). This method is described in [36] and [37]. Its implementation in UNI-DEM is discussed in [18]. Other numerical methods can also be used:

- **Pseudo-spectral discretization** ([51]). The pseudo-spectral discretization is based on expanding the unknown function in Fourier series and truncating this series after some finite number of terms. This method requires periodicity on the boundaries, which causes problems, because in many realistic situations this condition is not satisfied. Special techniques are to be used in order to satisfy this requirement (see [17]). The

pseudo-spectral method is able to produce very accurate results when the requirement for periodicity on the boundaries is satisfied; [16], [26], [51].

- **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 [30]. This method is well-know under the name method of characteristics among the mathematicians (see, for example, [40]).

- **Methods producing non-negative values of the concentrations.** The method proposed in [5] is often used in air pollution modelling. The method from [24] is based on a solid theoretical foundation. A thorough description of the principle used in all methods of this type is given in the monograph of Hundsdorfer and Verwer [25].

As mentioned above, there are no spatial derivatives in (4), which means that the system of ODEs (8) is trivially obtained by (4).

Much more details about the methods, which can be used in the space discretization, can be found in [51] and [25].

4. Time integration

It is necessary to couple the three ODE systems (5), (6) and (8). The coupling procedure is connected with the time-integration of these systems. Assume that the values of the concentrations have been found for $t = t_n$. These values can be considered as components of $g(t_n) \in R^{N_x \times N_y \times N_z \times N_s}$. 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 (5). The solution of (5) is used as an initial condition of (6). The solution of (6) is used as an initial condition of (8). The solution of the last system (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, (5), can be solved by using many classical time-integration methods. The so-called θ-method (see, for example, [27]) is currently used in UNI-DEM. The choice of numerical method is not very critical in this part, because as it will be shown Section 8, it is normally not very expensive.

Predictor-corrector methods with several different correctors are used in the solution of the ODE system (6). The correctors are carefully chosen so that the stability properties of the method are enhanced; see [48]. The reliability of

the algorithms used in the advection part was verified by using the well-known rotational test proposed simultaneously in 1968 in [8] and [33].

The solution of (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, [22] or [23]) 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 (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:

$$\frac{dg_1}{dt} = f_1(t, g_1, g_2), \tag{9}$$

$$0 = f_2(t, g_1, g_2). \tag{10}$$

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 [6]). 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 [39]. A method based on the solution of DAE for air pollution models was recently proposed in [13].

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; [27]. On the other hand, these methods can be implemented with an error control and perhaps with larger time-steps. The extrapolation methods, [9], 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, (5), (6) and (8), are to be treated successively.

Partitioning can also be used ([2]). Some convergence problems related to the implementation of partitioning are studied in [52].

It is important to apply good algorithms for treating general sparse matrices when (8) is solved. The methods discussed in [46], [47], [49] and [50] can successfully be used. Methods for sparse matrices developed especially for atmospheric chemistry sub-models are discussed in [2].

The experiments with different integration methods for the chemical sub-model are continuing. The QSSA with some enhancements based on ideas from [41] and [42] will be used here. The method is described in [2]. There are still very open questions related to the choice of method for the chemical part. The choice of the improved QSSA was made to get well-balanced parallel tasks.

5. Need for high performance computing

Many tasks, which several years ago had to be handled on powerful super-computers, can be handled at present on PCs or work-stations. There are still many tasks that can only be run on parallel computers. It will be shown in this section that large air pollution models are to be treated on parallel computers.

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

Only the two systems of ODEs (6) and (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 (6) and (8) is equal to the product of the grid points (9216) and the number of chemical species (35), i.e. 322560 equations have to be treated at each time-step when any of the systems (6) and (8) is handled. The time-stepsize used in the transport sub-model (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 UNI-DEM covers a period of one year (very often a period of extra five days is needed to start up the model). This means that 35520 time-steps are needed in the transport sub-model, while six times more time-steps, 213120 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 climate changes on the air pollution studies are 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 (6) and (8) is increased by a factor of 25 (compared with the previous case). This means that 8064000 equations are to be treated at each time-step when any of the systems (6) and (8) is handled. The time-stepsize remains 150 s when the chemical part is treated. The time-stepsize has to be reduced from 900 s to 150 s in the transport part. A typical run (one year + 5 days to start up the model) will require 213520 time-steps for each of the systems (6) and (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. This version of the model **must** be treated on powerful parallel architectures.

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

All three sub-models, (5), (6) and (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 (6) and (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 (5) is extra, but this part of the total computational work is much smaller than the parts needed to treat (6) and (8); see Section 8.

