1992

R.A. Trompert

Local uniform grid refinement and brine transport in porous media with inhomogeneities

Department of Numerical Mathematics Report NM-R9224 November

CWI is the research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a non-profit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands organization for scientific research (NWO).

Local Uniform Grid Refinement and Brine Transport in Porous Media with Inhomogeneities

Ron Trompert
CWI
P.O. Box 4079, 1009 AB, Amsterdam, The Netherlands

Abstract. The application of an adaptive grid method is discussed for a mathematical model for unsteady flow, coupled with transport of brine in porous media with inhomogeneities in two space dimensions. When the concentration of salt is large, the salt concentration profile in brine transport problems can show locally large gradients in space and in time. For this reason we have chosen an adaptive grid method to solve these problems. We consider a method based on local uniform grid refinement, where integration takes place on a series of nested, local uniform finer and finer subgrids. These subgrids are created up to a level of refinement where sufficient spatial accuracy is reached and their location and shape is adjusted after each time step. The space domain is considered to be a rectangle and all grids in use are uniform and cartesian. The interfaces, caused by the inhomogeneities, are assumed to coincide with cell edges in the numerical approximation. Special conditions are applied here, connecting the solutions on both sides of the interface. These interface conditions involve continuity of fluxes across the interfaces. The mesh refinement process and the variable time stepsizes are controlled by heuristic error monitors. The performance of the method is illustrated by two example problems.

1980 Mathematics subject classification: Primary:65M50. Secondary:65M20.
1991 CR Categories: G.1.8.
Key Words & Phrases: partial differential equations, numerical mathematics, time-dependent problems, adaptive grid methods, fluid-flow/solute-transport in porous media with inhomogeneities, brine transport.

Note: This work was carried out as a part of contract research by order of the Laboratory for Soil and Groundwater Research of RIVM - the Dutch National Institute of Public Health and Environmental Protection - in connection with project "Locatiespecifieke Modelvalidatie" 725205. Financial support for this project was provided by the Dutch Ministry of Economic Affairs.

1. INTRODUCTION

An adaptive grid finite difference method is applied to a model for unsteady, isothermal flow coupled with transport of brine in porous media with inhomogeneities. The origin of this work lies in a safety assessment study on disposal of high-level radioactive wastes in rock salt formations, like salt domes. The concentration of salt in the proximity of salt formations is known to be large and also in aquifers overlying these salt formations the salt content varies from fresh water to that of saturated brine [4].

The numerical simulations of groundwater flow near these salt formations may provide insight in what might happen in the event of contaminants escaping from such a repository. Also with respect to the mathematical modeling itself, numerical simulations can be useful. Since recent studies indicate that the

Report NM-R9224
ISSN 0169-0388
CWI
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

basic mathematical model, designed for low salt concentration cases, need to be re-examined when the salt concentration becomes large [3, 4]. Numerical simulations together with laboratory experiments can then be used to validate a new modified mathematical model.

It should be noted that the presence of a high concentration of salt in these natural situations gives rise to large concentration gradients as well. A typical situation one encounters is that of a sharp fresh-salt water interface that moves in time. For such problems, a single uniform space grid held fixed throughout the entire time evolution can be computationally very inefficient, since, to afford an accurate approximation, such a grid has to be very fine over the whole domain while a fine grid is only needed there where the sharp front is located. Adaptive grid methods prove to be very useful here, since these methods refine the space grid only there where it is really needed, hence, reducing the memory use and CPU time.

The applied adaptive grid method is based on local uniform grid refinement and is also discussed in our previous work [7-13]. The main feature of local uniform grid refinement is that integration takes place on a series of nested, local uniform finer and finer subgrids which are automatically adjusted at discrete times in order to follow the movement of rapid spatial transitions. The generation of these subgrids is continued until the spatial phenomena are described with sufficient accuracy.

This work can be regarded as a sequel to the work reported in [11], in which an adaptive grid code was discussed for computing brine transport problems in homogeneous porous media. Since in natural situations, soil parameters such as the permeability, the transversal and longitudinal dispersivity etc. can change abruptly from one region to another, this adaptive grid code has now been adapted so that it can handle brine transport in porous media with such inhomogeneities. In order to get consistent numerical approximations, interface conditions are applied at grid nodes in the vicinity of these abrupt changes. These conditions connect the numerical solution on both sides of the interface and are based on continuity of fluxes across these interfaces. The space domain is considered to be a rectangle and all grids in use are uniform and cartesian. Numerically, the interfaces are assumed to coincide with the cell edges.

The adaptive grid code can handle systems of PDEs of the following type, defined on a rectangular domain Ω with boundary $\partial\Omega$,

$$
\begin{aligned}
&G(x,y,t,u,u_t,u_x,u_y,u_{xx},u_{xy},u_{yy}) = 0, \quad (x,y) \in \Omega, \quad t > t_0,\\
&H(x,y,t,u,u_t,u_x,u_y) = 0, \quad (x,y) \in \partial\Omega, \quad t > t_0,\\
&u(x,y,t_0) = u_0(x,y), \quad (x,y) \in \Omega \cup \partial\Omega,
\end{aligned}
\tag{1.1}
$$

where the exact solution u may be vector valued.

For time integration we use implicit Euler for the first time step and the second order two-step implicit BDF method with variable coefficients for the following time steps where variable time stepsizes are taken. Standard second order finite differences are used for space discretization and the interpolation is linear. The discretization of the boundary conditions and the interface conditions are of first order. The resulting systems of equations are solved by modified Newton's method in combination with ILU preconditioned Bi-CGSTAB [15].

In Section 2, we discuss the mathematical model of brine transport in porous media we have used in this paper. An outline of the local uniform mesh refinement method is given in Section 3. The interface conditions are considered in Section 4 and the strategies concerning grid refinement, time stepping, and the Newton iteration process are given in Section 5. Two specific example problems are used to illustrate the performance of our research code. This is described in Section 6 and finally, a summary and concluding remarks are given in Section 7.

2. Model of Brine Transport in Porous Media

In this section, we are going to describe the mathematical model we have used to solve the two example problems (cf. Section 6). Following [11] we consider a model for unsteady, isothermal, single-phase, two-component saturated flow in a porous medium in two space dimensions. This model contains two conservation laws, namely one for the mass of the total fluid, i.e. water and salt and one for the mass of salt only. The mass conservation of the total fluid supplemented with Darcy's law for the velocity field is given by

$$\frac{\partial}{\partial t}(n\rho) + \nabla.(\rho \mathbf{q}) = 0, \quad \mathbf{q} = -\frac{k}{\mu}(\nabla p - \rho \mathbf{g}), \tag{2.1}$$

where n is the porosity of the porous medium, ρ is the mass density and $\mathbf{q}$ the velocity vector of the total fluid. The permeability of the porous medium is denoted by k, μ is the dynamic viscosity, p the pressure and $\mathbf{g}$ the acceleration of gravity vector. The mass conservation law of salt and Fick's law for the dispersive mass fluxes are given by

$$\frac{\partial}{\partial t}(n\rho\omega) + \nabla.(\rho\omega\mathbf{q} + \rho\mathbf{J}) = 0, \quad \mathbf{J} = -n\mathbf{D}\nabla\omega, \tag{2.2}$$

respectively, where ω is the concentration of salt and $\mathbf{J}$ the dispersive mass flux vector. $\mathbf{D}$ is the 2×2 dispersion tensor defined by

$$n\mathbf{D} = (nd_m + \alpha_t|\mathbf{q}|)\mathbf{I} + (\alpha_l - \alpha_t)\frac{\mathbf{q}\mathbf{q}^T}{|\mathbf{q}|}, \quad |\mathbf{q}| = (\mathbf{q}^T\mathbf{q})^{1/2}, \tag{2.3}$$

where α_l denotes the longitudinal and α_t the transversal dispersivity and d_m the molecular diffusion. $\mathbf{I}$ is the 2×2 identity matrix. The soil parameters in this model are n, d_m, α_l, α_t and k. They can assume different values in different porous media. Temperature and compressibility effects are neglected in this model, as well as sources, sinks and deformation of the porous medium. To complete the model we have an equation of state for the fluid mass density ρ and a polynomial expression for the dynamic viscosity μ which depends on the concentration of salt:

$$\rho = \rho_0 \exp(\gamma\omega), \tag{2.4}$$

$$\mu = \mu_0(1 + 1.85\omega - 4.10\omega^2 + 44.50\omega^3), \tag{2.5}$$

where ρ_0 and μ_0 are the reference density and dynamic viscosity and γ is a coefficient obtained from laboratory experiments.

In cases of a low salt concentration (2.1) and (2.2) are only weakly coupled and can be solved independently. The flow can then be regarded as independent from the density gradients caused by differences in the salt concentration since these gradients prove to be negligible. However, we consider cases of high salt concentration, in which case the flow is no longer independent from the density gradients, so these equations should be solved simultaneously. With this model we have followed [4] in the description of brine transport, except for Darcy's law and Fick's law. In this paper these laws are used in their classical formulation, valid for low concentration cases.

Using p and ω as independent variables, we have discretized the equations (2.1), (2.2) in the form

$$-\gamma\nabla.\mathbf{J} - \gamma^2\mathbf{J}.\nabla\omega + \nabla.\mathbf{q} = 0, \tag{2.6}$$

$$n\frac{\partial \omega}{\partial t} + \mathbf{q}.\nabla\omega + \gamma\mathbf{J}.\nabla\omega + \nabla.\mathbf{J} = 0,$$

which is obtained after some elementary calculations. At this stage we note that this model fits into format (1.1) and can, within the limits of (1.1), be modified by the user of the code, like, for example, by adding a temperature equation or by using different formulations for Darcy's law and Fick's law or by adding compressibility effects, etcetera.

3. Outline of the Adaptive-Grid Method

Although its elaboration readily becomes complicated, the idea behind local uniform grid refinement is simple. Starting from a coarse base grid, covering the whole domain, finer and finer uniform subgrids are created locally in a nested manner in regions of high spatial activity. Here, a set of interconnected grid cells, all having the same sizes, is called a subgrid. A set of subgrids having the same cell sizes is called a grid level or just grid. Hence, a grid level consists of a single subgrid or several disjunct non-overlapping subgrids. A new initial-boundary value problem is solved at each grid level and the integration takes place in a consecutive order, from coarse to fine. Each of these integrations spans the same time interval. Required initial values are defined by interpolation from the next coarser grid level or taken from a grid level from the previous time step when available. Internal boundaries, i.e. subgrid boundaries lying in the interior of the domain, are treated as Dirichlet boundaries and values are also interpolated from the next coarser grid level. Where the boundary of a fine subgrid coincides with the boundary of the domain, the given boundary conditions are used. The generation of grid levels is determined by the local refinement strategy and is continued until the spatial phenomena are described well enough by the finest grid. The fine grid cells are created by bisecting the sides of the cells of the next coarser grid. Note that the subgrids created this way need not be rectangles.

During each time step the following operations are performed:

1. *Solve PDEs on the coarse grid.*
2. *If the desired accuracy in space or the maximum number of grid levels is reached then go to 8.*
3. *Determine new finer grid level at forward time.*
4. *Interpolate internal boundary values at forward time.*
5. *Provide new initial values at backward time.*
6. *Solve PDEs on new grid level, using the same steplength.*
7. *go to 2.*
8. *Inject fine grid values in coinciding coarser grid points.*

Thus, for each time step the computation starts at the coarse base grid using the most accurate solution available, since fine grid solution values are always injected in coinciding coarse grid points and all grid levels are kept in storage for step continuation.

