Iterative Solvers of the Implicit Parallel Accurate Reservoir Simulator (IPARS), I: Single Processor Case[*]

Sebastien Lacroix[†] Yuri V. Vassilevski[‡] Mary F. Wheeler[§]

[*] The work was supported by DOE grant DE-FG03-99ER25371.

[†] IFP, 1 et 4 av. de Bois-Preau, BP 311, 92852 Rueil-Malmaison Cedex, FRANCE. Current address: TICAM, The University of Texas at Austin, 211 E. 24th Street, Austin, TX 78712, USA.

[‡] INM, 8 ul.Gubkina, Moscow 117333, RUSSIA. Current address: TICAM, The University of Texas at Austin, 211 E. 24th Street, Austin, TX 78712, USA.

[§] TICAM, The University of Texas at Austin, 211 E. 24th Street, Austin, TX 78712, USA.

Contents

1 Introduction

The IPARS software is research software developed mainly for the purposes of examining different physical models and numerical algorithms for modeling multi-phase flows in porous media. The IPARS framework supports three dimensional transient subsurface flow of multiple phases containing multiple components. A fully implicit solution of the coupled multicomponent nonlinear flow equations is used. Fully implicit time-stepping schemes are the most robust but expensive in the subsurface flow simulation. They result in nonlinear systems to be solved at each time step. The fully implicit solution algorithm is based upon a two-level iterative solution of the coupled nonlinear partial differential equations at each time step. The outer iteration is an inexact Newton method for the nonlinear equations with an inner iterative solution method for the Newton method's Jacobian. In the case of multi-phase flow, the Jacobian matrix is sparse, non-symmetric, and ill conditioned. The linear system is solved by a preconditioned generalized minimum residual (GMRES) iterative solution method.

The series of reports addresses computational considerations of both iterative methods applied to two different models, the hydrology model and the black oil model [14], [24]. In this report, we consider serial computations only. Parallelization techniques and parallel performance will be presented in the second report. Iterative solution for the alternative black-oil model as well as multiblock and multimodel iterative solution are to be reported in the following issues of the series.

The contents of the report are based on a set of basic computational concepts of the IPARS software. In Section 2 we consider the general model formulation and its linearization. The implicit approximation of parabolic type model equations is equipped with model constraints, in order to balance the number of unknowns and equations. The model equations comprise accumulation, transport, and well terms. Each of the terms is linearized via the respective Jacobian parts and the resulting linear system is written in the form of increments. The sparse linear system is to be solved by an iterative technique. Therefore, the issues of an adequate tolerance are to be examined. To this end, we take advantage of the forcing term technique for the inexact Newton and investigate its effect both on Newton method's convergence and the total arithmetical work. Since the Newton method is quadratically convergent in a vicinity of the solution, it is very important to choose an appropriate initial guess. We study two possibilities for the initial guess, to take the solution from the previous time step, or to extrapolate linearly the solution from the previous two time steps. In addition to initialization, we examine a completion step of the inexact Newton method. With a relatively low computational cost, we can improve the total mass conservation of the time stepping scheme. In Section 3, an iterative technique for the linear solver is considered. The basic linear solver within IPARS is chosen to be the right preconditioned GMRES method. The GMRES method is known to be the most robust method for solving non-symmetric systems, and it has a modification (flexible GMRES) capable to converge with a nonlinear preconditioner. The essential drawback of the GMRES method is its memory requirements. However, it is compensated readily by a good preconditioner resulting in a fast convergence as well as the restart version of GMRES. In this section, we examine the difference of GMRES and flexible GMRES methods with linear and nonlinear preconditioners applied to a Jacobian system. In the next Sections, we study systematically the preconditioning within IPARS. The preconditioning is two-stage. At the first stage the decoupling preconditioner is introduced. It allows us to precondition the diagonal pressure block of the Jacobian *independently* of the saturation blocks. Also, construction of a global preconditioner implies certain coupling between saturations and pressure which is a complementary issue to decoupling. At the second stage, the pressure block is preconditioned. Different techniques for decoupling and pressure block preconditioning are considered in Sections 4 and 5, respectively. In Section 6, we summarize our computational experiments.

2 General model formulation and its linearization

2.1 General model equations

In this Section, we follow the framework presented in [7]. A multi-phase flow model consists of $n+m$ equations associated to each grid block (grid cell). The first n equations are those for conservation of n species M_i:

$$\Delta_t M_i = Q_i \Delta t, \quad i = 1, \ldots, n. \tag{1}$$

Here, Q_i represents inter-block flow and well terms:

$$Q_i = \sum_{\kappa} T_{i\kappa}(p_\kappa - p) - q_i, \tag{2}$$

$\sum\limits_{\kappa}$ denotes the summation over all neighbor grid blocks κ, p and p_κ stand for a grid block and a neighbor block pressure, q_i denotes the production rate of species i, and $T_{i\kappa}$ is a transmissibility for flow of species i between a grid block and its neighbor κ. Although the capillary pressure and gravity terms are taken into account in IPARS, for the sake of brevity we neglect them in the course of the presentation.

In the case of fully implicit schemes, both $\Delta_t M_i = M_i^{k+1} - M_i^k$ and $Q_i = Q_i^{k+1}$ are unknown. They are computed by the Newton method. Let M_i^{l+1}, Q_i^{l+1} be the new iterates approximating M_i^{k+1}, Q_i^{k+1}, respectively. Then equation (1) may be rewritten as

$$M_i^{l+1} - M_i^l + M_i^l - M_i^k - Q_i^l \Delta t = (Q_i^{l+1} - Q_i^l)\Delta t. \tag{3}$$

Since $M_i^{k+1} - M_i^k = Q_i^{k+1}\Delta t$, the residual of Newton iteration is

$$r_i = M_i^l - M_i^k - Q_i^l \Delta t,$$

and (3) may be written in the form of increments:

$$\delta M_i + r_i = \delta Q_i \Delta t, \quad i = 1, \ldots, n. \tag{4}$$

Given a set of n species, there always exists a set of $n+m$ variables $\{Y_j\}$, $j = 1, \ldots, n+m$, such that each M_i is a unique function of $\{Y_j\}$. The first n variables from $\{Y_j\}$ are called primary, and the remained variables are referred to as secondary. In order to close the system (4), we need additional m constraint equations. They may express phase equilibrium, saturation constraint, and other model constraints. A general form of the additional equations is

$$\delta L_i + r_i = 0, \quad i = n+1, \dots, n+m. \tag{5}$$

2.2 Newton linearization

Linearization of (4) yields linear equations

$$\sum_{j=1}^{n+m} g_{ij}^M \delta Y_j + r_i = \sum_{j=1}^{n+m} g_{ij}^Q \delta Y_j, \quad i = 1, \dots, n, \tag{6}$$

where g_{ij}^M, g_{ij}^Q are the entries of the accumulation and transport-well Jacobian's terms.