It is preferable to run the 3-D version on high-speed parallel computers even when the coarse grid is used. The runs are very heavy when the 3-D version is to be run on a fine grid. More powerful parallel computers are needed if meaningful studies with the 3-D version of UNI-DEM discretized on a fine grid are to be carried out.

6. Exploiting the cache memory of the computer

In the modern computers the time needed for performing arithmetic operations is reduced dramatically (compared with computers available 10-15 years ago). 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 performed and the time needed to store the results back in the memory are much smaller. This is why most of the computers have different caches. It is much more efficient to use data which is in cache than to make references to the memory. 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. The rules from [12], [34] and [35] are outlined below. These rules are performing rather well when Sun parallel computers are used.

Consider the 2-D versions of UNI-DEM. Assume that the concentrations are stored in 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). Cache problems may appear when the fine grid is used, because the arrays are very long.

Great problems appear in the chemical part, because when the concentration of a given species in a given row is modified, other species in the same row are needed in the computations. This 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 vector machines (like CRAY); see [12]. However, if cache memory is available, then the computations, will become 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 I'th 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
```

In the second code, both the operations that are performed in the beginning and in the end of the first loop 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. This additional benefit of using chunks was quite unexpected.

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 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 $CONS(N_x \times N_y, N_s, N_z)$. 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 (6) and (8) are treated.

The use of chunks leads to considerable savings in computing time.

7. Achieving parallelism

It was explained in Section 5 that the discretization of an air pollution model is resulting in huge computational tasks. Therefore, it is important to prepare parallel codes which run efficiently on modern parallel computers.

7.1. Basic principles used in 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 tools, OpenMP ([44]) and MPI ([19]), are used in UNI-DEM.

7.2. Development of OpenMP versions of UNI-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.

- **Parallel tasks in the transport sub-model.** This sub-model, which is described by (6), consists 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. There are $q \times N_z$ parallel tasks. Each task is a system of $N_x \times N_y$ ODEs. There are 32 independent chemical species in UNI-DEM. N_z is equal to 1 in the 2-D case and to 10 in the 3-D case. The 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 is perfect 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.

- **Parallel tasks in the chemical sub-model.** This sub-model, which is described by (8), consists 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 tasks are very small (each task is a system of q ODEs). It is necessary to group them in clusters. 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 previous section).

- **Parallel tasks in the vertical exchange sub-model.** This sub-model, which is described by (5), consists 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 O_x and O_y. $N_s = q$ is the number of chemical species. The number of parallel tasks is very large (8064000 when the $480 \times 480 \times 10$ grid is used with 35 chemical species) and the loading balance is perfect. However, the tasks are small (each parallel task is a system of N_z ODEs). It is necessary to group the tasks in an appropriate way. A very long array (its leading dimension being $N_x \times N_y \times N_s$) has to handled by rows. It is desirable to use chunks in order to avoid a large number of cache misses. No chunks are used at present, but there are plans to introduce chunks in the near future.

It is easy to achieve parallelism when OpenMP is used; the parallel runs depend on the splitting, but not very much on the numerical methods.

7.3. Development of MPI versions of UNI-DEM

The approach used with MPI tools is based in dividing the space domain into p sub-domains, where p is the number of processors. Two modules are needed: (i) a pre-processing module and (ii) a post-processing module.

- **The pre-processing module.** The input data are divided into p portions corresponding to the p sub-domains. In this way, each processor will work during the whole computational process with its own set of input data.

- **The post-processing module.** Each processor prepares its own set of output data. During the post-processing the p sets of output data are collected and common output files are prepared.

