INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 10, 651–681 (1990)

IMPLICIT FINITE ELEMENT METHODS FOR TWO-PHASE FLOW IN OIL RESERVOIRS

HANS PETTER LANGTANGEN

Department of Mathematics, University of Oslo, PO Box 1053, N-0316 Oslo 3, Norway

SUMMARY

The equations governing immiscible, incompressible, two-phase, porous media flow are discretized by generalized streamline diffusion Petrov–Galerkin methods in space and by implicit differences in time. Systems of non-linear algebraic equations are solved by Newton–Raphson iteration employing ILU-preconditioned conjugate-gradient-like methods to the non-symmetric matrix system in each iteration. The resulting solution methods are robust, enable complex grids with irregular nodal orderings and allow capillary effects.

Several numerical formulations are tested and compared for one-, two- and three-dimensional flow cases, with emphasis on problems involving saturation shocks, heterogeneous media and curved boundaries. For reservoirs consisting of multiple rock types with differing capillary pressure properties, it is shown that traditional Bubnov–Galerkin methods give poor results and the new Petrov–Galerkin formulations are required. Investigations regarding the behaviour of several preconditioned conjugate-gradient-like methods in these type of problems are also reported.

KEY WORDS Porous media Two-phase flow Oil recovery Finite elements Preconditioning Conjugate gradients

1. INTRODUCTION

In this paper we consider numerical methods for the simultaneous flow of two immiscible fluids in a porous medium. Such models are relevant to, for instance, the petroleum industry for predicting the displacement of oil by injected water in reservoir rocks. Traditionally, oil reservoir simulators have been based on finite difference discretization of the governing partial differential equations. Since these equations are highly non-linear and may develop sharp fronts, special care must be taken to represent non-linearities and suppress non-physical oscillations in the vicinity of fronts. The latter problem is usually circumvented by applying upwind difference schemes. Unfortunately, such schemes also artificially dissipate sharp fronts and lead to results which are sensitive to the direction of the computational grid. For problems in one spatial dimension numerous sophisticated higher-order difference schemes have been developed and used with success. Nevertheless, first-order methods are still dominant in two- and three-dimensional problems, probably owing to their simplicity and their ability to damp non-physical oscillations.

Most commercial reservoir simulators employ explicit treatment in time of non-linearities (see e.g. IMPES formulation[1]). Recently, fully implicit methods have increased in popularity because they offer robustness and unconditional stability. In some problems they may also be more efficient than explicit methods because of less restrictions on the time increments. Implicit time schemes give rise to sets of coupled non-linear algebraic equations at each time step. Application of iterative Newton-like methods to the non-linear equations leads to large, sparse, non-symmetric matrix systems in each iteration. The efficiency of implicit methods is therefore

0271–2091/90/060651–31$15.50 *Received March 1989*

crucially dependent on fast (iterative) methods for solving systems of linear equations. We refer to Aziz and Settari[1] for a review and comparison of finite difference schemes and iterative sparse matrix solution techniques that are widely used in commercial simulators.

Reservoir geometries are generally complex and therefore the application of the finite element method is desirable. The flexibility of finite elements is especially valuable in 2D and 3D where one can construct, in a consistent manner, methods for irregular grids with, for example, local mesh refinements. The finite element method also offers a convenient spatial treatment of non-linearities. Contributions concerning application of the standard Bubnov–Galerkin finite element method to immiscible two-phase porous media flow[2–7] report improved accuracy over finite differences and absence of the grid orientation effect. Unfortunately, problems with oscillations around sharp fronts and convergence to solutions with wrong shock velocity and strength have been reported. Similar problems occur in miscible displacements.[8–10] To overcome these difficulties, several authors have applied 'upwinding' techniques,[7,9–13] for example Petrov–Galerkin or discontinuous Galerkin methods (see Hughes[14] for an overview). Experience with these modifications has been promising.

Variations in pressure and saturation occur on different time and space scales in multiphase porous media flow. The pressure and saturations are also qualitatively different in that the former is smoothly varying while the latter may contain propagating sharp fronts. The different qualitative behaviour is of course reflected in the governing equations which are of parabolic/elliptic versus (almost-)hyperbolic type respectively. It seems reasonable to apply different numerical methods for updating the pressure and the saturation. Front-tracking methods[15] are schemes that efficiently exploit the different nature of the equations. However, serious problems are associated with these methods when applied to 3D flow and in cases where diffusive effects are significant.

The main contributions of the present paper are related to the development and testing of an implicit finite element formulation for two-phase flow. It is particularly focused on topics such as a new 'upwind' space discretization, capillary heterogeneities, fully implicit versus sequentially implicit solution techniques, and efficient iterative solvers for systems of linear equations. These topics have previously received minor attention in the literature on finite elements in immiscible porous media flow. We will formulate the system of governing partial differential equations in terms of a 'pressure' equation and a 'saturation' equation. The two equations are discretized by the finite element method in space and by implicit backward differences in time. Weighting functions that have been successful in multidimensional linear convection–diffusion problems[16,17] are adapted to the present two-phase flow equations for accurate treatment of sharp fronts. The philosophy behind the present model is generality and robustness. We therefore employ simultaneous solution of all unknowns at each time level. However, a simpler scheme where the two differential equations are solved in sequence at each time level may suffice in many instances, and we have implemented and tested this approach as well. Comparison of these two solution schemes is of interest, since in extensions to, for example, compositional flow, a sequential solution of the equations is often required owing to limited computer resources.

Other authors on finite element methods for reservoir flow have tested their methods mostly in 1D or on rectangular 2D domains with homogeneous rock properties. In this paper special emphasis is devoted to heterogeneous media with discontinuous rock properties and curved boundaries. Application of the method in 3D is also reported.

Generally, the development and testing of efficient iterative solution techniques for large, sparse, non-symmetric matrix systems have been restricted to matrices generated by finite difference schemes on grids with structured orderings of the unknowns (natural, D2, D4[18]). Finite element methods give rise to less sparse matrices than standard finite differences, especially in 3D.

Practical reservoir flow problems also frequently involve complicated geometries and hence complicated grids with an irregular node numbering. The resulting coefficient matrix will in such cases have a correspondingly complicated sparsity pattern which prevents application of many popular preconditioners (e.g. nested factorization[19]). In this work we test a family of iterative ILU-preconditioned generalized conjugate gradient methods applied to the matrix systems generated by our finite element schemes. These iterative methods are well suited for complicated mesh structures and nodal orderings.

Section 2 treats the partial differential equations that govern two-phase porous media flow. Discretization procedures in space and time are introduced in Section 3. Section 4 is devoted to iterative solution of the matrix systems in our implicit scheme. A heterogeneous one-dimensional model problem is studied in Section 5, while experiments concerning two-dimensional flow are presented in Section 6. An example of three-dimensional flow is reported in Section 7. Results from investigations regarding the optimal choice of iterative equation solvers for linear systems are given in Section 8, while Section 9 contains concluding remarks.

2. MATHEMATICAL MODEL

We consider the flow of two immiscible fluids in a porous medium.[1,20] Mass conservation for each fluid phase leads to the continuity equations

$$\frac{\partial}{\partial t}(\phi \rho_i S_i) + \nabla \cdot (\rho_i \mathbf{v}_i) + q_i = 0, \quad i = \mathrm{n}, \mathrm{w}, \tag{1}$$

where S_i is the saturation in phase i (i.e. the fraction of the pore volume occupied by phase i), the subscripts 'w' and 'n' refer to the wetting and non-wetting phases respectively, ϕ is the porosity and $\mathbf{v}_i$, ρ_i and q_i are the filtration velocity, the density and the production rate respectively of phase i. By definition of the saturations we have

$$S_\mathrm{n} + S_\mathrm{w} = 1,$$

while the capillary pressure p_c relates the phase pressures P_n and P_w:

$$p_\mathrm{c} = P_\mathrm{n} - P_\mathrm{w}, \tag{2}$$

with p_c a non-increasing function of S_w.

The momentum equations follow from Darcy's law with relative permeabilities:

$$\mathbf{v}_i = -\lambda_i(\nabla P_i - \rho_i \mathbf{g}), \quad i = \mathrm{n}, \mathrm{w}, \tag{3}$$

which is valid as long as the local pore Reynolds number is less than unity. The acceleration due to gravity is denoted by $\mathbf{g}$. The phase mobility λ_i equals $K k_{ri}/\mu_i$, where K is the absolute permeability, μ_i is the viscosity of phase i and k_{ri} is the relative permeability of phase i. k_{ri} is a function of S_i. K is in general a tensor quantity, which for simplicity is assumed isotropic in the rest of the paper. μ_i and ϕ are here assumed to be independent of the phase pressures. The absolute permeability and the porosity are allowed to be space-varying but time-independent. We also assume both phases to be incompressible (constant ρ_i). All these assumptions can be relaxed in the mathematical as well as in the proposed numerical formulations without any major difficulties.

The equation system above can be reduced to two equations in two scalar unknowns. Here we choose P_n and S_w as the primary unknowns. Adding the two continuity equations yields the equation

$$\nabla \cdot \mathbf{v}_\mathrm{t} + Q_\mathrm{t} = 0, \tag{4}$$

where $Q_t = Q_n + Q_w$, with $Q_i = q_i/\rho_i$. The total filtration velocity expressed as a function of P_n and S_w reads

$$\mathbf{v}_t \equiv \mathbf{v}_n + \mathbf{v}_w = -\lambda_t \nabla P_n + \lambda_w p_c' \nabla S_w + (\lambda_w \rho_w + \lambda_n \rho_n)\mathbf{g}, \tag{5}$$

where $\lambda_t = \lambda_w + \lambda_n$ and the prime denotes derivation with respect to S_w.

As companion equation to (4) we use the wetting fluid continuity equation

$$\frac{\partial}{\partial t}(\phi S_w) + \nabla \cdot \mathbf{v}_w + Q_w = 0, \tag{6}$$

where $\mathbf{v}_w$ as a function of the primary unknowns reads

$$\mathbf{v}_w = -\lambda_w(\nabla P_n - \rho_w \mathbf{g}) + \lambda_w p_c' \nabla S_w.$$

In order to make it possible to apply discontinuous weighting functions in the finite element formulations, equation (6) must be rewritten in 'hyperbolic' form:[20]

$$\frac{\partial}{\partial t}(\phi S_w) + f_w' \mathbf{v}_t \cdot \nabla S_w - \nabla \cdot (h_w p_c' \nabla S_w) + \nabla \cdot (G_w \mathbf{g}) - f_w Q_t + Q_w = 0, \tag{7}$$

where $f_w = \lambda_w/\lambda_t$, $h_w = -\lambda_n f_w$ and $G_w = h_w(\rho_n - \rho_w)$ are known functions of S_w. Note that λ_i, f_w and $p_c' h_w$ are positive since $k_{ri} > 0$ and $p_c' \leqslant 0$. Thus the term $\nabla \cdot (h_w p_c' \nabla S_w)$ is mathematically equivalent to a diffusion term. The hyperbolic nature of equation (7) becomes evident when the capillary pressure is regarded as independent of S_w. The equation system consisting of (4) and (7) will be referred to as the *hyperbolic formulation*. In the finite element context the hyperbolic formulation has previously been applied by Morgan *et al.*,[5] but in a pure Bubnov–Galerkin framework.

A widely used condition at wells is that sources and sinks inject and extract fluids in proportion to their local mobilities,

$$Q_w = f_w Q_t,$$

so that the last two terms in (7) cancel. The similar condition at the boundary reads

$$\mathbf{v}_w \cdot \mathbf{n} = f_w \mathbf{v}_t \cdot \mathbf{n}, \tag{8}$$

which corresponds to $\partial p_c/\partial n = 0$ (provided that gravity effects can be neglected near wells). $\mathbf{n}$ is the outward unit normal to the boundary.