4. Interface Conditions

The soil parameters in porous media can show abrupt changes from one region to another. Moreover, across these interfaces, i.e. there where the sudden changes occur, p and ω are continuous but their profiles may be kinked. We will assume that, mathematically, the soil parameters are piecewise constant functions and that p and ω are both continuous functions, not differentiable in space, at an interface. This means that in order to get consistent numerical approximations, we have to take care that numerical differentiation does not take place across such an interface. Therefore, the numerical solution at an interface is obtained by fulfilling interface conditions which connect the solution on both sides of the interface and involve only one-

sided difference schemes. Since (2.1) and (2.2) represent two conservation laws, it is natural impose continuity of the spatial fluxes $\rho\mathbf{q}.\mathbf{n}$ and $(\rho\omega\mathbf{q} + \rho\mathbf{J}).\mathbf{n}$ at interfaces as interface conditions, where $\mathbf{n}$ is a vector locally perpendicular to the interface. It suffices to impose continuity of $\mathbf{q}.\mathbf{n}$ and $\mathbf{J}.\mathbf{n}$, since ρ and ω are both continuous functions.

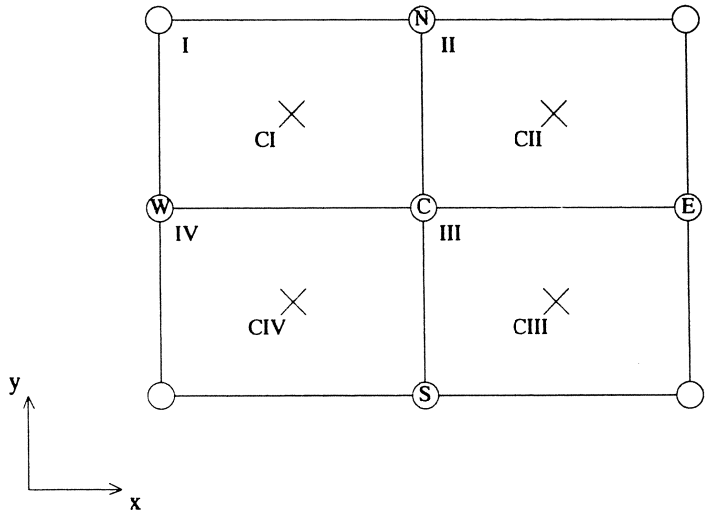

FIGURE 4.1 Four arbitrary grid cells with cell edges, parallel to the co-ordinate axes.

Consider the four grid cells shown in Figure. 4.1, numbered I till IV, possessing cell edges, parallel to the co-ordinate axes. First, the soil parameters are evaluated in all cell centers and are supposed to be constant over each cell. Hence, the interfaces are assumed to coincide with cell edges in the numerical approximation. When the soil parameters are constant over these four cells then none of these cells are intersected by an interface and (2.6) is discretized at grid node C using the standard second order finite differences in space. Now suppose that, for example, the soil parameters in CI are different from those in CII. Then the component of $\mathbf{q}$ and $\mathbf{J}$ in x-direction which is perpendicular to the cell edge, separating the upper left cell I from the upper right cell II, must be continuous. This cell edge is denoted as CN. Due to (2.1)-(2.3) we have,

$$
\begin{aligned}
q_1 &= -\frac{k}{\mu}(p_x - \rho g_1),\\
q_2 &= -\frac{k}{\mu}(p_y - \rho g_2),\\
J_1 &= -nD_{11}\omega_x - nD_{12}\omega_y,\\
nD_{11} &= nd_m + \alpha_t|\mathbf{q}| + (\alpha_l - \alpha_t)\frac{q_1^2}{|\mathbf{q}|},\\
nD_{12} &= (\alpha_l - \alpha_t)\frac{q_1 q_2}{|\mathbf{q}|},
\end{aligned}
\tag{4.1}
$$

where q_1, J_1 and g_1 are the components in x-direction of $\mathbf{q}$, $\mathbf{J}$ and $\mathbf{g}$, respectively and nD_{11} and nD_{12} are elements of the first row of the dispersion tensor $n\mathbf{D}$; q_2 and g_2 are the components in y-direction of $\mathbf{q}$ and $\mathbf{g}$. The derivatives of (4.1) are discretized using the grid nodes N, S, E, W and C, which yields a first order accurate discretization. The fluxes q_1 and J_1 on the left and righthand side of CN are denoted as $q_{1,CN,I}$,

$J_{1,CN,I}$ and $q_{1,CN,II}, J_{1,CN,II}$ respectively. They are now approximated as

$$
\begin{aligned}
q_{1,CN,I} &= -\frac{k_{CI}}{\mu_C}\left(\frac{p_C - p_W}{\Delta x} - \rho_C g_1\right),\\
J_{1,CN,I} &= -nD_{11,CN,I}\frac{\omega_C - \omega_W}{\Delta x} - nD_{12,CN,I}\frac{\omega_N - \omega_C}{\Delta y},\\
q_{1,CN,II} &= -\frac{k_{CII}}{\mu_C}\left(\frac{p_E - p_C}{\Delta x} - \rho_C g_1\right),\\
J_{1,CN,II} &= -nD_{11,CN,II}\frac{\omega_E - \omega_C}{\Delta x} - nD_{12,CN,II}\frac{\omega_N - \omega_C}{\Delta y},
\end{aligned}
\tag{4.2}
$$

where

$$
\begin{aligned}
nD_{11,CN,I} &= nd_{m,CI} + \alpha_{t,CI}|\mathbf{q}_{CN,I}| + (\alpha_{l,CI} - \alpha_{t,CI})\frac{q_{1,CN,I}^2}{|\mathbf{q}_{CN,I}|},\\
nD_{12,CN,I} &= (\alpha_{l,CI} - \alpha_{t,CI})\frac{q_{1,CN,I}\, q_{2,CN,I}}{|\mathbf{q}_{CN,I}|},\\
nD_{11,CN,II} &= nd_{m,CII} + \alpha_{t,CII}|\mathbf{q}_{CN,II}| + (\alpha_{l,CII} - \alpha_{t,CII})\frac{q_{1,CN,II}^2}{|\mathbf{q}_{CN,II}|},\\
nD_{12,CN,II} &= (\alpha_{l,CII} - \alpha_{t,CII})\frac{q_{1,CN,II}\, q_{2,CN,II}}{|\mathbf{q}_{CN,II}|},\\
q_{2,CN,I} &= -\frac{k_{CI}}{\mu_C}\left(\frac{p_N - p_C}{\Delta y} - \rho_C g_2\right),\\
q_{2,CN,II} &= -\frac{k_{CII}}{\mu_C}\left(\frac{p_N - p_C}{\Delta y} - \rho_C g_2\right),\\
|\mathbf{q}_{CN,I}| &= (q_{1,CN,I}^2 + q_{2,CN,I}^2)^{1/2}, \quad |\mathbf{q}_{CN,II}| = (q_{1,CN,II}^2 + q_{2,CN,II}^2)^{1/2}.
\end{aligned}
\tag{4.3}
$$

Here $q_{2,CN,I}$, $q_{2,CN,II}$ represent velocities parallel to CN and $nD_{11,CN,I}$, $nD_{12,CN,I}$, $nD_{11,CN,II}$, $nD_{12,CN,II}$ the elements of the first row of the dispersion tensor, on each side of CN. Constants like k_{CII} and $\alpha_{l,CII}$ denote the permeability and longitudinal dispersivity at cell II and entries like, for example, ρ_C and μ_C are the mass density and the dynamic viscosity in C. Continuity of $\mathbf{q}.\,\mathbf{n}$ and $\mathbf{J}.\,\mathbf{n}$ across CN yields the following system of flux continuity equations for p and ω in C

$$
\begin{aligned}
q_{1,CN,I} - q_{1,CN,II} &= 0,\\
J_{1,CN,I} - J_{1,CN,II} &= 0.
\end{aligned}
\tag{4.4}
$$

When not only CN is an interface but also CW, CE or CS then the flux continuity equations are generated for each interface. The equations we then solve is the sum of these flux continuity equations.

5. STRATEGIES

5.1. REFINEMENT STRATEGY

The reason why the local uniform grid refinement method is an interesting method for solving PDEs with steep solutions is that it can solve these PDEs just as accurately as on a very fine grid, but with considerably less computational effort, since the involved fine subgrids cover only a part of the domain. Moreover, it creates extra refinements when necessary and removes these when they are no longer needed. This refinement process is controlled by a refinement strategy. In [7-9, 12, 13], the refinement strategy is based on a comprehensive error analysis taking into account space discretization and interpolation error estimates. The intention of this strategy is that the overall spatial accuracy is dominated by the spatial accuracy at the finest grid level. When the number of grid levels is constant for all times, this strategy should lead to a spatial accuracy which is comparable to the one achieved with a single uniform grid having cell sizes identical to those of the finest grid level in use in the adaptive grid method. The success of this refinement strategy is very much dependent on the accuracy of error estimates. It's clear that these error estimates can only be accurate when the solution is sufficiently smooth, i.e. it may be steep but it should be sufficiently differentiable in space. Since nonsmoothness in the boundary conditions, or even in the solution itself, is a well known phenomenon in brine transport problems, the approach above was dropped and replaced by a more heuristic approach. In [10, 11] the refinement strategy was based on a curvature monitor. This monitor is also used here.

We are now going to introducc some notation. Let the vector U^n denote the numerical approximation to the solution u of (1.1) at time t_n on a space grid. Suppose that (1.1) consists of *npde* PDEs, so the solution vector u has length *npde*. In this case, u_i represents the i^{th} component of u and U_i^n the numerical approximation to u_i at time t_n on a space grid. Let the component of U_i^n associated with the grid node (k,l) be written as $U_i^n(k,l)$, where k and l are indices related to the space co-ordinates x and y of this node. The curvature monitor value corresponding with the i^{th} solution component in (k,l) is now defined as

$$ESTS_i(k,l) = \frac{1}{scale(i)} \{ |U_i^n(k+1,l)-2U_i^n(k,l) + U_i^n(k-1,l)| + |U_i^n(k,l+1)-2U_i^n(k,l) + U_i^n(k,l-1)| \}. \tag{5.1}$$

The user defined array *scale* has length *npde* and holds characteristic values of each solution component. At every grid node $ESTS_i(k,l)$ is computed for each solution component i. At boundary nodes, the difference formulas in (5.1) are replaced by equivalent one-sided formulas.

Suppose we have just completed a time step on grid level m; grid level 1 is the coarsest grid level or the base grid, grid level 2 is the next coarsest grid level and so on. After this time step the maximum values of $ESTS_i(k,l)$ are computed over grid level m for each component i. These maxima are denoted as $ESTSmax_i$. If for some i, $ESTSmax_i > TOLS$, then a new grid level $m+1$ is created within the current time step, provided $m+1$ does not exceed the user specified maximum number of grid levels. Here, TOLS is a user defined tolerance. Grid level $m+1$ is now determined as follows. For each i, for which $ESTSmax_i > TOLS$ holds, the cells around the nodes of grid level m where $ESTS_i(k,l) > \frac{1}{4} \times TOLS$ will be subdivided in four identical cells. The set of these finer cells makes up grid level $m+1$, on which the current time step will now be repeated.

Finally, we have built in an extra condition to smoothen the behaviour of the code. Suppose that the maximum number of grid levels during the previous time step is *levtop* and that at grid level $m<levtop$, $ESTSmax_i \le TOLS$. Although this means that a new finer grid level $m+1$ is actually not necessary, it will still be created when $ESTSmax_i > 0.9 \times TOLS$. This way fluctuation of the maximum number of grid levels from one time point to the next is likely to be avoided.

5.2. *Time Integration Aspects*

We have implemented the two-step BDF method of order two which we apply in the variable stepsize mode. The time derivative in (1.1) is then approximated as

$$U_t \approx \frac{U^n - a_1 U^{n-1} - a_2 U^{n-2}}{\theta_2 \Delta t_n}, \tag{5.2}$$

where

$$a_1 = \frac{(c+1)^2}{c^2+2c}, \quad a_2 = \frac{-1}{c^2+2c}, \quad \theta_2 = \frac{c+1}{c+2}, \tag{5.3}$$
$$c = \frac{\Delta t_{n-1}}{\Delta t_n}, \quad \Delta t_n = t_n - t_{n-1}.$$

Here, U_t represents the pointwise restriction of u_t to a space grid. We note that variable time stepping is a prerequisite for brine transport problems in porous media, as they can exhibit a highly distinct behaviour in time. As starting formula we employ the one-step BDF method of order one (implicit Euler).

