INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 15, 175–191 (1992)

APPLICATION OF A MOVING GRID METHOD TO A CLASS OF 1D BRINE TRANSPORT PROBLEMS IN POROUS MEDIA

P. A. ZEGELING* AND J. G. VERWER

Centre for Mathematics and Computer Science (CWI), PO Box 4079, NL-1009 AB Amsterdam, The Netherlands

AND

J. C. H. VAN EIJKEREN

National Institute of Public Health and Environmental Hygiene (RIVM), PO Box 1, NL-3720 BA Bilthoven, The Netherlands

SUMMARY

The background of this paper is the study of transport of pollutants by groundwater flow when released from a repository in a rock salt formation. Flow in regions surrounding such formations may be strongly influenced by variations in salt concentrations, a factor requiring special attention in the development of realistic mathematical models for predicting transport of pollutants. Indispensable for this development are advanced numerical methods. The aim of this paper is to illustrate the application of such a method to a class of non-linear brine transport problems in one space dimension. Our method is based on the method of lines for solving time-dependent partial differential equations. The method is of the finite difference type, implicit and thus applicable to wide classes of (one-space-dimensional) partial differential equation systems. The main feature of the method, however, is that it can automatically move the spatial grid for evolving time and thus is able to refine the grid in regions with large, special transitions. The grid refinement has proven to be a very valuable facility in the numerical modelling of brine transport problems involving low and high salt concentrations. From the user's point of view an additional advantage of the moving grid method is that it can be implemented in advanced, user-oriented method-of-lines software packages based on implicit stiff ODE solvers. In the brine transport application discussed here we have used the package SPRINT.

KEY WORDS Fluid flow/solute transport in porous media Groundwater flow Moving grid Finite difference method Method of lines

1. INTRODUCTION

The subject of this paper originates from the problem of disposal of hazardous waste, e.g. high-level radioactive waste, in salt formations. The most probable mechanism for release of these wastes to the biosphere is by transport via groundwater. Existing standard mathematical models for the study of groundwater flow and brine transport assume that the salt concentration is less than or equal to seawater concentration. This, however, is not true for flows in the vicinity of rock

* This work has been carried out in connection with a joint CWI/Shell project on 'Adaptive grids'. For this project Paul Zegeling has received support from the Netherlands Foundation for the Technical Sciences (STW), Contract CWI 59.0922.

0271-2091/92/140175-17$13.50

Received July 1991
Revised November 1991

salt formations. In the vicinity of these formations, e.g. salt domes, the salt concentration may become very large and in fact to an extent that the groundwater flow is really influenced by the salt concentration. Recent theoretical and experimental hydrological studies indicate that for such high-concentration situations the basic equations of flow and transport involved need to be modified.[1,2] This incurs a significant effort in numerical modelling since the partial differential equations (PDEs) which show up cannot be solved by analytical means. The content of the current paper has its origin in part of these numerical modelling studies.

We discuss the application of a numerical moving grid method, originally developed for general time-dependent PDEs in one space dimension, to a specific class of non-linear brine transport problems borrowed from Reference 3. Our purpose is twofold. Firstly, while focusing on the application, we wish to show that this numerical method is a valuable tool for modelling non-linear (brine) transport problems in one space dimension, specifically so for problems having solutions with rapid transitions, such as a solute front transported in the soil or a sharp fresh–salt water interface. Secondly, while now focusing on the numerical analysis aspects, we wish to show that for the class of transport problems chosen, the grid movement approach is successful and may provide a notable improvement compared to the more traditional approach of time stepping on a fixed spatial grid.

The numerical method is based on the method-of-lines (MOL) approach for solving time-dependent PDEs (see e.g. Reference 4, Chap. 10, and Reference 5). The method is of the finite difference type, implicit and thus applicable to wide classes of one-space-dimensional PDE systems. In addition, the main feature of the method is that for evolving time it automatically refines the spatial grid in regions with large spatial transitions. Since it is a Lagrangian-type method, in many cases of practical interest the grid movement also softens the solution behaviour in time, so that larger time steps can be taken than on a fixed spatial grid. The actual moving grid algorithm underlies the principle of spatial equidistribution and is provided with appropriate grid regularization procedures to cater for smooth grid trajectories. The principal ideas for this regularization emanate from Reference 6 and a further comprehensive discussion of the complete moving grid algorithm can be found in Reference 7 (see also Reference 8 and references cited therein).

An advantage of the moving grid method is that it can be implemented in most of the MOL software packages based on sophisticated implicit stiff ODE/DAE solvers. We mention the BDF solvers developed by Gear, Byrne, Hindmarsh, Petzold and others (see e.g. Reference 9). In the brine transport problem application we have used the FORTRAN package SPRINT.[10] SPRINT is a package developed for solving general algebraic, ordinary and partial differential equations. So far SPRINT has been used mainly for one-space-dimensional problems, since its core is formed by implicit stiff ODE/DAE solvers (of BDF type). In Reference 11 SPRINT has been provided with a software interface based on the moving grid method considered here. This MGI (moving grid interface), being an extension of the fixed grid interface based on Reference 12, is a most convenient tool for researchers who wish to concentrate on modelling their physics, since it automatically carries out the spatial discretization, thus relieving them from numerical choices to be made and saving programming time. The use of MGI merely requires that the mathematical problem be formulated in terms of FORTRAN statements. Consequently, both the spatial discretization and the temporal integration can then be left to the package and the user only has to set to some numerical control parameters, such as a local tolerance parameter for the numerical integration in time, the number of points for the spatial discretization and some parameters controlling the grid movement. In the experiments reported here we have used the MGI from Reference 11.

Section 2 is devoted to the moving grid method. An outline is given of important properties and principles of this method. The class of fluid flow/salt transport problems we focus on is discussed in Section 3. The physical properties involved here are advection–dispersion; in the case of dominant advection, solutions with rapid spatial and temporal transitions arise. In Section 4 we present results of numerical tests, emphasizing the occurrence of the rapid transitions and the use of the moving grid method. Section 5 is devoted to concluding remarks.

2. THE MOVING GRID ALGORITHM

2.1. *The moving grid algorithm*

We will present the algorithm along the lines of the numerical method-of-lines (MOL) approach for solving time-dependent PDEs. Consider an abstract Cauchy problem for a system of PDEs in one space dimension,

$$\frac{\partial u}{\partial t}=f(u), \quad x_{\mathrm{L}}<x<x_{\mathrm{R}}, \quad t>0, \tag{1}$$

where $u=u(x, t)$ and f is a spatial operator of at most order two. We do not discuss boundary conditions here since these are dealt with in the usual way. It is assumed that the solution u has (a sufficient number of) finite temporal and spatial derivatives and these are allowed to be very large. We thus focus on problems possessing solutions u with very large spatial and temporal variations but do not consider problems with genuine discontinuous solutions.