All calculations reported in this paper were carried out using dimensionless quantities. Details concerning the scaling are now given. We introduce the characteristic length l_c, characteristic pressure P_c (not to be confused with the capillary pressure p_c!), characteristic viscosity μ_c, characteristic density ρ_c, characteristic gravity g_c, characteristic permeability K_c and characteristic time t_c. Using an asterisk as superscript for representing dimensionless quantities, we have

$$\mathbf{x} = l_c \mathbf{x}^*, \qquad t = t_c t^*, \qquad K = K_c K^*,$$
$$P_i = P_c P_i^*, \qquad \mathbf{g} = g_c \mathbf{g}^*, \qquad \rho_i = \rho_c \rho_i^*, \qquad \mu_i = \mu_c \mu_i^*,$$
$$p_c' = P_c p_c^{*\prime}, \qquad Q_i = Q_i^*/t_c,$$
$$\lambda_i = (K_c/\mu_c)\lambda_i^*, \qquad \lambda_i^* = K^* k_{ri}/\mu_i^*, \qquad \lambda_t^* = \lambda_n^* + \lambda_w^*,$$
$$\mathbf{v}_i = (K_c P_c/\mu_c l_c)\mathbf{v}_i^*, \qquad \mathbf{v}_i^* = -\lambda_i^*[\nabla^* P_i^* - (W/V)\rho_i^* \mathbf{g}^*],$$
$$\mathbf{v}_t = -\lambda_t^* \nabla^* P_n^* + \lambda_w^* p_c^{*\prime} \nabla^* S_w + (W/V)(\lambda_w^* \rho_w^* + \lambda_n^* \rho_n^*)\mathbf{g}^*,$$
$$h_w = (K_c/\mu_c)h_w^*, \qquad h_w^* = -\lambda_n^* f_w,$$
$$G_w = (K_c \sigma_c/\mu_c)G_w^*, \qquad G_w^* = h_w^*(\rho_n^* - \rho_w^*).$$

Observe that f_w, ϕ and S_i are dimensionless by definition. The two dimensionless numbers W and V are given by

$$V = P_c K_c t_c / \mu_c l_c^2, \qquad W = \rho_c q_c K_c t_c / \mu_c l_c.$$

Equations (4) and (7) in dimensionless form then read

$$V\nabla^* \cdot \mathbf{v}_t^* + Q_t^* = 0, \tag{9}$$

$$\frac{\partial}{\partial t^*}(\phi S_w) + V[f_w' \mathbf{v}_t^* \cdot \nabla^* S_w - \nabla^* \cdot (h_w^* p_c^{*\prime} \nabla^* S_w)] + W\nabla^* \cdot (G_w^* \mathbf{g}^*) - f_w Q_t^* + Q_w^* = 0. \tag{10}$$

In the rest of the paper the asterisk will be dropped and all quantities will be assumed dimensionless.

3. FINITE ELEMENT FORMULATIONS

Space discretization

The coupled non-linear partial differential equation system outlined in the previous section is discretized in space by the finite element method. In recent years, mixed finite elements have seemed to be the most popular finite element discretization procedure.[12,13] Besides having the advantage of approximating velocity and pressure to the same order of accuracy, mixed finite elements give rise to difference equations that are closely related to the dominating block-centred finite difference methods.[13] Compared to a standard approach, mixed formulations lead to larger matrix systems since $\mathbf{v}_t$ is also introduced as an unknown. It is a serious disadvantage that the matrix systems associated with mixed finite elements have a structure which makes it difficult to find efficient iterative equation solvers. Owing to this linear algebra difficulty we have in this work used the following approach. Let P_n and S_w be approximated as

$$P_n = \sum_{j=1}^{m} N_j(\mathbf{x})\pi_j(t), \tag{11}$$

$$S_w = \sum_{j=1}^{m} N_j(\mathbf{x})\sigma_j(t), \tag{12}$$

where N_j are trial functions and π_j and σ_j are nodal values of P_n and S_w, respectively. Multilinear trial functions have here been employed for both pressure and saturation. Inserting the approximate form of P_n and S_w in the partial differential equations results in a residual which is forced to vanish in a weighted mean over the reservoir. The terms with second-order derivatives are integrated by parts to relax the regularity requirements on the trial functions. Owing to the hyperbolic nature of equation (10), a Petrov–Galerkin formulation is adopted for this equation with weighting functions of the form

$$\hat{N}_i = N_i + \kappa_1 V f_w' \mathbf{v}_t \cdot \nabla N_i + \kappa_2 V \frac{f_w' \mathbf{v}_t \cdot \nabla S_w}{\| \nabla S_w \|_2^2} \nabla S_w \cdot \nabla N_i, \tag{13}$$

where κ_1 and κ_2 are scalar parameters. Choosing $\kappa_1 = \kappa_2 = 0$ results in the standard Bubnov–Galerkin method. The term associated with κ_1 is the well known streamline–diffusion term from Reference 16 and adds diffusion in the direction parallel to the total filtration velocity. The κ_2-term represents a new shock-capturing operator as proposed by Hughes *et al.*[17] and adds diffusion in the direction normal to fronts in S_w. We have tried four formulations of the type (13) corresponding to different choices of κ_1 and κ_2. These formulations are abbreviated PG1, PG2, PG3 and PG4. Details of their mathematical forms are given in Appendix I. PG1 and PG2 use

spatial information (element dimensions, etc.) for determining κ_1 and κ_2. In PG1 only the κ_1-term is included. PG3 and PG4 are much simpler in that a global temporal criterion is used: $\kappa_1 = \Delta t/2$, $\kappa_2 = 0$ (PG3) or $\kappa_1 = \kappa_2 = \Delta t/2$ (PG4). The product of the discontinuous weighting function perturbations and terms in the equations involving second-order derivatives are, as usual,[16] simply omitted. Therefore the Petrov–Galerkin formulation in conjunction with multilinear elements cannot be applied to the saturation equation in the form (6). An important advantage of using discontinuous weighting functions on the form (13) is that one obtains a stable spatial discretization similar to upwind finite difference but with minor grid orientation effects. Previous authors[10,11] have investigated grid orientation effects of 'upwind' finite element methods in detail, and their conclusion was that the effect was negligible for all types of finite element methods tested.

Time discretization

The space discretization procedure yields equations for π_i and σ_i that can be written in the form

$$\mathscr{F}_i^{(1)}(\pi_1, \ldots, \pi_m, \sigma_1, \ldots, \sigma_m) = 0, \quad i = 1, \ldots, m,$$

$$\sum_{j=1}^{m} M_{ij} \frac{\mathrm{d}\sigma_j}{\mathrm{d}t} + \mathscr{F}_i^{(2)}(\pi_1, \ldots, \pi_m, \sigma_1, \ldots, \sigma_m) = 0, \quad i = 1, \ldots, m,$$

where $\mathscr{F}_i^{(1)}$ and $\mathscr{F}_i^{(2)}$ are non-linear functions and M_{ij} is a mass matrix. The first equation corresponds to the pressure equation and the second to the saturation equation. For stability and robustness purposes we have employed an implicit discretization in time:

$$\mathscr{F}_i^{(1)}(\pi_1, \ldots, \pi_m, \sigma_1, \ldots, \sigma_m) = 0, \quad i = 1, \ldots, m,$$

$$\alpha \sum_{j=1}^{m} M_{ij}\sigma_j + \Delta t \mathscr{F}_i^{(2)}(\pi_1, \ldots, \pi_m, \sigma_1, \ldots, \sigma_m) = \sum_{j=1}^{m} M_{ij}\tau(\bar{\sigma}_j, \hat{\sigma}_j), \quad i = 1, \ldots, m.$$

Here σ_i denotes the value of S_w at node number i at time level $t = t_n$, $\bar{\sigma}_i$ is the corresponding value at $t = t_{n-1}$ and $\hat{\sigma}_i$ is the value at $t = t_{n-2}$. The scalar α and the function τ are given as

$$\tau(\bar{\beta}, \hat{\beta}) = \bar{\beta}, \qquad \alpha = 1 \tag{14}$$

for a two-level backward Euler scheme and as

$$\tau(\bar{\beta}, \hat{\beta}) = 2\bar{\beta} - \tfrac{1}{2}\hat{\beta}, \qquad \alpha = \tfrac{3}{2} \tag{15}$$

for a three-level backward scheme. Both these schemes are A-stable.[21] The two-level scheme (14) has a truncation error $O(\Delta t)$, while the truncation error of the three-level scheme is $O(\Delta t^2)$. The backward schemes lead to efficient formation of the equation system since the non-linear functions of π_i and σ_i are only evaluated at *one* time level.

The three-level time scheme should be started with a two-level scheme with accuracy $O(\Delta t^2)$. Herein we use (14) as a starter for (15) with Δt replaced by $\Delta t/2$ in the discrete equations. Provided that the injection rates are constant during the first time step, $\lambda_w = 0$ at $t = 0$ and either $\nabla P_n = 0$ or $\lambda_w' = 0$ at $t = 0$, one can show that the two-level backward Euler scheme generates a solution at $t = \Delta t/2$ that coincides with an $O(\Delta t^2)$-accurate Crank–Nicholson scheme for the interval $[0, \Delta t]$. These assumptions hold for all simulations reported in this paper.

Let us define $\bar{S}_w = \Sigma_j N_j \bar{\sigma}_j$ and $\hat{S}_w = \Sigma_j N_j \hat{\sigma}_j$. The discretized form of the pressure equation (4) becomes

$$F_i^{(p)} \equiv \int_\Omega (-V\nabla N_i \cdot \mathbf{v}_t + N_i Q_t)\mathrm{d}\Omega + \int_{\partial\Omega} V N_i \mathbf{v}_t \cdot \mathbf{n}\mathrm{d}\Gamma = 0. \tag{16}$$

The saturation equation (10) is discretized with a Petrov–Galerkin method which leads to

$$F_i^{(s)} \equiv \int_\Omega [\hat{N}_i \alpha \phi S_w + \Delta t(V\hat{N}_i f'_w \mathbf{v}_t \cdot \nabla S_w + V h_w p'_c \nabla N_i \cdot \nabla S_w + W G_w \mathbf{g} \cdot \nabla N_i - \hat{N}_i f_w Q_t)] d\Omega - \int_\Omega \hat{N}_i(\phi\tau(\bar{S}_w, \hat{S}_w) + Q_w) d\Omega - \Delta t \int_{\partial\Omega} V N_i(\mathbf{v}_w - f_w \mathbf{v}_t) \cdot \mathbf{n} d\Gamma = 0. \tag{17}$$

Besides enabling us to apply discontinuous weighting functions, the hyperbolic formulations also offers the advantage of having (8) as a natural boundary condition. Observe that the conditions at impermeable boundaries are also included as natural boundary conditions. All integrals on multilinear elements are computed numerically using Gauss quadrature with 2^d sampling points.

Solution of systems of non-linear equations

The system (16), (17) is highly non-linear in the unknowns σ_j and π_j. Two methods will be outlined for the solution of the non-linear algebraic equations. First a fully implicit Newton–Raphson scheme is presented and then we consider a method where the pressure and saturation equations are solved sequentially at each time level.

Letting $\mathbf{u} = (\pi_1, \sigma_1, \pi_2, \sigma_2, \ldots, \pi_m, \sigma_m)^T$ and $\mathbf{f} = (F_1^{(p)}, F_1^{(s)}, \ldots, F_m^{(p)}, F_m^{(s)})^T$, the Newton–Raphson scheme for the simultaneous solution of the pressure and saturation equations takes the form

$$\mathbf{u}_{i+1} = \mathbf{u}_i + \delta\mathbf{u}_{i+1}, \quad i = 0, 1, \ldots,$$

where $\delta\mathbf{u}_{i+1}$ solves

$$\mathbf{J}(\mathbf{u}_i)\delta\mathbf{u}_{i+1} = -\mathbf{f}(\mathbf{u}_i). \tag{18}$$

Here $\mathbf{J}$ is the Jacobian of $\mathbf{f}$ with respect to $\mathbf{u}$. This implicit solution method is later referred to as the Fully Implicit (FI) algorithm. Details of $\mathbf{J}$ are given in Appendix II. As start vector $\mathbf{u}_0$ for the iteration, the solution $\bar{P}_n$ and $\bar{S}_w$ at the previous time level is used. The iteration is stopped when the Euclidean norm of $\mathbf{f}$ is less than ε_u. The simulations presented in this paper were carried out using $\varepsilon_u = 5 \times 10^{-2}$. Numerical experiments have shown that this is a sufficiently accurate value in our problems.

The Sequentially Implicit (SI) algorithm can be described as follows. At each time level we solve $F_i^{(p)} = 0$, $i = 1, \ldots, n$, with respect to $\pi_1, \ldots, \pi_m$ using saturation values from the previous time step. This approach yields a (linear) Poisson equation for the pressure which in discretized form reads

$$\sum_{j=1}^{m} B_{ij}\pi_j = b_i, \quad i = 1, \ldots, m. \tag{19}$$

B_{ij} and b_i are given explicitly in Appendix II. The non-linear saturation equation $F_i^{(s)} = 0$ is then solved by Newton–Raphson iteration using the pressure values found from equation (19):

$$\sum_{j=1}^{m} \left(\frac{\partial F_i^{(s)}}{\partial \sigma_j}\right)(\delta\sigma_j^{(k)}) = -F_i^{(s)}, \qquad \sigma_j^{(k+1)} = \sigma_j^{(k)} + \delta\sigma_j^{(k)}, \qquad \sigma_j^{(0)} = \bar{\sigma}_j. \tag{20}$$

The advantage with such a solution algorithm is that the size of the matrix system to be solved is reduced from $2m \times 2m$ to $m \times m$. In some cases we will solve equation (19) only at each νth time

step if P_n changes much more slowly (in time) compared to S_w. We have applied the SI algorithm in conjunction with the two-level time discretization scheme exclusively. The SI algorithm has previously been used by Morgan *et al.*[5]

It should be emphasized that the non-linearities due to the modified weighting functions in (13) are treated explicitly when forming the Jacobians in (18) or (20). The impact of this explicit treatment on the convergence of the Newton–Raphson iteration is reported in later sections.