The time stepsize is controlled by the time error monitor value

$$ESTT = \| \frac{U_i^n - U_i^{n-1}}{scale(i)} \|_\infty, \quad i=1,.....,npde, \tag{5.4}$$

which is computed only over the interior grid nodes of each grid level for reason of robustness of the code. We will not elaborate this further here. After each time step on all grid levels, defined in Section 3, the maximum value of *ESTT* is computed over all grid levels. If this maximum exceeds a user specified tolerance *TOLT*, then the time step is rejected, otherwise accepted. For each grid level a new time stepsize is predicted such that the predicted value of *ESTT* for the new time step is equal to 0.5 x *TOLT*. The minimum of these new time stepsize estimates is taken to be the time stepsize for the next time step. However, in case of a step rejection, the new time stepsize will be taken as 0.8 x this estimated value. In all cases we require that the new time stepsize is not smaller than ⅓ x and not larger than 2 x the old time stepsize to avoid too large jumps in the stepsize selection. Finally, the new time stepsize is corrected with a small value to assure that the next output point is reached exactly.

5.3. *Solution of the Linear and Nonlinear Systems*

Because we use an implicit integration method and treat PDEs like (1.1) fully coupled, we are facing the task of solving large coupled systems of nonlinear algebraic equations. In our code we use modified Newton in combination with the preconditioned Bi-CGSTAB [15] for solving these equations. In the remainder of this section we will explain the implemented solution procedure.

For any system of PDEs like (1.1), the required Jacobian matrix for the Newton process is computed in a completely automatic manner. To illustrate this, consider the 1D form

$$G(u_t, u_x, u_{xx}) = 0, \tag{5.5}$$

for which the Jacobian matrix is tridiagonal. Recall that we use central 3-point finite differencing on a

uniform space grid with cell size Δx. The diagonal entries, corresponding with internal grid points, are then easily seen to be defined by the functional

$$\frac{\partial G}{\partial u_t}\frac{1}{\theta_2 \Delta t} - \frac{\partial G}{\partial u_{xx}}\frac{2}{(\Delta x)^2}, \tag{5.6}$$

where Δt is the time stepsize. Similar expressions are easily found for the nondiagonal entries and for grid nodes whose finite difference expression depends on the boundary conditions. This way of constructing a Jacobian was borrowed from [5]. In our code the partial derivatives are estimated by a simple first order difference formula, so that the user does not need to specify these. The procedure we followed, described below, was obtained from [1]. For example, we use the approximation

$$\frac{\partial G}{\partial u_{xx}} \approx \frac{G(u_t, u_x, u_{xx} + \varepsilon) - G(u_t, u_x, u_{xx})}{\varepsilon}, \tag{5.7}$$

$$\varepsilon = (uround)^{1/2} \max(|u_{xx}|, \text{typ}\,(u))\,\text{sign}(u_{xx}).$$

Here *uround* is the machine roundoff error. The value "typ (u)" represents a characteristic value of u. In our code we use the user defined array *scale* for these values. In (5.7) we use the recomputed value for ε, given by

$$\varepsilon := (u_{xx} + \varepsilon) - u_{xx}. \tag{5.8}$$

This is a trick to enhance the accuracy of approximation (5.7).

Let the nonlinear system of equations to be solved be denoted as,

$$F(U) = 0. \tag{5.9}$$

In the modified Newton approach, the linear system,

$$\begin{aligned} J(U^0)\delta^k &= -F(U^{k-1}), \\ U^k &= U^{k-1} + \delta^k, \end{aligned} \tag{5.10}$$

is subsequently solved, starting with $k=1$, until a stopping criterion is fulfilled. Here, J is the Jacobian matrix, U^0 is the initial guess, U^k is the k^{th} iterate. The stopping criterion in our code is a relative error test, based on the correction δ^k. It reads

$$\max_i\{\max_j\{\frac{|\delta^k_{i,j}|}{w_{i,j}}\}\} < 1, \tag{5.11}$$

$$w_{i,j} = 10^{-3} \min\{TOLT^2, TOLS\} \max\{|U^k_{i,j}|, 10^{-2}\, scale(i)\}.$$

The lower index i refers to the i^{th} PDE solution component and j to the nodes of the current grid level. Since the accuracy of computed solutions increase when *TOLT* or *TOLS* decrease, it is natural to let the stopping criterion depend on these tolerances. We have used $TOLT^2$ here because the time error behaves like $O(\Delta t^2)$

while the time error monitor value *ESTT* from (5.4) only behaves like $O(\Delta t)$ as the time stepsize $\Delta t \to 0$. For this reason the stopping criterion should depend quadratically on *TOLT*. The order of convergence of the space error monitor (5.1), however, is in agreement with the order of the space discretization so there is only a linear dependence on *TOLS*. The linear system (5.10) is solved by ILU preconditioned Bi-CGSTAB. This iterative solver was obtained by modifying the public domain CGS code from the SLAP library written by Anne Greenbaum and Mark K. Seager which is available from netlib [2]. Also some alterations were made to the stopping criterion of this code to better prepare it for solving subsequent iterations in a Newton process [14]. For example, the stopping criterion in the netlib code is relative to the righthand side vector of (5.10) which vanishes in a Newton iteration. After l Bi-CGSTAB iterations we have

$$J(U^0)\delta^{k,l} = -F(U^{k-1}) + r^l, \tag{5.12}$$

where r^l is the residue of the Bi-CGSTAB process and $\delta^{k,l}$, the approximation of δ^k after l iterations. Following [14], the stopping criterion now reads

$$\max_i\{\max_j\{\frac{|(K^{-1}r^l)_{i,j}|}{w_{i,j}}\}\} < \frac{1}{20 \times \text{maximum \# Newton iterations}}, \tag{5.13}$$

where K is the ILU decomposition of the Jacobian matrix.

Suppose we solve a system of time-dependent nonlinear PDEs on a single space grid. In this case, the standard modified Newton procedure for solving a system of nonlinear algebraic equations stemming from such a system of PDEs would be; first compute a Jacobian at the beginning of a time step, then iterate and when the iteration fails to converge, start again with a smaller time stepsize. This is a good approach for the problem above because when the time stepsize decreases, the solution of the nonlinear equations will be closer to the solution at the beginning of the time step which is used as initial guess for the iteration process. This procedure, however, doesn't always work in case a system of PDEs is solved with the local uniform grid refinement method. In this adaptive grid method, the local subgrids can move or grow in space or be newly created. This means that it frequently happens that the initial values for an initial-boundary value problem, defined on a subgrid, have to be interpolated from a next coarser subgrid. When the subgrid is newly created, this interpolation takes place over the whole subgrid and when the subgrid moves or grows, interpolation only takes place over the part of the subgrid which did not exist at the backward time point. Although the refinement strategy attempts to create subgrids in such a way that interpolation only takes place in regions where the solution is smooth, it can happen that the interpolated initial values are not accurate enough. If this is the case, then it's possible that the Newton iteration does not converge because of this, since, these (partially) interpolated initial solution is used as initial guess for the iteration as well as for computing the Jacobian. When solving brine transport problems in porous media with inhomogeneities one encounters these problems. For example when a subgrid is newly created or moves in space from one time point to the next, initial values for the new fine subgrid cells are interpolated from the next coarser subgrid solution which may be kinked. This way initial data is obtained which doesn't look like the fine grid solution to the PDEs at the backward time point. To our experience, when the Newton iteration fails to converge, time stepsize reduction works very poorly, or not at all, in such a case. For this reason the modified Newton procedure has to be adapted. The adaptations we have made will now be elaborated.

The solution at the backward time point is taken as the initial guess for the finest grid level. With respect to the initial guess for the coarser grid levels, we note that, in spite of the fact that injection of fine grid values in coinciding nodes improves the accuracy of the solution at the coarser grid level (cf. Section 2, step 8), the updated coarser grid solution is usually not a very good initial guess for the solution at this grid at the future time point. Therefore, we also keep the original, not-updated solution at the backward time point in storage which is used as initial guess for the next time step. After this, the linear system (5.10) is generated and iteratively solved. In case this iteration process terminates unsuccessfully, (5.10) is generated all over

again, employing a smaller time stepsize. When (5.10) is solved at least twice (i.e. after two nonlinear iterations), we check for convergence and convergence speed as follows. Let *itmax* be the specified maximum number of nonlinear iterations, k the number of completed iterations and let $errit_k$ be defined as

$$errit_k = \max_i\{\max_j\{\frac{|\delta_{i,j}^k|}{w_{i,j}}\}\}. \tag{5.14}$$

The convergence rate is defined as $errit_k/errit_{k-1}$. Assuming linear convergence of the iteration process, i.e. the convergence rate does not tend to zero when $k\to\infty$, the value of $errit_{itmax}$, the value of *errit* after the maximum number of iterations have been performed, can then be approximated as

$$errit_{itmax} \approx errit_k \left(\frac{errit_k}{errit_{k-1}}\right)^{itmax-k}. \tag{5.15}$$

Our convergence/convergence speed criterion now reads $errit_{itmax} < 1$. When this criterion is satisfied, (5.11) is expected to be fulfilled after *itmax* iterations. Note that this criterion terminates a diverging as well as a slowly converging iteration process. When the convergence/convergence speed criterion is fulfilled, we check if the stopping criterion (5.11) is satisfied. If this is the case then we are finished, otherwise we proceed with the next iteration. In case that the convergence/convergence speed criterion is not satisfied, a new Jacobian is computed. There are two ways to compute a new Jacobian. First, the Jacobian can be computed using the previous iterate U^{k-1} as initial guess and employing the same time stepsize. Second, we can compute the new Jacobian using the original initial guess U^0 with a reduced time stepsize, just like in the standard modified Newton approach. How the Jacobian is going to be calculated depends on a number of criteria. First, the number of new Jacobians with the same time stepsize during the whole iteration process is limited to a user defined maximum. If this maximum is reached then the new Jacobian is computed with a reduced time stepsize. When a new Jacobian with the same time stepsize was already obtained during the previous iteration and the convergence/convergence speed criterion is still not satisfied, the new Jacobian is also calculated with a smaller time stepsize. Suppose that the last iteration where a new Jacobian was computed with the same time stepsize is denoted by j. We assume that when $\|F(U^{k-1})\|_\infty < \|F(U^{j-1})\|_\infty$, the iterate U^{k-1} is a "better" solution to (5.9) than U^{j-1}. A new Jacobian is only computed with the same time stepsize if this is the case, and computed with a reduced time stepsize, otherwise.

This algorithm is more complicated than standard modified Newton. It's behaviour ranges from standard modified Newton to a genuine Newton-Raphson process. The idea behind it is that when the convergence criteria are not fulfilled, the iteration is not immediately repeated with a smaller time stepsize, like in the standard modified Newton approach, but a new Jacobian is tried first, based on the last accepted iterate and with the same time stepsize. Should this fail too, then the iteration is repeated with a smaller time stepsize. The maximum number of Newton iterations and Jacobians with the same time stepsize in our code are chosen to be 10 and 5 respectively. When the time stepsize needs to be decreased, we take the new stepsize to be ¼ × the previous one.

6. NUMERICAL ILLUSTRATIONS

Two example problems are presented dealing with the displacement of fresh water by brine in a thin vertical column, filled with a porous medium and measuring one by one meter. Here we assume that $\mathbf{g} = (0,-g)^T$. The values of the parameters which are the same for both problems are chosen as

$$n = 0.4, \quad d_m = 0\,m^2.s, \quad \rho_0 = 10^3\,kg.m^{-3}, \quad p_0 = 10^5\,N.m^{-2}, \tag{6.1}$$

$$\gamma = \log(2.0), \quad g = 9.81\, m.s^{-2}, \quad \mu_0 = 10^{-3} kg.m.s^{-1}.$$

6.1. PROBLEM 1

In this example problem, the vertical column is completely open at the top as well as at the bottom and closed at its vertical sides. This configuration is shown in Figure 6.1. Inside this column, two elliptic shaped regions, denoted by I, can be distinguished.