Linearization of (5) results in

$$\sum_{j=1}^{n+m} g_{ij}^L \delta Y_j + r_i = 0, \quad i = n+1, \dots, n+m. \tag{7}$$

The system (6),(7) may be presented in an algebraic form

$$\begin{pmatrix} B & C \\ D & E \end{pmatrix} \begin{pmatrix} \delta Y_I \\ \delta Y_{II} \end{pmatrix} = \begin{pmatrix} \delta Z_I \\ \delta Z_{II} \end{pmatrix}. \tag{8}$$

The dependence of the secondary variables is eliminated by the reduction to the Schur complement counterpart of the system (8)

$$A := B - CE^{-1}D, Z := Z_I - CE^{-1}Z_{II}, Y := \delta Y_I,$$

$$AY = Z. \tag{9}$$

The system (9) is obtained by the reduction of linearized equations to the primary variables. These equations are the linearization of the residual formulation for the system of conservation equations. Since Y stands for the vector of primary variables, (9) may not be reduced to a smaller system. It is to be solved by an iterative technique. Although (9) is a Schur complement reduction of the Jacobian system (8), for the sake of brevity we shall refer to it as the Jacobian system.

2.3 Choice of an initial guess

In view of the quadratic convergence of the Newton method the choice of an initial guess is important. Since IPARS is applied to evolution problems, it is natural to take the initial guess to be equal the solution from previous time steps. Basically, there are two popular choices: to take the solution from the last time step (constant extrapolation), or to take a linear combination of the solutions from the previous two time steps (linear extrapolation). In the absence of abrupt changes in the well term (sinks and sources), the latter choice is more preferable, though it may result in worse initial guess at the instants of strong load variation (turn on/off the wells). In Table 1 we present the L_∞-norm of the initial residual for the Newton method at several time steps due to constant and linear extrapolations. We have considered the ninth SPE comparison problem ($15 \times 24 \times 25$) grid blocks, the first time step is chosen to be one day. The linear extrapolation provides much better initial guess even in the case of dynamic problem such as the ninth SPE comparison problem.

Time step	time (days)	Constant extr.	Linear extr.
4	5.4	12254	3403
5	7.4	11864	5917
10	26	16001	4239

Table 1: Constant and linear extrapolation effects on the initial Newtonian residual.

2.4 Inexact Newton method

Since the system (9) is to be solved iteratively, the problem of an adequate tolerance appears. The higher accuracy for the solution of (9) is, the faster Newton method converges, and vice versa. The optimal balance between the linear and nonlinear iterations is provided by the forcing term technique [10]. On each Newton step (k) the Jacobian system is recommended to be solved so that

$$\|A^k Y^k - Z^k\| \le \eta^k \|Z^k\|,$$

$$\eta^k = \min\left\{0.9999, \max\left\{\tilde{\eta}_k, (\eta^{k-1})^2\right\}\right\}, \quad \tilde{\eta}_k = |\|Z^k\| - \|A^{k-1}Y^{k-1} - Z^{k-1}\||/\|Z^{k-1}\|.$$

In addition, the updated Newton iterate should be modified using a line-search backtracking method:

$$Y_I^k = Y_I^{k-1} + \lambda_k Y^k,$$

in order to provide the global convergence of the inexact Newton method. The scaling factor λ_k is computed from a sequence of the nonlinear vector function evaluations. Since such an evaluation is not implemented in IPARS, we take $\lambda_k = 1$ and suggest to oversolve the Jacobian system in case of lack of the convergence: $\eta^k \to f \cdot \eta^k$, $0 < f \le 1$. In Table 2 we present the number of linear and nonlinear iterations for performing the first time step (1 day) in the hydrology model ($20\times 40\times 40$) and in the black oil model ($20\times 40\times 40$). The physical properties of the media are similar. The Newton convergence tolerance is 10^{-4}, $f = 0.1$. Since generation of the Jacobian takes less time compared to the linear solver, the total number of GMRES iteration is a good measure for the computational work. We fix the preconditioner to be the combinative one, with 6 LSOR iterations for pressure equation (for details we refer to Sections 4,5). We see that the smallest number

	Hydrology		Black oil	
η^k	#Newton it.	#GMRES it.	#Newton it.	#GMRES it.
10^{-4}	3	49	5	81
10^{-3}	3	35	5	56
10^{-2}	4	30	6	43
Forcing	3	20	6	34

Table 2: Inexact Newton convergence and estimation of computational work.

of linear iterations is provided by the forcing term technique. The computational price for that may be an increase of the Newton iterations and the complexity of the Jacobian generation.

2.5 Completion of the Newton method

The Newtonian stopping criterion featured by IPARS requires computation of the new residual which is accompanied by computation of the new Jacobian. It implies that the next Newton iteration is virtually prepared. We suggest to replace the most expensive operation of the Newton step, solving a linear system for the Newtonian correction, by its low rank counterpart. It results in a low rank Newtonian correction to the previous guess. The low rank correction may be defined so that certain integral characteristic of the solution be improved. For instance, it may be the total mass correction yielding better total mass balance, as it is implemented in IPARS.