Table I defines some convenient abbreviations. Let H denote the hyperbolic formulation with a two-step time scheme and the FI algorithm. S(ν) denotes the SI algorithm with the two-step time scheme and pressure computation by (19) at each νth time step. H3 is defined as H except that the three-step time scheme is used. We add the symbol BG to indicate that the Bubnov–Galerkin formulation is used ($\kappa_1 = \kappa_2 = 0$). Distinction between consistent and lumped mass matrices is represented by a suffix C or L, such as H-BG-L. Similar formulations with PG1, PG2, PG3 and PG4 are denoted as H-PG1-C, S(ν)-PG2-L and so on.

4. ITERATIVE SOLUTION OF MATRIX SYSTEMS

A disadvantage of our implicit time discretization is that several matrix systems must be solved at each time level. These matrix systems are characterized by being large, sparse and non-symmetric. It is crucial for the overall efficiency and practical applicability of a simulator to apply fast equation solvers that utilize the sparseness. Earlier finite element investigators of reservoir flow have tended to use direct methods, such as Gaussian elimination. Although direct methods are

Table I. Definition of abbreviations for numerical formulations used in the paper

Abbreviation	Description
FI	Fully Implicit algorithm; simultaneous solution for P_n and S_w
SI	Sequentially Implicit algorithm; P_n is found first, then it is implicitly solved for S_w
H	FI algorithm based on the hyperbolic formulation, (9) and (10), with two-level time scheme
S(ν)	SI algorithm based on the hyperbolic formulation, (9) and (10), with two-level time scheme and solution of P_n at each νth time step
H3	As H, but with three-level time scheme
BG	Bubnov–Galerkin formulation ($\kappa_1 = \kappa_2 = 0$)
PG1	Petrov–Galerkin formulation with $\kappa_2 = 0$ and spatial criterion for κ_1
PG2	Petrov–Galerkin formulation with spatial criteria for κ_1 and κ_2
PG3	Petrov–Galerkin formulation with temporal criteria: $\kappa_2 = 0$, $\kappa_1 = \Delta t/2$
PG4	Petrov–Galerkin formulation with temporal criteria: $\kappa_1 = \kappa_2 = \Delta t/2$
C	Suffix for indicating use of consistent mass matrix
L	Suffix for indicating use of lumped mass matrix
ILU(l)	ILU preconditioning with fill-in level l (no fill-in: $l = 0$)
MILU(l)	Modified ILU preconditioning with fill-in level l
OMR(k)	Orthominres iterative equation solver with k recurrence terms
OM(k)	Orthomin method with k recurrence terms
GCR(k)	Generalized Conjugate Residual method with k recurrence terms
OR(k)	Orthores method with k recurrence terms
T	Suffix for indicating use of truncated method (for OMR and OR)
R	Suffix for indicating use of restarted method (for OMR and OR)
BiCG	BiConjugate Gradient method
CGS	Conjugate-Gradient-Squared method

very robust, the computer time and storage are much larger than for iterative methods, especially in 3D problems.[22] In practical reservoir flow problems the finite element mesh, and hence the nodal numbering, is in general expected to be complicated. We will therefore exclusively be concerned with iterative techniques that handle arbitrary matrix sparsity patterns.

The demands made on an iterative method are generality with respect to nodal numbering, robustness, ability to converge quickly on a variety of problems, few user-given parameters and low storage requirements. A family of techniques, known as conjugate-gradient-like iterative methods, meet our demands. These methods include the use of a preconditioner coupled with an acceleration scheme. In contrast to the possibly more efficient multigrid methods, conjugate-gradient-like algorithms are easily implemented to enable any nodal ordering and any number of unknowns per node. The original Conjugate Gradient method is known to be a simple, robust and effective iterative method for problems involving symmetric, positive definite matrices arising from self-adjoint differential operators. During the last decade numerous generalizations to non-symmetric matrix systems have been developed.[23] In reservoir simulation, Orthomin[24] (OM) and its restarted version Generalized Conjugate Residuals (GCR) are the most popular conjugate-gradient-like schemes. It is known from experiments involving linear, scalar convection–diffusion problems[25,26] that other conjugate-gradient-like algorithms may be more reliable and effective than OM/GCR. Therefore it is of interest to test the performance of other methods in two-phase porous media flow. In accordance with previous work[25,26] we have concentrated on five schemes that have proven to be effective and reliable in a wide range of convection–diffusion problems. These are Orthomin (OM), Orthominres[27] (OMR), Orthores[28] (OR), Conjugate Gradients Squared[29] (CGS) and BiConjugate Gradients[30,31] (BiCG). In the ith iteration, OMR, OM and OR require the use of $2i$ previously computed vectors of length equal to the number of unknowns. To reduce storage demands and increase computational efficiency, only vectors from the last $k \leqslant i$ iterations are used. This gives rise to *truncated* versions, here denoted by T-OMR(k), OM(k) and T-OR(k). Another possibility is to restart the methods after every kth iteration, which leads to *restarted* versions: R-OMR(k), GCR(k) and R-OR(k). Observe that GCR(k) is the restarted version of OM(k). Algorithms for CGS, OMR and OM are listed in Appendix III in a form for straightforward implementation. Convenient algorithms for BiCG and OR can be found in References 31 and 32 respectively. Mathematical properties of our five conjugate-gradient-like methods are covered in References 23, 24, 27, 28 and 33.

The efficiency of a basic conjugate-gradient-like method is usually considerably increased by applying a preconditioner. That is, instead of solving the original matrix system $\mathbf{Ax} = \mathbf{b}$, the methods are applied to the equivalent system $\mathbf{M}^{-1}\mathbf{Ax} = \mathbf{M}^{-1}\mathbf{b}$. The preconditioning matrix $\mathbf{M}$ should be a good approximation to $\mathbf{A}$, cheap to compute and have low storage requirements, and systems with $\mathbf{M}$ as coefficient matrix should be effectively solved. A preconditioner suited for matrix systems with arbitrary sparsity patterns is the incomplete LU factorization method. This technique consists of letting $\mathbf{M} = \mathbf{LU}$, where $\mathbf{LU}$ is an approximate LU factorization of $\mathbf{A}$ computed by Gaussian elimination with omission of all fill-in. The factors $\mathbf{L}$ and $\mathbf{U}$, which are lower and upper triangular respectively, have thus the same sparsity pattern as $\mathbf{A}$. To achieve a better approximation to $\mathbf{A}$ one can allow a certain amount of fill-in in $\mathbf{L}$ and $\mathbf{U}$. Let ILU(l) denote incomplete LU factorization with fill-in level l. Rejection of all fill-in corresponds to $l = 0$. The sparsity of level $l \geqslant 1$ factors $\mathbf{L}$ and $\mathbf{U}$ results from multiplication of level $l-1$ factors. Gustafsson[34] suggested adding the fill-in terms to the main diagonal. This technique is known as modified incomplete LU (MILU) factorization. The preconditioners ILU(l) and MILU(l) for matrices with arbitrary sparsity patterns are described and tested on linear convection–diffusion problems in Reference 26. The same methods and computer implementation are employed here. Abbreviations for preconditioners and acceleration schemes are included in Table I.

Comparison and brief descriptions of ours and other popular preconditioning techniques in reservoir simulation are given in e.g. References 18, 19 and 35–37. We emphasize that many of the cited papers employ preconditioners not readily applicable to matrices with arbitrary sparsity patterns. This is particularly the case for nested factorization and other block preconditioners.

The SI formulation requires solution of the (linear) pressure equation (19) where the coefficient matrix is symmetric and positive definite. This system is solved by the standard conjugate gradient method with incomplete Cholesky factorization. Since the equation is linear, values of the pressure at the previous time level are used as start vector for the iteration. We use the null vector as initial guess for the conjugate-gradient-like methods when applied to the equation systems occurring in each Newton–Raphson iteration. The computational labour of an equation solver applied to a specific problem is measured in *work units*. One work unit is defined as one multiplication (division) per unknown.[18] Work units are given to three significant digits. An iterative equation solver is considered as converged when the Euclidean norm of the residual is less than ε_r. Automatic adjustment of ε_r during the Newton iterations is possible,[38] but we have here used a fixed value of $\varepsilon_r = 5 \times 10^{-3}$. Numerical experiments have confirmed that this value gives sufficient accuracy for the problems reported in this paper.

5. ONE-DIMENSIONAL FLOW

The standard Buckley–Leverett test example in a homogeneous rock has been run, but this test was found to be too simple for achieving a clear differentiation between our formulations. Instead, a more complicated one-dimensional test problem has been constructed. Consider a horizontal heterogeneous reservoir with two different rock types as shown in Figure 1(a). The non-wetting fluid (oil) was displaced by the wetting fluid (water). The flow was driven by a prescribed pressure difference ΔP between the injection (source) and production (sink) wells, and S_w was fixed at unity at the injection well. Known pressure values are incorporated as essential boundary conditions in the pressure equation, while known saturation values are incorporated in the saturation equation. In this case essential conditions for the saturation equation occur only at the sink, and the H formulation automatically gives the condition (8) at the source, which can be applied also after water breakthrough. At $t = 0$, $S_w = 0$. Only pressure differences, and not the absolute values, are of physical significance in incompressible flow. For convenience the initial level of P_n was therefore chosen to be zero.

Tables II and III display the values of the physical parameters. Observe that implicit in the formulae for the relative permeabilities are values of zero for the irreducible water saturation and residual oil saturation. The dimensional rock properties in this test problem are of the same order as those used by Spivak *et al.*[2] Owing to the discontinuity in the capillary pressure derivative, and the requirement of continuous phase pressures, a stationary discontinuity in the saturation arises at the interface between rock 1 and rock 2 ($x = 12$ in Figure 1(a)). There will be a saturation shock wave in rock 1. This wave is smeared, owing to capillary effects, in rock 2.

The different numerical formulations were tested on a grid consisting of 40×1 bilinear elements, giving the element length $\Delta x = 1$. Representative saturation profiles for the wetting phase are shown in Figures 1(b)–1(h) at three different time levels, $t = 0{\cdot}1, 0{\cdot}2$ and $0{\cdot}3$ (corresponding to 1000, 2000 and 3000 days respectively). The value of Δt was held fixed at 5×10^{-3} in all simulations. A refined grid with 300×1 bilinear elements and $\Delta t = 5 \times 10^{-5}$ was used for obtaining a reference solution. All formulations based on the two-level time scheme seemed to converge to this reference solution. Banded Gaussian elimination was employed for solving the very narrow band matrix systems in this problem.