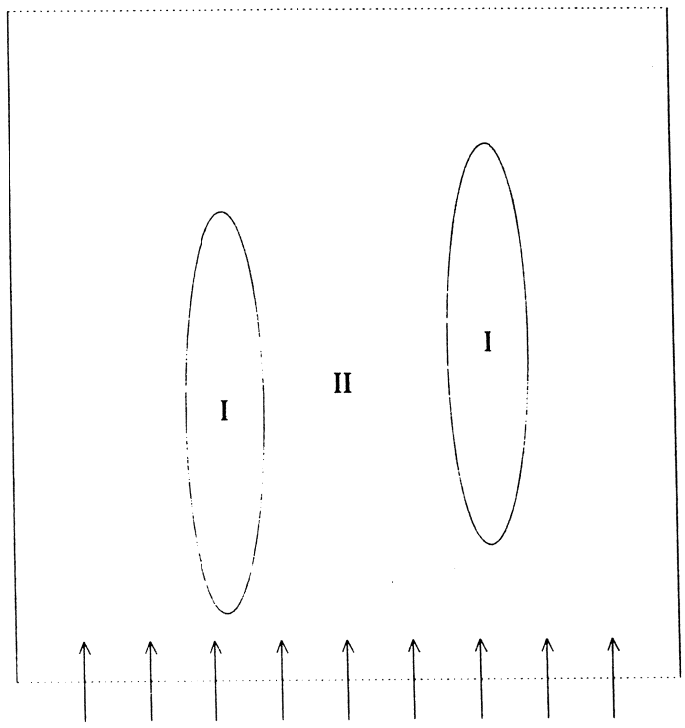

FIGURE 6.1 The vertical column of problem 1.

The permeability and the longitudinal and transversal dispersivity in I are chosen to be different from those in the remainder of the column, II. The initial values and boundary conditions are:

$$\begin{aligned}
&p(x,y,0) = p_0 + (1-y)\rho_0 g, \quad \omega(x,y,0) = 0, \quad 0\,m < x,y < 1\,m,\\
&q_1 = 0\,m.s^{-1}, \quad \omega_x = 0\,m^{-1}, \quad x = 0{,}1\,m \text{ and } 0\,m < y < 1\,m,\\
&q_2 = 10^{-4}\,m.s^{-1}, \quad \omega = 0.25 \times (1 - \exp(-10t)), \quad 0\,m < x < 1\,m \text{ and } y = 0\,m,\\
&p = p_0, \quad \omega_y = 0\,m^{-1}, \quad 0\,m < x < 1\,m \text{ and } y = 1\,m.
\end{aligned} \tag{6.2}$$

The soil parameters are given by:

$$\begin{aligned} &\text{I:} \quad k = 10^{-13}\, m^2, \quad \alpha_l = 0.005\, m, \quad \alpha_t = 0.001\, m, \\ &\text{II:} \quad k = 10^{-10}\, m^2, \quad \alpha_l = 0.010\, m, \quad \alpha_t = 0.002\, m. \end{aligned} \tag{6.3}$$

The ellipses are defined as:

$$\begin{aligned} &(\frac{x-0.32}{0.05})^2 + (\frac{y-0.5}{0.3})^2 = 1, \\ &(\frac{x-0.72}{0.05})^2 + (\frac{y-0.4}{0.3})^2 = 1. \end{aligned} \tag{6.4}$$

respectively. At the bottom, saturated brine is injected into the column. As this happens, a steep front in the salt concentration developes which starts to move slowly towards the top of the column. While moving, the front tends to smooth out due to dispersion. When the salt front collides with the elliptic regions, the salt moves around these regions, so the salt concentration in I remains virtually unchanged. When the salt front reaches the top of the column, there is still almost no salt present in I. Only much later we see a noticable penetration of salt in I. This continues until a steady state situation is reached and the salt concentration is at its saturation level everywhere in the column.

We have computed the solution to this problem using two and three grid levels, of which the coarsest a 20×20 grid. For both cases we have chosen *TOLT* = 0.1 and *TOLS* = 0.25. The salt concentration at two times is shown in Figure 6.2 together with the corresponding grid levels. We have also computed the solution on a single uniform 40×40 and 80×80 grid to compare those with the solution obtained with the two-level and three-level computations, respectively. The salt concentration computed on these grids are shown in Figure 6.3. Since the elliptic regions are rather thin, the space error monitor computed at the coarse 20×20 base grid experiences difficulties in seeing these regions in time, i.e. before the salt front gets there. In order to get more accurate results in this example problem, refinements were created at the interfaces, irrespective of the space error monitor values there. When we compare Figure 6.2 with Figure 6.3 we see only minor differences, indicating that the adaptive grid computations produce results with a comparable accuracy as the results on a single uniform grid.

6.2. PROBLEM 2

In the vertical column we use in this example problem, there are four different regions, indicated as I till IV. This is shown in Figure 6.4. Each of these regions has its own permeability and longitudinal and transversal dispersivity. These are given in (6.3). The column is completely open at the top and only half open at the bottom. The vertical sides are just like in the previous example, closed. The initial values, boundary conditions and soil parameters are:

$$\begin{aligned} &p(x,y,0) = p_0 + (1-y)\rho_0 g, \quad \omega(x,y,0) = 0, \quad 0\, m < x,y < 1\, m, \\ &q_1 = 0\, m.s^{-1}, \quad \omega_x = 0\, m^{-1}, \quad x = 0,1\, m \text{ and } 0\, m < y < 1\, m, \\ &q_2 = 10^{-4}\, m.s^{-1}, \quad \omega = 0.25 \times (1 - \exp(-10t)), \quad 0\, m < x \leq 0.5\, m \text{ and } y = 0\, m, \\ &q_2 = 0\, m.s^{-1}, \quad \omega = 0, \quad 0.5\, m < x < 1\, m \text{ and } y = 0\, m, \\ &p = p_0, \quad \omega_y = 0\, m^{-1}, \quad 0\, m < x < 1\, m \text{ and } y = 1\, m. \end{aligned} \tag{6.5}$$

The soil parameters are given by:

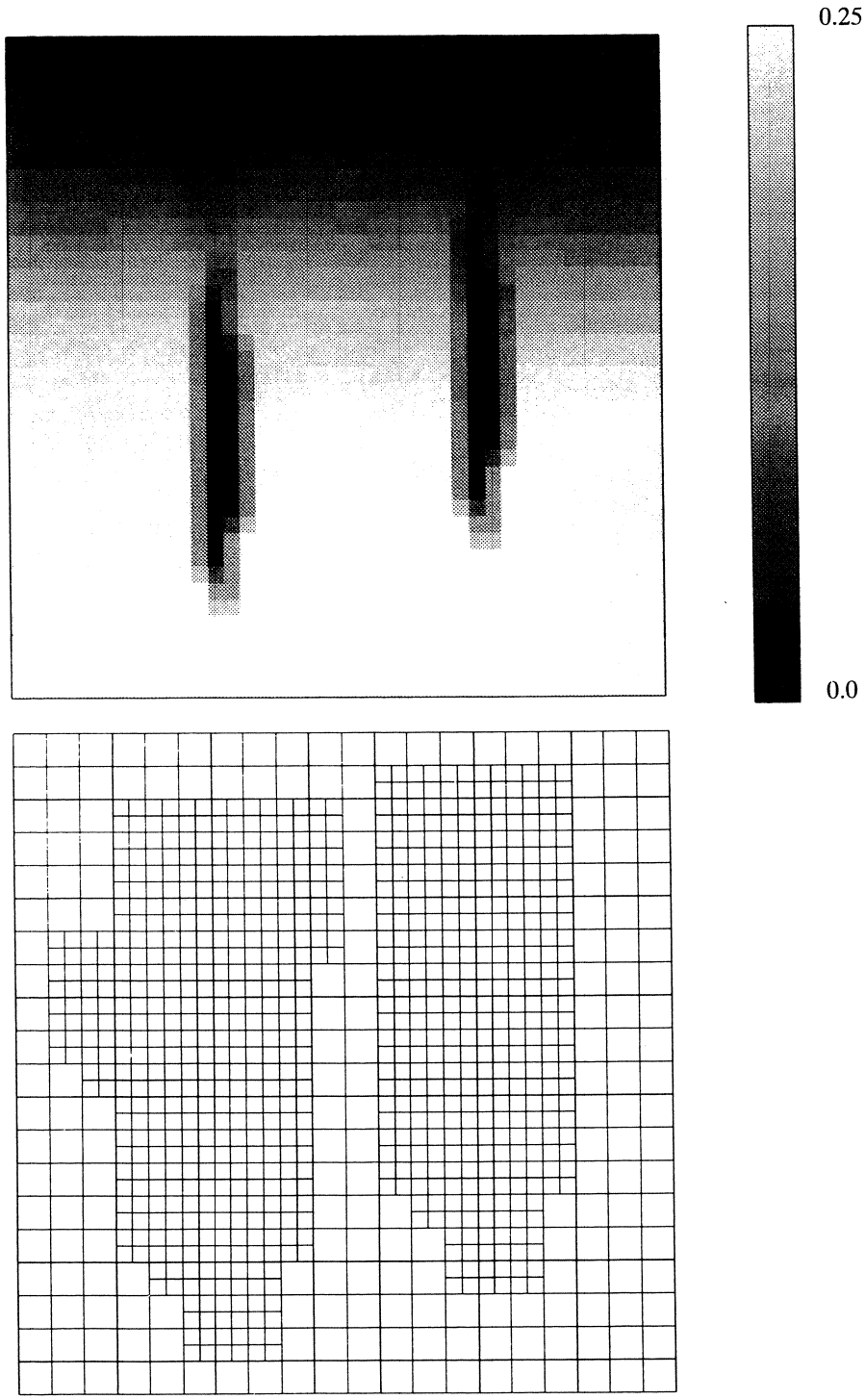

FIGURE 6.2 The salt concentration with two grid levels at t=2000.

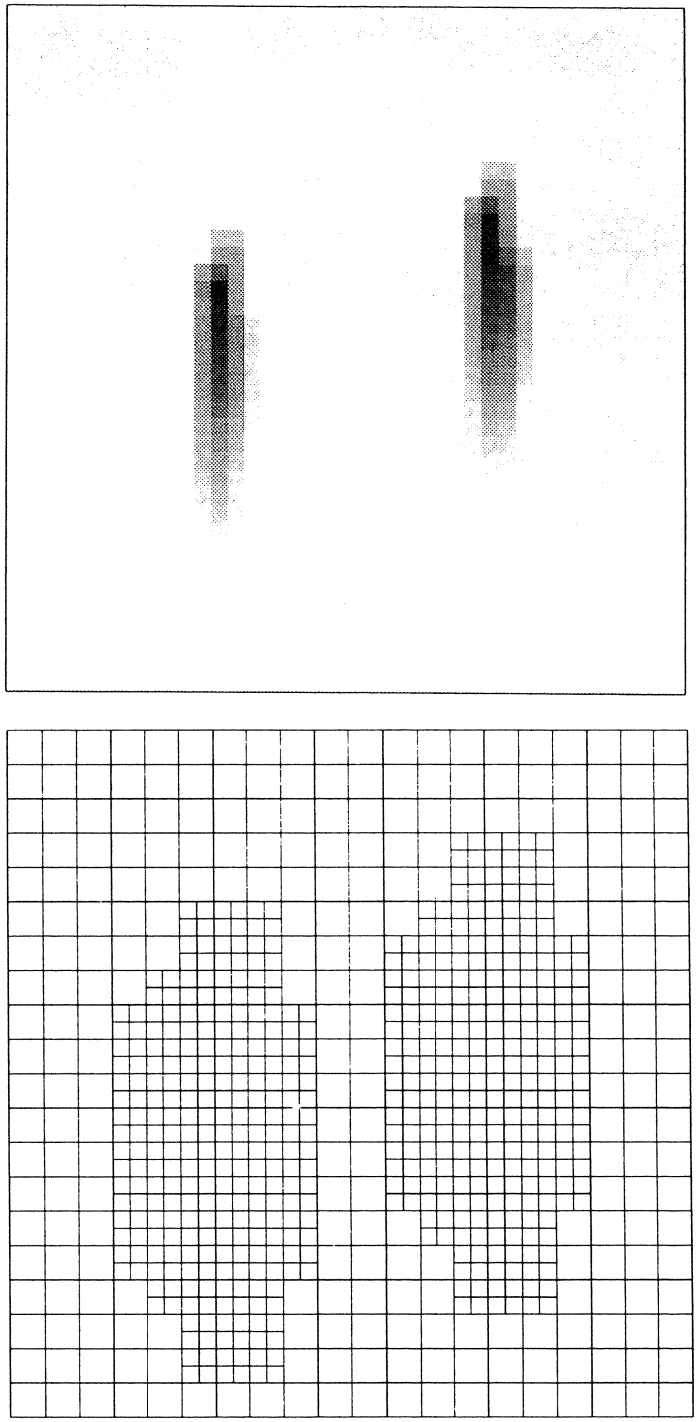

FIGURE 6.2 Continued. The salt concentration with two grid levels at t =200000.