Let the system (9) at the Newton step l is written as

$$A^l \delta Y^l = Z^l, \tag{10}$$

where δY^l is the l-th correction to be added to Y_I^l to obtain

$$Y_I^{l+1} = Y_I^l + \delta Y^l. \tag{11}$$

Let Z^l and A^l be computed, and Z^l be small enough to exit from the Newton loop. (Due to (3),(4), Z^l is the Newtonian residual, and its norm is used in the Newton stopping criterion.) Instead of the full rank correction δY to Y_I^l, we find a low rank correction DY^l to Y_I^l so that

$$\sum_{\kappa} (A_i^l DY^l)_\kappa = \sum_{\kappa} (Z_i^l)_\kappa, \quad i = 1, \ldots, n, \tag{12}$$

where $\sum\limits_{\kappa}$ denotes the summation over all grid blocks and A_i^l is the row block of matrix A^l corresponding to i-th equation (1). In fact, Z_i^l is the mass residual of the Y_I^l iterate. The physical meaning of (12) is to find a correction DY^l such that (10) is satisfied for the total masses. For the sake of consistency of (12), the rank of of the correction DY^l

has to be equal to the number of conservation equation, n. The simplest representation of DY^l is

$$DY^l = \sum_{i=1}^{n} y_i E_i$$

where entries of vector E_i are zeros except those units associated to the i-th primary variable. In other words, on exit from the Newton loop, we perform an additional low rank Newton step and correct each of the components of Y^l *evenly* over the domain so that the total mass residual for each species M_i becomes smaller. In Table 3 we present the component balances (defined as one plus relative change of total mass of the species with respect to initial values) for four time steps (initial time step is 1 day) in the black oil model ($20 \times 40 \times 40$). The Newton convergence tolerance is 10^{-3}, the linear iterations stop when the initial residual is reduced by factor 10^2. The low rank correction produces better mass balance. This allows us to relax the tolerance of the inexact Newton method.

Low rank	Time	Black oil		
correction	step	oil	water	gas
No	1	0.9999995	0.9999992	0.9999983
No	2	0.9999993	0.9999959	0.9999970
No	3	0.9999994	0.9999943	0.9999967
No	4	0.9999994	0.9999940	0.9999980
Yes	1	0.9999995	0.9999992	0.9999983
Yes	2	0.9999997	0.9999959	0.9999986
Yes	3	1.0000000	0.9999964	0.9999995
Yes	4	1.0000000	0.9999972	1.0000010

Table 3: Effect of the low rank correction onto the mass balances.

3 Preconditioned GMRES method

In general, the matrix A of the Jacobian system (9) is sparse, highly non-symmetric and ill conditioned. The order of the matrix may range from hundreds to many millions. Such a wide range implies usage of Krylov subspace iterative methods. The preconditioned generalized minimum residual (GMRES) method [20, 13] is known to be the most robust and efficient method for the general sparse matrices. Besides, its slight modification, the flexible GMRES [21], is capable to converge fast with nonlinear preconditioners. It is very important if the preconditioner comprises Krylov subspace iterations. The only essential drawback of GMRES and FGMRES methods is that it is necessary to store all the Krylov space in a computer memory. However, it may be cured by the restart versions of GMRES and FGMRES. Moreover, in case of a good preconditioner the convergence of (F)GMRES is so good, that there is no need in restarting. In order to compare GMRES(20) and FGMRES(20) methods, we consider the total number of linear iterations for the first time step (1 day) for two models: hydrology ($20 \times 40 \times 40$) and black oil ($20 \times 40 \times 40$).

The structure of the media is similar, and the tolerance for the linear solver is chosen to be 10^{-4} and for the nonlinear solver 10^{-3}. We apply the Householder decoupling combinative preconditioner (Section 4) with the algebraic multigrid preconditioner for the pressure block (linear preconditioner) and the truncated Neumann series pressure preconditioner with very rough[1] approximate GMRES inversion of the Schur complement (nonlinear preconditioner, Section 5). In Table 4 we show the number of iterations needed to solve the linear problem. The number of iterations for the nonlinear preconditioner is larger than that for the linear one, since the linear preconditioner is almost the best (pressure block is preconditioned very well). The GMRES iterations may be sensitive (up to divergence) to a nonlinearity of preconditioner. The FGMRES iterations remain robust in all the cases.

	Hydrology		Black oil	
Preconditioner	#GMRES it.	#FGMRES it.	#GMRES it.	#FGMRES it.
Linear	10	10	59	59
Nonlinear	div.	24	123	89

Table 4: Linear and nonlinear preconditioning for GMRES and FGMRES methods.

4 Decoupling preconditioners

In the case of multi-phase flow, the system matrix is sparse, non-symmetric, ill conditioned, and its blocks have different nature. The sensible approach to the construction of a preconditioner is to precondition different blocks separately, taking the advantage of their nature. Since the blocks are coupled through nontrivial off-diagonal blocks, the issues of decoupling the blocks are to be considered.

4.1 Properties of the Jacobian system

Properties of the Jacobian system depend on the nature of equations (1) and (5). Since we do not specify particular characteristics of (1) and (5), we make *assumptions* on the reduced Jacobian matrix A. An algebraic property has a relation with physics of equations. Therefore, we try to illustrate algebraic assumptions by physical considerations.

Primary variables

Although a wide set of primary variables is available [16], we restrict our attention to a very particular set of primary variables.

Assumption 1. We assume that Y_1 is the grid block pressure and $\{Y_j\}$, $j = 2, \dots, n+m$, are the grid block saturations (or concentrations).

We remark no special phase pressure has been chosen. However, the optimal choice of the component turns out to be very important in computational practice.

[1] Relative tolerance is 0.6

Local interaction

Additional constraint equations (7) used to close (6), may be chosen to possess local properties. Thus, we may assume that the constraint equations (7) state relationships between our variables in each grid block independently of other grid blocks. On the other hand, the equations of conservation (1)-(2) contain three terms: accumulation $\Delta_t M_i$, transport $\sum\limits_{\kappa} T_{i\kappa}(p_\kappa - p)$, and well terms q_i. By definition, the transport term provides interaction between grid blocks through the pressure differences. Accumulation term $\Delta_t M_i$, responsible for a change in amount of a given species, is likely to have a dominant local interaction within a grid block. The well term may yield an inter-block coupling but be dominated mainly by the pressure variable.