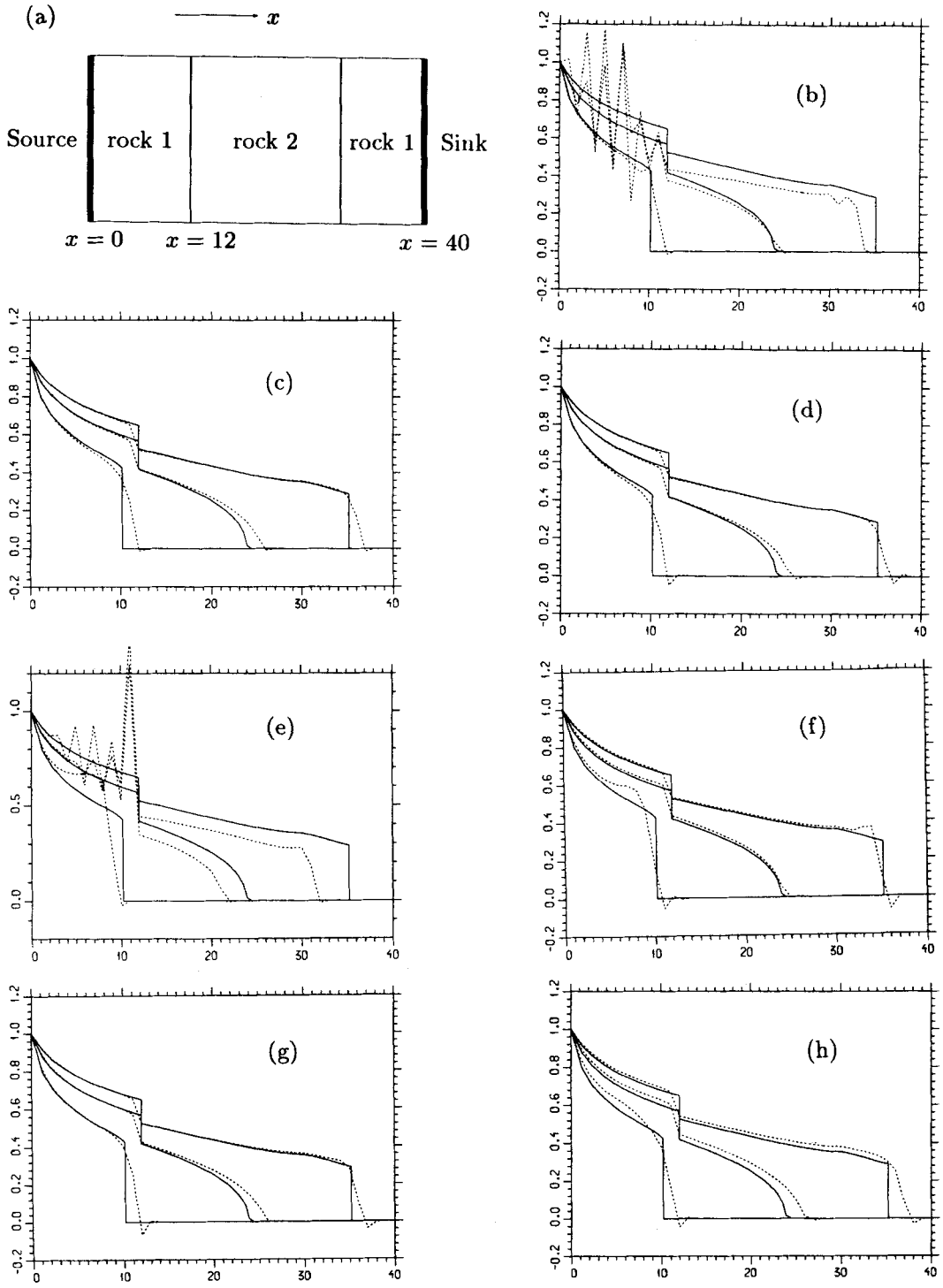

Figure 1. Unidirectional flow in a horizontal, heterogeneous reservoir. $S_w(x, t)$ is plotted as a function of x for three different time levels, corresponding to $t = 0{\cdot}1$, $0{\cdot}2$ and $0{\cdot}3$. The dotted lines represent the numerical solution with $\Delta x = 1$ and $\Delta t = 5 \times 10^{-3}$. A reference solution is indicated by the solid line. (a) Sketch of the one-dimensional test example involving two porous materials with different rock properties; (b) S_w computed by H-BG-L; (c) H-PG1-L; (d) H-PG1-C; (e) H3-BG-C; (f) H3-PG1-L; (g) S(1)-PG1-L; (h) S(10)-PG1-L

Table II. Values of characteristic quantities and rock-independent dimensionless physical, parameters

Parameter	Value
l_c	10·0 m
t_c	$8{\cdot}64 \times 10^8$ s
K_c	$2{\cdot}10 \times 10^{-14}$ m^2
μ_c	10^{-3} kg m^{-1} s^{-1}
P_c	5·5 kPa
μ_w	1·0
μ_n	4·0
ΔP	2000
Δt	5×10^{-3}
V	1·0
W	0

Table III. Values of dimensionless rock properties for the test problem in Section 5

Parameter	Rock 1	Rock 2
Porosity ϕ	0·25	0·35
Absolute permeability K	1·0	1·5
Relative permeability k_{rw}	S_w^2	S_w^2
Relative permeability k_{rn}	$(1 - S_w)^2$	$(1 - S_w)^2$
Capillary pressure p_c'	0	−25·0

All methods based on the standard Bubnov–Galerkin approximation led in this example to node-to-node oscillations in S_w after the shock entered rock 2; see Figure 1(b). The PG4 formulation damped the oscillations slightly. No considerable improvement was achieved by using PG3 in comparison to BG. Satisfactory results were obtained by PG1 and PG2, which in 1D give identical results (see Appendix I). Figure 1(c) shows that H-PG1-L produced saturation profiles where all discontinuities were resolved within two elements, with negligible over- and undershoots. We have found that for the other formulations, which lead to non-physical oscillations around discontinuities, these oscillations were significantly reduced with diminishing element size. Small oscillations downstream of the shock, which were present when applying the consistent mass matrix, were suppressed using the lumped mass matrix. This effect can be seen by comparison of Figures 1(c) and 1(d). The improved performance of mass lumping in this type of non-linear problem is in accordance with the findings of other investigators.[7,11]

The three-level time discretization scheme was clearly inferior to the two-level scheme. H3-BG-C and H-3-BG-L were the only formulations that suffered from wrong shock velocity and shock strength (Figure 1(e)). As the element size was diminished, H3-BG-C and H3-BG-L converged to the wrong solution. Application of the discontinuous weighting functions improved the results considerably, as seen in Figure 1(f). Lumping the mass matrix seemed more demanded for the three-level scheme than for the two-level scheme; for example, H3-PG1-C was clearly inferior to H3-PG1-L. Convergence to the wrong solution for the three-level scheme also occurred in the standard Buckley–Leverett problem.

The variants of the FI algorithm required two to three Newton iterations per time step, except when combined with PG2 which demanded about one extra Newton iteration. This slower convergence is expected since the PG2 formulation introduces significant non-linearities which presently are treated explicitly in the Newton–Raphson scheme (see Appendix II). The SI methods needed up to twice as many Newton iterations as the corresponding FI formulations. For both families of methods the convergence rate was slightly improved when the (smeared) saturation front moved through rock 2. The results of the SI algorithm were in accordance with those obtained by the FI algorithm, and the accuracy was only slightly inferior to the latter, as seen in Figure 1(g). Large values of v led to further reduced accuracy, and Figure 1(h) shows an example of S(10)-PG1-L.

It is known *a priori* that S_w must lie in an interval $[S_{wr}, S_{wM}]$. In the rest of the paper we have employed this information and consequently set non-physical S_w values equal the appropriate upper or lower limit. This approach avoids undershoots downstream of the front, for example.

6. TWO-DIMENSIONAL FLOW

In this section the different numerical formulations are applied to a two-dimensional analogue to our one-dimensional test example in Section 5. Figure 2 shows a horizontal reservoir containing two porous media. The wetting fluid was injected along a line on the boundary and production took place in two point-wells. The rest of the boundary was assumed to be impermeable. Physical parameters are as in Table II and the rock properties are listed in Table IV. Values of Δt smaller than 5×10^{-3} were necessary at the beginning of the simulations. Initially, $S_w = P_n = 0$. During the initial stages of the displacement a saturation shock propagates through rock 1. As this shock enters rock 2, where capillary effects are significant, it is diffused. However, a part of the shock wave moves outside rock 2 and maintains its sharp front. The discontinuity in p_c' leads also in this problem to a stationary saturation discontinuity at parts of the boundary between the two media. The finite element mesh depicted in Figure 3 is refined along the boundary between rock 1 and 2 in order to resolve the rapid variations in the saturation function tangentially to the front. Numerical comparison with a similar physical problem in a geometry with straight boundaries showed that the present problem made greater demands on the numerical solution methods.

The different formulations resulted in almost identical pressure fields. Figure 4 gives an example of the smoothly varying P_n function. We mention that special considerations must often be taken to numerically represent steep pressure gradients in the vicinity of wells.[13, 39] This is not strictly required here since the gradients are moderate owing to finitely prescribed well pressures.

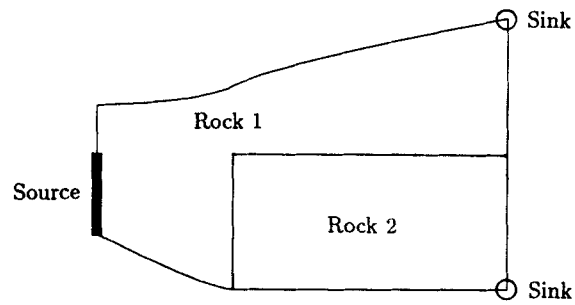

Figure 2. Sketch of 2D flow problem in a reservoir containing rocks with different properties. The area of the domain is 478 (dimensionless)

Table IV. Values of dimensionless rock properties for the test problem in Section 6

Parameter	Rock 1	Rock 2
Porosity ϕ	0·25	0·40
Absolute permeability K	1·0	1·5
Relative permeability k_{rw}	$S_w^{3.5}$	$S_w^{3.5}$
Relative permeability k_{rn}	$(1-S_w)^2$	$(1-S_w)^2$
Capillary pressure p_c'	0	$-15{\cdot}0$

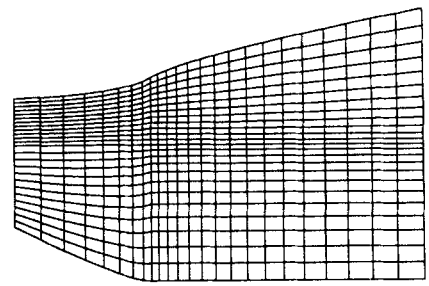

Figure 3. Finite element mesh corresponding to the problem depicted in Figure 2. There are 576 elements with 625 nodes

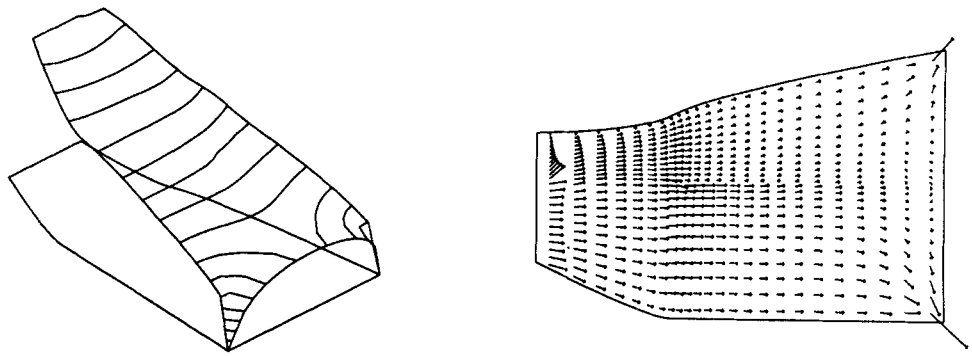

Figure 4. Pressure and total filtration velocity at $t = 0{\cdot}2$ computed by H-PG2-L for the problem in Figure 2. Left: elevated contour lines representing P_n. Right: velocity vectors representing $\mathbf{v}_t$

The standard H-BG-L formulation produced poor results for the saturation field. Typical features were node-to-node oscillations as shown in Figure 5. The location and shape of the front were also inaccurate. We have implemented and tested the FI algorithm with the saturation equation of the form (6). This formulation gave better results with respect to the front propagation velocity. The node-to-node oscillations were nevertheless still present. The same conclusion also applied to the 1D problem in Section 5. The PG1 formulation led to considerable improvements (Figure 6), but PG2 clearly gave the best results (Figure 7). The superior behaviour of PG2 compared with PG1 in this flow case can be explained. From the definition of κ_2 in the PG2 formulation (see Appendix I) we see that κ_2 vanishes if the saturation front is normal to the

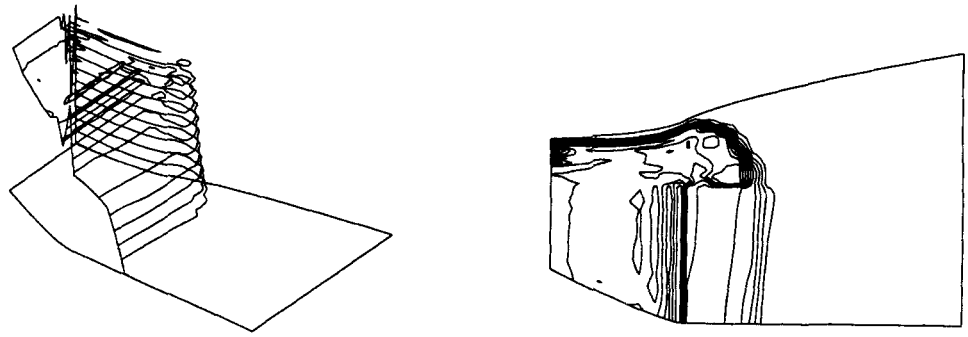

Figure 5. S_w computed by H-BG-L plotted as a function of the space co-ordinates for $t = 0{\cdot}2$ (2000 days). The physical problem is depicted in Figure 2. Left: elevated equidistant contour lines. Right: plane equidistant contour lines

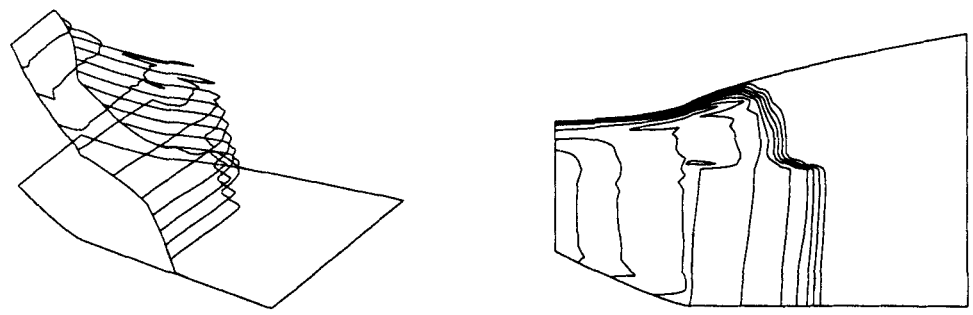

Figure 6. S_w computed by H-PG1-L. See caption to Figure 5

streamlines. This is to a large extent the situation in many simple flow problems, e.g. the five-spot well pattern. However, in this flow problem the steep saturation gradients are not always normal to the streamlines and therefore the κ_2-term will have a significant effect. In contrast to the κ_1-term, the κ_2-term adds diffusion normal to the thin saturation boundary layer and hence suppresses the oscillations seen in Figure 6. Temporal criteria for κ_1 and κ_2 were again clearly inferior to the spatial criteria.