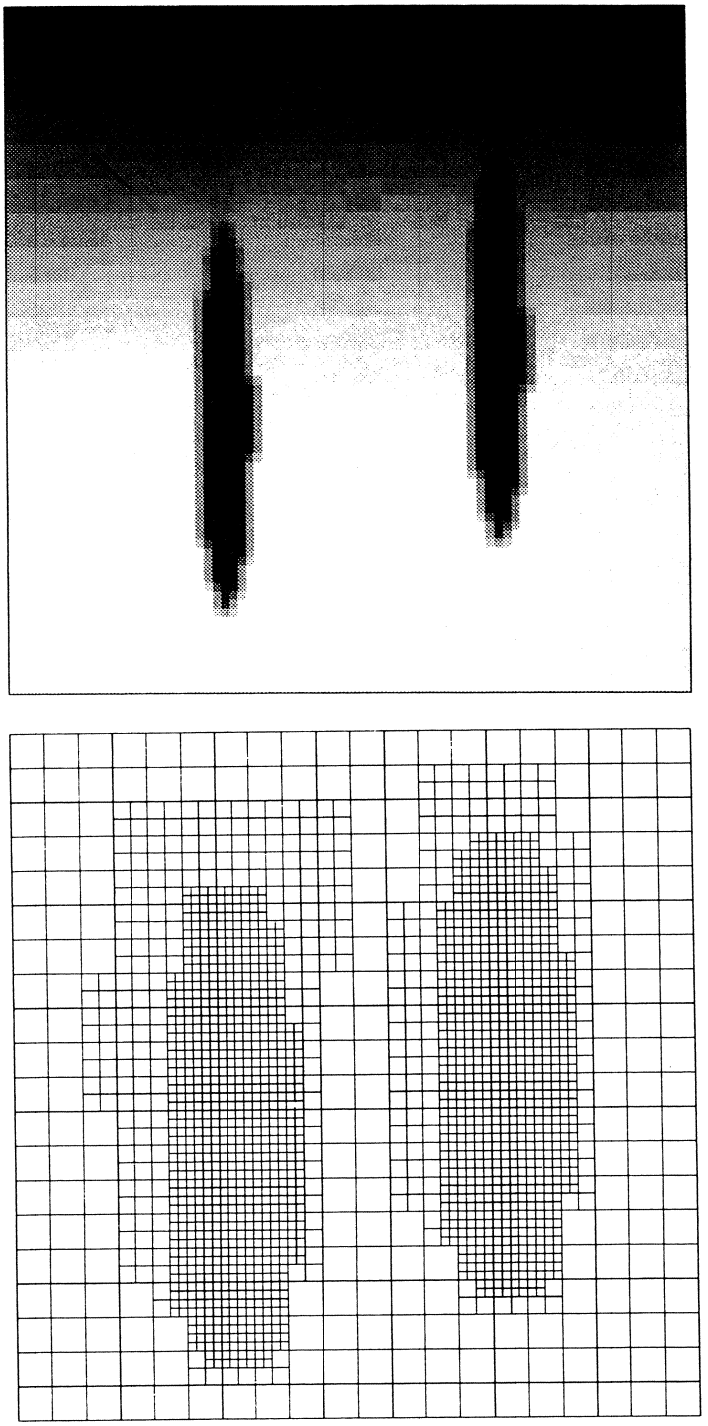

FIGURE 6.2 Continued. The salt concentration with three grid levels at t=2000.

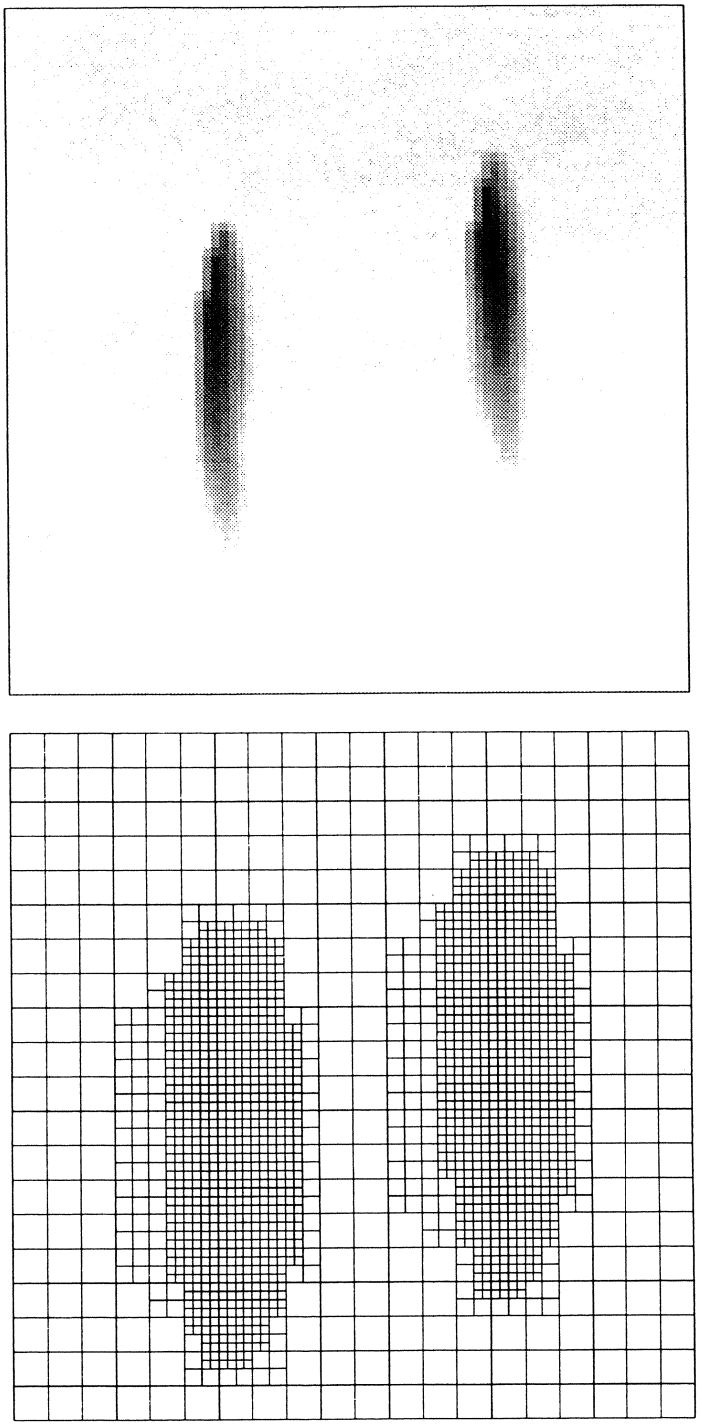

FIGURE 6.2 Continued. The salt concentration with three grid levels at t=200000.

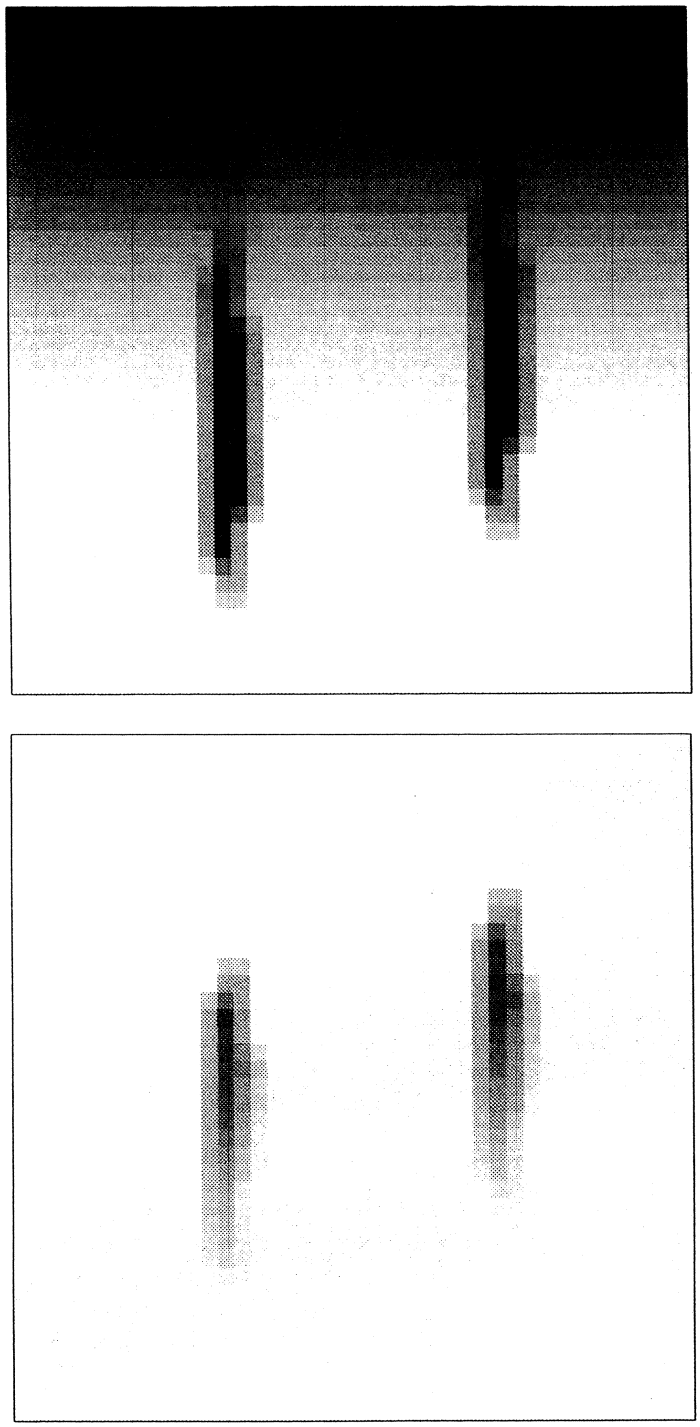

FIGURE 6.3 The salt concentration with 40 × 40 grid t =2000 and t =200000.