Taking into account the above considerations we conclude that *the interaction between variables is chiefly local.* In algebraic terms, it allows us to make

Assumption 2. Consider the block representation of matrix A associated with grid cell blocks. The off-diagonal block entries responsible for interaction between different variables, are small compared to the respective entries of the diagonal block.

Pressure "governs" saturations

According to Assumption 1, our formulation is presented in terms of pressure and saturations. At least for the black oil isothermal models, the studies [2, 15, 5] show that: the pressure equation is essentially parabolic or elliptic and the saturation equations are hyperbolic or transport dominated parabolic. These features are expected to be inherited by compositional models as well [27]. A well known consequence is that the pressure equation must be treated implicitly and the saturation equations *may* be treated explicitly (Impes models).

Applicability of the Impes models is a starting point of our considerations. We note that implicit pressure and explicit saturation advancing in time approximates the original parabolic equations. It implies that the solutions due to Impes and fully implicit time stepping are close to each other. Therefore, the respective time step nonlinear operators are close in a sense, and their linearizations (Jacobian) are expected to possess a similar nature as well. Thus, given a meaningful guess to the pressure variable, an explicit update of the saturations hopefully yields a meaningful guess to the saturation variables. It means that an explicit saturation calculation based on physically reasonable pressure computation, results in a meaningful approximation for the inversion of fully implicit Jacobian.

Assumption 3. Consider a reduced system with the fully implicit Jacobian (9). Let the matrix A and the vectors Y, Z be split into pressure and saturation blocks:

$$A = \begin{pmatrix} A_p & A_{ps} \\ A_{sp} & A_s \end{pmatrix}, \quad Y = \begin{pmatrix} Y_p \\ Y_s \end{pmatrix}, \quad Z = \begin{pmatrix} Z_p \\ Z_s \end{pmatrix},$$

and let a meaningful approximation $\tilde{Y}_p$ to Y_p and an easy-to-invert approximation $\tilde{A}_s$ to A_s be known. Then $\left(\tilde{Y}_p, \tilde{A}_s^{-1}(Z_s - A_{sp}\tilde{Y}_p)\right)^T$ is a meaningful approximation to $(Y_p, Y_s)^T$.

The choice $\tilde{A}_s = A_s$ implies solution of a saturation system. Lesser stiffness of A_s allows us to replace A_s by a simple approximation (ILU(0) or cell block Jacobi). As we shall see, the latter choice results in moderate convergence dependence on the number of grid blocks.

We note, however, that Assumption 3 is not applicable to the solution of (9) directly, since a meaningful guess $\tilde{Y}_p$ is to be found. Computation of such a guess is the main target of decoupling techniques.

4.2 Decoupling techniques

Basic framework

Our goal is an efficient iterative solution of system (9). To this end, we need a physically meaningful preconditioner for the system matrix. In this section, we address those preconditioners which minimize the number of systems to be solved at each precondition step, and do not require high accuracy for such systems. This reduces both computer memory requirements and CPU time for solving a system with the preconditioner. In fact, we shall focus on preconditioners based on the pressure equation solution and block Gauss-Seidel update of saturation. Different types of such an update as well as more advanced preconditioners [11, 28], are considered in Section 4.4. Different types of the pressure equation preconditioner will be addressed in Section 5.

According to Assumption 3, we need a meaningful guess $\tilde{Y}_p$ to Y_p. The pressure equation reads as

$$A_p Y_p + A_{ps} Y_s = Z_p.$$

Here the pressure variable is coupled to the saturation variables by matrix A_{ps}. This coupling is chiefly local (Assumption 2) which means that the entries of matrix A_{ps} not belonging to the diagonal cell blocks $\{A\}_{ii}$ of A may be neglected. Therefore, any transformation of system (9) which makes the diagonal cell blocks $\{A_{ps}\}_{ii}$ of A_{ps} to be zero, essentially decouples pressure from saturation and allows us to find $\tilde{Y}_p$. We consider several such transformations. Hereinafter, we denote by $\{A\}_{ii}$ the diagonal blocks of a matrix A which is decomposed into grid cell blocks. Within these notations we consider transformations of (9) such that $\{A_{ps}\}_{ii} = 0$.

Constrained pressure decoupling

The approach [11, 26] is based on inversion of local matrices $\{A\}_{ii}$. Let $e_1 = (1, 0, \dots, 0)^T \in \mathbf{R}^n$, I be the identity matrix of order n, and

$$G_{ii}^W = I + e_1 e_1^T \left(\{A_p\}_{ii} \{A\}_{ii}^{-1} - I \right). \qquad (13)$$

It is easy to check that

$$G_{ii}^W \{A\}_{ii} = \left\{ \begin{array}{cc} A_p^W & O \\ A_{sp}^W & A_s^W \end{array} \right\}_{ii} \qquad (14)$$

which implies decoupling pressure from saturations within the diagonal cell block $\{A\}_{ii}$. Introducing the block diagonal matrix

$$G^W = blockdiag\{G^W_{ii}\} \tag{15}$$

and multiplying (9) by G^W, we obtain the transformed system

$$A^W = G^W A, \quad A^W Y = G^W Z.$$

Decomposition of A^W into blocks corresponding to primary variables

$$A^W = \begin{pmatrix} A^W_p & A^W_{ps} \\ A^W_{sp} & A^W_s \end{pmatrix},$$

(14)-(15), and Assumption 3 result in the CPR preconditioner

$$\tilde{A}^W = \begin{pmatrix} A^W_p & O \\ A^W_{sp} & \tilde{A}^W_s \end{pmatrix} \tag{16}$$

to matrix A^W. Here, $\tilde{A}^W_s$ denotes a preconditioner to A^W_s (cell block Jacobi). In order to solve a system $\tilde{A}^W x = r$, one has to solve the pressure equation $A^W_p x_p = r_p$, compute the residual $r_s - A^W_{sp} x_p$ and precondition the residual $(\tilde{A}^W_s)^{-1}(r_s - A^W_{sp} x_p)$. We note that inverting $\tilde{A}^W_s$ requires either additional storage for keeping $(\tilde{A}^W_s)^{-1}$ or to invert $\tilde{A}^W_s$ whenever we solve a system with A^W. In the latter case the inversion may be performed cell-by-cell resulting in a sequence of inversions of order $n-1$.