The discretization of the PDE system is carried out in two stages. In the first stage $f(u)$ is discretized on a selected space mesh, which converts (1) into a Cauchy problem for an ODE system. The second stage then deals with the numerical integration in time of this semidiscrete system. Let us discuss the first stage, which takes place here in a moving reference frame. First we choose N time-dependent grid points $X_i(t)$, $1\leqslant i\leqslant N$, defining the space grid

$$X: \quad x_{\mathrm{L}}=X_0<\ldots<X_i(t)<X_{i+1}(t)<\ldots<X_{N+1}=x_{\mathrm{R}}, \quad t\geqslant 0. \tag{2}$$

As yet the trajectories $X_i(t)$ are unknown, but they are assumed to be (sufficiently often) differentiable. Next, along each trajectory $x(t)=X_i(t)$ we introduce the total derivative

$$u'=x'u_x+u_t=X_i'u_x+f(u), \quad 1\leqslant i\leqslant N, \tag{3}$$

and spatially discretize the space operators $\partial/\partial x$ and f so as to obtain the Lagrangian semidiscrete system

$$U_i'=X_i'[(U_{i+1}-U_{i-1})/(X_{i+1}-X_{i-1})]+F_i, \quad t>0, \quad 1\leqslant i\leqslant N. \tag{4}$$

Here U_i and F_i represent the semidiscrete approximations to their exact counterparts u and f at the point $(x, t)=(X_i(t), t)$. The finite difference replacement for f is in principle still free to be choosen. We discuss this in Section 2.3. Note that we use the standard, central finite difference approximation for u_x. Also note that the boundary values U_0 and U_{N+1} are to be defined from the semidiscretization of the physical boundary conditions.

The internal grid points X_i are still free to be choosen. The purpose is to let them move automatically such that X becomes fine in regions of high spatial activity and coarse in regions where the spatial variation is low. One way to accomplish this is to apply *equidistribution*. For this purpose we introduce the point concentration values[6]

$$n_i=(\Delta X_i)^{-1}, \quad \Delta X_i=X_{i+1}-X_i, \quad 0\leqslant i\leqslant N, \tag{5}$$

and the equidistribution equation

$$n_{i-1}/M_{i-1} = n_i/M_i, \quad 1 \leqslant i \leqslant N, \tag{6}$$

where $M_i \geqslant \sqrt{\alpha} > 0$ represents a so-called monitor value that reflects the variation in space. The parameter $\alpha > 0$ serves to ensure that M_i remains positive. Trivially, n_i is proportional to M_i. Thus the equidistribution idea assumes that if some measure of the spatial error is available, here taken to be represented by M_i, then a good choice for the grid X would be one for which the error is equidistributed over X.

In applications the monitor M_i is usually taken to be a semidiscrete replacement of a solution functional $m(u)$ containing one or more spatial derivatives (note that the variables U_i and X_i are still time-continuous). Lest we miss the obvious, the choice of monitor is important because it plays a decisive role in the actual local grid refinement. Following References 7, 8 and 13, in the present implicit MOL approach we advocate the first-derivative monitor

$$M_i = \left(\alpha + \frac{1}{\text{NPDE}} \sum_{j=1}^{\text{NPDE}} \beta_j (\Delta U_i^j)^2 (\Delta X_i^j)^{-2} \right)^{1/2}, \quad \Delta U_i^j = U_{i+1}^j - U_i^j, \tag{7}$$

where NPDE denotes the number of PDEs in (1) and U_i^j is the jth component of the vector variable U_i. Note that at a given point of time, (7) is a semidiscrete replacement of $m(u) = (\alpha + \|u_x\|^2)^{1/2}$, where $\|.\|$ is the weighted Euclidean norm involved. With $\alpha = 1$ we have the well known arc-length monitor which places grid points along uniform arc-length intervals. We use α as a parameter which can eventually be used for tuning purposes. In fact, the main purpose of this tuning parameter is to keep the monitor values positive, saying that a small value of α suffices. Clearly, α should not be taken too 'large' compared to the maximum of $\|u_x\|^2$, since this would result in a uniform grid, approximately. The weighting parameters β_i in (7) serve to make it possible to let certain components dominate the equidistribution. This may be desirable in the case of a badly scaled problem, for example. The actual choice of the monitor parameters $\alpha, \beta_1, \ldots, \beta_{\text{NPDE}}$ will influence the outcome of a numerical simulation and therefore their optimal choice is problem-dependent. On the other hand, our experience is that with the monitor (7) the method is quite robust and a bad choice merely affects the resulting accuracy. This means that given a well described problem class such as the brine transport problems, a close-to-optimal choice is normally easy to determine.

2.2. *Grid smoothing*

The Cauchy ODE problem resulting from the first MOL stage thus reads

$$U_i' = X_i'[(U_{i+1} - U_{i-1})/(X_{i+1} - X_{i-1})] + F_i, \quad t > 0, \quad 1 \leqslant i \leqslant N, \tag{8a}$$

$$n_{i-1}/M_{i-1} = n_i/M_i, \quad t > 0, \quad 1 \leqslant i \leqslant N. \tag{8b}$$

After prescribing initial data for U_i and X_i, $1 \leqslant i \leqslant N$, and the boundary values U_0 and U_{N+1} from a semidiscretization of the physical boundary conditions, system (8) can be numerically integrated in time so as to obtain the final fully discrete solution on the moving grid X. However, since (8b) prescribes X in an implicit way in terms of the unknowns U_i, there is little control over the grid movement. For example, it may happen that the grid distance ΔX_i varies extremely rapidly over X or that for evolving time the trajectories $X_i(t)$ tend to oscillate. A too large variation in ΔX_i may be detrimental to spatial accuracy, and temporal grid oscillations do hinder the numerical time stepping since the grid trajectories are computed automatically by numerical integration. Following References 6–8, we therefore employ two so-called grid-smoothing procedures, one for generating a spatially smooth grid and the other for avoiding temporal grid oscillations. This involves a modification of the grid equation system (8b).

The modified grid equation system is given by

$$\left(\boldsymbol{n}_{i-1}+\tau\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{n}_{i-1}\right)\Big/M_{i-1}=\left(\boldsymbol{n}_i+\tau\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{n}_i\right)\Big/M_i,\quad t>0,\quad 1\leqslant i\leqslant N, \tag{9}$$

where $\boldsymbol{n}_i=n_i-\kappa(\kappa+1)(n_{i+1}-2n_i+n_{i-1})$, with $n_{-1}=n_0$ and $n_{N+1}=n_N$. We note in passing that in the actual implementation n_i is replaced by $(\Delta X_i)^{-1}$ and n_i' by $-\Delta X_i'/(\Delta X_i)^2$. The modification thus results in a five-point-coupled, time-dependent grid equation system. A consequence of the grid smoothing is that in addition to the monitor parameters $\alpha, \beta_1, \ldots, \beta_{\mathrm{NPDE}}$, two new grid parameters have been introduced, namely κ and τ.

The parameter $\kappa\geqslant 0$ is connected with the spatial grid smoothing. Any grid X solving (9) satisfies

$$\frac{\kappa}{\kappa+1}\leqslant\frac{n_{i-1}}{n_i}\leqslant\frac{\kappa+1}{\kappa}, \tag{10}$$

showing that we have control over the variation in ΔX_i. Through κ we can control grid clustering and grid expansion. Loosely speaking, the monitor function still determines the shape of X, and κ the level of clustering. Note that the extreme value $\kappa=\infty$ yields a uniform grid. Of importance is to emphasize that for a given number of points N and any given distribution of monitor function values M_i, κ determines the minimal and maximal interval lengths (see e.g. Reference 7). In actual application the minimum should of course be related to the expected small-scale features in the sought solution. In our application we choose $\kappa=2$. With this value of κ we not only obtain a rather modestly graded space grid but also keep a sufficient number of points within the actual transitions of $\partial u/\partial x$.