The three-level scheme in combination with PG2 led to inferior results compared to the two-level scheme; see Figure 8 for an example of H3-PG2-L. Figure 9 shows a fine mesh and a saturation field to be compared with that in Figure 7. We see that the wavy contour lines behind the front are still present.

All results in Figures 4–9 were computed with Δt fixed at 5×10^{-3}. With this value the Newton–Raphson method needed one to four iterations. Longer steps were possible in parts of the simulation, but the value of Δt then fluctuated, which was less convenient (and not always more efficient) when several methods were to be compared. Smaller values of Δt gave results in accordance with those for $\Delta t = 5 \times 10^{-3}$. In this example the SI algorithms were significantly less robust than the fully implicit schemes, especially when combined with PG2. The SI algorithms required smaller time steps in order for the Newton iteration to converge: $\Delta t \leqslant 1{\cdot}25 \times 10^{-3}$. The Newton–Raphson method also required up to twice as many iterations as needed in the FI algorithm. Particularly behind the front, the S(v)PG2-L method gave less accurate

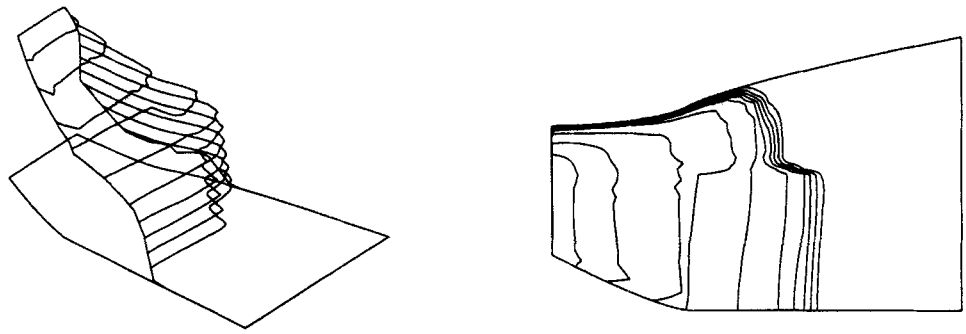

Figure 7. S_w computed by H-PG2-L. See caption to Figure 5

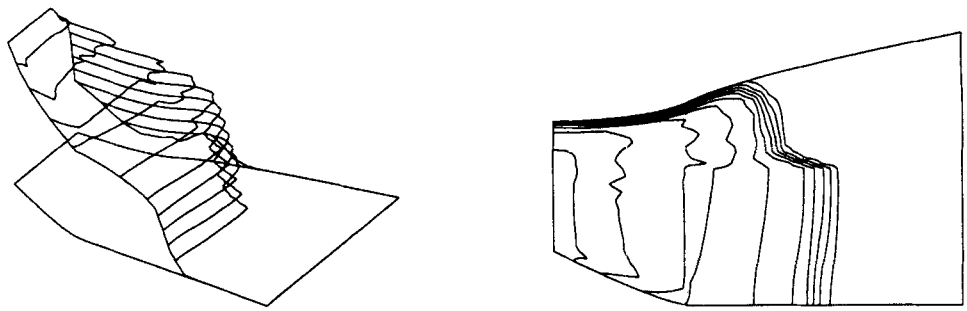

Figure 8. S_w computed by H3-PG2-L. See caption to Figure 5

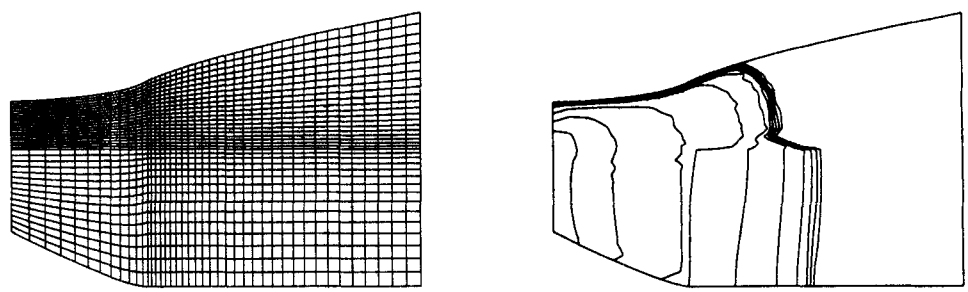

Figure 9. Left: refined mesh with 1470 elements. Right: S_w at $t = 0{\cdot}2$ computed by H-PG2-L on the refined mesh

results than H-PG2-L. Our experience indicates that v may be chosen large, say $v = 10$ in this example, without further significant reduction of accuracy.

In our implementation the SI method is, for large values of v, about twice as fast as the FI algorithm with respect to computation of the matrix systems. The time spent on iterative solution of the linear systems in the SI algorithm is in this problem almost negligible in comparison with the calculation of the matrices. In the FI algorithm, iterative solution of the systems is about as time-consuming as the computation of the systems. Taking into account that the SI algorithm usually needs more Newton iterations per time step than the FI method, the SI solution strategy is then two to three times faster than the FI strategy, provided that Δt is

the same for both. In this particular problem where the SI algorithm required about four times as many time steps as the FI version, the latter turned out to be the most efficient approach.

7. THREE-DIMENSIONAL FLOW

Since there is no major dependence on the number of space dimensions in the proposed numerical methods, they can easily be implemented to treat both two- and three-dimensional flow cases within the same code lines. It is also to be expected that the accuracy of the methods should be the same in 3D as in 2D. In order to demonstrate this assertion, a simple three-dimensional flow problem is considered next. Figure 10 shows a homogeneous reservoir shaped as a deformed cube. A relatively coarse mesh was used, consisting of $10 \times 10 \times 10$ trilinear bricks. At $t = 0$ the reservoir contained the non-wetting fluid only, at hydrostatic pressure. The

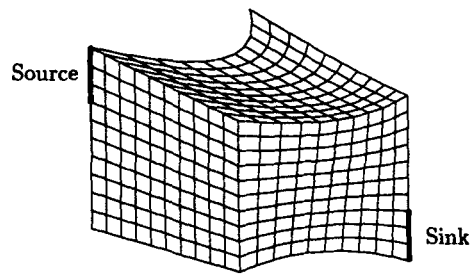

Figure 10. Sketch of three-dimensional flow problem in a homogeneous medium. The volume of the reservoir is 29 400 (dimensionless)

Table V. Values of physical parameters for the test problem in Section 7

Parameter	Value
l_c	10·0 m
t_c	$1{\cdot}73 \times 10^6$ s
K_c	$1{\cdot}77 \times 10^{-12}$ m^2
μ_c	10^{-3} kg m^{-1} s^{-1}
P_c	32·7 kPa
ρ_c	1000 kg m^{-3}
g_c	9·81 m s^{-2}
μ_w	1·0
μ_n	4·0
ρ_w	1·0
ρ_n	0·833
ΔP	20
Δt	0·5
V	1·0
W	0·03
ϕ	0·25
K	1·0
k_{rw}	S_w^2
k_{rn}	$(1 - S_w)^2$
P_c'	0

wetting fluid was injected at four nodes (where $S_w = 1$). The production well also consisted of four nodes. Values of the physical parameters are listed in Table V.

The ILU(0) preconditioned T-OMR(5) was used as equation solver in this and in the previous section. A report on the behaviour of other conjugate-gradient-like algorithms for these problems is given in the next section. Figure 11 shows the intersection of the reservoir boundary and equidistant contour surfaces of pressure and saturation. Taking into account the coarse mesh and that $p_c' \approx 0$, the results must be regarded as good and indicate that the numerical methods work as expected from similar 2D simulations. H-PG1-L was used for producing the results in Figure 11. The differences between PG1 and PG2 were small since the saturation front was approximately normal to the streamlines (see the discussion in Section 6). S(25)-PG1-L was also tested and gave results almost identical to those in Figure 11. However, the SI algorithm required twice as many Newton iterations per time step and occasionally smaller time steps than the FI version. The overall efficiency was therefore about the same for the two approaches when using the grid in Figure 10. Superior efficiency of SI compared with FI is expected as the number of elements is increased. Algthough the present problem involved more elements than the problem in Section 6, it required only about four times as much work per time step.

8. BEHAVIOUR OF ITERATIVE EQUATION SOLVERS

Development and comparison of iterative equation solvers frequently use the stationary, linear, scalar convection–diffusion equation as model problem. One of the clearest conclusions from comparison experiments with the convection–diffusion equation is that the results are highly problem-dependent.[25, 26] The main purpose of this section is therefore to present some experimental convergence results for a more physically complicated problem. Two-phase

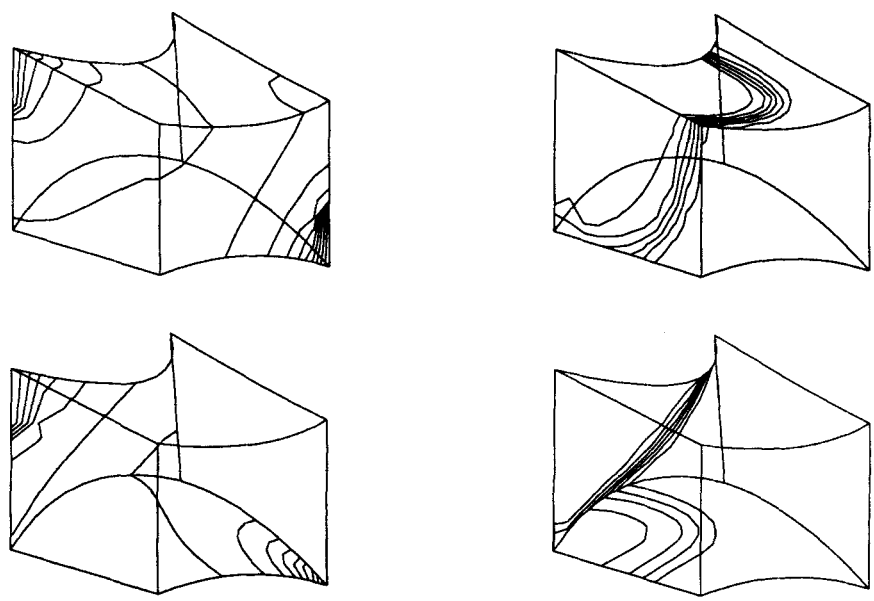

Figure 11. Intersection of the reservoir boundary and equidistant pressure (left) and saturation (right) contour surfaces. Results are computed by H-PG2-L and shown for $t = 100$ (2000 days). Top: facing boundary. Bottom: hidden boundary

porous media flow is described by differential equations of convection–diffusion nature but with two unknowns per node (FI algorithm). This leads to less sparse matrices which implies that the matrix–vector product(s) will consume a larger portion of the total work per iteration compared to the case with a single unknown per node.

Besides the problems described in Sections 6 and 7, a simpler homogeneous two-dimensional problem has to a large extent been used for testing equation solvers. Figure 12 depicts this problem. Owing to symmetry, only the upper half of the reservoir needs to be discretized. Source and sink nodes with specified P_n values are indicated by the thick lines in Figure 12. S_w was kept equal to unity at the source. Values of the physical parameters are as in Table II and Figure 12. For all simulations we used a grid with 40×20 bilinear elements and fixed Δt. Figure 13 shows examples of the wetting fluid saturation field. Cases with significant capillary effects were satisfactorily treated by all formulations. The differences between Bubnov–Galerkin and Petrov–Galerkin formulations for vanishing capillary pressure derivative were much smaller here than in the problems in Sections 5 and 6. For many practical purposes the Bubnov–Galerkin formulation gives acceptable results in simple problems like this.

Average work units (and iterations) over some time steps have been calculated for different combinations of preconditioners and acceleration schemes in several reservoir flow cases. The main results are summarized below.

