INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2002; **38**:849–859 (DOI: 10.1002/fld.239)

Parallel solution of a coupled flow and transport model for shallow water†

B. P. Sommeijer* and P. J. van der Houwen

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

SUMMARY

We present an integrated approach for the concurrent solution of a 3D hydrodynamical model coupled with a 3D transport model. Since both models are quite similar in nature, the same numerical method has been employed. This leads to a code that is more efficient than when two existing codes would have been combined.

Discretization of the spatial differential operators, and the boundary conditions, results in a stiff initial value problem. To cope with the stiffness, we select an implicit time-integration formula, viz. the second-order, L-stable BDF method because of its excellent stability properties. To reduce the huge amount of linear algebra involved in solving the implicit relations, an Approximate Factorization technique has been used. Essentially, this technique replaces a 'multi-dimensional' system by a series of 'one-dimensional' systems.

Since the output of the hydrodynamical model (i.e., the flow field) serves as input for the transport model, we solve the hydrodynamical model one time step ahead in time. This allows us to solve the models in parallel, using two different groups of processors. By a little tuning of the parameters in the algorithm, a load-balancing has been obtained that is close to optimal. As a result, both models require roughly the same amount of CPU time, so that one of them, effectively, can be considered as obtained 'for free'. Copyright © 2002 John Wiley & Sons, Ltd.

KEY WORDS: numerical analysis; partial differential equations; iteration methods; approximate factorization; parallelism

1. INTRODUCTION

Modeling the dynamics of shallow seas is of great importance to many aspects of human interest, such as land protection, shipping, recreation, exploitation of oil and gas fields, etc. In addition to the dynamics, there is much interest in environmental issues, for example in levels of pollution. In general, the ecological situation as a result of man's use (and misuse) of shallow seas generates many questions that are still unanswered. For example, questions

*Correspondence to: B. P. Sommeijer, Centre for Maths and Computer Science, Dept. MAS, CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands.

† The investigations reported in this paper were supported by the NCF (Foundation Nationale Computer Faciliteiten).

Received April 2000
Revised July 2001

with respect to long-term trends clearly have a political impact and definitely require more research before scientists can produce reliable predictions.

Along with the two classical scientific lines of experimentation and theoretical analysis, the value of computer simulation of marine flow and transport problems has now been established. In fact, by adding more and more physical aspects to the model, numerical simulation seems to become increasingly important. Although modern high-speed computers indeed provoke the inclusion of an increased number of physical phenomena, the resulting CPU time is still the limiting factor in many realistic simulations. Therefore, further research to design numerical techniques that are most appropriate for the problem at hand is still of crucial importance.

Undoubtedly, the ever-increasing capacity of high-performance computers has given an enormous impulse to the development of codes for performing real-life simulations. However, in spite of this, the complexity of the full ecosystem in a shallow sea is so large that a number of simplifications still has to be incorporated in the current models, simply to keep the amount of storage and CPU time at a realistic level. In Reference [1] it is indicated that realistic models would require unacceptably large simulation times, even on 'Teraflop' (10^{12} flops) machines. Since fast computers like the CRAY C916 and the NEC SX4 perform in the Gigaflop (10^9 flops) range, it is clear that we should lower our demands, at least for the next decade. Nevertheless, there is a possibility to bridge part of the gap between present-day practice and the ultimate goal. This possibility is provided by efficient, tailor-made numerical algorithms in combination with innovative computer science techniques. In this way, satisfactory results are feasible to make a significant step towards understanding the full complexity.

At CWI, several transport solvers have been designed during the last few years [2–5]. These models describe the advective and diffusive transport of contaminants in shallow water combined with chemical or biological interaction. The present status of this research is that an arbitrary number of species can be dealt with.