The parameter $\tau\geqslant 0$ is connected with the temporal grid smoothing and serves to act as a delay factor for the grid movement. More precisely, the introduction of the temporal derivative of the grid X forces the grid to adjust over a time interval of length τ from old to new monitor values, which provides a tool for suppressing grid oscillations and hence to obtain a smoother progression of $X(t)$. However, choosing τ too large will result in a grid X that lags too far behind any moving steep spatial transition. In fact, it can be shown that for $\tau\to\infty$ a non-moving grid results. In situations where temporal grid smoothing is really advisable, one should therefore choose τ not too large. For practical purposes a good choice is one which is close to the minimal temporal step size taken in the numerical integration, so that the influence of past monitor values is felt only over one or a few time steps.

2.3. Integration in time

We have now semidiscretized (1) on a moving grid. The semidiscrete formulation consists of the combined equations (8a)–(9), where $n_i=(\Delta X_i)^{-1}$ and $n_i'=-\Delta X_i'/(\Delta X_i)^2$ are used to convert the dependence on the point concentration values into a 'natural' dependence on the grid points X_i. Recall that the boundary values U_0 and U_{N+1} are to be defined from the spatial discretization of physical boundary conditions. The equations can be written in the linearly implicit ODE system form

$$A(Y)Y'=L(t,Y),\quad t>0,\quad Y(0)\text{ given}, \tag{11}$$

where Y assumes the natural ordering of unknowns U_i^j and X_i, i.e. $Y=(\ldots, U_i^1,\ldots, U_i^{\mathrm{NPDE}}, X_i,\ldots)$. Form (11) is a standard format for various well known stiff ODE/DAE solvers. Note that without temporal grid smoothing, (9) is of purely algebraic form, so that (11) then becomes a DAE system. The numerical results in this paper have been obtained with the LSODI-based BDF solver of the SPRINT package. A similar solver is DASSL,[9] which we have also applied elsewhere.[7] It is of interest to note that in our moving grid application these solvers are employed in essentially the same way as in the conventional non-moving MOL approach.

2.4. A moving grid interface

Since the integration in time is done automatically by the stiff integrator, it makes sense to also automize the spatial discretization of the PDE operator with its boundary conditions. This is particularly attractive for researchers who wish to concentrate on modelling their physics, since it saves programming time and relieves them from numerical choices to be made. Such a FORTRAN interface for use with the moving grid method has been developed in Reference 11. We have also used this interface, called MGI, in the tests reported in Section 4.

MGI is an extension of the fixed grid interface from Reference 12, which is available in the SPRINT package. The discretization is based on a central second-order finite volume scheme and covers the following PDE system:

$$\sum_{k=1}^{\mathrm{NPDE}} C_{jk}(x,t,u,u_x)\frac{\partial u^k}{\partial t}=x^{-m}\frac{\partial}{\partial x}(x^m R_j(x,t,u,u_x))-Q_j(x,t,u,u_x),\quad x_{\mathrm{L}}<x<x_{\mathrm{R}},\quad t>0. \tag{12}$$

Index j runs from 1 to NPDE, u^k is the kth component of the vector-valued function u, and R_j and Q_j can be thought of as flux and source or sink terms respectively. The parameter m serves to cover polar co-ordinates ($m=1$ or 2). In our present application we work in Cartesian co-ordinates and thus $m=0$. The coefficient functions C_{jk}, R_j and Q_j are assumed to be at least continuous. The boundary conditions should fit the MGI master form

$$\chi_j(x,t)R_j(x,t,u,u_x)=\gamma_j(x,t,u,u_t,u_x),\quad x=x_{\mathrm{L}},x_{\mathrm{R}}, \tag{13}$$

and the standard initial condition $u(x,0)=u_0$ is assumed. For a description of MGI with its underlying spatial discretization we refer to Reference 11. The fluid flow/salt transport problem fits this master form.

3. THE 1D FLUID FLOW/SALT TRANSPORT PROBLEM

Disposal of radioactive wastes in rock salt formations is being considered as a serious possibility by a number of countries. An integral part of the safety assessment of waste disposal is the study of mathematical models for nuclide transport to the geosphere via groundwater flow. Existing standard models for groundwater flow and salt transport assume that the salt concentration is less than or equal to seawater concentration. In such low-concentration situations the models in use have been sufficiently validated and in many cases of interest the fluid flow and the salt concentration equation can be treated uncoupled. However, for flows in the vicinity of rock salt formations the salt concentration may become high and influence the fluid density to an extent that it effects the fluid flow. On the other hand, salt is transported by the fluid and thus fluid flow and salt transport are mutually coupled. The existing standard models and their uncoupled treatment are then no longer adequate for safety assessment, which makes it interesting to study this intricate situation. Recent theoretical and experimental hydrological studies[1,2] indicate that for such high-concentration situations the basic equations of flow and transport involved need to be modified, which requires a significant effort in numerical modelling. Here the moving grid method enters the scene, because in the high-concentration situations large concentration gradients also prevail, making the use of fixed grid methods inefficient.

In modelling the transport of M solutes by groundwater flow, generally $M+1$ sets of equations appear, i.e. one set for each solute and a set for the flowing fluid.[1] The set for the fluid (brine) constitutes the fundamental balance-of-mass property of the fluid supplemented by a Darcy law expressing conservation of momentum. Similarly, for each solute the associated set constitutes the balance-of-mass property supplemented by conservation of momentum through a Fickian-type law. If temperature changes are important, then an energy equation should be added. Also, if deformation effects of the porous medium and porosity changes are important, then an additional set of equations for the solid phase of the porous medium has to be provided. In the present study we do not consider temperature or deformation effects and assume only one solute, the salt. We thus consider an isothermal, single-phase, two-component saturated flow model in the idealized case of one space dimension. It is further assumed that no external body forces except gravity exist and that the two brine components, water and salt, do not react or adsorb. This specific model, which we have borrowed from an RIVM report,[3] has been selected for demonstration purposes.

The model comprises the following set of equations. For the fluid and salt we have respectively

$$\frac{\partial}{\partial t}(n\rho)+\frac{\partial}{\partial x}(\rho q)=0, \qquad q=-\frac{k}{\mu}\left(\frac{\partial p}{\partial x}+\rho g\right) \tag{14a, b}$$

and

$$\frac{\partial}{\partial t}(n\rho\omega)+\frac{\partial}{\partial x}(\rho\omega q+\rho J)=0, \qquad J=-\lambda|q|\frac{\partial\omega}{\partial x} \tag{15a, b}$$

and the fluid density ρ is assumed to obey the equation of state

$$\rho=\rho_0\exp[\beta(p-p_0)+\gamma\omega], \tag{16}$$

with constant reference density ρ_0, constant reference pressure p_0, constant compressibility coefficient β and constant salt coefficient γ. Other constants are porosity n, permeability k, viscosity μ, gravity g and dispersion length λ. The various variables are the (Darcy) velocity q of the fluid, the hydrodynamic pressure p and the salt mass fraction ω. We thus consider the medium to be homogeneous with respect to porosity, permeability and viscosity. However, inhomogeneities, and also sources and sinks, can easily be taken into account.