Householder reflection decoupling

An alternative to CPR decoupling is the Householder reflection [9]. Let G^H_{ii} be a product of $n-1$ Householder matrices:

$$G^H_{ii} = P_{1,ii} \cdot P_{2,ii} \ldots P_{n-1,ii}. \tag{17}$$

Multiplication of a matrix by $P_{k,ii}$ zero the k-th row of the upper triangular part of $\{A\}_{ii}$. Hence,

$$G^H_{ii}\{A\}_{ii} = \left\{ \begin{matrix} A^H_p & O \\ A^H_{sp} & A^H_s \end{matrix} \right\}_{ii} \tag{18}$$

where A^H_s is lower triangular matrix. This implies not only decoupling pressure from saturations, but a virtual factorization of the saturation block A^H_s within a grid block. Multiplication of (9) by the block diagonal matrix

$$G^H = blockdiag\{G^H_{ii}\}$$

result in the transformed system

$$A^H = G^H A, \quad A^H Y = G^H Z.$$

Block representation of A^H and its preconditioner $\tilde{A}^H$ related to primary variables are

$$A^H = \begin{pmatrix} A_p^H & A_{ps}^H \\ A_{sp}^H & A_s^H \end{pmatrix}, \quad \tilde{A}^H = \begin{pmatrix} A_p^H & O \\ A_{sp}^H & \tilde{A}_s^H \end{pmatrix}. \tag{19}$$

Here $\tilde{A}_s^H$ denotes the cell block Jacobi approximation to A_s^H. The solution procedure for matrix $\tilde{A}^H$ is similar to that for the matrix $\tilde{A}^W$. The first advantage is that neither additional memory nor additional inversion is needed to evaluate $(\tilde{A}_s^H)^{-1}$, since it is lower triangular. Another profit of Householder reflections is that they preserve the L_2 norm of a vector. The property is important in the case of the inexact Newton method, when the forcing term technique is used to relax the tolerance of the linear iterative solver. The L_2-norm conservation implies direct applicability of advanced modifications of the Newton method.

Quasi-Impes decoupling

Quasi-Impes decoupling uses an Impes reduction approach to zero the block $\{A_{ps}\}_{ii}$. Let $X_i \in \mathbf{R}^n$ satisfy the system

$$\{A\}_{ii}^T X_i = e_1. \tag{20}$$

Due to (20) multiplication of $\{A\}_{ii}$ by X_i^T yields

$$X_i^T \left\{ \begin{array}{cc} A_p & A_{ps} \\ A_{sp} & A_s \end{array} \right\}_{ii} = \left\{ \begin{array}{cc} A_p^X & O \end{array} \right\}_{ii}.$$

Therefore, if we define the cell block diagonal matrix

$$G^X = blockdiag \left\{ \begin{array}{c} X_i^T \\ \overset{\circ}{I}_{n-1} \end{array} \right\}, \quad \overset{\circ}{I}_{n-1} := \begin{pmatrix} O & I_{n-1} \end{pmatrix} \in \mathbf{R}^{(n-1)\times n}$$

and multiply by it both sides of (9), we obtain the transformed system

$$A^X = G^X A, \quad A^X Y = G^X Z.$$

Block representations of A^X and its preconditioner $\tilde{A}^X$ are similar to those of A^W and $\tilde{A}^W$:

$$A^X = \begin{pmatrix} A_p^X & A_{ps}^X \\ A_{sp}^X & A_s^X \end{pmatrix}, \quad \tilde{A}^X = \begin{pmatrix} A_p^X & O \\ A_{sp}^X & \tilde{A}_s^X \end{pmatrix}, \tag{21}$$

where $\tilde{A}_s^X$ is the cell block Jacobi preconditioner to A_s^X. The solution procedure for the matrix $\tilde{A}^X$ is just the same as for $\tilde{A}^W$. The only difference is that $A_{sp}^X = A_{sp}$, $A_s^X = A_s$, that is, large part of system (9) remains unchanged.

True Impes decoupling

The main idea of the above approaches is to extract the pressure equation which is not coupled to saturations locally within grid cells. Then the construction of the preconditioner for the modified system matrix is performed in two steps: neglecting the remained pressure-saturation ties in the pressure equation; replacing the saturation block by an easy-to-invert approximation. All the approaches are similar in a sense that they construct the *slightly* coupled pressure equation algebraically, basing on the system (9). An alternative is to construct a decoupled pressure equation along with the generation of matrix A. Such an equation may be obtained in the framework of the Impes approach [7]. We remind that if only accumulation term is linearized in (6), the reduction procedure (6)-(9) yields a matrix denoted by A^M. Let us find such a linear combination of rows of the cell diagonal blocks $\{A^M\}_{ii}$, that the pressure be decoupled within the cells. Let vector $X_i^M \in \mathbf{R}^n$ satisfy the system

$$\{A^M\}_{ii}^T X_i^M = e_1. \tag{22}$$

Analogously to the quasi-Impes decoupling, multiplication by $(X_i^M)^T$ eliminates dependency of pressure on saturations:

$$(X_i^M)^T \left\{ \begin{array}{cc} A_p^M & A_{ps}^M \\ A_{sp}^M & A_s^M \end{array} \right\}_{ii} = \left\{ \begin{array}{cc} A_p^M & O \end{array} \right\}_{ii}.$$

The modified system is obtained by the multiplication of the system (9) by the cell block diagonal matrix

$$G^M = blockdiag \left\{ \begin{array}{c} (X_i^M)^T \\ \circ \\ I_{n-1} \end{array} \right\}.$$

If we assume that the well terms are implicit in pressure only, the pressure equation of the modified system is the Impes pressure equation [7]. The modified matrix and its preconditioner are

$$A^I = \begin{pmatrix} A_p^I & A_{ps}^I \\ A_{sp}^I & A_s^I \end{pmatrix}, \quad \tilde{A}^I = \begin{pmatrix} A_p^I & O \\ A_{sp}^I & \tilde{A}_s^I \end{pmatrix}. \tag{23}$$

The true Impes reduction is different from the quasi-Impes one in the vectors X_i^M and X_i only. Vector X_i^M is defined on the basis of accumulation term, while X_i depends on all three terms of the Jacobian. Therefore, the quasi-Impes decoupling is more efficient from the algebraic point of view, though the true Impes decoupling is more physically meaningful.