Fully implicit formulation

First we discuss the equation solvers applied to the matrix systems associated with the FI algorithm. The behaviour of the equation solvers showed little sensitivity to various values of the time step length. The number of Newton iterations was of course independent of the equation solver being used as long as the equation solver converged properly and fulfilled the termination criterion. Divergence of the five *preconditioned* equation solvers used here (OMR, OM, CGS, OR and BiCG) was due to an underlying divergence of the Newton method. This assertion was justified by applying banded Gaussian elimination as equation solver in selected cases. Reduction of the time step length made the Newton method converge and thus also the conjugate-gradient-like schemes. The number of iterations required by the conjugate-gradient-like methods was usually about the same for each Newton iteration.

Table VI shows the average performance of some ILU(l) preconditioner and conjugate-gradient-like schemes in the problem from Figure 12. We have also tested the MILU(l)

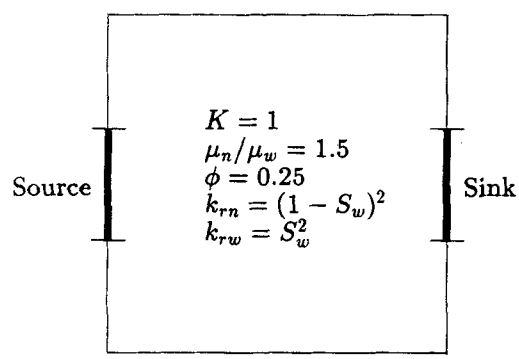

Figure 12. Sketch of a two-dimensional test example involving a horizontal homogeneous reservoir. The size of the reservoir is $[0, 40] \times [0, 40]$

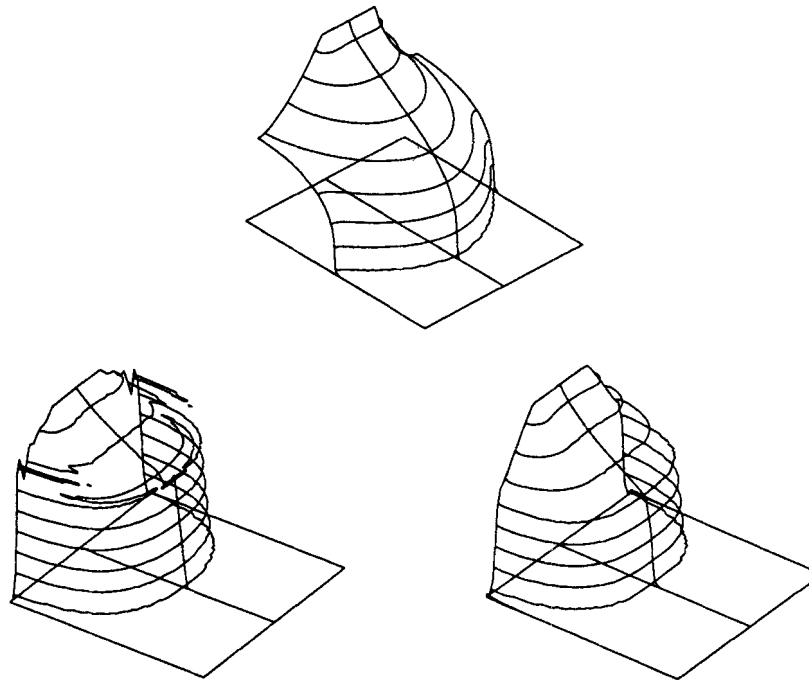

Figure 13. S_w at $t = 0{\cdot}2$ for the problem depicted in Figure 12. Functions are visualized by elevated equidistant contour lines. Top: $p'_c = -10$, computed by H-BG-L. Bottom left: $p'_c = 0$, computed by H-BG-L. Bottom right: $p'_c = 0$, computed by H-PG1-L

Table VI. Work units, with number of iterations shown in parentheses, based on average over three time steps in a simulation corresponding to Figure 12; 800 bilinear elements, 1722 unknowns, $\Delta t = 5 \times 10^{-3}$ and $p'_c = 0$; H-PG1-L formulation

	Preconditioning		
Method	ILU(0)	ILU(1)	ILU(2)
CGS	1040 (12)	1040 (9)	1030 (6)
T-OMR(5)	887 (8)	842 (5)	921 (4)
BiCG	1410 (17)	1240 (11)	1210 (8)
OM(5)	913 (15)	835 (10)	869 (7)
T-OR(5)	1170 (20)	1000 (13)	881 (8)

preconditioner, but ILU performed in general clearly better than MILU. However, in combination with OM(k), MILU was occasionally competitive with ILU and in a few cases even superior. All formulations except H-PG2-C gave similar convergence results for the iterative methods (H-PG2-C led to slower convergence). Tables VII and VIII show the average performance of conjugate-gradient-like methods applied to the problems in Sections 6 and 7 respectively. ILU(l) increases the storage of the preconditioning matrix by a factor of $1{\cdot}44^l$ for bilinear elements and natural node numbering. The corresponding factor for trilinear

elements is $2{\cdot}1^l$. Previous investigations[26] showed clear inefficiency of $l \geqslant 1$ in 3D owing to the significant decrease in sparsity with increasing l. Therefore only ILU(l) for $l = 0$ was tested in 3D. Comparing Tables VI and VII we see that ILU(1) is the most efficient preconditioner. However, the small differences between ILU(1) and ILU(0) favour ILU(0) because of its lower storage requirements. None of the tested equation solvers has been observed to diverge (or converge very slowly) in combination with ILU(l) preconditioning, provided the Newton iteration converged. Without preconditioning, the equation solvers converged much more slowly, and occasionally several hundred iterations were needed to fulfil the termination criterion. In some cases non-preconditioned equation solver algorithms broke down.

A difficulty in applying methods with long recurrences is to choose between the restarted and the truncated version and to find an optimal value of k. Our numerical experiments indicate that OM is most efficient in the truncated version, while OR and OMR are less sensitive to the choice of restart or truncation. The optimal value of k for OM(k), T-OMR(k) and R-OMR(k) was usually $k \leqslant 10$, and $k = 5$ is recommended as an all-round choice. GCR(k) performed most efficiently for somewhat larger k. OR was in general less effective than OMR and OM. In linear, scalar convection–diffusion problems $k = 1$ usually represents the optimal k value.[25] This is seldom the case in multiphase reservoir problems. For example, GCR(1), which is very effective in linear convection–diffusion problems, converged slowly in our test examples. Table IX–XI show how the numbers of work units and iterations change with restart/truncation and the value of k for ILU(0)-preconditioned OM/GCR and OMR.

Table VII. Work units, with number of iterations shown in parentheses, based on average over three time steps in a simulation corresponding to Figure 2; 576 bilinear elements, 1250 unknowns, $\Delta t = 2{\cdot}5 \times 10^{-3}$ and physical parameters as in Section 6; H-PG2-L formulation

	Preconditioning		
Method	ILU(0)	ILU(1)	ILU(2)
CGS	791 (9)	723 (6)	775 (4)
T-OMR(5)	657 (5)	691 (4)	730 (3)
BiCG	1000 (12)	906 (8)	931 (5)
OM(5)	770 (12)	672 (8)	697 (5)
T-OR(5)	927 (15)	664 (8)	691 (5)

Table VIII. Work units, with number of iterations shown in parentheses, based on average over three time steps in a 3D simulation corresponding to Figure 10; 1000 bilinear elements, 2662 unknowns, $\Delta t = 0{\cdot}25$ and physical parameters as in Section 7; H-PG1-L formulation

Method	ILU(0)
CGS	2130 (8)
T-OMR(5)	1420 (4)
BiCG	2070 (8)
OM(5)	1520 (8)
T-OR(5)	2180 (15)

Table IX. Work units, with number of iterations shown in parentheses, based on average over three time steps in a simulation corresponding to Figure 12; 800 bilinear elements, 1722 unknowns, $\Delta t = 5 \times 10^{-3}$ and $p'_c = 0$; ILU(0) preconditioning and H-PG1-L formulation

Method	$k = 1$	$k = 3$	$k = 5$	$k = 7$	$k = 9$
T-OMR(k)	2050 (14)	992 (10)	887 (8)	952 (8)	1050 (8)
R-OMR(k)	1980 (23)	1120 (12)	992 (10)	895 (8)	917 (8)
OM(k)	808 (16)	865 (15)	913 (15)	994 (14)	1090 (15)
GCR(k)	1430 (32)	1310 (27)	1080 (21)	917 (16)	1060 (18)

Table X. Work units, with number of iterations shown in parentheses, based on average over three time steps in a simulation corresponding to Figure 2; 576 bilinear elements, 1250 unknowns, $\Delta t = 2{\cdot}5 \times 10^{-3}$ and physical parameters as in Section 6; ILU(0) preconditioning and H-PG2-L formulation

Method	$k = 1$	$k = 3$	$k = 5$	$k = 7$	$k = 9$
T-OMR(k)	1460 (17)	805 (8)	657 (5)	703 (5)	750 (5)
R-OMR(k)	1580 (19)	822 (8)	668 (6)	622 (5)	645 (5)
OM(k)	738 (15)	791 (14)	770 (12)	748 (10)	813 (10)
GCR(k)	1190 (26)	997 (20)	760 (14)	873 (15)	688 (11)

Table XI. Work units, with number of iterations shown in parentheses, based on average over three time steps in a 3D simulation corresponding to Figure 10; 1000 bilinear elements, 2662 unknowns, $\Delta t = 0{\cdot}25$ and physical parameters as in Section 7; ILU(0) preconditioning and H-PG1-L formulation

Method	$k = 1$	$k = 3$	$k = 5$	$k = 7$	$k = 9$
T-OMR(k)	5480 (25)	1420 (4)	1420 (4)	1450 (4)	1480 (4)
R-OMR(k)	5150 (24)	1610 (5)	1470 (4)	1390 (4)	1410 (4)
OM(k)	1450 (9)	1540 (9)	1520 (8)	1590 (9)	1620 (8)
GCR(k)	3090 (26)	2550 (20)	1530 (9)	1810 (12)	1680 (10)

Our experiments indicate that OM and OMR were the best acceleration schemes, especially in 3D. The CGS method, which is among the very best when solving scalar convection–diffusion problems, was somewhat inferior here. Since a very large portion of the work per iteration in the CGS method is spent on the two matrix–vector products, CGS is more sensitive to the decrease in sparsity that occurs when the number of unknowns per node is increased from one to two, and when the number of non-zeros per row is increased by a factor of three when going from two to three space dimensions. Without preconditioning, CGS was the best method. However, preconditioning seems in general to be required both for stabilizing the iterative schemes and for increasing the efficiency.

It is of interest to investigate how the preconditioners behave when the matrix sparsity pattern becomes irregular, which may happen for complicated geometries or adaptively refined grids.[26] The results in Table VII correspond to a regular (row-by-row) node numbering and hence a standard nine-band sparsity pattern, where each band consists of 2×2 blocks. An irregular node numbering produced by a general preprocessor technique[26] gave the sparsity pattern shown in Figure 14. This type of sparsity pattern is also generated by certain adaptive mesh techniques.[26] The preprocessor assumes that the domain is divided into several simple

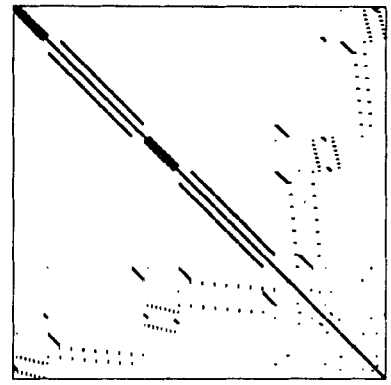

Figure 14. Matrix sparsity pattern associated with a node numbering produced by a general preprocessor technique applied to the problem in Figure 2

subdomains (four in the present case). The internal nodes within each subdomain are regularly numbered and then the nodes on the boundaries between the subdomains are given numbers. Compared to the case with regular node numbering, the ILU(0) preconditioner was still robust, but we experienced roughly a doubling of the number of iterations. This must be regarded as satisfactory taking into account the large increase in fill-in entries when the sparsity pattern becomes complex. The decrease in efficiency was however larger than in the case with only one unknown per node.[26]

Sequentially implicit formulation

The picture changes drastically when we consider iterative solution of the matrix systems occurring in the SI algorithm. To update the pressure only one to two iterations are necessary, occasionally slightly increasing with ν. The solution of the matrix system in equation (20) usually required one to four iterations when $p'_c = 0$. A couple of additional iterations were needed when the saturation front moved through the medium with significant diffusion (capillary) effects. Most of the work is thus spent on calculating the matrix system and not on the solution process. It is therefore somewhat irrelevant to search for an optimal equation solver, and the most reliable method mentioned above is recommended. In some cases it is even beneficial to apply the equation solvers without preconditioning, at a cost of considerably decreased reliability.