The set of equations can be formulated as a system of two PDEs with pressure p and salt concentration ω as independent variables. To this end we compute from (16)

$$\frac{\partial\rho}{\partial t}=\rho\beta\frac{\partial p}{\partial t}+\rho\gamma\frac{\partial\omega}{\partial t} \tag{17}$$

and substitute into (14) to obtain the fluid mass balance equation

$$n\rho\beta\frac{\partial p}{\partial t}+n\rho\gamma\frac{\partial\omega}{\partial t}=-\frac{\partial}{\partial x}(\rho q). \tag{18}$$

A further substitution yields the salt transport equation

$$n\rho\frac{\partial\omega}{\partial t}=-\rho q\frac{\partial\omega}{\partial x}-\frac{\partial}{\partial x}(\rho J). \tag{19}$$

We have used this form as input for the numerical solution method. A few comments are in order. First, substitution of the expression for J into (19) yields the advection–dispersion equation

$$n\rho\frac{\partial\omega}{\partial t}=-\rho q\frac{\partial\omega}{\partial x}+\frac{\partial}{\partial x}\left(\rho\lambda|q|\frac{\partial\omega}{\partial x}\right), \tag{20}$$

showing that in the present model the physical salt transport phenomena are advection and dispersion. Molecular diffusion is absent here. It is easily built in, however, since this merely amounts to adding a small constant to $\lambda|q|$. Assuming 'frozen' coefficients, we see that the Peclet number is

$$Pe=\left|\frac{L\rho q}{\rho\lambda|q|}\right|=\frac{L}{\lambda}, \tag{21}$$

where L denotes the physical length of the medium. Hence for $\lambda \ll L$ advection dominates and this is just the physical situation that gives rise to steep concentration gradients. Another point worth mentioning is that the compressibility coefficient β is very small compared to the salt coefficient γ. In fact, it is often zero, in which case the balance equation (18) reduces to

$$n\rho\gamma\frac{\partial\omega}{\partial t}=-\frac{\partial}{\partial x}(\rho q) \tag{22}$$

and $\partial p/\partial t$ is absent. We then have two equations for $\partial\omega/\partial t$, of which (18) can be rewritten as a PDE containing only spatial derivatives. Hence this rules out the possibility of explicit time stepping. Note that if we also put $\gamma=0$, then the density ρ is constant and the mass balance equation reduces to the simple pressure equation $p_{xx}=0$. Of course, a zero salt coefficient γ is not realistic in our application.

To complete the model description, we must give the initial and boundary conditions. Defining the space–time domain as $[0, L]\times[0, T]$, the initial and boundary conditions we have imposed for $\omega(x, t)$ and $p(x, t)$ are respectively

$$\omega(x,0)=0,\quad 0\leqslant x\leqslant L, \tag{23a}$$

$$\omega(0,t)=\omega_0>0 \quad\text{and}\quad \frac{\partial\omega}{\partial x}(1,t)=0,\quad 0<t\leqslant T, \tag{23b}$$

and

$$p(x,0)=p_0[(1-x/L)p_{\text{left}}+(x/L)p_{\text{right}}],\quad 0\leqslant x\leqslant L, \tag{24a}$$

$$p(0,t)=p_0p_{\text{left}} \quad\text{and}\quad p(L,t)=p_0p_{\text{right}},\quad 0<t\leqslant T, \tag{24b}$$

where ω_0 is the left-end salt concentration and p_{left} and p_{right} are pressure coefficients.

We have selected these conditions with the aim of generating a travelling salt front. Note that at $t=0$ there is no salt in the medium and that the inflow value $\omega_0>0$. Hence, assuming appropriate model data, this should give rise to a travelling front. The steepness and speed of the front will of course be determined by the complete set of physical data. A characteristic set is given in Table I which comprises all data needed to run the problem, except for the end time T, the dispersion length λ and the pressure coefficients p_{left} and p_{right}. Finally, numerically we have treated the problem in scaled, dimensionless form. We refer to Table I for the scaling relations with the dimensionless values of all quantities involved. From these relations one can check that all equations are left invariant (note that this also holds for (16) owing to the fact that after scaling, $p_0=1$). Hence in the remainder we have worked with the same set of equations as discussed above. The pressure coefficients p_{left} and p_{right} are left unchanged and will be specified with the numerical examples.

4. NUMERICAL EXAMPLES

We will present results of three numerical examples. To simplify the demonstration, these results

Table I. Model data. Boldface notation is used for the non-scaled quantities

	Non-scaled	Scaled
Time	$\mathbf{0}<\mathbf{t}<\mathbf{T}$ (s)	$t=\mathbf{t}/\mathbf{t_0}$, $\mathbf{t_0}=\boldsymbol{\mu}\mathbf{L}^2/\mathbf{k_0}\mathbf{p_0}=10^4$
Space	$\mathbf{0}<\mathbf{x}<\mathbf{L}$ (m)	$x=\mathbf{x}/\mathbf{L}$
End time	$\mathbf{T}$ (s)	$T=\mathbf{T}/\mathbf{t_0}$
Domain length	$\mathbf{L}=1$ m	$L=1$
Pressure	$\mathbf{p}$ (kg m^{-1} s^{-2})	$p=\mathbf{p}/\mathbf{p_0}$
Salt concentration	$\boldsymbol{\omega}$	$\omega=\boldsymbol{\omega}/\boldsymbol{\omega_0}$
Density	$\boldsymbol{\rho}$ (kg m^{-3})	$\rho=\boldsymbol{\rho}/\boldsymbol{\rho_0}$
Permeability	$\mathbf{k}=\mathbf{k_0}=10^{-12}$ m^2	$k=\mathbf{k}/\mathbf{k_0}=1$
Viscosity	$\boldsymbol{\mu}=\boldsymbol{\mu_0}=10^{-3}$ kg m^{-1} s^{-1}	$\mu=\boldsymbol{\mu}/\boldsymbol{\mu_0}=1$
Reference pressure	$\mathbf{p_0}=10^5$ kg m^{-1} s^{-2},	$p_0=1$
Salt inflow concentration	$\boldsymbol{\omega_0}=0{\cdot}26$	$\omega_0=1$
Reference density	$\boldsymbol{\rho_0}=10^3$ kg m^{-32}	$\rho_0=1$
Gravity force	$\mathbf{g}=9{\cdot}81$ m s^{-2}	$g=\boldsymbol{\rho_0}\mathbf{L}\mathbf{g}/\mathbf{p_0}=0{\cdot}0981$
Porosity	$\mathbf{n}=0{\cdot}2$	$n=0{\cdot}2$
Salt coefficient	$\boldsymbol{\gamma}=0{\cdot}69$,	$\gamma=\boldsymbol{\gamma}\boldsymbol{\omega_0}=0{\cdot}1794$
Dispersion length	$\boldsymbol{\lambda}$ (m)	$\lambda=\boldsymbol{\lambda}/\mathbf{L}$
Compressibility coefficient	$\boldsymbol{\beta}=10^{-10}$ m s^2 kg^{-1}	$\beta=\boldsymbol{\beta}\mathbf{p_0}=10^{-5}$

have been obtained with a fixed set of numerical control parameters:

$$\mathrm{TOL}=10^{-5} \quad \text{(temporal integration)}; \tag{25a}$$

$$\kappa=2, \quad \tau=10^{-3}, \quad \alpha=10^{-2}, \quad \beta_1=0, \quad \beta_2=1 \quad \text{(grid movement)}; \tag{25b}$$

$$X_i(0)=\frac{i}{N+1}, \quad 0\leqslant i\leqslant N+1 \quad \text{(uniform initial grid)}. \tag{25c}$$

SPRINT was called in standard mode, thus providing automatically an initial step size and Jacobian evaluation. The Euclidean norm was used for local error control, while (25a) was imposed for all components of the vector Y (NPDE$=2$, $u_1=p$, $u_2=\omega$). Note that TOL$=10^{-5}$ is quite small. However, to accurately simulate the rapid birth of the salt front, which arises from the inconsistency between the initial and left-end salt concentrations, a small tolerance value is natural. We also emphasize that we always started on a uniform grid, just for convenience of use. This means that immediately after the start the method should rapidly cluster most of the grid points near the left boundary.