The output of the corresponding code consists of species concentrations, in space and time. As input, any transport solver needs the velocity field. Up to now, the velocity field was considered to be given (in fact, we used an analytically prescribed expression). In real practice, however, the unknown velocities have to be computed by a 3D hydrodynamical solver. The output of this solver then serves as input for the transport solver. Common practice nowadays is to calculate the flow field *a priori* over the whole time interval and to store the output. Especially on fine, three-dimensional grids and long simulation intervals, this approach requires an enormous amount of storage. Moreover, the transport solver will spend a lot of I/O to read all this pre-computed data from file. This is of course a very combersome approach. A possibility to avoid all this data transfer is to let the hydrodynamical solver run concurrently with the transport solver. However, many existing hydrodynamical solvers have been designed in the previous decade (see e.g. Reference [6]) and are based on algorithms different from the one we use in the transport solver. Moreover, completely different data structures have been used so that the conversion of the data from one solver to the other will certainly decrease the overall performance. Therefore, coupling the transport solver with an existing hydrodynamical solver will lead to an 'unbalanced combination'. To avoid this situation, we will discuss a hydrodynamical solver that is based on the same algorithm as used in the transport solver. This is a natural choice since the underlying partial differential equations are to a large extent of the same nature. Choosing, in both solvers, the same spatial grids and time steps enables us to use the same data structures, which is important to increase the performance on a supercomputer.

As an additional advantage, we automatically achieve a divergence-free velocity field in the grid points where it is needed by the transport solver. This property is of great relevance for the transport solver and can only be realized at high costs in the case that we use one of the existing hydrodynamical solvers, which usually produces output in 'unwanted' points.

Hence, the proposed approach leads to an integrated code which is much more 'balanced' than a combination of two existing codes.

2. MATHEMATICAL DESCRIPTION

We start with the mathematical model formulation for the transport process and the hydrodynamics in shallow water. As said in the Introduction, both models possess similar features. Next, we will briefly discuss the spatial discretization, resulting in a large system of ordinary differential equations (ODEs).

2.1. The transport model

The model for the transport of pollutants etc. combined with their chemical or bio-chemical interactions is defined by an initial-boundary value problem for the system of 3D advection–diffusion–reaction equations (cf. [7])

$$\frac{\partial c_i}{\partial t} = L(u, v, w; \varepsilon_x, \varepsilon_y, \varepsilon_z)\, c_i + g_i(t, x, y, z, c_1, \ldots, c_m), \quad i = 1, \ldots, m \tag{1}$$

where the differential operator L is defined by

$$L(u, v, w; \varepsilon_x, \varepsilon_y, \varepsilon_z) := -u\frac{\partial}{\partial x} - v\frac{\partial}{\partial y} - w\frac{\partial}{\partial z} + \frac{\partial \varepsilon_x \partial}{\partial x^2} + \frac{\partial \varepsilon_y \partial}{\partial y^2} + \frac{\partial \varepsilon_z \partial}{\partial z^2} \tag{2}$$

Here, the c_i denote the (unknown) concentrations of the contaminants, u, v, w are the local fluid velocities in the x, y, z direction respectively, the ε's are the diffusion coefficients, and the functions g_i describe the chemical reactions, emissions from sources, etc. and therefore depend on the concentrations. Note that the mutual coupling in the system (1) is due to these functions g_i.

2.2. The hydrodynamical model

The mathematical model describing the hydrodynamics in shallow water is defined by an initial-boundary value problem for the system of 3D equations (cf. [7])

$$\begin{aligned}
\frac{\partial u}{\partial t} &= L(u, v, w; \delta_x, \delta_y, \delta_z)u + \omega v - g\,\frac{\partial}{\partial x}\,\zeta + \tau_x \\
\frac{\partial v}{\partial t} &= L(u, v, w; \delta_x, \delta_y, \delta_z)v - \omega u - g\,\frac{\partial}{\partial y}\,\zeta + \tau_y \\
\frac{\partial \zeta}{\partial t} &= -\int_{-d}^{\zeta} \frac{\partial}{\partial x}\, u(t, x, y, s)\, ds - \int_{-d}^{\zeta} \frac{\partial}{\partial y}\, v(t, x, y, s)\, ds
\end{aligned} \tag{3}$$

where L is the same as in the transport case (notice that the diffusion parameters may be different in the hydrodynamical model). This similarity, combined with the fact that the oper-

ator L plays a dominant role in Equations (1) and (3), allows us to use the same numerical methods in both models. The vertical velocity component w is defined by requiring that the velocity field is divergence free, i.e.,

$$w(t,x,y,z) = -\int_{-d}^{z} \frac{\partial}{\partial x} u(t,x,y,s)\, ds - \int_{-d}^{z} \frac{\partial}{\partial y} v(t,x,y,s)\, ds \tag{4}$$