It is known that second-order elliptic equations, such as our pressure equation, give rise to matrix systems where the coefficient matrices have a condition number $O(h^{-2})$, h being a measure of the element size. First-order hyperbolic equations lead to better conditioned matrices, typically $O(h^{-1})$. Therefore iterative solution of matrix systems arising from the saturation equation will be more effective than iterative solution of systems from the pressure equation, provided that $|p'_c|$ is small. When using the FI algorithm the pressure equation dictates the conditioning of the Jacobian, while in the SI algorithm the two equations give rise to two differently conditioned matrix systems. Thus a sequential solution method diminishes the size of the matrix systems, takes advantage of the good conditioning of the saturation equation and enables the use of a very good start vector for the iterative solution of the pressure equation.

All numerical results in this paper have been obtained by a FORTRAN 77 code developed by the author and run in single precision on an FPS-164 array processor (64-bit arithmetic).

9. CONCLUSIONS

Several implicit and robust finite-element-based methods for the equations governing incompressible, immiscible, two-phase porous media flow have been described and tested. The main subjects of this paper concern adaptation and investigation of a family of discontinuous weighting functions, demonstration of the very poor behaviour of traditional finite element formulations in a certain class of heterogeneous problems, and use of general and effective preconditioners in combination with conjugate-gradient-like acceleration schemes for solving matrix systems. The main results can be summarized as follows.

In problems where capillary effects were significant throughout the reservoir, the pressure and saturation fields were smooth and any of the proposed formulations worked well. Difficulties were encountered when sharp saturation fronts developed in rocks with negligible capillary effects. In this case standard Bubnov–Galerkin formulations gave oscillations in the vicinity of propagating saturation fronts. These oscillations were to a large extent suppressed by any of the proposed Petrov–Galerkin formulations. The pressure field seemed to be handled accurately by all methods tested. In problems where the rock properties were homogeneous throughout the domain, the differences between the results from the various numerical formulations were relatively small provided the entropy condition was fulfilled. However, this picture changed drastically for the heterogeneous problems considered. Discontinuous capillary pressure properties introduced significant oscillations and other inaccuracies, such as wrong propagation velocity, in the saturation fields for many of the methods, including all standard Bubnov–Galerkin formulations. Promising results were obtained with a new Petrov–Galerkin method employing weighting functions that add diffusion both along streamlines and in the direction normal to the saturation front. This Petrov–Galerkin formulation did not increase the width of the shock fronts in comparison to the standard Bubnov–Galerkin formulation, and fronts usually extended throughout two multilinear elements. Optimizing parameters in the weighting functions should be based on spatial rather than temporal criteria. It is important to note that in order to apply the present discontinuous weighting functions, the saturation equation must be written in 'hyperbolic' form.

The S(v) formulation, where the pressure and saturation equations are numerically decoupled and P_n is solved for only at each vth time step, worked very efficiently in simpler flow problems. The main advantages of this approach are smaller matrix systems to be solved and the possibility for pressure computation at only a few time levels. In addition, the matrix systems are more effectively solved owing to good starting values for the pressure and good conditioning of the coefficient matrix arising from the saturation equation (provided $|p_c'|$ is small). The seqential formulation also offers the possibility of using different numerical methods for the elliptic pressure equation and the (almost-)hyperbolic saturation equation. However, the convergence of the Newton–Raphson iteration was considerably slower and the accuracy was usually inferior compared to the fully simultaneous H formulation. In more complicated problems, such as that in Section 6, the decoupling procedure led to decreased reliability and a requirement for smaller time steps. The SI algorithm is usually more efficient than the FI algorithm provided that the time step length is the same for both. In simpler problems the SI algorithm is generally the recommended solution scheme, but in physically complicated cases a fully implicit approach is much more reliable and occasionally also more efficient. In a particular problem it is difficult to evaluate the performance of a sequential approach without having a fully implicit simulator for comparison.

The time discretization was carried out by A-stable backward differences. It seemed that the simple two-level backward Euler scheme performed better than the three-level second-order scheme. For the linear, scalar convection–diffusion equation the Euler scheme can be shown to

produce an $O(\Delta t)$ diffusion in the streamline direction. We think that this diffusion, which is not provided by the three-level scheme, is advantageous in the present non-linear problems to stabilize solutions around sharp fronts and contribute to the fulfilment of the entropy condition.

Five acceleration schemes were tested in combination with ILU(l) preconditioning. Taking into account both efficiency and storage requirements, ILU(0) is the recommended preconditioner. However, in difficult problems $l = 1$ or $l = 2$ may be more reliable choices. T/R-OMR(k) and OM(k) turned out to be the most effective acceleration schemes, with R-OMR(k) slightly superior to the others. In contrast to scalar convection–diffusion problems where $k = 1$ is frequently the optimal k value, larger values of k represented the optimal choice in our test problems, especially if curved boundaries and heterogeneities were present. The recommended all-round value for OMR and OM is $k = 5$. The main impression is that OM(5) and T/R-OMR(5) with ILU(0) preconditioning are very reliable and effective methods in all two-phase flows that have been tested here.

The clearest conclusion to be drawn from the equation solver experiments is that the behaviour of the methods in 3D is different from that in 2D. This is due to the smaller number of non-zeros per matrix row in 3D, which leads to a significant increase in the work spent on calculating matrix–vector products. Finite difference methods do not exhibit the significant decrease in sparsity when going from 2D to 3D. Convergence results for equation solvers applied to finite-difference-generated matrices in 3D may therefore not be relevant to matrix systems arising from finite element discretization.

The advantages of the present finite element methods over corresponding traditional finite difference methods are trivial implementation of boundary conditions at impermeable boundaries and at wells with production proportional to mobility, less cross-wind diffusion and thus less grid orientation effects associated with 'upwinding' techniques, and, finally (and maybe most important), easy handling of complex reservoir geometries and potentially greater flexibility in construction of grids with local refinements. Our iterative matrix system solvers allow any structure of the non-zeros and hence any nodal numbering. However, this high degree of generality and robustness decreases the efficiency in comparison with finite diference methods on rectangle- or box-shaped domains. Two strategies are suggested for improving the efficiency. First, integrals should be calculated by one-point quadrature. This will particularly increase the efficiency of 3D simulations. Straightforward application of one-point Gauss quadrature has been tested here without success. Modifications in terms of stabilization matrices[40] seem therefore to be required. Secondly, a more vectorizable preconditioner for irregular matrix sparsity patterns should be found. Adaptive, local grid refinements around saturation fronts would represent another important improvement.

Finally, we mention that the solution methods in this paper are in principle straightforwardly applicable to more physically complicated problems, such as black-oil or compositional models.

APPENDIX I: PARAMETERS FOR THE WEIGHTING FUNCTIONS

In the weighting functions given in (13), κ_1 and κ_2 are scalar parameters for which we have adapted ideas from the literature on convection–diffusion equations.[16,17,41]

Spatial criteria

Let ξ_i denote the co-ordinates in the local element frame. The Jacobian D_{ij} of the mapping from the local frame onto the physical frame can then be written as

$$\tilde{D}_{ij} = \partial x_j / \partial \xi_i.$$

The Jacobian of the inverse transformation, i.e. the inverse of $\tilde{D}_{ij}$, is denoted

$$D_{ij} = \partial \xi_j / \partial x_i.$$

A 'diffusion' parameter ε is defined as

$$\varepsilon = |p_c'| h_w.$$

Let

$$\Theta(\gamma) = \begin{cases} \gamma/3, & |\gamma| \leqslant 3, \\ \operatorname{sign} \gamma, & |\gamma| > 3. \end{cases}$$

Adapting ideas from Hughes *et al.*[17] we have

$$\kappa_1 = (1/\beta)\Theta(\gamma), \tag{21}$$

$$\gamma = (f_w')^2 \| \mathbf{v}_t \|_2^2 / \varepsilon \beta, \tag{22}$$

$$\beta = \left(\sum_{k=1}^{d} b_k^p \right)^{1/p}, \tag{23}$$

$$b_i = f_w' \sum_{k=1}^{d} D_{ik} (\mathbf{v}_t)_i \tag{24}$$

and

$$\kappa_2 = \max(0, \kappa_\| - \kappa_1), \tag{25}$$

$$\kappa_\| = (1/\beta)\Theta(\gamma), \tag{26}$$

$$\gamma = (1/\varepsilon\beta)\, \delta f_w' \mathbf{v}_t \cdot \nabla S_w, \tag{27}$$

$$\delta = f_w' \mathbf{v}_t \cdot \nabla S_w / \| \nabla S_w \|_2^2, \tag{28}$$

$$\beta = \left(\sum_{k=1}^{d} b_k^p \right)^{1/p}, \tag{29}$$

$$b_i = \delta \sum_{k=1}^{d} D_{ik} \frac{\partial S_w}{\partial x_i}. \tag{30}$$

In this paper we have used $p = 2$. $(\mathbf{v}_t)_i$ denotes the ith component of $\mathbf{v}_t$.

In the PG2 formulation both κ_1 and κ_2 are as given above. κ_2 is set to zero in the PG1 formulation. Observe that PG1 and especially PG2 increase the non-linearity in the basic discrete equations. Non-linearities due to the weighting functions are treated explicitly when forming Jacobi matrices in the Newton–Raphson scheme.

Temporal criteria

A method related to the Taylor–Galerkin schemes for hyperbolic systems[42] is obtained by setting $\kappa_2 = 0$ and $\kappa_1 = \Delta t/2$; see also Reference 41. This choice is here termed PG3. An obvious extension is the PG4 formulation for which $\kappa_1 = \kappa_2 = \Delta t/2$.

APPENDIX II: EXPLICIT EXPRESSIONS FOR THE MATRIX SYSTEMS

The Jacobian matrix **J** in the FI algorithm is an $m \times m$ block matrix where the contribution from element number e to block (i, j) can be written

$$\begin{bmatrix} \partial F_i^{(p)}/\partial \pi_j & \partial F_i^{(p)}/\partial \sigma_j \\ \partial F_i^{(s)}/\partial \pi_j & \partial F_i^{(s)}/\partial \sigma_j \end{bmatrix}.$$

The formulae for the matrix entries become

$$\frac{\partial F_i^{(p)}}{\partial \pi_j} = V \int_{\Omega^e} \lambda_t \nabla N_i \cdot \nabla N_j \mathrm{d}\Omega,$$

$$\frac{\partial F_i^{(p)}}{\partial \sigma_j} = -V \int_{\Omega^e} \nabla N_i \cdot \frac{\partial \mathbf{v}}{\partial \sigma_j} \mathrm{d}\Omega,$$

$$\frac{\partial F_i^{(s)}}{\partial \pi_j} = -V \Delta t \int_{\Omega^e} \hat{N}_i f'_w \lambda_t \nabla N_j \cdot \nabla S_w \mathrm{d}\Omega,$$

$$\begin{aligned} \frac{\partial F_i^{(s)}}{\partial \sigma_j} = \int_{\Omega^e} \Bigg\{ & \phi \alpha \hat{N}_i M + \Delta t \Bigg[V \hat{N}_i \left(f''_w N_j \mathbf{v}_t \cdot \nabla S_w + f'_w \frac{\partial \mathbf{v}_t}{\partial \sigma_j} \cdot \nabla S_w + f'_w \mathbf{v}_t \cdot \nabla N_j \right) \\ & + V [p''_c N_j h_w \nabla N_i \cdot \nabla S_w + p'_c (h'_w N_j \nabla N_i \cdot \nabla S_w + h_w \nabla N_i \cdot \nabla N_j)] \\ & + W G'_w N_j \mathbf{g} \cdot \nabla N_i - f'_w Q_t \hat{N}_i M \Bigg] \Bigg\} \mathrm{d}\Omega, \end{aligned}$$

$$\frac{\partial \mathbf{v}_t}{\partial \sigma_j} = -\lambda'_t N_j \nabla P_n + p''_c N_j \lambda_w \nabla S^w + p'_c (\lambda'_w N_j \nabla S_w + \lambda_w \nabla N_j) + \frac{W}{V} N_j (\lambda'_w \rho_w + \lambda'_n \rho_n) \mathbf{g}.$$

M is defined as

$$M = \begin{cases} N_j & \text{(consistent mass matrix)}, \\ \delta_{ij} & \text{(lumped mass matrix)}. \end{cases}$$

In the SI algorithm the pressure is found from equation (19), where we have

$$B_{ij} = V \int_{\Omega} \lambda_t \nabla N_i \cdot \nabla N_j \mathrm{d}\Omega,$$

$$b_i = \int_{\Omega} [V \lambda_w p'_c \nabla N_i \cdot \nabla S_w - W(\lambda_w \rho_w + \lambda_n \rho_n) \mathbf{g} \cdot \nabla N_i + N_i Q_t] \mathrm{d}\Omega - \int_{\partial\Omega} V N_i \mathbf{v}_t \cdot \mathbf{n} \mathrm{d}\Gamma.$$

B_{ij} is easily seen to be symmetric and positive definite since we always have $\lambda_t > 0$.