The grid parameters take on more or less standard values, except for β_1. The choice $\beta_1=0$ means that the pressure gradient $\partial p/\partial x$ is not taken into account in the monitor (7). We decided to omit $\partial p/\partial x$ in the monitor, since in our examples $\partial p/\partial x$ varies very slowly and thus acts more or less in the same manner as the constant regularization parameter α. In such cases, too large a value for the near-constant pressure gradient yields an unnecessarily large regularization effect. This in turn would imply that variations in the concentration gradient $\partial\omega/\partial x$ become of lesser importance in the spatial equidistribution than desired.

4.1. *Example I*

The first example is defined by the data of Table I together with $\lambda=0{\cdot}001$, $T=5$, $p_{\mathrm{left}}=1{\cdot}7$ and $p_{\mathrm{right}}=1{\cdot}0$. With this choice initial pressure function the arising salt front travels to the right boundary and finally renders a steady state for p and ω with p equal to the linear initial pressure and $\omega=\omega_0=1$. The steady state starts to settle at about $t=2$, long before the end time $T=5$ has been reached. Consequently, owing to the uniform salt concentration, at about $t=2$ the grid should again become uniform. Hence this example provides an interesting test for the moving grid method. The pressure p undergoes only a marginal change for $t>0$ and below we will therefore only plot ω.

Figure 1 depicts the grid and salt concentrations at some values of t for $N_2=N+2=25$ and 50. We see that the grid accurately reflects the anticipated solution behaviour. At very early times the grid points rapidly cluster near $x=0$, then the cluster travels with the front, and when the steady state is reached, a uniform grid appears. While $N_2=25$ results in a little overshoot at the top and a little smearing at the foot, $N_2=50$ gives already very accurate salt concentration profiles. The profiles for $N=100$ (not shown here) do equal those for $N=50$ up to plotting accuracy.

Table II shows the integration history for $N_2=25$, 50 and 100 and serves to provide insight on the costs of the implicit numerical integration method. The given data have the following meaning: STEPS, number of integration steps; JACS, number of Jacobian updates; RESIDS, total number of evaluations of the ODE system including those needed for the Jacobian updates; NITER, total number of Newton iterations; CPU, central processing time on an ALLIANT/FX4 computer using one processor. Note that our decision to start on a uniform initial grid has its price. For example, for $N_2=100$ more than half the number of steps is used to reach $t=0{\cdot}1$. In fact, at $t=10^{-4}$, 10^{-3} and 10^{-2} we have STEPS = 39, 152 and 271 respectively. A large number of these step are needed simply to adjust the initial grid to the very steep concentraion profile at the very early times (see the right upper plot in Figure 1). Therefore somehow adjusting the initial grid to the expected solution profile at the first forward time level will reduce STEPS significantly. We also wish to remark that the method efficiently detects the steady state, since for $t>2$ the temporal step sizes are rapidly increased and very few steps are required to complete the integration. Finally, we have also tabulated $\omega_{\max}-\omega_0$, which is the maximal overshoot at the given points of time. We see that already for $N_2=25$ the overshoot is very little.

A further inspection of the salt concentraion plots shows that, as expected, the first-derivative monitor (7) places quite a number of points just within the front where $\partial\omega/\partial x$ is largest. Fortunately, the spatial grid smoothing resulting in relation (10) has the nice side effect of keeping a substantial number of points at the foot and top of the front, where $\partial\omega/\partial x$ becomes smaller and finally zero. This only works of course if κ is taken not too small. Note that there should be enough points at the foot and top so as to avoid wiggles, since the spatial discretization is based on a common central finite volume scheme. For a comparison of results obtained with a second-derivative monitor based on $m(u)=(\alpha+\|u_{xx}\|^2)^{1/4}$ and with a fixed grid, the reader is referred to Reference 14. There it is concluded that the first-derivative monitor results are generally significantly better.

4.2. Example II

The second example is also defined by the data of Table I, but now $p_{\text{left}}=1{\cdot}11$, $p_{\text{right}}=1{\cdot}0$, $T=500$ and $\lambda=0{\cdot}0001$. The smaller pressure gradient in the initial function has two effects. First, it yields a smaller fluid velocity, resulting in a larger time scale, which explains the larger value for T. The second and more interesting effect is that the travelling salt front now comes to a standstill before it has reached the right boundary. This happens at about $t=150$, at which point of time the front lies near $x=0{\cdot}6$. The reason is that the fluid velocity q tends to zero, uniformly in x, which settles the system into a steady state and this takes place long before the salt front has reached the right boundary. We note that this phenomenon is rather special in the sense that it depends