- **Benefits of using the two modules.** Excessive communications during the computational process are avoided when the two modules are used. Some communications along the inner boundaries of the sub-domains are still needed. These communications are to be carried only once per step and only a few data are to be communicated. Thus, the actual communications are rather cheap when the pre-processing and the post-processing modules are properly implemented.

The introduction of p sub-domains leads to a reduction of the main arrays by a factor of p. Consider 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 $(N_x \times N_y/p, N_s)$. This is equivalent to the use of p chunks. Chunks of length $N_x \times N_y/p$ are still very large. Chunks have to be used in each sub-domain. 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 6. 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. The MPI versions of UNI-DEM are often performing better than the corresponding OpenMP versions (see Section 8).

8. Numerical results

Some results will be presented 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).

8.1. Description of the grid of Sun computers

Sun computers at DCSC (the Danish Centre for Scientific Computing) were used in the runs. The computers and the their characteristics are shown in Table 1 (see [45]). All computers were connected with a 1Gbit/s Switch. They are united in a grid (216 processors) so that a job sent without a special

Table 1: The computers available at the Sun grid.

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

demand will be assigned on the computer on which there are sufficiently many free processors. Only "Newton" was used in the runs.

8.2. Running the MPI versions of UNI-DEM

Four MPI versions of UNI-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. Runs with three different sizes of chunks were carried out: (a) $NSIZE = 1$, (b) $NSIZE = 24$ and (c) the maximal size of the chunks when 8 processors are used, which is $NSIZE = 1152$ for the coarse grid and $NSIZE = 28800$ for the fine grid. In most of the cases both 1 processor and 8 processors were used. Some of the jobs were also run on more than 8 processors.

All runs of the versions discretized on the coarse grid were carried out for the typical period of one year (in which case it is possible to study seasonal variations). The 2-D version of UNI-DEM discretized on the fine grid was run over a period of one month. Finally, the 3-D version of UNI-DEM discretized on the fine grid was run over a time-period of 42 hours. 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 chemical reactions, the treatment of the emissions and the deposition part,
- COMM stands for communications along the inner boundaries,
- VERT stands for the vertical exchange processes

Table 2: Running UNI-DEM on a $96 \times 96 \times 1$ grid on one processor.

	$NSIZE = 1$		$NSIZE = 24$		$NSIZE = 1152$	
Process	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%

- TOTAL stands for the total computing time (ADV + CHEM + COMM + VERT + 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 used).

Running the 2-D MPI version discretized on the coarse grid. Results from runs performed by using three values of $NSIZE$ are shown in Table 2 (runs on one processor) and Table 3 (runs on 8 processors).

Running the 3-D MPI version on the coarse grid. Results from runs performed by using three values of $NSIZE$ are shown in Table 4 (runs on one processor) and Table 5 (runs on 8 processors).

Running the 2-D MPI version on the fine grid. Results from runs performed by using three values of $NSIZE$ are shown in Table 6 (runs on one processor) and Table 7 (runs on 8 processors).

Running the 3-D MPI version on the fine grid. Results from runs performed by using three values of $NSIZE$ are shown in Table 2 (runs on one processor) and Table 3 (runs on 8 processors).

Major conclusions from the runs. The exploitation of the cache memory is always giving good results; compare the results for $NSIZE = 24$ with the results for $NSIZE = 1$ and $NSIZE = 1152(28800)$. The speed-ups for the physical processes are super-linear (greater for ADV and VERT than for CHEM). The speed-ups for the total computing time are lower, but they are still close to linear.

8.3. Scaling results for the MPI versions

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

Table 3: Running UNI-DEM on a $96 \times 96 \times 1$ grid on eight processors.

	$NSIZE = 1$			$NSIZE = 24$			$NSIZE = 1152$		
Process	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 UNI-DEM on a $96 \times 96 \times 10$ grid on one processor.

	$NSIZE = 1$		$NSIZE = 24$		$NSIZE = 1152$	
Process	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%

Table 5: Running UNI-DEM on a $96 \times 96 \times 10$ grid on eight processors.

	$NSIZE = 1$			$NSIZE = 24$			$NSIZE = 1152$		
Process	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

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 8 processors. Some results, which are obtained when the 3-D version of UNI-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.

Several runs were performed by using up to 60 processors. The 3-D refined version, where high efficiency is most desirable, was used in this runs.

Table 6: Running UNI-DEM on a $480 \times 480 \times 1$ grid on one processor.

	$NSIZE = 1$		$NSIZE = 24$		$NSIZE = 28800$	