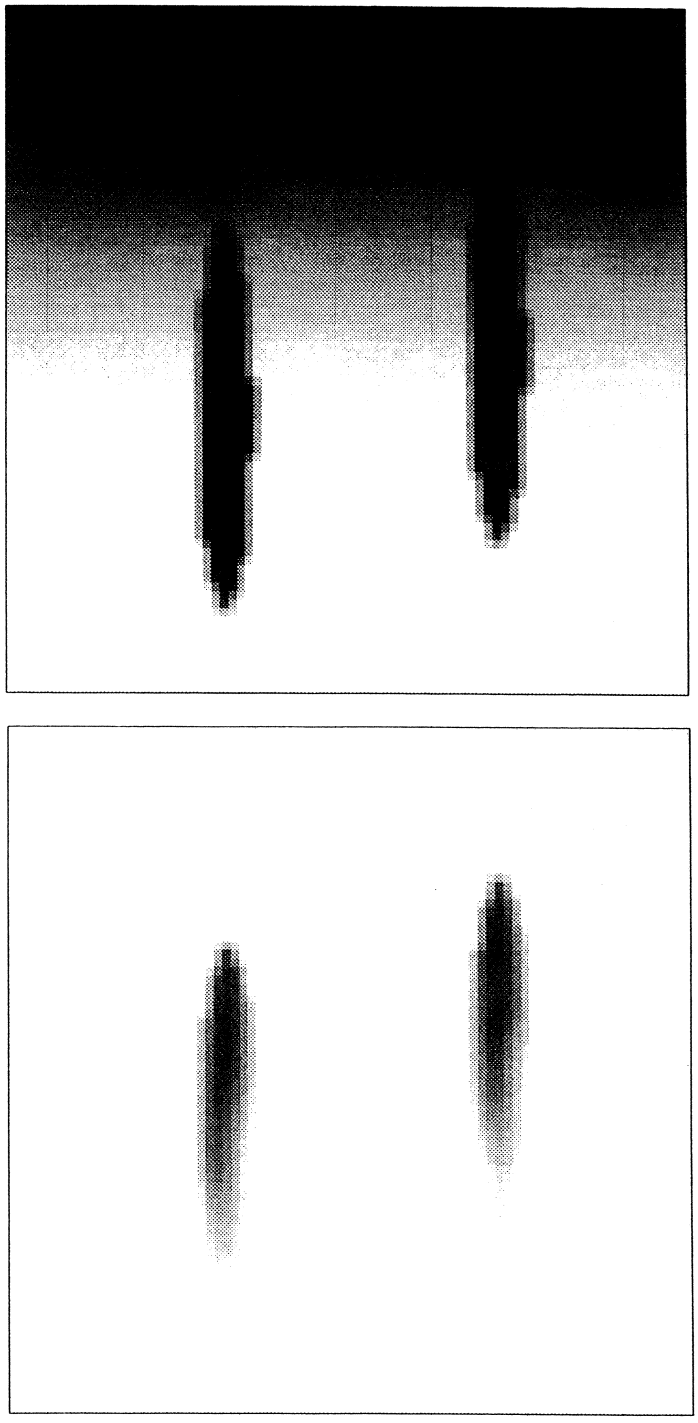

FIGURE 6.3 Continued. The salt concentration with 80 × 80 grid t=2000 and t=200000.

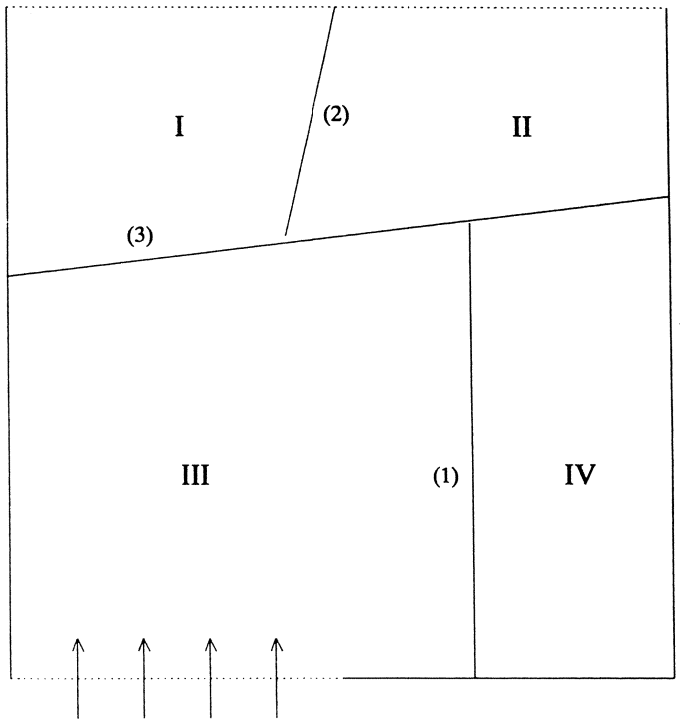

FIGURE 6.4 The vertical column of problem 2.

$$
\begin{aligned}
&\text{I:} && k = 10^{-13}\,m^2, && \alpha_l = 0.008\,m, && \alpha_t = 0.0016\,m,\\
&\text{II:} && k = 10^{-15}\,m^2, && \alpha_l = 0.005\,m, && \alpha_t = 0.0010\,m,\\
&\text{III:} && k = 10^{-10}\,m^2, && \alpha_l = 0.010\,m, && \alpha_t = 0.0020\,m,\\
&\text{IV:} && k = 10^{-13}\,m^2, && \alpha_l = 0.008\,m, && \alpha_t = 0.0016\,m.
\end{aligned}
\tag{6.6}
$$

The interfaces are defined as:

$$
\begin{aligned}
&1: && x = 0.7\,m,\\
&2: && x = 0.3\,m + 0.2 \times y,\\
&3: && y = 0.6\,m + 0.1 \times x.
\end{aligned}
\tag{6.7}
$$

Again, saturated brine is injected into the column at the opening in the bottom and a steep front in the salt concentration will develop, moving slowly towards the top of the column. The front appears to move completely past the interface on its righthand side. So, there is almost no penetration of salt into region IV and a very sharp transition arises at the interface between III from IV. Also in this case, the front smooths out while moving. Later on the front will move from region III to I. As the salt moves from I to II, it also seems to penetrate II from III. At the same time the salt enters IV, first at its top left corner and later at the entire interface, seperating IV from III. Steady state is reached when eventually the saturated brine has spread out

over the entire domain.

We have followed the same approach as in problem 1, that is, we computed the solution to this problem with two and three grid levels, of which the coarsest a 20 x 20 grid. Again we have chosen *TOLT* = 0.1 and *TOLS* = 0.25 for both cases. The salt concentration at two times is shown in Figure 6.5 together with the associated grid levels. We have computed the solution on a single uniform 40 x 40 and 80 x 80 grid for comparison. The salt concentration computed on these grids are shown in Figure 6.6. Just like in problem 1, we see very little difference between the adaptive grid solution and the uniform grid solution. In Figure 6.7, we shown 3D plots of the pressure and the salt concentration at t=4000 s, computed on the 40 x 40 grid. The kinked solution profile at some interfaces are clearly visible here. Note that at the interface between III and IV there is a sharp kink in the salt concentration and no visible kink in the pressure.

# grid levels	# acc. time steps	# rej. time steps	# Newton failures
2	290	2	1
3	318	3	1

TABLE 6.1. Example problem 2. # of accepted timesteps, rejected time steps and Newton failures of the adaptive grid computations.

grid level	# Newton it.	# jacobians
1	973	294
2	1051	291

TABLE 6.2. Example problem 2. # of Newton iterations and jacobians, needed for the two grid computation.

grid level	# Newton it.	# jacobians
1	1074	322
2	1175	326
3	1189	318

TABLE 6.3. Example problem 2. # of Newton iterations and jacobians, needed for the three grid computation.

The Tables 6.1, 6.2 and 6.3 contain information about the time stepping and Newton iteration process of both adaptive grid computations. When the time stepsize needs to be decreased for Newton-convergence reasons, we call this a Newton failure. We can see that for both the two and the three grid level computation the number of Jacobians is only moderately larger than the number of accepted time steps. So, we can draw the conclusion that our Newton iteration strategy performs well in combination with the local uniform grid refinement method.

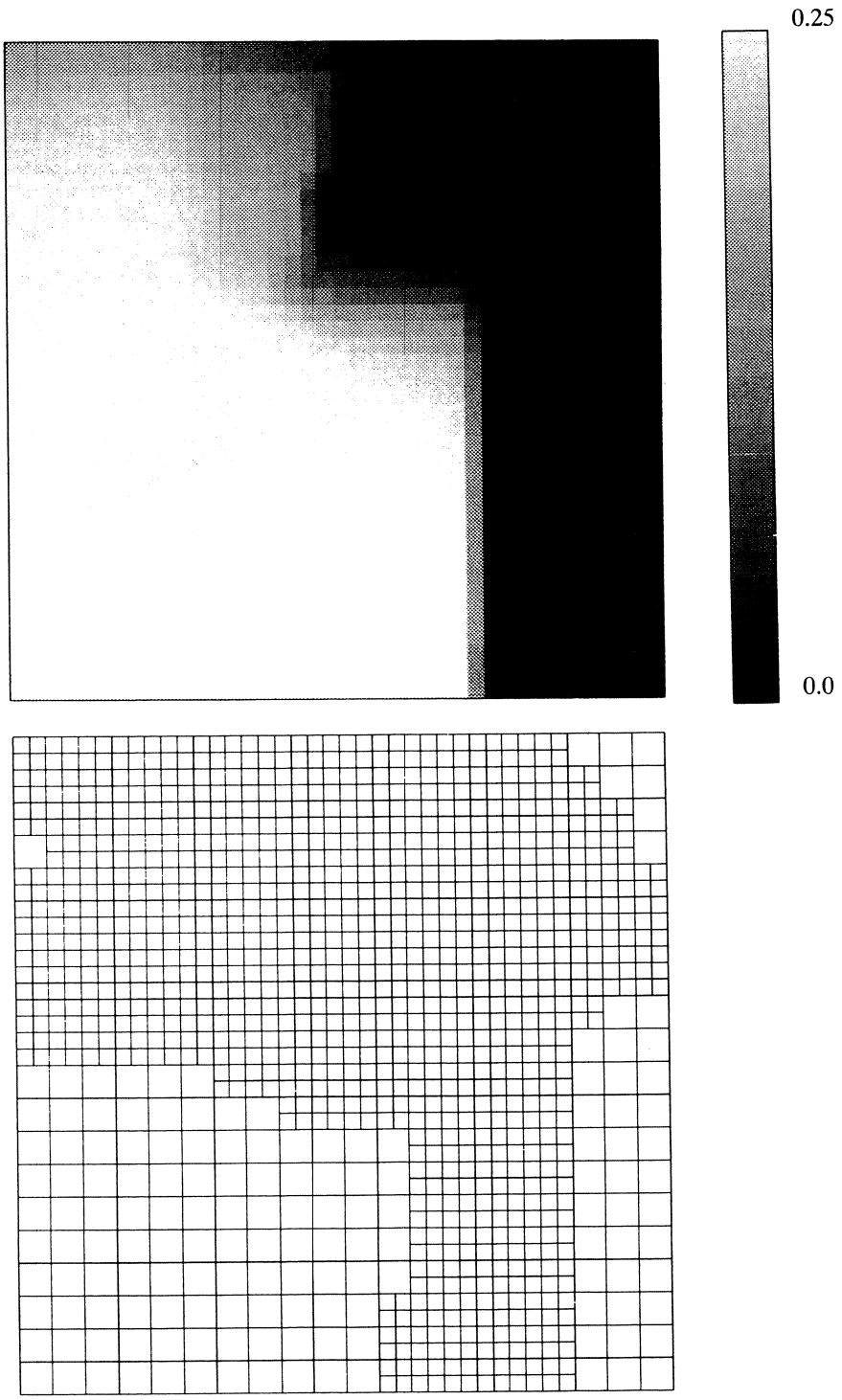

FIGURE 6.5 The salt concentration with two grid levels at t=4000.