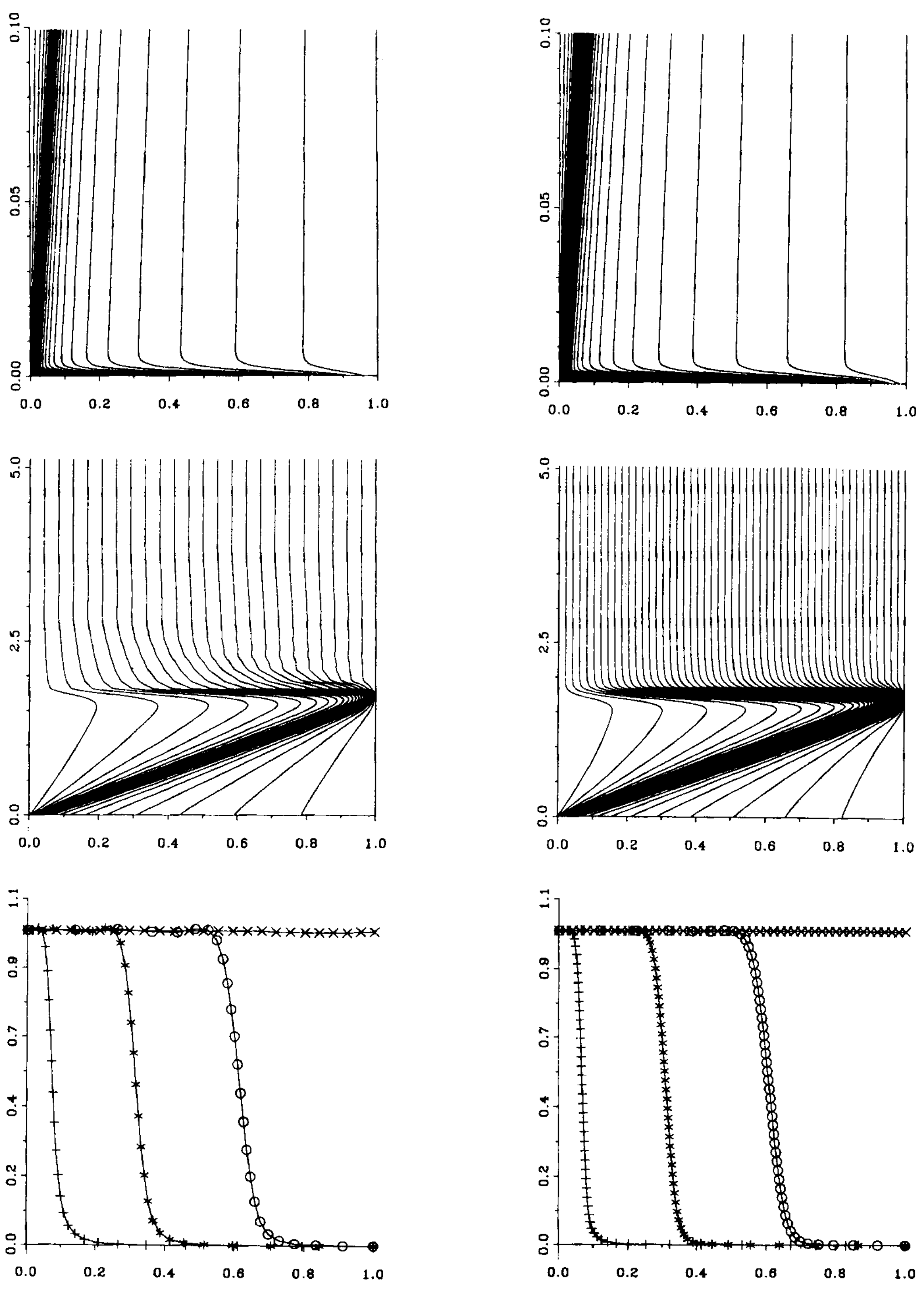

Figure 1. Example I: grid lines and salt concentration profiles at $t = 0{\cdot}1$, $0{\cdot}5$, $1{\cdot}0$ and $5{\cdot}0$. The left part of each pair corresponds to $N_2 = 25$ and the right part to $N_2 = 50$. Note the difference in scaling in each of the two grid line plots

Table II. Example I: integration histories

N_2	t	STEPS	JACS	RESIDS	NITER	$\omega_{\max} - 1{\cdot}0$	CPU (s)
25	0·1	149	39	931	407	$7{\cdot}0 \times 10^{-3}$	—
	0·5	198	47	1175	545	$7{\cdot}0 \times 10^{-3}$	—
	1·0	220	52	1302	607	$6{\cdot}0 \times 10^{-3}$	—
	5·0	565	144	3532	1610	—	140
50	0·1	202	53	1282	574	$8{\cdot}0 \times 10^{-4}$	—
	0·5	225	58	1413	638	$1{\cdot}0 \times 10^{-3}$	—
	1·0	234	61	1481	667	$1{\cdot}0 \times 10^{-3}$	—
	5·0	450	118	2904	1335	—	236
100	0·1	301	83	1988	882	$3{\cdot}0 \times 10^{-4}$	—
	0·5	317	87	2085	927	$3{\cdot}0 \times 10^{-4}$	—
	1·0	326	89	2140	956	$6{\cdot}0 \times 10^{-4}$	—
	5·0	529	142	3533	1648	—	566

heavily on the initial pressure gradient. The standstill of the salt front is lost with a relatively slight change in this gradient. Also note that this standstill requires zero molecular diffusion, which in reality is of course not true. However, the simulation of this rather subtle situation provides a nice numerical test, since it requires an accurate balancing of gravity force ρg and pressure gradient force $\partial p/\partial x$ in the Darcy velocity expression (14b). Finally, we have made the dispersion length 10 times smaller than in the previous example, giving a Peclet number of 10 000 and a much steeper front (recall that the spatial discretization of MGI is based on a common central finite volume scheme).

Figure 2 shows the computed grid and salt concentration profiles at some values of time for $N_2 = N+2 = 25$ and 50. As in the previous example, we see that the grid movement accurately reflects the anticipated solution behaviour. For early times it is completely similar, while for later times the cluster around the steep salt front remains in position. We also see that $N_2 = 25$ now results in more overshoot owing to the fact that the dispersion length is 10 times smaller than in the previous example. However, $N_2 = 50$ again gives a very accurate solution and the profiles for $N_2 = 100$ (not shown here) do equal those for $N_2 = 50$ up to plotting accuracy.

Table III contains part of the integration history for $N_2 = 25$, 50 and 100, providing the same information as before. With this table we wish to draw attention to an inherent model difficult stemming from the absolute value function in the dispersion flux expression $\rho J = -\rho\lambda|q|\omega_x$. This difficulty manifests itself in the large number of time steps and Jacobian updates used over the 'near-steady-state interval' [200, 500] for $N_2 = 100$ (recall that the steady state starts to settle at about $t = 150$). While the code easily detects the numerical steady state solution with 25 and 50 points, which can be concluded from the small number of steps needed to integrate from $t = 200$ to 500, this is clearly not the case with 100 points. In fact, with 100 points this 'near-steady-state part' of the integration interval requires $1038 - 423 = 615$ integration steps and $707 - 105 = 602$ Jacobian updates, which is rather extreme. What has happened here is that the iterative Newton algorithm has repeatedly failed to converge, so that the strategy of the code keeps the temporal step size down and keeps asking for new Jacobians.

The cause of the Newton convergence failure lies with $|q|$ if $q \approx 0$. The following observations explain this. Owing to the absolute value function, entries of the Jacobian matrix contain sign(q). Consequently, if $q \approx 0$, then during the Newton iteration approximate values for q readily change