The various quantities in $\{(3),(4)\}$ have the following meaning:

u, v, w	local fluid velocities in x, y, z directions (divergence free by virtue of Equation (4)),
ζ	water elevation,
τ_x, τ_y	external forcing functions in the horizontal directions, like wind forces,
$\delta_x, \delta_y, \delta_z$	diffusion coefficients in x, y, z directions,
g	acceleration due to gravity,
ω	Coriolis parameter,
d	depth function.

Both models are defined on an arbitrary domain, the boundaries of which consist of coastal lines and ocean boundaries, which are both assumed vertical. In the transport model $\{(1),(2)\}$ we will be interested in the transport of the pollutants, locally induced in the 'middle of the sea'. Hence, as boundary conditions we will use vanishing concentrations at the boundaries of the domain.

For the hydrodynamical model $\{(2),(3),(4)\}$, the water elevation at the ocean boundary is prescribed and at the coastal boundaries the velocity normal to the coast is required to be zero. Furthermore, at the sea surface and at the sea bed we impose the usual free surface and bottom friction condition (see Reference [7]).

For the spatial discretization we replace the physical domain by a set of $N := N_x N_y N_z$ Cartesian grid points with mesh sizes $\Delta x, \Delta y$, and Δz, and approximate the transport model and its boundary conditions by the semi-discrete mN-dimensional initial value problem (IVP)

$$\frac{\mathrm{d}\mathbf{C}(t)}{\mathrm{d}t} = \mathbf{F}(t, \mathbf{C}(t)), \qquad \mathbf{C}(t_0) = \mathbf{C}_0 \tag{5}$$

Here $\mathbf{C}$ contains the m concentrations c_i at all N grid points and $\mathbf{C}_0$ defines the initial values. The advection terms have been discretized by third-order upwind-biased $\kappa = 1/3$ discretizations and the diffusion terms by symmetric three-point discretizations (see e.g. Reference [8]).

Similarly, for the hydrodynamical model we obtain the $N_x N_y (2N_z + 1)$-dimensional IVP

$$\begin{aligned}
\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}t} &= \Lambda_{xyz}(\mathbf{U},\mathbf{V},\mathbf{W})\mathbf{U} + \omega\mathbf{V} - gD_x\mathbf{Z} + \mathbf{T}_x, \quad \mathbf{U}(t_0) = \mathbf{U}_0 \\
\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t} &= \Lambda_{xyz}(\mathbf{U},\mathbf{V},\mathbf{W})\mathbf{V} - \omega\mathbf{U} - gD_y\mathbf{Z} + \mathbf{T}_y, \quad \mathbf{V}(t_0) = \mathbf{V}_0 \\
\frac{\mathrm{d}\mathbf{Z}}{\mathrm{d}t} &= -A_x(\mathbf{Z})\mathbf{U} - A_y(\mathbf{Z})\mathbf{V}, \quad \mathbf{Z}(t_0) = \mathbf{Z}_0
\end{aligned} \tag{6}$$

where $\mathbf{W}$ is defined by

$$\mathbf{W} = -C_x\mathbf{U} - C_y\mathbf{V} \tag{7}$$

Here, $\mathbf{U}$ and $\mathbf{V}$ contain the horizontal velocity components at all $N_xN_yN_z$ grid points, $\mathbf{Z}$ contains the elevation at the N_xN_y horizontal grid points in the upper plane of the three-dimensional grid, $\mathbf{T}_x$ and $\mathbf{T}_y$ represent the external forces at the grid points including the inhomogeneous parts of the boundary conditions, Λ_{xyz}, A_x, and A_y are matrices depending on the velocity or elevation values, and C_x, C_y, D_x and D_y are constant matrices. The matrix Λ_{xyz} also takes the coastal, free surface and bottom friction conditions into account.