4.3 Numerical comparison for the decoupling techniques

The decoupling preconditioners have been tested for several matrix equations (9). The comparative characteristic is the number of GMRES(20) iterations needed to reduce the residual L_2-norm by a factor of 10^3 (initial guess is supposed to be trivial). We consider the black oil model (water pressure as a primary variable). Case 1 is the first Newton iteration of the first time step of the ninth SPE comparison problem ($15 \times 24 \times 25$ grid blocks), with a one day time step. Case 2 is different from Case 1 only in the time step increased to 10 days. Case 3 is the same as Case 2 but for the second Newton iteration. Cases 4, 5, 6 are similar to Cases 1, 2, 3 but correspond to a finer mesh ($30 \times 48 \times 50$ grid blocks). Table 5 summarizes the performance of the preconditioners $\tilde{A}^W$, $\tilde{A}^H$, $\tilde{A}^X$, $\tilde{A}^I$, with the cell block Jacobi approximations of saturation blocks A_s^W, A_s^H, A_s^X, A_s^I, and almost exact solution of the pressure equation.

Case	$\tilde{A}^W$	$\tilde{A}^H$	$\tilde{A}^X$	$\tilde{A}^I$
1	4	4	4	5
2	4	4	4	5
3	5	5	5	27
4	7	7	7	12
5	7	7	7	12
6	14	14	15	> 100

Table 5: Performance of decoupling preconditioners.

We may conclude that the true Impes results in larger number of iterations compared to other types of decoupling which perform similarly.

In the above experiments we used the cell block Jacobi preconditioner $\tilde{A}_s$ in the block Gauss-Seidel update of saturations. However, it is not clear how accurate should be the saturation preconditioner $\tilde{A}_s$, or, in other words, what is the price for the replacement of the saturation block A_s by a computationally cheap preconditioner. In Table 6 we compare two block Gauss-Seidel preconditioners for the Householder decoupling (19) and the above described data set. The first one takes the cell block Jacobi approximation $\tilde{A}_s^H$ for the saturation block A_s^H, and the second, $\tilde{A}_{exact}^H$, uses $\tilde{A}_s^H = A_s^H$.

Case	$\tilde{A}^H$	$\tilde{A}_{exact}^H$
1	4	4
2	4	4
3	5	5
4	7	6
5	7	6
6	14	14

Table 6: Exact saturation solve versus the block Jacobi approximation.

It is clear that the usage of cell block Jacobi approximation to the saturation block almost does not affect the convergence rate. Hence, it is decoupling preconditioner that makes the convergence sensitive to the mesh size.

4.4 Combinative techniques

The assumption that pressure "governs" saturations but is not "governed" by saturations may be too strong. The preconditioner providing a feedback for the pressure-saturation interaction is likely to converge faster. An example of such a preconditioner is the combinative two-stage preconditioner [11, 26, 4]. Consider, for example, a Jacobian system transformed by the Householder reflection decoupling (19). The action of the two-stage combinative preconditioner $Y = (\tilde{A}_2^H)^{-1} Z$ is

1. Solve the pressure equation $A_p^H Y_p = Z_p$

2. Compute the total residual

$$\begin{pmatrix} R_p \\ R_s \end{pmatrix} = \begin{pmatrix} Z_p \\ Z_s \end{pmatrix} - \begin{pmatrix} A_p^H \\ A_{sp}^H \end{pmatrix} Y_p$$

3. Precondition the total residual and update the pressure

$$\begin{pmatrix} Y_p \\ Y_s \end{pmatrix} := (\hat{A}^H)^{-1} \begin{pmatrix} R_p \\ R_s \end{pmatrix} + \begin{pmatrix} Y_p \\ O \end{pmatrix}$$

Here, $\hat{A}^H$ stands for a preconditioner to A^H providing a pressure dependence of saturations. The difference between the combinative $\tilde{A}_2^H$ and block Gauss-Seidel preconditioner $\tilde{A}^H$ (19) is in computing and preconditioning the residual, as well as the presence of the feedback update of the pressure. The algebraic form of the combinative preconditioner is

$$\left(\tilde{A}_2^H\right)^{-1} = \begin{pmatrix} (A_p^H)^{-1} & 0 \\ 0 & 0 \end{pmatrix} + \left(\hat{A}^H\right)^{-1} \left(I - \begin{pmatrix} A_p^H \\ A_{sp}^H \end{pmatrix} (A_p^H)^{-1} \right). \tag{24}$$

Two important remarks are pertinent here. First, the block $(A_p^H)^{-1}$ may be replaced by any pressure preconditioner. Second, according to numerical evidence, the preconditioner $\hat{A}^H$ to the whole matrix may be chosen to be rather weak, since its goal is to provide a pressure-saturation feedback. Possible candidates are $ILU(1)$ [26], $DILU$ [8], or one LSOR iteration, or even a couple of Richardson iterations with a block Jacobi preconditioner.

We compare the combinative preconditioner (24) with the block Gauss-Seidel preconditioner (19). The preconditioner $\tilde{A}^H$ uses the cell block Jacobi approximation $\tilde{A}_s^H$ of A_s^H. The global preconditioner $\hat{A}^H$ in the combinative method $\tilde{A}_2^H$ is just two Richardson iterations with matrix A^H and the cell block Jacobi preconditioner and zero initial guess $(\tilde{A}_{2,R}^H)$, or one LSOR iteration with blocks associated to vertical grid lines $(\tilde{A}_{2,L}^H)$. We note that in the case of the black oil (and compositional) model the cost of $\hat{A}^H$ evaluation approaches the cost of multiplication by the Jacobian matrix A^H. Therefore, the cost of one GMRES iteration with the combinative preconditioners $\tilde{A}_{2,R}^H$, $\tilde{A}_{2,L}^H$ exceeds that for $\tilde{A}^H$ by an additional matrix-vector multiplication for A^H. In the case of the hydrology model the relative weight of $\hat{A}^H$ becomes larger in the overall cost of the combinative preconditioner.