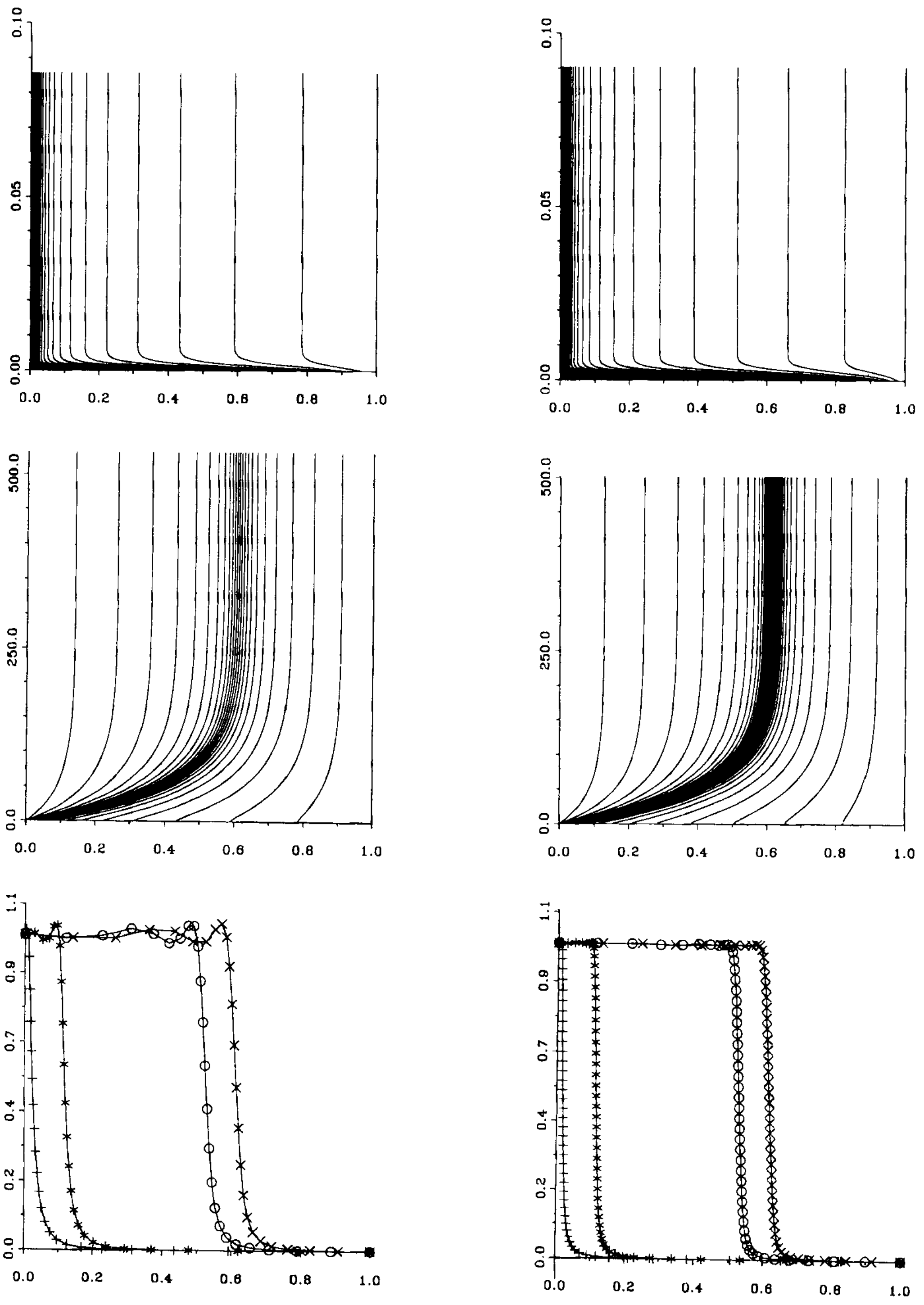

Figure 2. Example II: grid lines and salt concentration profiles at $t=1$, 10, 200 and 500. The left part of each pair corresponds to $N_2=25$ and the right part to $N_2=50$. Note the difference in scaling in each of the two grid line plots

Table III. Example II: integration histories

N_2	t	STEPS	JACS	RESIDS	NITER	$\omega_{\max} - 1{\cdot}0$	CPU (s)
25	1	160	40	958	421	$2{\cdot}0 \times 10^{-2}$	—
	10	239	64	1509	654	$3{\cdot}0 \times 10^{-2}$	—
	100	335	92	2155	930	$3{\cdot}0 \times 10^{-2}$	—
	200	354	95	2244	980	$4{\cdot}0 \times 10^{-2}$	—
	500	373	101	2371	1072	$4{\cdot}0 \times 10^{-2}$	95
50	1	249	59	1473	685	$4{\cdot}0 \times 10^{-3}$	—
	10	300	72	1787	826	$2{\cdot}0 \times 10^{-3}$	—
	100	338	79	1983	931	$2{\cdot}0 \times 10^{-3}$	—
	200	355	81	2048	968	$2{\cdot}0 \times 10^{-3}$	—
	500	379	88	2204	1033	$2{\cdot}0 \times 10^{-3}$	185
100	1	312	85	2058	926	$4{\cdot}0 \times 10^{-4}$	
	10	346	93	2253	1015	$4{\cdot}0 \times 10^{-4}$	—
	100	410	103	2538	1168	$7{\cdot}0 \times 10^{-4}$	—
	200	423	105	2592	1196	$8{\cdot}0 \times 10^{-4}$	—
	500	*1038*	*707*	*12700*	*3344*	$9{\cdot}0 \times 10^{-4}$	*1753*

sign. Because the size of entries is large, since they contain terms $(\Delta x_i)^{-2}$ and Δx_i can be very small, it happens that during the iteration process entries frequently change their value from large positive to large negative or vice versa. No doubt this severely hinders the convergence of the iterative Newton process and, as we have observed, will often lead to convergence failures and requests for a Jacobian update. This explains why the march to steady state in the case of 100 points is so troublesome. However, we stipulate that also with 25 and 50 points the march to steady state eventually becomes troublesome. It all depends on the size of the computed velocities q and is a matter of accuracy. With fewer points the computed velocities arrive in the troublesome regime for larger values of time when the system has become sufficiently stationary or, in other words, when the numerical velocities have become sufficiently small. Ironically, with 100 points the accuracy is sufficiently good to have the troublesome Newton convergence behaviour already for $200 < t < 500$.

We emphasize that the troublesome march to steady state originates from the Jacobian matrix needed in the iterative solution process and not from the integration formula itself. In fact, we have also run the problem with $|q|$ replaced by $\sqrt{(q^2+\varepsilon)}$, with $\varepsilon = 10^{-5}$, which completely remedies the situation, and a normal march to steady state is observed with very large step sizes towards the end value T, even up to $T = 10^{12}$. When the modelling does not allow this slight modification in the dispersion flux expression, an alternative remedy is to change the expression for $|q|$ only in the entries of the Jacobian so as to avoid the sign changes. This involves little change in the Jacobian matrix and thus should not interfere significantly with the convergence behaviour of the iterative Newton method.

4.3. Example III

This example is derived from Example I by changing the salt concentration value $\omega(0, t) = 1$ to the step function

$$\omega(0, t) = \begin{cases} 1, & 0 < t \leqslant 0{\cdot}75, \\ 0, & 0{\cdot}75 < t \leqslant 5{\cdot}0. \end{cases} \tag{26}$$

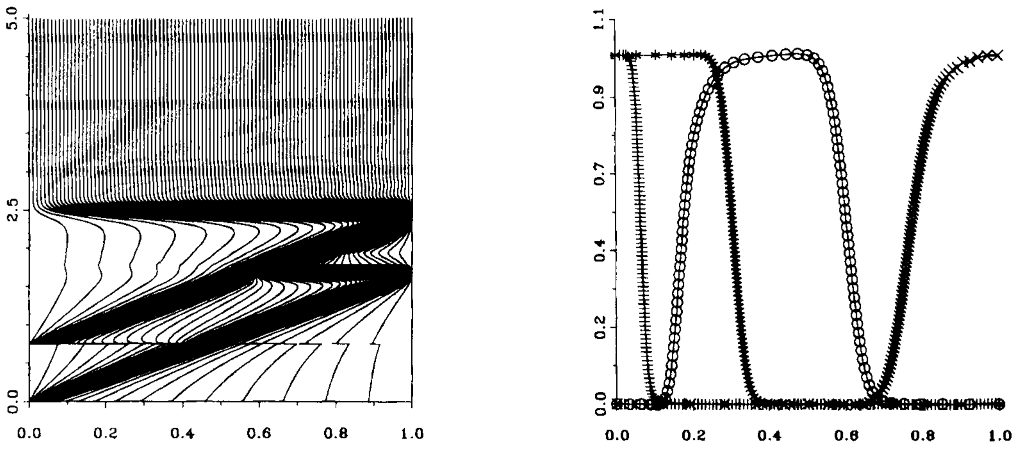

Figure 3. Example III: grid lines and salt concentration profiles at $t=0{\cdot}1$, $0{\cdot}5$, $1{\cdot}0$, $2{\cdot}0$ and $5{\cdot}0$ for $N_2=100$