3. TIME INTEGRATION

In order to cope with the stiffness of the IVPs (5) and (6), we shall use an implicit formula for the time discretization. Since both systems are advection dominated, this implicit formula should at least be A-stable and preferably L-stable (see, e.g. Reference [9] for a description of these concepts). The specific choice of such a highly stable time discretization formula depends on the required order of accuracy in time. Assuming that second-order accuracy suffices, we shall use the second-order, L-stable Backward Differentiation Formula (BDF) [9].

For the description of the BDF method and its iterative solution process, we will use the compact notation

$$\frac{\mathrm{d}\mathbf{Y}(t)}{\mathrm{d}t} = \mathbf{F}(t, \mathbf{Y}(t)), \quad \mathbf{Y}(t_0) = \mathbf{Y}_0 \tag{8}$$

where $\mathbf{Y}(t) = \mathbf{C}(t)$ in the transport case and $\mathbf{Y}(t) = (\mathbf{U}(t)^T, \mathbf{V}(t)^T, \mathbf{Z}(t)^T)^T$ in the hydrodynamical case. The BDF discretization is defined by

$$\mathbf{R}(t_{n+1}, \mathbf{Y}_{n+1}) = \mathbf{0} \tag{9}$$

where

$$\mathbf{R}(t_1, \mathbf{Y}) := \mathbf{Y} - \Delta t\, \mathbf{F}(t_1, \mathbf{Y}) - \mathbf{Y}_0 \tag{10}$$

$$\mathbf{R}(t_{n+1}, \mathbf{Y}) := \mathbf{Y} - \tfrac{2}{3}\Delta t\, \mathbf{F}(t_{n+1}, \mathbf{Y}) - \tfrac{1}{3}[4\mathbf{Y}_n - \mathbf{Y}_{n-1}], \quad n \geqslant 1 \tag{11}$$

Here, $\Delta t := t_{n+1} - t_n$ is the (constant) time step and $\mathbf{Y}_n$ is an approximation to the solution $\mathbf{Y}(t_n)$. Clearly, both models have to solve the implicit relation (9) in each time step. Since the dimension of these systems is usually extremely large (10^6 unknowns is certainly not an exception), and because we are dealing with a multidimensional coupling, systems of the form (9) can only be solved by using advanced iterative solution techniques that are tuned to modern parallel vector machines. In the next section we will discuss such an iteration method.

4. THE ITERATION PROCESS

The most simple iteration process that one can think of is fixed point iteration, defined by

$$\mathbf{Y}^{(j)} = \mathbf{Y}^{(j-1)} - \mathbf{R}(t_{n+1}, \mathbf{Y}^{(j-1)}), \quad j = 1, 2, \ldots \tag{12}$$

Although this iteration process is relatively cheap, highly vectorizable, and highly parallelizable it is not suitable for our purpose since the large Lipschitz constant associated with the

residual function $\mathbf{R}$, will force us to use extremely small Δt in order to obtain convergence. Therefore, we have to discard Equation (12).

Next we consider the preconditioned process

$$P(\mathbf{Y}^{(j)} - \mathbf{Y}^{(j-1)}) = -\mathbf{R}(t_{n+1}, \mathbf{Y}^{(j-1)}), \quad j = 1, 2, \ldots \tag{13}$$

where the preconditioning matrix P should compensate for the large Lipschitz constant. For example, choosing

$$P = I - \tfrac{2}{3}\Delta t J, \quad J := \frac{\partial \mathbf{F}(t, \mathbf{Y})}{\partial \mathbf{Y}} \tag{14}$$

yields the well known modified Newton process when the Jacobian matrix J is evaluated at t_n and kept fixed during the iteration process. This process is expected to converge under rather mild conditions on the time step Δt. However, each iteration requires the solution of a large linear system for which the linear algebra is so expensive (due to coupling in the spatial directions) that we also have to drop this approach.

To arrive at a manageable level of computations we propose to replace P by its so-called 'Approximate Factorization' (AF) (see References [5, 10, 11]) defined by

$$P := (I - \tfrac{2}{3}\Delta t J_x)(I - \tfrac{2}{3}\Delta t J_y)(I - \tfrac{2}{3}\Delta t J_z) \tag{15}$$

with $J = J_x + J_y + J_z$. The matrices J_x, J_y, and J_z correspond to the terms in the various spatial directions. The effect of this factorization is that now, successively, three linear systems have to be solved in each iteration. However, each of these systems is much simpler since they have a banded structure. Because these systems are easily vectorizable and parallelizable, they can be solved very efficiently. Indeed, an optimal implementation on the CRAY C916 shows a high performance (cf. [3]).