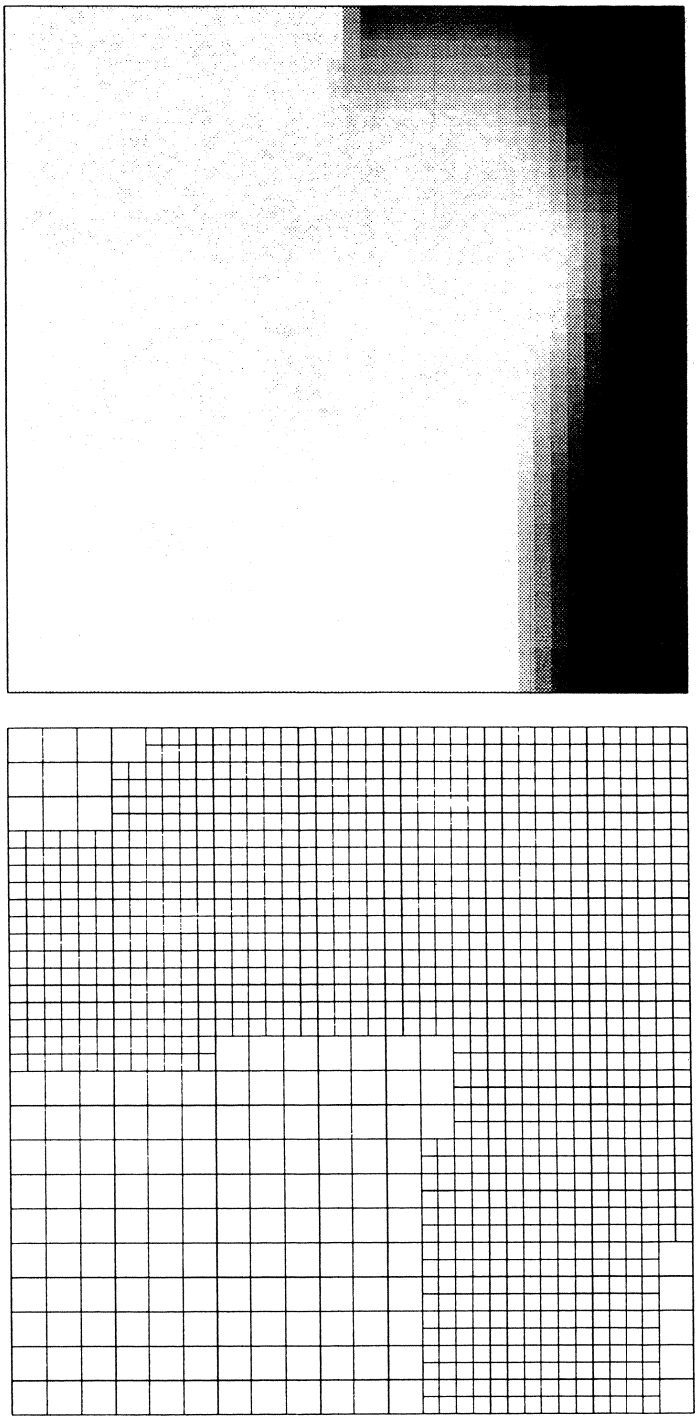

FIGURE 6.5 Continued. The salt concentration with two grid levels at t=60000.

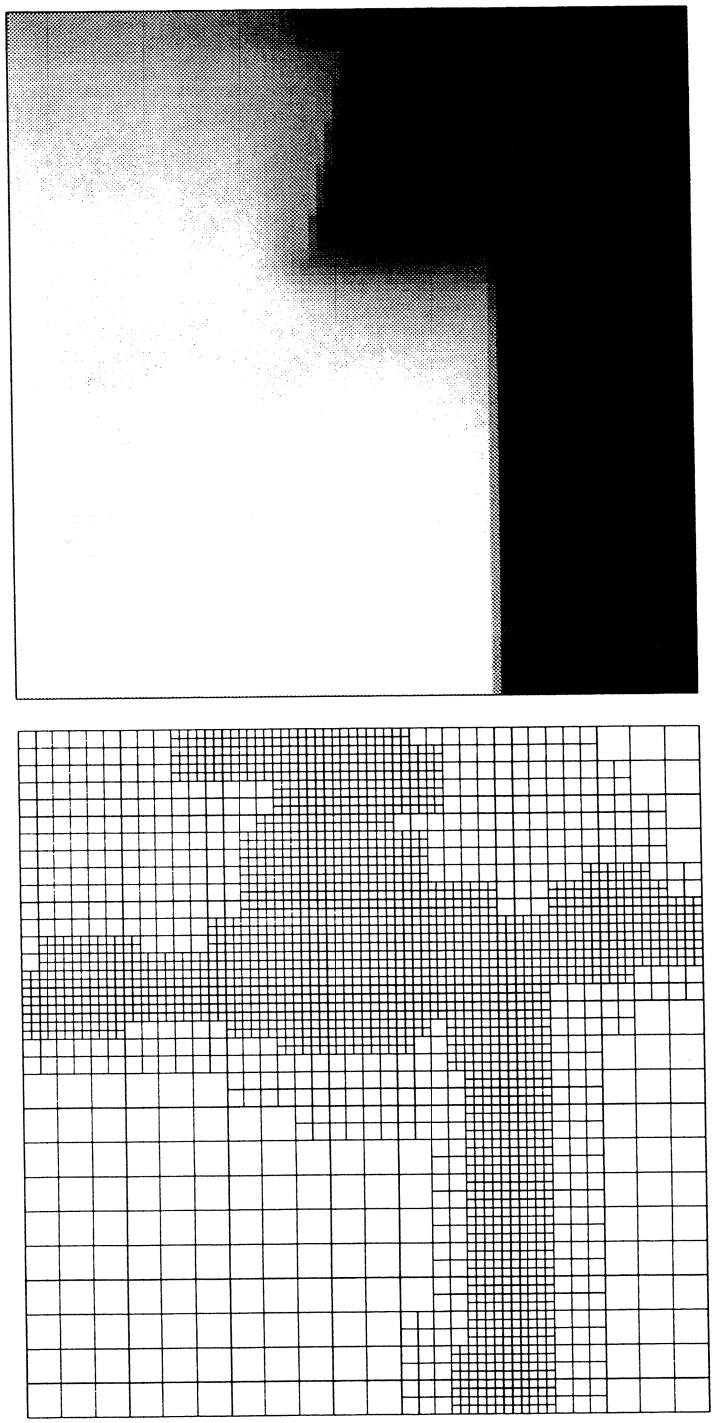

FIGURE 6.5 Continued. The salt concentration with three grid levels at t=4000.

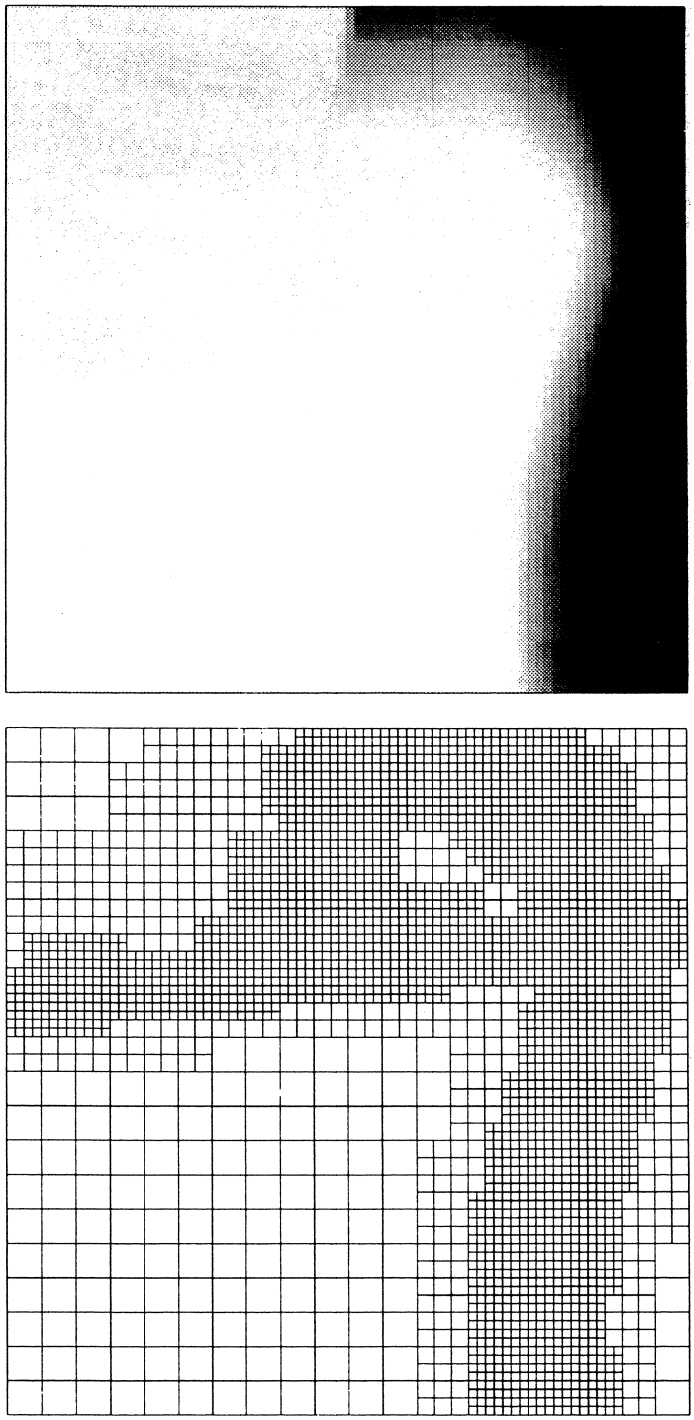

FIGURE 6.5 Continued. The salt concentration with three grid levels at t =60000.

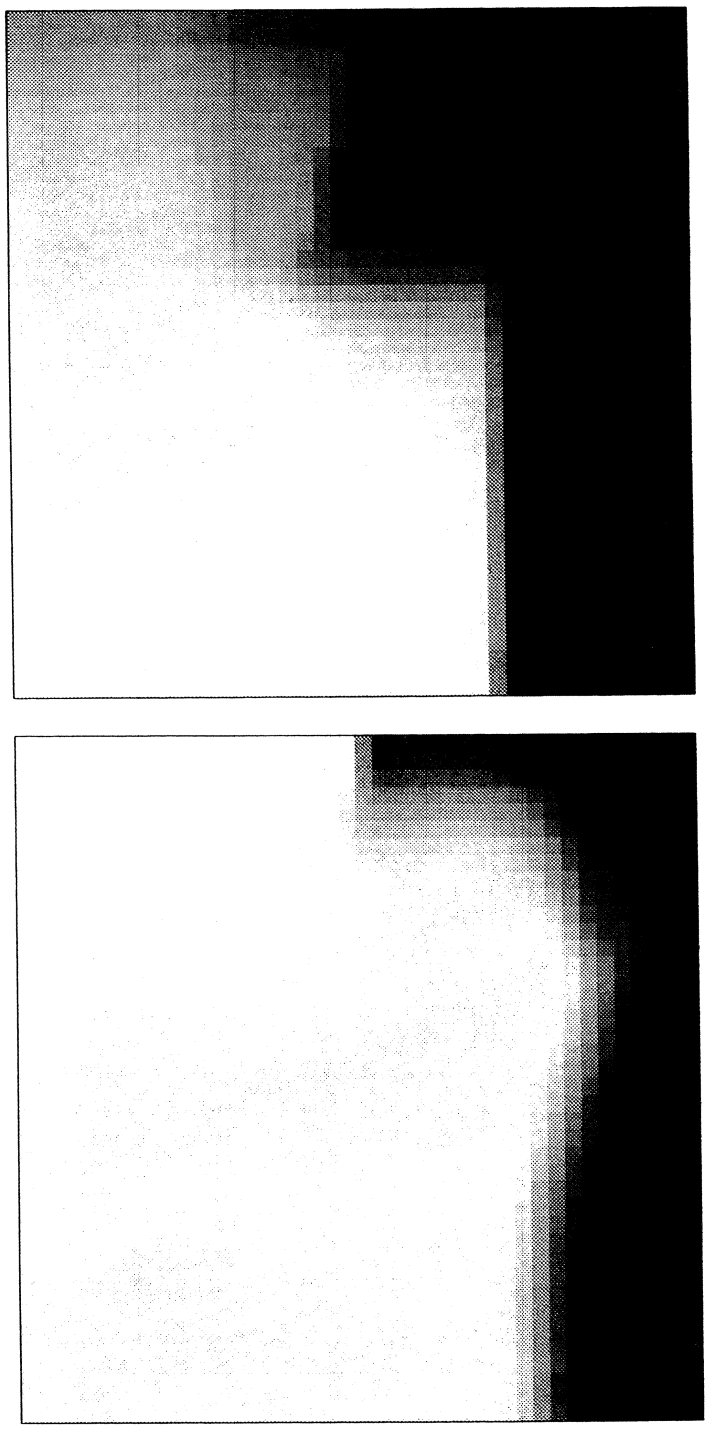

FIGURE 6.6 The salt concentration with 40 × 40 grid t =4000 and t =60000.