Table IV. Example III: integration costs at $t=1{\cdot}0$, $2{\cdot}0$ and $5{\cdot}0$ for $N_2=100$ (see Table II for corresponding data at $t=0{\cdot}1$ and $0{\cdot}5$)

N_2	t	STEPS	JACS	RESIDS	NITER	CPU (s)
100	1·0	554	159	3734	1624	—
	2·0	819	262	6008	2511	—
	5·0	1009	308	7259	3154	1146

Thus for $0<t\leqslant 0{\cdot}75$ the two solutions are equal and at $t=0{\cdot}75$ the step function generates a second front at $x=0$, resulting in a block-form concentration profile. The block then travels to the right boundary and eventually the system runs into steady state with uniform zero salt concentration. For the moving grid method this solution is more difficult to compute, since now two travelling fronts are present which appear and disappear at different values of t. Hence, instead of two times the solution shape is drastically changed four times and the automatic grid movement and step size control should be able to cope with these drastic changes. For example, without neglecting the already existing first front, at $t=0{\cdot}75$ the method must rapidly cluster grid points at the left boundary and decrease the time step to timely see the onset of the second front. Therefore, for the same accuracy, roughly twice the number of grid points and time-stepping effort will be needed as for Example I.

We have used $N_2=25$, 50 and 100. Apparently, 25 points is not enough, but with 50 points the solution is already fairly accurate. A comparison for 50 and 100 points reveals only minor differences at the top of the computed salt block profile and we may conclude that the results are very satisfactory. The grid line plot in Figure 3 for $N_2=100$ nicely reveals the onset of the second front where very small time steps have been taken, similarly as at $t=0$ (see Figure 1). The arrival of the two fronts at the right boundary can also be clearly recovered from the plot, like the change to the uniform steady state grid. Note that here also small time steps are needed to accurately simulate the rapid solution change. The integration costs given in Table IV indeed show that the time-stepping effort is about twice as large as for Example I. As anticipated, comparison of

Tables II and IV reveals that it is mainly the drastic changes in the solution shape that determine the costs. Once the front is in existence, the time stepping is done very efficiently, as can be deduced from the number of Jacobian updates listed in Table II at $t = 0{\cdot}1$ and $1{\cdot}0$.

5. CONCLUDING REMARKS

We have applied a moving grid, finite difference method to a particular class of one-space-dimensional fluid flow/salt transport problems with rapid spatial and temporal transitions in the salt concentration. The success of this method rests on two sorts of automatic grid adaptation. The first adaptation is connected with the space grid and serves to cope with the rapid spatial transitions. These are dealt with by integrating on grids that spatially equidistribute a relevant measure of the error. The equidistribution is realized in a dynamic Lagrangian approach where the grid is adapted continuously in time. This feature is important since it makes it possible to follow steep travelling fronts accurately and efficiently. The second adaptation serves to cope with rapid temporal transitions and is just the use of variable step sizes in the numerical integration. Variable step sizes are a prerequisite when drastic solution changes have to be dealt with, such as the onset of a steep front. The numerical integration has been performed with the LSODI-based stiff ODE solver of the SPRINT package.[10]

Our findings reported in Section 4 have convincingly shown that the method is very well suited to solve 1D brine transport models involving high concentration gradients. Because we have worked with an *a priori* chosen set of numerical control parameters, it is most likely that tuning of these parameters will further enhance the efficiency and accuracy for the specific model at hand. Since the method was originally developed for general one-space-dimensional PDE systems,[6,7] it is also an excellent candidate for solving fluid flow/solute transport problems from other fields of application. In this connection it is worth emphasizing the user-friendly computational environment of the SPRINT package and the moving grid interface MGI,[11] which together provide a numerical software tool that requires a minimum of programming effort.

REFERENCES

1. S. M. Hassanizadeh and T. Leijnse, 'On the modeling of brine transport in porous media', *Water Resources Res.*, **24**, 321–330 (1988).
2. S. M. Hassanizadeh, 'Experimental study of coupled flow and mass transport: a model validation exercise', in K. Kovar (ed.), *Calibration and Reliability in Groundwater Modeling, IAHS Publ. 195*, IAHS, Wallingford, Oxfordshire, 1990.
3. J. C. H. van Eijkeren, P. A. Zegeling and S. M. Hassanizadeh, 'Practical use of SPRINT and a moving-grid interface for a class of 1D nonlinear transport problems', *Rep. 959101001*, RIVM, Bilthoven, 1991.
4. K. Dekker and J. G. Verwer, *Stability of Runge–Kutta Methods for Stiff Nonlinear Differential Equations*, North-Holland, Amsterdam, 1984.
5. M. Sanz-Serna and J. G. Verwer, 'Stability and convergence at the PDE/stiff ODE interface', *Appl. Numer. Math.*, **5**, 117–132 (1989).
6. E. A. Dorfi and L. O'C. Durry, 'Simple adaptive grids for 1D initial-value problems', *J. Comput. Phys.*, **69**, 175–195 (1987).
7. J. G. Verwer, J. G. Blom, R. M. Furzeland and P. A. Zegeling, 'A moving-grid method for one-dimensional PDEs based on the method of lines', In J. E. Flaherty, P. J. Paslow, M. S. Shephard and J. D. Vasilakis (eds), *Adaptive Methods for Partial Differential Equations*, SIAM, Philadelphia, PA, 1989, pp. 160–175.
8. R. M. Furzeland, J. G. Verwer and P. A. Zegeling, 'A numerical study of three moving-grid methods for one-dimensional partial differential equations which are based on the method of lines', *J. Comput. Phys.*, **89**, 349–388 (1990).
9. K. E. Brenan, S. L. Campbell and L. R. Petzold, *Numerical Solution of Initial-value Problems in Differential Algebraic Equations*, North-Holland, Amsterdam, 1989.
10. M. Berzins, P. M. Dew and R. M. Furzeland, 'Developing software for time-dependent problems using the method of lines', *Appl. Numer. Math.*, **5**, 375–398 (1989).

11. J. G. Blom and P. A. Zegeling, 'A moving-grid interface for systems of one-dimensional time-dependent partial differential equations', *Rep. NM-R8904*, CWI, Amsterdam, 1989.
12. R. D. Skeel and M. Berzins, 'A method for the spatial discretization of parabolic equations in one space variable', *SIAM J. Sci. Stat. Comput.*, **11**, 1–32 (1990).
13. J. G. Blom and J. G. Verwer, 'On the use of the arclength and curvature monitor in a moving-grid method which is based on the method of lines', *Rep. NM-N8902*, CWI, Amsterdam, 1989.
14. P. A. Zegeling, J. G. Verwer and J. C. H. van Eijkeren, 'Application of a moving-grid method to a class of 1D brine transport problems in porous media', *Rep. NMR-9112*, CWI, Amsterdam, 1991.