For the transport model it is obvious how to choose the matrices J_x, J_y, and J_z. The corresponding AF iteration method has extensively been analyzed and tested in [2–5]. In these papers we examined the situation that the vertical mesh size Δz did not impose a condition on the time step Δt. This is a nice property since in shallow water Δz is small. The main result obtained in these papers is that the time step has to satisfy a condition of the form

$$\Delta t \leqslant \frac{\gamma}{\max\{\rho(J_x), \rho(J_y)\}} \tag{16}$$

in order to obtain convergence. Here, ρ denotes the spectral radius and γ is a constant depending on the underlying method. For the second-order BDF, this constant equals 0.96.

For the hydrodynamical model we have several options how to choose the matrices J_x, J_y, and J_z. The full Jacobian matrix J is given by (cf. Equation (6))

$$J := \begin{pmatrix} \Lambda_{xyz} & \omega I & -gD_x \\ -\omega I & \Lambda_{xyz} & -gD_y \\ -A_x & -A_y & O \end{pmatrix} \tag{17}$$

where we ignored that Λ_{xyz}, A_x, and A_y depend on the velocity or elevation values. To apply the AF technique, we suggest to choose the block-triangular matrices

$$J_x := \begin{pmatrix} \Lambda_x & O & O \\ -\omega I & \Lambda_x & O \\ -A_x & -A_y & O \end{pmatrix}, \quad J_y := \begin{pmatrix} \Lambda_y & \omega I & -gD_x \\ O & \Lambda_y & -gD_y \\ O & O & O \end{pmatrix}, \quad J_z := \begin{pmatrix} \Lambda_z & O & O \\ O & \Lambda_z & O \\ O & O & O \end{pmatrix} \tag{18}$$

with $\Lambda_{xyz} = \Lambda_x + \Lambda_y + \Lambda_z$, where Λ_x, Λ_y and Λ_z represent the coupling in the x, y and z direction, respectively. The scheme defined by {(13), (15), (18)} requires the solution of six linear systems per iteration. Notice that the two systems corresponding with J_z can be solved in parallel. Hence, effectively, only five systems have to be solved. Each of these systems is only 'one-dimensional', which allows for an extremely fast solution on a parallel vector computer (cf. References [3, 12]). Notice that after each iteration the vertical velocity component **W** has to be updated according to

$$\mathbf{W}^{(j)} = -C_x\mathbf{U}^{(j)} - C_y\mathbf{V}^{(j)} \tag{19}$$

since this quantity is needed in the right-hand side function for the horizontal velocity components **U** and **V** (cf. Equation (6)).

The convergence analysis of the resulting AF method is beyond the scope of this paper and is subject of future research.

5. NUMERICAL ILLUSTRATION

In this section we will describe a numerical test with the coupled hydrodynamical and transport solver. The domain of interest is defined by a rectangle in the horizontal: $0 \leqslant x \leqslant L_x$, $0 \leqslant y \leqslant L_y$, and we use a constant depth: $-L_z \leqslant z \leqslant 0$. At the east, south and west boundary we assume coastal lines, whereas the north boundary is formed by the ocean. For the spatial grid (uniform in each direction) we use $N_x = 41$, $N_y = 81$, $N_z = 31$ grid points. Hence, in the hydrodynamical problem we have approximately 2×10^5 unknowns. In the transport part we consider 10 different species resulting in more than 10^6 unknowns.

For each of these species we assume an initial distribution with a Gauss-shaped form, centred around the point $(x, y) = (L_x/4, L_y/4)$:

$$c_i(t=0,x,y,z) = \exp\left(\mu_i \frac{z}{L_z} - \gamma_i \left[\left(\frac{x}{L_x} - \frac{1}{4}\right)^2 + \left(\frac{y}{L_y} - \frac{1}{4}\right)^2\right]\right), \quad i = 1, \ldots, 10 \tag{20}$$

with μ_i in the range $[0.5, 1]$ and γ_i in the interval $[20, 120]$. The inhomogeneous terms g_i in Equation (1) are defined by non-stiff (i.e., slow), nonlinear reaction equations.