APPENDIX III: ALGORITHMS FOR CGS, OM AND OMR

The following two procedures for computing the initial residual and the matrix–vector product respectively are used later in the algorithms.

function $\mathscr{R}(\mathbf{x}_0)$;
begin
$\mathscr{R} := \mathbf{b} - \mathbf{A}\mathbf{x}_0$;

```
    if preconditioning then
      ℛ := M^{-1} ℛ;
  end

  function 𝒱(q);
  begin
    𝒱 := Aq;
    if preconditioning then
      𝒱 := M^{-1} 𝒱;
  end
```

Conjugate gradients squared

```
  begin
    Compute M;
    r := ℛ(x);
    q := 0; p := 0;
    r̃ := r;
    ρ_1 := 1;
    s := 0;
    while not converged do
    begin
      s := s + 1;
      ρ_2 := (r, r̃);
      β := ρ_2/ρ_1;
      ρ_1 := ρ_2;
      v := βq;
      u := r + v;
      h := v + β^2 p;
      p := u + h;
      v := 𝒱(p);
      σ := (r̃, v); α := ρ_1/σ;
      q := u − αv;
      h := u + q;
      v := 𝒱(h);
      x := x + αh;
      r := r − αv;
    end
  end
```

Orthomin(k)

```
  begin
    Compute M;
    r := ℛ(x);
    label start:
    q_0 := 0; p_0 := 0;
    ψ_0 := 0;
    s := 0;
```

```
    while not converged do
    begin
      m := min(s, k); η := mod(s, + 1, k + 1); s := s + 1;
      u := 𝒱(r);
      α_i := (p_i, u)ψ_i;   i = 0, . . . , m,   i ≠ η
      q_η := r − Σ_{j=0, j≠η}^{m} α_j q_j;
      p_η := u − Σ_{j=0, j≠η}^{m} α_j p_j;
      ψ_η := 1/(p_η, p_η);
      ω := (r, p_η)ψ_η;
      x := x + ω q_η;
      r := r − ω p_η;
      if restarted version and s = k + 1 then
        go to start;
    end
  end
```

Orthominres(k)

```
  begin
    Compute M;
    r_0 := ℛ(x_0);
    ρ_0 := (r_0, r_0);
    label start:
    s := 0;
    while not converged do
    begin
      m := min(s, k);
      η := mod(s + 1, k + 1);
      j := mod(s, k + 1); s := s + 1;
      p := 𝒱(r_j);
      v := 𝒱(p);
      τ_i := (v, r_i); ξ_i := (p, r_i);   i = 0, . . . , m,   i ≠ η
      s_1 := Σ_{i=0, i≠η}^{m} τ_i/ρ_i; s_2 := Σ_{i=0, i≠η}^{m} ξ_i/ρ_i;
      γ_i := (s_2 τ_i − s_1 ξ_i)/s_2;   i = 0, . . . , m,   i ≠ η
      h_1 := (p, v); h_2 := (p, p); h_3 := (v, v);
      θ_1 := Σ_{i=0, i≠η}^{m} γ_i ξ_i/ρ_i; θ_2 := Σ_{i=0, i≠η}^{m} γ_i^2/ρ_i;
      ω_1 := s_2(−θ_2 − 2h_1 s_1/s_2 + h_3 + (s_1/s_2)^2 h_2);
      ω_1 := (θ_1 − h_1 + h_2 s_1/s_2)/ω_1;
      ω_2 := (1 − ω_1 s_1)/s_2;
      α_i := (ξ_i/s_2 + ω_1 γ_i)/ρ_i;   i = 0, . . . , m,   i ≠ η
      x_η := ω_1 p + ω_2 r_j + r_j + Σ_{i=0, i≠η}^{m} α_i x_i;
      r_η := −ω_1 v − ω_2 p + r_j + Σ_{i=0, i≠η}^{m} α_i r_i;
      ρ_η := (r_η, r_η);
```

```
    if restarted version and s = k + 1 then
      r_0 := r_η;
      x_0 := x_η;
      go to start;
  end
end
```

The solution is contained in $\mathbf{x}_\eta$.

REFERENCES

1. Aziz and Settari, *Petroleum Reservoir Simulation*, Applied Science Publishers, London, 1979.
2. A. Spivak, H. S. Price and A. Settari, 'Solution of the equations for multidimensional, two-phase, immiscible flow by variational methods', *Soc. Petrol. Eng. J.*, **17**, 27–41 (1977).
3. C. L. McMichael and G. W. Thomas, 'Reservoir simulation by Galerkin's method', *Soc. Petrol. Eng. J.*, 125–1138 (1973).
4. R. W. Lewis, I. R. White and W. L. Wood, 'A starting algorithm for the numerical solution of two-phase flow problems', *Int. j. numer. methods eng.*, **12**, 319–329 (1978).
5. K. Morgan, R. W. Lewis and P. M. Roberts, 'Solution of two-phase flow problems in porous media via an alternating-direction finite element method', *Appl. Math. Model.*, **8**, 391–396 (1984).
6. S. Gulbrandsen and S. Ø. Wille, 'A finite element formulation of the two-phase flow equations for oil reservoirs', *Proc. 8th SPE Reservoir Simulation Symp.* Dallas, 1985, Paper SPE 13516.
7. Ø. Langsrud, 'Simulation of two-phase flow by finite element methods', *Proc. 4th SPE Symp. of Numerical Simulation of Reservoir Performance*, Los Angeles, 1976, Paper SPE 5725.
8. A. Settari, H. S. Price and T. Dupont, 'Development and application of variational methods for simulation of miscible displacement in porous media', *Soc. Petrol. Eng. J.*, 228–246 (1977).
9. M. F. Cohen, 'Application of the Petrov–Galerkin method to chemical-flooding reservoir simulation in one dimension', *Comput. Methods Appl. Mech. Eng.*, **41**, 195–218 (1983).
10. M. F. Cohen, 'Finite element methods for enhanced oil recovery simulation', *Proc. 8th SPE Reservoir Simulation Symp.*, Dallas, 1985, Paper SPE 13512.
11. P. S. Huyakorn and G. F. Pinder, 'A new finite element technique for the solution of two-phase flow through porous media', *Adv. Water Resources*, **1**, 285–298 (1978).
12. G. Chavent, C. Cohen and J. Jaffre, 'Discontinuous upwinding and mixed finite elements for two-phase flows in reservoir simulation', *Comput. Methods Appl. Mech. Eng.*, **47**, 93–118 (1984).
13. T. F. Russell and M. F. Wheeler, 'Finite element and finite difference methods for continuous flows in porous media', in R. E. Ewing (ed.), *The Mathematics of Reservoir Simulation*, SIAM, Philadelphia 1983.
14. T. J. R. Hughes, 'Recent progress in the development and understanding of SUPG methods with special reference to the compressible Euler and Navier–Stokes equations', *Int. j. numer. methods fluids*, **7**, 1261–1275 (1987).
15. J. Glimm, B. Lindquist, O, McBryan and L. Padmanabhan, 'A front tracking reservoir simulator, five-spot validation studies and the water coning problem', in R. E.. Ewing (ed.), *The Mathematics of Reservoir Simulation*, SIAM, Philadelphia 1983.
16. A. Brooks and T. J. R. Hughes, 'Streamline upwind/Petrov–Galerkin formulation for convection-dominated flows with particular emphasis on the incompressible Navier–Stokes equations', *Comput. Methods Appl. Mech. Eng.*, **32**, 199–259 (1982).
17. T. J. R. Hughes, M. Mallet and A. Mizukami, 'A new finite element formulation for computational fluid dynamics: II. Beyond SUPG', *Comput. Methods Appl. Mech. Eng.*, **54**, 341–355 (1986).
18. G. A. Behie and P. A. Forsyth, 'Incomplete factorization methods for fully implicit simulation of enhanced oil recovery', *SIAM J. Sci. Stat. Comput.*, **5**, 543–561 (1984).
19. A. Behie, 'Comparison of nested factorization, constrained pressure residual, and incomplete factorization pre-conditionings', *Proc. 8th SPE Reservoir Simulation Symp.*, Dallas, 1985, Paper SPE 13531.
20. D. Peaceman, *Fundamentals of Numerical Reservoir Simulation*. Elsevier, Amsterdam, 1977.
21. C. A. J. Fletcher, *Computational Techniques for Fluid Dynamics, Vols I and II*, Springer, Berlin Heidelberg, 1988.
22. M. B. Allen III, G. A. Behie and J. A. Trangenstein, 'Multiphase flow in porous media', *Lecture Notes in Engineering*, Springer, New York, 1988.
23. Y. Saad and M. H. Schultz, 'Conjugate gradient-like algorithms for solving non-symmetric linear systems', *Math. Comput.*, **44**, 417–424 (1985).
24. S. C. Eisenstat, H. C. Elman and M. C. Schultz, 'Variational iterative methods for non-symmetric systems of linear equations', *SIAM J. Numer. Anal.*, **20**, 345–357 (1983).
25. H. P. Langtangen and A. Tveito, 'A numerical comparison of conjugate gradient-like methods', *Comm. Appl. Numer. Methods*, **4**, 793–798 (1988).

26. H. P. Langtangen, 'Conjugate gradient methods and ILU preconditioning of non-symmetric matrix systems with arbitrary sparsity patterns', *Int. j. numer. methods fluids*, **9**, 213–233 (1989).
27. H. P. Langtangen, T. Rusten, A. Tveito and S. Ø. Wille, 'An element by element preconditioner for iterative equation solvers', in A. Sa de Costa *et al.* (eds), *Proc. 6th Conf. on Finite Elements in Water Resources*, Springer-Verlag, Lisboa, Portugal, 1986.
28. D. M. Young and K. C. Jea, 'Generalized conjugate-gradient acceleration of non-symmetrizable iterative methods', *Lin. Alg. App.*, **34**, 159–194 (1980).
29 P. Sonneveld, 'CGS, a fast Lanzcos-type solver for nonsymmetric linear systems', *Report 84-16*, Department of Mathematics and Information, Delft University of Technology, 1984.
30. R. Fletcher, 'Conjugate gradient methods for indefinite systems', in G. A. Watson (ed.), *Proc. Dundee Biennial Conf. of Numerical Analysis*, Springer, New York, 1975, pp. 73–89.
31. K. C. Jea and D. M. Young, 'On the simplification of generalized conjugate-gradient methods for nonsymmetrizable linear systems', *Lin. Alg. Appl.*, **52/53**, 399–417 (1983).
32. J. R. Wallis, 'Incomplete Gaussian elimination as a preconditioning for generalized conjugate gradient acceleration', *Proc. 7th SPE Reservoir Simulation Symp.*, San Fransisco, 1983, Paper SPE 12265.
33. H. C. Elman, 'Iterative methods for non-self-adjoint elliptic problems', in *Elliptic Problem Solvers II*, Academic Press, Inc. 1984, pp. 271–283.
34. I. Gustafsson, 'A class of first order factorizations', *BIT*, **18**, 142–156 (1978).
35. S. C. Eisenstat, H. C. Elman and M. H. Schultz, 'Block-preconditioned conjugate-gradient-like methods for numerical reservoir simulation', *Proc. 8th SPE Reservoir Simulation Symp.*, Dallas, 1985, Paper SPE 13534.
36. E. J. Northrup and P. T. Woo, 'Application of preconditioned conjugate-gradient-like methods in reservoir simulation', *Proc. 8th SPE Reservoir Simulation Symp.*, Dallas, 1985, Paper SPE 13532.
37. B. F. Towler and J. E. Killough, 'Comparison of preconditioners for the conjugate gradient method in reservoir simulation', *Proc. 6th SPE Reservoir Simulation Symp.*, New Orleans, 1982, Paper SPE 10490.
38. J. C. Diaz, W. R. Jines and T. Steihaug, 'On a convergence criterion for linear (inner) iterative solvers for reservoir simulation', *Proc. 8th SPE Reservoir Simulation Symp.*, Dallas, 1985, Paper SPE 13500.
39. L. C. Young, 'A finite element method for reservoir simulation', *Soc. Petrol. Eng. J.*, 115–128 (1981).
40. W. K. Liu and T. Belytschko, 'Efficient linear and nonlinear heat conduction with a quadrilateral element', *Int. j. numer. methods eng.*, **20**, 931–948 (1984).
41. T. J. R. Hughes and T. E. Tezduyar, 'Finite element methods for first-order hyperbolic systems with particular emphasis on the compressible Euler equations', *Comput. Methods Appl. Mech. Eng.*, **45**, 217–284 (1984).
42. R. Lohner, K. Morgan and O. C. Zienkiewicz, 'The solution of non-linear hyperbolic equation systems by the finite element method', *Int. j. numer. methods fluids*, **4**, 1043–1063 (1984).