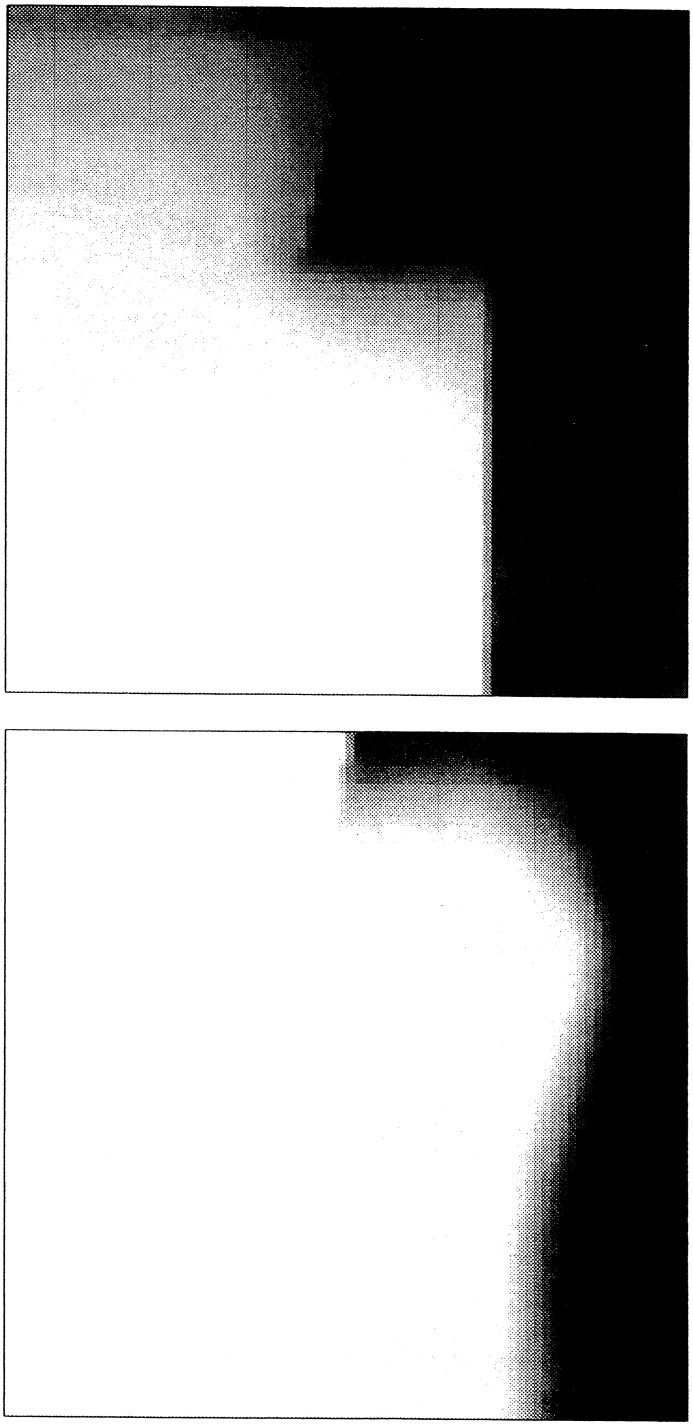

FIGURE 6.6 Continued. The salt concentration with 80 × 80 grid t=4000 and t=60000.

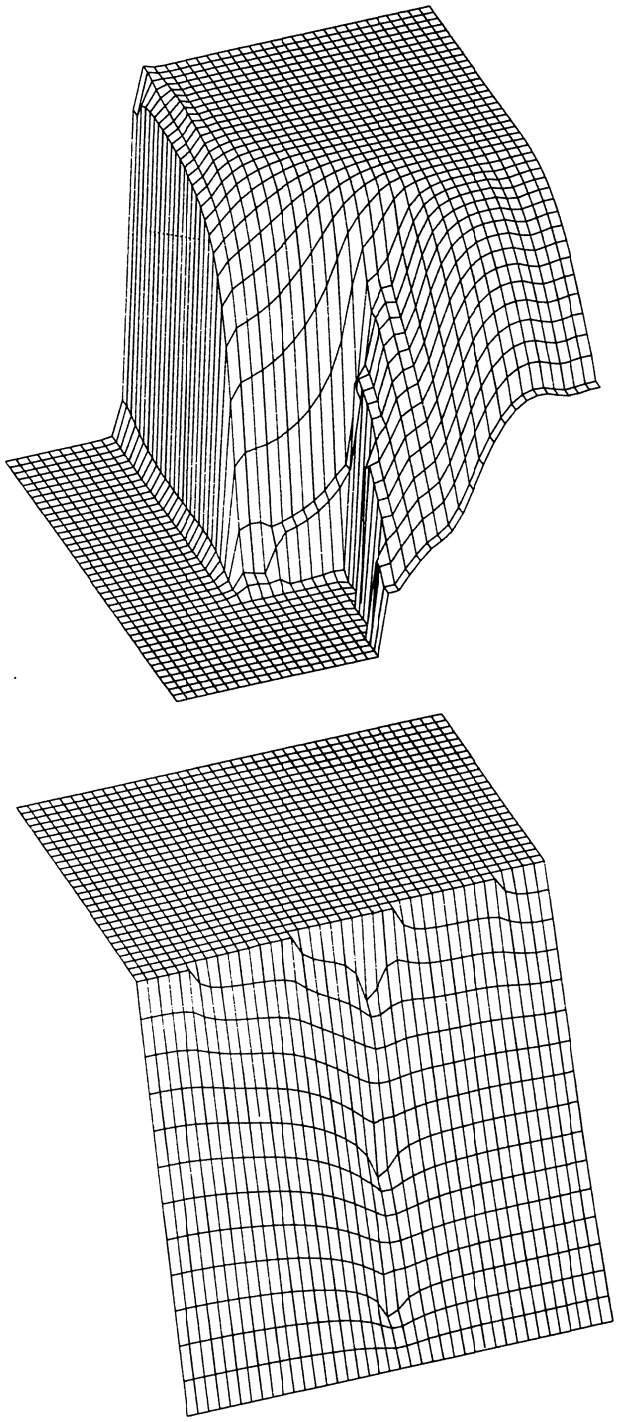

FIGURE 6.7 The salt concentration and the pressure at t =4000 with a 40 × 40 grid.

7. Summary and Concluding Remarks

In this paper we have discussed the application of an adaptive grid method, based on local uniform grid refinement, to brine transport problems in porous media with inhomogeneities. For such problems, where locally steep fronts in the salt concentration occur, adaptive grid methods are valuable, since they can compute a solution to these problems with locally the same resolution as on a very fine uniform grid, but with much less computational costs.

In natural circumstances, the soil parameters of the porous medium can change very suddenly from one region to another. At these sudden changes the solution profile may be kinked. Consequently, interface conditions, implying continuity of fluxes across these interfaces and involving only one-sided difference schemes, have been applied here to obtain consistent numerical approximations. The "numerical" interfaces are supposed to coincide with grid cell edges. Further, compared to our previous publication [11], the modified Newton method for solving the systems of nonlinear equations has been adapted to increase the robustness of the code. The results of the two test problems indicate that the solution computed with the local uniform grid refinement method is of the same accuracy as that of the associated uniform grid solution. The results also indicate that the adapted modified Newton method and the linear iterative solver Bi-CGSTAB work satisfactorily.

To sum up, we consider our results as satisfactory but nevertheless we think that two warnings are appropriate here. First, although the adaptation of the modified Newton method has improved the robustness of the code considerably, there still is a possibility that the code breaks down, simply because the (partially) interpolated initial guess for the iteration process (cf. Section 5) is too far away from the solution of the system of nonlinear equations at hand. A remedy to this could be to create finer grids always at interfaces, whether this is necessary, regarding the space error monitor values, or not. The second warning has to do with the dispersion tensor. In [6] we show that the mathematical formulation of the dispersion tensor can cause serious difficulties for the iterative solution of the systems of nonlinear equations, even to the extent that a code can break down. This occurs when the velocity exhibits large changes in direction during the iterative solution process. This is likely to occur when the velocity becomes relatively small, for instance, near stagnation points or near the kernel of a vortex. When brine transport problems in porous media with inhomogeneities are solved, stagnation points or points where the velocities are very small are not uncommon, so there is a real danger that the problems above occur.

References

1. J.E. Dennis and R.B. Schnabel (1983). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall Series in Computational Mathematics.
2. J.J. Dongarra and E. Grosse. Distribution of Mathematical Software via Electronic Mail, *Communications of the ACM*, 30, 403-407 (netlib@research.att.com).
3. S.M. Hassanizadeh. Experimental Study of Coupled Flow and Mass Transport: A Model Validation Exercise, in *Calibration and Reliability in Groundwater Modeling*, ed. K. Kovar, IAHS Publication No. 195, Wallington, Oxfordshire, U.K..
4. S.M. Hassanizadeh and T. Leijnse (1988). On the Modelling of Brine Transport in Porous Media, *Water Resources Research*, 24, 321-330.
5. W. Schönauer and E. Schnepf (1987). Software Considerations for the Black Box Solver FIDISOL for Partial Differential Equations, *ACM Trans. on Math. Softw.*, 13, 333-349.
6. R.A. Trompert (1992). *A Note on Singularities Caused by the Hydrodynamic Dispersion Tensor*, (in preparation), Centre for Mathematics and Computer Science, Amsterdam.
7. R.A. Trompert (1992). *Local Uniform Grid Refinement and Systems of Coupled Partial Differential Equations with and without Time Derivatives*, (in preparation), Centre for Mathematics and Computer Science, Amsterdam.
8. R.A. Trompert and J.G. Verwer (1990). *Analysis of the Implicit Euler Local Uniform Grid Refinement Method*, Report NM-R9011 (to appear in SIAM J. Sci. Stat. Comput.), Centre for Mathematics and Computer Science, Amsterdam.

9. R.A. TROMPERT and J.G. VERWER (1990). *Runge-Kutta Methods and Local Uniform Grid Refinement,* Report NM-R9022 (to appear in Math. Comp.), Centre for Mathematics and Computer Science, Amsterdam.
10. R.A. TROMPERT and J.G. VERWER (1991). A Static-Regridding Method for Two Dimensional Parabolic Partial Differential Equations, *Appl. Numer. Math.*, 8, 65-90.
11. R.A. TROMPERT and J.G. VERWER (1992). *Computing Brine Transport in Porous Media with an Adaptive-Grid Method,* Report NM-R9201 (to appear in Int. J. Numer. Meths. in Fluids), Centre for Mathematics and Computer Science, Amsterdam.
12. J.G. VERWER and R.A. TROMPERT (1991). *Local Uniform Grid Refinement for Time-Dependent Partial Differential Equations,* Report NM-R9105, Centre for Mathematics and Computer Science, Amsterdam.
13. J.G. VERWER and R.A. TROMPERT (1992). *Analysis of Local Uniform Grid Refinement,* Report NM-R9211, Centre for Mathematics and Computer Science, Amsterdam.
14. J.G. BLOM, J.G. VERWER and R.A. TROMPERT (1992). *A Comparison between Direct and Iterative Methods to solve the Linear Systems arising from a Time-Dependent 2D Groundwater Flow Model,* Report NM-R9205, Centre for Mathematics and Computer Science, Amsterdam.
15. H.A. VAN DER VORST (1992). BI-CGSTAB: A Fast and Smoothly Converging Variant of BI-CG for the Solution of Nonsymmetric Linear Systems, *SIAM J. Sci. Stat. Comput.*, 13(2), 631-644.