Initially, the sea is in rest ($\mathbf{U}=\mathbf{V}=\mathbf{W}=\mathbf{Z}=\mathbf{0}$ at $t=0$). The whole process is driven by the wind field defined by

$$\begin{aligned}\tau_x(t,x,y) &= 10^{-5}\left(1.5+0.75\sin\left(\frac{2\pi t}{24*3600}\right)\right)\exp\left(-10\left(\frac{x}{L_x}-\frac{1}{2}\right)^2\right)\\ \tau_y(t,x,y) &= 10^{-5}\left(1.5+0.75\cos\left(\frac{2\pi t}{24*3600}\right)\right)\exp\left(-2.5\left(\frac{y}{L_y}-1\right)^2\right)\end{aligned} \tag{21}$$

Hence, this 'south-western' wind will cause a velocity field, which in turn will activate the transport. This process will be simulated during five hours 'real time', i.e., $0\leqslant t\leqslant T_{\text{end}}=18\,000$.

In this experiment we take the following values for the physical parameters:

$$\begin{array}{lll} L_x=100\ 000 & L_y=200\ 000 & L_z=100\\ \varepsilon_x=0.5 & \varepsilon_y=0.5 & \varepsilon_z=0.05\\ \delta_x=0.05 & \delta_y=0.05 & \delta_z=0.01\\ \omega=7.27\times10^{-5}*2\sin(50^\circ), & g=9.81 & \end{array}$$

The idea to exploit parallelism is that – in the combined solution process – the hydrodynamical model is solved *concurrently* with the transport model. Because the flow field is input for the transport model, the hydrodynamical solver should be ahead in time by (at least) one time step. Thus, one group of processors integrates the hydrodynamical equations over a step Δt from t_{n+1} until t_{n+2}, while the other group of processors integrates, in parallel, the transport equations from t_n until t_{n+1}. Hence, compared with the original, stand-alone transport solver, the calculation of the flow field is 'for free', due to parallelism.

In passing, we remark that the processors within each group can be exploited to obtain a further amount of parallelism. Both the hydrodynamical solver and the transport solver allow for intrinsic concurrency. For example, all the 'one-dimensional' linear systems that have to be solved are independent along the grid lines in that particular spatial direction. Another possibility is offered in the transport part where the term $L(u,v,w;\varepsilon_x,\varepsilon_y,\varepsilon_z)c_i$ on the right-hand side of Equation (1) can be calculated concurrently for all species.

We recall that the number of unknowns in the hydrodynamical and in the transport model is given by $N_xN_y(2N_z+1)$ and $mN_xN_yN_z$, respectively, m denoting the number of species. Hence, the ratio is approximately given by $m/2$. This ratio also holds for the number of systems to be solved in each iteration. Assuming that both solvers need the same number of iterations to solve their respective implicit relations, we see that the transport solver is expected to be $m/2$ times more expensive per time step. To obtain a good balance, the program has been organized in such a way that the hydrodynamical solver takes time steps which are $m/2$ times smaller than the steps used in the transport solver. Hence, in our test example with $m=10$, the transport solver takes one step of size Δt from t_n until t_{n+1} while at the same time the hydrodynamical solver takes five steps of size $\Delta t/5$ from t_{n+1} until t_{n+2}. In this way we expect both solvers to arrive at the same time at their target points t_{n+1} and t_{n+2}. In Table I we list the ratio of the CPU times needed by both solvers to advance the solution over a distance Δt. This ratio should be close to one for a good load-balancing.

Table I. Performance results of the coupled models.

Number of time steps	$\frac{\text{CPU}_{\text{hydro}}}{\text{CPU}_{\text{trans}}}$	Averaged no. of iter. in hydrodynamical solver	Averaged no. of iter. in transport solver
30	1.14	3.23	4.13
60	1.18	2.77	3.28
90	1.17	2.56	3.04
120	1.16	2.45	3.01
180	1.15	2.32	3.00

As an example, we show in Figure 1 the initial and final state of c_1, the first component in the system of transport equations. At $t = 0$, this concentration is defined by Equation (20), where we used $\gamma_1 = 120$ and $\mu_1 = 1$. At $t = T_{\text{end}}$ we see that the solution has been transported in 'north-east' direction. Furthermore, we observe that c_1 has been spread and decreased in magnitude, due to diffusion and chemical interaction with other species.