In our comparison, we consider four cases related to the hydrology (Cases 1,2) and to the black oil (Cases 3,4) models. The physical properties of the reservoir are similar in all the cases: vertical permeability has a 4-fold jump in a thin horizontal layer (Fig.1), and in two opposite corners there are injection and production wells. The mesh in Cases 1 and 3 has $10 \times 20 \times 20$ cells, while in Cases 2 and 4 the mesh has $20 \times 40 \times 40$ cells. The simulation is done for 18 days within 10 time steps. The relative tolerance for the Newton iterations is 10^{-4} and for the linear solver 10^{-2}. The pressure equation is solved by 6 LSOR iterations. In Table 7 we show the total number of linear iterations accumulated in the whole simulation and the average number of GMRES(20) iterations per Newton step, as well as CPU time of all linear solves measured on a PC-II(400 MH).

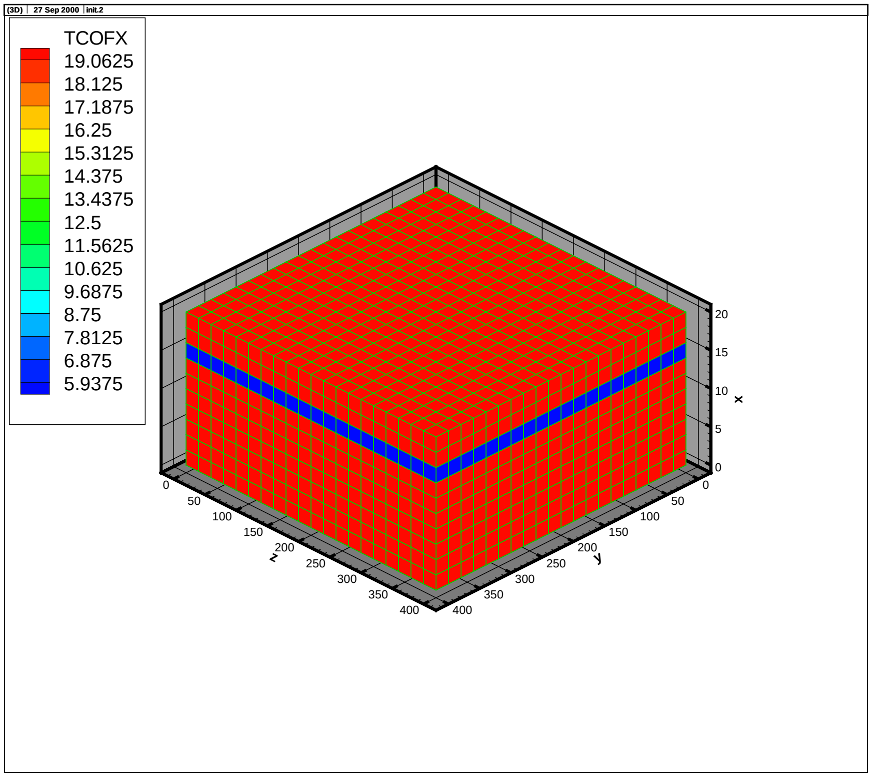

Figure 1: Layered media.

As it stems from the data in Table 7, the combinative preconditioner results in a faster convergence although one GMRES iteration is more costly then that for the block Gauss-Seidel. The advantage of the combinative preconditioner becomes more evident for large number of unknowns. The drawback of the considered two Richardson iterations is that the iterative parameter is not known *a priori*. The value of the parameter affects the convergence. The chosen value (1.0) accelerates the method considerably for the above cases. But in other cases, the convergence may be even worse compared to the block

Case	#GMRES it.			#GMRES per Newton st.			CPU time		
	$\tilde{A}^H$	$\tilde{A}^H_{2,R}$	$\tilde{A}^H_{2,L}$	$\tilde{A}^H$	$\tilde{A}^H_{2,R}$	$\tilde{A}^H_{2,L}$	$\tilde{A}^H$	$\tilde{A}^H_{2,R}$	$\tilde{A}^H_{2,L}$
Case 1	158	130	120	7.9	6.5	6	8.5	9.5	9.6
Case 2	361	257	241	17.2	11.7	10.5	177	166	168
Case 3	141	114	108	5.6	4.6	4.3	11.3	12.9	15.4
Case 4	653	363	345	15.5	9.5	8.6	431	355	397

Table 7: Block Gauss-Seidel and the combinative preconditioners. Pressure block is preconditioned by LSOR(6).

Gauss-Seidel preconditioner. LSOR preconditioning is robust and may be considered to be parameter independent. Our experience shows that the combinative technique is more efficient than the block Gauss-Seidel method, if the pressure block is not preconditioned very well. Table 8 illustrates this by the results of the same experiments with algebraic multigrid preconditioner (coupled with one LSOR iteration) for the pressure block, which is the best preconditioner at hand. Since many preconditioners to be examined are not as good as the latter one, we fix the combinative preconditioner as the basic one.

Case	#GMRES it.			#GMRES per Newton st.			CPU time		
	$\tilde{A}^H$	$\tilde{A}^H_{2,R}$	$\tilde{A}^H_{2,L}$	$\tilde{A}^H$	$\tilde{A}^H_{2,R}$	$\tilde{A}^H_{2,L}$	$\tilde{A}^H$	$\tilde{A}^H_{2,R}$	$\tilde{A}^H_{2,L}$
Case 1	58	49	52	2.9	2.5	2.6	4.8	5.6	6.2
Case 2	115	82	72	5.5	3.7	3.4	72	76	73
Case 3	147	112	108	5.9	4.5	4.3	11.5	13.5	16
Case 4	604	372	343	15.5	9.5	8.3	417	384	423

Table 8: Block Gauss-Seidel and the combinative preconditioners. Pressure block is preconditioned by AMG coupled with LSOR(1).