Process	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 UNI-DEM on a $480 \times 480 \times 1$ grid on eight processors.

	$NSIZE = 1$			$NSIZE = 24$			$NSIZE = 28800$		
Process	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 UNI-DEM on a $480 \times 480 \times 10$ grid on one processor.

	$NSIZE = 1$		$NSIZE = 24$		$NSIZE = 28800$	
Process	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%

The results are given in Table 11. Comparing the results in Table 10 and Table 11, it is seen that the fact that very long arrays are split into many much shorter arrays is leading to higher efficiency when the problem discretized on a $480 \times 480 \times 10$ 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 9), is also seen in Table 11 for up to 60 processors.

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

8.4. Comparing MPI versions with OpenMP versions

The Sun computers, which are used in this paper, are shared memory machines. Therefore, one should expect the OpenMP versions of the code to

Table 9: Running UNI-DEM on a $480 \times 480 \times 10$ grid on eight processors.

	$NSIZE = 1$			$NSIZE = 24$			$NSIZE = 28800$		
Process	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 UNI-DEM on a $96 \times 96 \times 10$ grid on 16 processors. The Speed-up-8 factors are calculated as ratios of the computing times obtained when 8 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.

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

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 reduction of the leading dimension of of arrays when the MPI versions are used results also in

Table 11: Running UNI-DEM on a $480 \times 480 \times 10$ grid 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 UNI-DEM on a $480 \times 480 \times 1$ grid on 8 processors by using the MPI version and the OpenMP version. The time period for these two runs was one year.

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

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 were 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.

9. Plans for further improvements of the performance

The improvement of the fine resolution versions of UNI-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 UNI-DEM.

Improvements of the performance of the model on high-speed computers are important because the computational work will be further increased by introducing additional options, such as (i) an option for treating particles (see [1] and [38]) and (ii) options for integrating measurements in the computational process by using variational data assimilation techniques (see [14], [29] and [43]).

Introduction of such additional options in UNI-DEM is urgently needed.

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.

This project was also supported by the NATO Scientific Programme (Grant No. 980505).

References

[1] I.Ackermann, H.Hass, B.Schell and F.S. Binkowski, *Regional modelling of particulate matter with MADE*, Environmental Management and Health, **10**, 1999, 201-208.

[2] 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**, 1997, 365–377.

[3] C. Ambelas Skjøth, A. Bastrup-Birk, J. Brandt and Z. Zlatev, *Studying variations of pollution levels in a given region of Europe during a long time-period*, Systems Analysis Modelling Simulation, **37**, 2000, 297-311.

[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**, 1997, 23917-23935.

[5] A. Bott, *A positive definite advection scheme obtained by non-linear renormalization of the advective fluxes*, Monthly Weather Review, **117**, 1989, 1006-1015.

[6] K. Brenan, S. Campbell and L. Petzold, *Numerical Solution of Initial Value Problems in Differential-Algebraic Equations*, SIAM, Philadelphia, 1996.

[7] R. A. Brost, *The sensitivity to input parameters of atmospheric concentrations simulated by a regional chemical model*, Journal of Geophysical Research, **93**, 1988, 2371-2387.

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

[9] P. Deuflhard, (1985). *Recent progress in extrapolation methods for ordinary differential equations*. SIAM Review, **27** , 1985, 505-535.

[10] I. Dimov, I. Faragó, Á. Havasi and Z. Zlatev, *L-Commutativity of the operators in splitting methods for air pollution models*, Annales Univ. Sci. Budapest, **44**, 2001, 129-150.

[11] I. Dimov, I. Faragó, Á. Havasi and Z. Zlatev, *Operator splitting and commutativity analysis in the Danish Eulerian Model*, Mathematics and Computers in Simulation, **67**, 2004, 217-233.

[12] I. Dimov, K. Georgiev, Tz. Ostromsky and Z. Zlatev, *Computational challenges in the numerical treatment of large scale air pollution models*, Ecological Modelling, **179**, 2004, 187-203.

[13] R. Djouad and B. Sportisse, *Solving reduced chemical models in air pollution modelling*, Applied Numerical Mathematics, **40**, 2003, 49-61.

[14] H. Elbern, H. Schmidt and A. Ebel. *Variational data assimilation for tropospheric chemistry modelling*, Journal of Geophysical Research, **104**, 1997, 15967-15985.

[15] I. Farago and Á. Havasi, *On the convergence and local splitting error of different splitting schemes*, Progress in Computational Fluid Dynamics, to appear.

[16] B. Fornberg, *On a Fourier Method for the integration of hyperbolic equations*, SIAM J. Numer. Anal., **12**, 1975, 509-528.