For the iteration process we have implemented the following strategy: in both models we iterate until 'convergence', thus allowing for a varying number of iterations. Here, convergence is defined as: the residual function $\mathbf{R}(t_{n+1}, \mathbf{Y}^{(j-1)})$ (cf. Equation (13)), measured in the maximum norm, should be less than a prescribed tolerance value, which is chosen equal to 10^{-5}. Table I lists the number of iterations needed by both solvers, averaged over all time steps. We see that the numbers of iterations needed by the hydrodynamical solver are slightly smaller than the ones needed by the transport solver. A possible explanation is that the hydrodynamical solver uses a stepsize that is five times as small, which improves the rate of convergence, of course.

We remark that it turns out that the restriction on the time step to obtain a convergent iteration process is more stringent for the hydrodynamical solver than for the transport solver. Hence, the load-balancing requirement to apply the hydrodynamical solver with a smaller time step is in good harmony with the convergence requirements.

6. CONCLUSION

In this paper we considered the coupled solution of a 3D hydrodynamical model and a 3D transport model, including chemical interactions. Both models are solved using the same numerical algorithms. We have choosen the second-order BDF method for the time integration because of its excellent stability behavior. The implicit relations are solved iteratively, using an Approximate Factorization technique. As a result, only 'one-dimensional' linear systems have to be solved. This can be implemented extremely efficiently on a multi-processor vector computer.

The aim was to organize the computations in such a way that the hydrodynamical model and the transport model could be solved concurrently. This goal has been achieved by solving the hydrodynamical model slightly ahead of time. This is a natural approach since the output of the hydrodynamical solver (i.e., the flow field) serves as input for the transport solver. In this way we can avoid the usual approach where the flow field is calculated *a priori* and stored in