[17] B. Fornberg, *A Practical Guide to Pseudospectral Methods*, Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, Cambridge, 1996.

[18] K. Georgiev and Z. Zlatev, *Parallel Sparse Matrix Algorithms for Air Pollution Models*, Parallel and Distributed Computing Practices, **2**, 1999, 429-442.

[19] W. Gropp, E. Lusk and A. Skjellum, *Using MPI: Portable Programming with the Message Passing Interface*, MIT Press, Cambridge, Massachusetts (1994).

[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**, 1989, 12925–12956.

[21] A. Havasi and Z. Zlatev, *Trends of Hungarian air pollution levels on a long time-scale*, Atmospheric Environment, **36**, 2002, 4145-4156.

[22] 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**, 1978, 971–994.

[23] 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**, 1988, 967–983.

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

[25] W. Hundsdorfer and J. G. Verwer, *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*, Springer, Berlin, 2003

[26] H. O. Kreiss and J. Oliger, *Comparison of accurate methods for the integration of hyperbolic equations*, Tellus, **24**, 1972, 199-215.

[27] J. D. Lambert, *Numerical Methods for Ordinary Differential Equations.* Wiley, New York (1991).

[28] D. Lanser and J. G. Verwer, *Analysis of operators splitting for advection-diffusion-reaction problems in air pollution modelling*, J. Comput.Appl. Math., **111**, 1999, 201-216.

[29] Z. Li and I. M. Navon, *Optimality of 4D-Var and its relationship with the Kalman filter and the Kalman smoother*, Quarterly Journal of the Royal Meteorological Society, **127B**, 2001, 661-684.

[30] M. van Loon, *Testing interpolation and filtering techniques in connection with a semi-Lagrangian method*, Atmospheric Environment, **27A**, 1993, 2351-2364.

[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**, 1984, 1–42.

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

[34] W. Owczarz and Z. Zlatev, *Running a large air pollution model on an IBM SMP computer*, International Journal of Computer Research, **10**, 2001, No. 4, 321-330.

[35] W. Owczarz and Z. Zlatev, *Parallel matrix computations in air pollution modelling*, Parallel Computing, **28**, 2002, 355-368.

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

[37] 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**, 1979, 223–237.

[38] B. Schell, I. J. Ackermann, H. Hass, F. S. Binkowski and A. Ebel, *Modelling the formation of secondary organic aerosol within a comprehensive air quality model system*, Journal of Geophysical Research, **106**, 2001, 28275-28293.

[39] L. F. Shampine, M. W. Reichelt and J. A. Kierzenka, *Solving Index-1 DAEs in MATLAB and Simulink*. SIAM Rev., **41**, 1999, 538-552.

[40] G. D. Smith, *Numerical Solution of Partial Differential Equations: Finite Difference Methods*, Oxford Applied Mathematics and Computing Science Series, Clarendon Press, Oxford, 1978.

[41] J. G. Verwer and 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.

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

[43] K. Y. Wang, D. J. Lary, D. E. Shallcross, S. M. Hall and J. A. Pyle, *A review on the use of the adjoint method in four-dimensional atmospheric-chemistry data assimilation*, Quarterly Journal of the Royal Meteorological Society, **127B**, 2001, 2181-2204.

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

[45] 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.

[46] Z. Zlatev, *On some pivotal strategies in Gaussian elimination by sparse technique*, SIAM J. Numer. Anal., **17**, 1980, 18-30.

[47] Z. Zlatev, *Use of iterative refinement in the solution of sparse linear systems*, SIAM J. Numer. Anal., **19**, 1982, 381-399.

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

[49] Z. Zlatev, *Survey of the advances in exploiting the sparsity in the solution of large problems*, J. Comput. Appl. Math., **20**, 1987, 83-105.

[50] Z. Zlatev, *Computational Methods for General Sparse Matrices*, Kluwer Academic Publishers, Dordrecht-Boston-London (1991).

[51] Z. Zlatev, *Computer Treatment of Large Air Pollution Models*, Kluwer Academic Publishers, Dordrecht-Boston-London (1995).

[52] Z. Zlatev, *Partitioning ODE systems with an application to air pollution models*, Computers and Mathematics with Applications, **42**, 2001, 817-832.

[53] Z. Zlatev, *Massive data set issues in air pollution modelling*, In: Handbook on Massive Data Sets (J. Abello, P. M. Pardalos and M. G. C. Resende, eds.), pp. 1169-1220, Kluwer Academic Publishers, Dordrecht-Boston-London, 2002.

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

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

[56] Z. Zlatev, I. Dimov and K. Georgiev, *Studying long-range transport of air pollutants*, Computational Science and Engineering, **1**, 1994, No. 3, 45-52.

[57] 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 Modelling and Assessment, **6**, 2001, 35-55.

[58] Z. Zlatev, J. Fenger and L. Mortensen, *Relationships between emission sources and excess ozone concentrations*, Computers and Mathematics with Applications, **32**, 1996, No. 11, 101-123.

National Environmental Research Inst.
Frederiksborgvej 399 P.O.Box 358
DK-4000 Roskilde, Denmark
e-mail: zz@dmu.dk

Received 31.01.2005