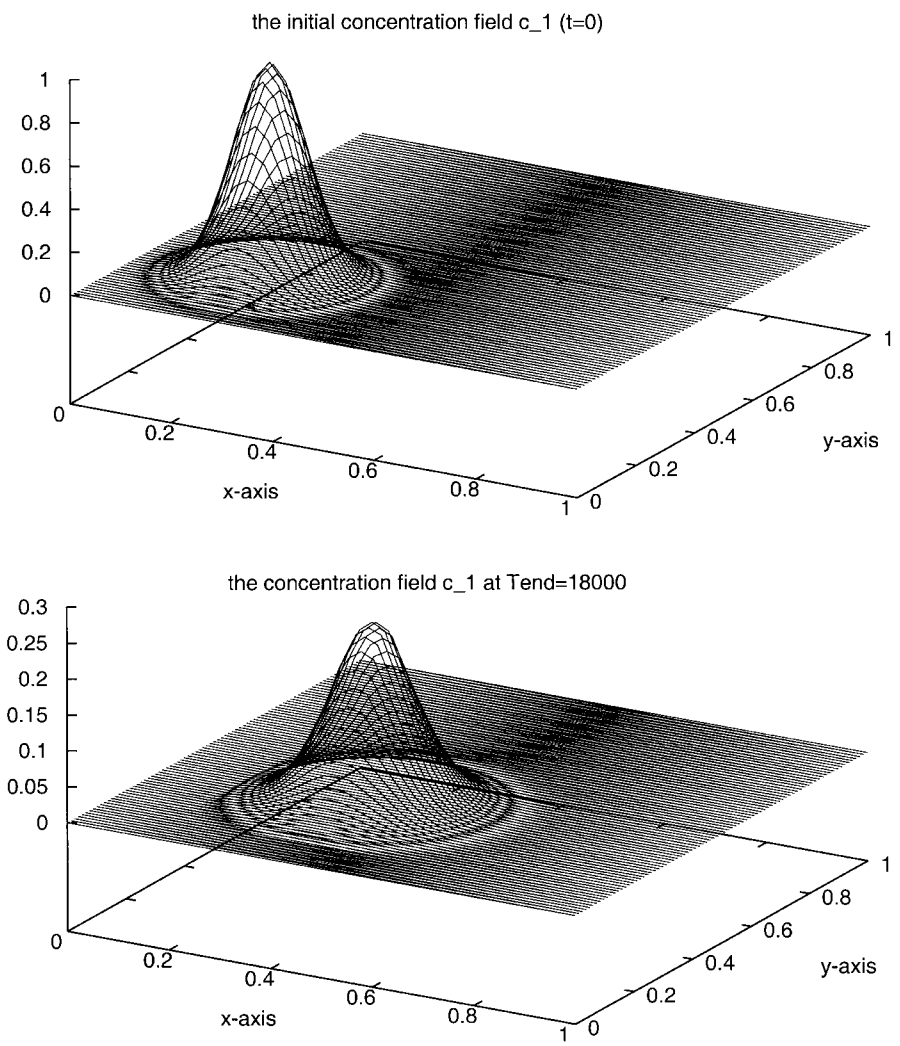

Figure 1. Top: the initial concentration field c_1 $(t = 0)$; bottom: the concentration field c_1 at the endpoint $T_{\text{end}} = 18\,000$. Notice that the x- and y-coordinates have been scaled by L_x and L_y, respectively.

large files. This latter approach forces the transport solver to read all this precomputed data which has a strong negative influence on the performance on a supercomputer.

By a little tuning of the parameters in the algorithm a load-balancing could be obtained that is close to optimal. As a result, the hydrodynamical solver, running in parallel on different processors, requires approximately the same amount of CPU time and can, effectively, be considered as obtained 'for free'.

As an extension to the work discussed here, we could mention to consider factorizations that differ from the one proposed in Equation (18) in the hydrodynamical part and to analyze the convergence of the resulting variants. This will be subject of future research.

REFERENCES

1. Stelling GS, Roose D, Sommeijer BP, van der Houwen PJ, Kok J, Lin HX, Tan KH. New generation shelf flux solvers. In *Final NOWESP Report*; 1997; also: Report MAS-R9815, CWI, Amsterdam.
2. Eichler-Liebenow C, van der Houwen PJ, Sommeijer BP. Analysis of approximate factorization in iteration methods. *Applied Numerical Mathematics* 1998; **28**:245–258.
3. Sommeijer BP. The iterative solution of fully implicit discretizations of three-dimensional transport models. In *Parallel Computational Fluid Dynamics, Development and Applications of Parallel Technology*, Lin CA, Ecer A, Periaux J, Satofuka N, Fox P (eds). Proceedings of the 10th parallel CFD Conference, Hsinchu, Taiwan, Elsevier, 1999; pp. 67–74.
4. van der Houwen PJ, Sommeijer BP. Factorization in block-triangularly implicit methods for shallow water applications. *Applied Numerical Mathematics* 2001; **36**:113–128.
5. van der Houwen PJ, Sommeijer BP. Approximate factorization for time-dependent partial differential equations. *Journal of Computational and Applied Mathematics* 2001; **128**(VII):447–466.
6. de Goede ED. Numerical methods for the three-dimensional shallow water equations on supercomputers. *Thesis*, University of Amsterdam, 1992.
7. Vreugdenhil CB. Numerical methods for shallow-water flow. *Water Science and Technology Library*, Vol. 13. Kluwer Academic Publishers, 1994.
8. Hirsch C. Numerical computation of internal and external flows I. *Fundamentals of Numerical Discretization*. Wiley: Chichester, 1988.
9. Hairer E, Wanner G. Solving ordinary differential equations II. Stiff and differential-algebraic problems, 2nd edn., *Springer Series in Computational Mathematics*, Vol. 14. Springer: Berlin, 1996.
10. Beam RM, Warming RF. An implicit finite-difference algorithm for hyperbolic systems in conservation-law form. *Journal of Computational Physics* 1976; **22**:87–110.
11. Warming RF, Beam RM. An extension of A-stability to alternating direction implicit methods. *BIT* 1979; **19**:395–417.
12. Sommeijer BP, Kok J. Implementation and performance of a three-dimensional numerical transport model. *International Journal for Numerical Methods in Fluids* 1995; **21**:349–367.