5 Pressure block preconditioners

The modified pressure block $\bar{A} \equiv A_p^{W,H,X}$ of the Jacobian matrix is sparse and highly non-symmetric. Moreover, it is stiff due to its elliptic nature. In this Section, we consider several preconditioners for the modified pressure block which satisfy the following requirements:

- they are implemented in IPARS,
- they are parallelizable,
- they have low arithmetical cost which is proportional to the order of the pressure block.

Before we proceed to description of preconditioners, we discuss the structure of the pressure block. Since the underlying meshes in IPARS are 3D rectangular, a preconditioner to the pressure block may be constructed for the finite difference 7-point stencil. Therefore, in case of necessity, one could consider the pressure block as a block matrix with 2D (5-point) areal stencil. Each diagonal block is a tridiagonal matrix associated to ties within a vertical line. If we order the vertical grid lines in the red-black areal fashion, we obtain the following block representation:

$$\bar{A} = \begin{pmatrix} \bar{A}_r & \bar{A}_{rb} \\ \bar{A}_{br} & \bar{A}_b \end{pmatrix}.$$

Here, $\bar{A}_r$, $\bar{A}_b$ are the block diagonal matrices. Each block, being a tridiagonal matrix, represents the interaction of a column of grid cells with itself (a one dimensional problem). Therefore, evaluation of $\bar{A}_r^{-1}$, $\bar{A}_b^{-1}$ is computationally cheap.

The majority of the preconditioners to be considered below have two modes: the basic algorithm, and its improved version. By improvement we understand certain modification which reduces the number of iterations considerably though may increase the arithmetical complexity of one iteration. According to this scheme we consider:

- block Gauss-Seidel method,
- truncated Neumann series method,
- separable preconditioning.

Besides, we consider modifications of algebraic multilevel methods which have less expensive iterations and initialization but may result in worse convergence. Hereinafter, we shall use the Householder decoupling in order to obtain the modified pressure block, and the combinative method with one LSOR iteration for preconditioning the modified Jacobian.

In the comparative numerical experiments we shall consider four cases related to the hydrology (Cases 1,2) and to the black oil (Cases 3,4) models, similarly to the cases from Section 4.4. The physical properties of the reservoir are similar in all the cases: vertical permeability has a 4-fold jump in a thin horizontal layer, and in two opposite corners there are injection and production wells. The mesh in Cases 1 and 3 has $10 \times 20 \times 20$ cells, while in Cases 2 and 4 the mesh has $20 \times 40 \times 40$ cells. The simulation is done for 18 days within 10 time steps. The relative tolerance for the Newton iterations is 10^{-4} and for the linear solver 10^{-2}. In Tables 9, 10, 11, 12, 14 we present the total number of linear iterations accumulated in the whole simulation and the average number of GMRES(20) iterations per Newton step, as well as CPU time of all linear iterations measured on a PC-II(400 MH).

5.1 Block Gauss-Seidel preconditioner and LSOR iterative acceleration

The block Gauss-Seidel preconditioner is a block lower triangular part of the pressure block $\bar{A}$:

$$\bar{A}_{GS} = \begin{pmatrix} \bar{A}_r & O \\ \bar{A}_{br} & \bar{A}_b \end{pmatrix}.$$

The solve operation $\bar{A}_{GS}^{-1}$ comprises evaluations of $\bar{A}_r^{-1}$, $\bar{A}_b^{-1}$ and a vector multiplication by $\bar{A}_{br}$. The iterative acceleration of the block Gauss-Seidel is the line SOR[2] (Successive Overrelaxation Method). It is devised by applying extrapolation to the block Gauss-Seidel method. This extrapolation takes the form of a weighted average between the previous iterate x^{k-1} and the computed block Gauss-Seidel iterate $\bar{x}^k$:

$$x^k = \omega \bar{x}^k + (1-\omega)x^{k-1}, \quad k = 1, \ldots, L_{SOR}.$$

The optimal value of ω is computed according to heuristic formulae. The resulting linear operator $\bar{A}_{LSOR}^{-1}$ is a better preconditioner to $\bar{A}$ but more expensive than $\bar{A}_{GS}$. The larger number of LSOR iterations L_{SOR} is, the better $\bar{A}_{LSOR}$ approximates $\bar{A}$. In Table 9 we show the performance data for $\bar{A}_{GS}$ and $\bar{A}_{LSOR}$ preconditioners. For the hydrology model, the iterative acceleration yields the 2-fold speed-up, whereas in the case of the black oil model the speed-up is not considerable.

Case	#GMRES it.		#GMRES per Newton st.		CPU time	
	$\bar{A}_{GS}$	$\bar{A}_{LSOR}$	$\bar{A}_{GS}$	$\bar{A}_{LSOR}$	$\bar{A}_{GS}$	$\bar{A}_{LSOR}$
Case 1	269	120	13.5	6	12.5	9.6
Case 2	963	241	40.1	10.5	419	168
Case 3	162	108	6.5	4.3	16.6	15.4
Case 4	471	345	12.7	8.6	414	397

Table 9: Block Gauss-Seidel and LSOR ($L_{SOR} = 6$) inner iterations as preconditioners for the pressure block.

5.2 Truncated Neumann series

The block factorization of matrix $\bar{A}$ is

$$\begin{pmatrix} \bar{A}_r & \bar{A}_{rb} \\ \bar{A}_{br} & \bar{A}_b \end{pmatrix} = \begin{pmatrix} \bar{A}_r & O \\ \bar{A}_{br} & \bar{A}_b^S \end{pmatrix} \begin{pmatrix} I & \bar{A}_r^{-1}\bar{A}_{rb} \\ O & I \end{pmatrix},$$

where

$$\bar{A}_b^S = \bar{A}_b - \bar{A}_{br}\bar{A}_r^{-1}\bar{A}_{rb}.$$

Due to the red-black reordering, the evaluation of $\bar{A}_r^{-1}$, $\bar{A}_b^{-1}$ may be performed efficiently and parallelized. However, the evaluation of $(\bar{A}_b^S)^{-1}$, the inverse Schur complement, may not be implemented efficiently since $\bar{A}_b^S$ is a dense matrix. A substitution for $(\bar{A}_b^S)^{-1}$ by a reasonable approximation yields a preconditioner to $\bar{A}$. The truncated Neumann series, $(1-x)^{-1} = 1 + x + \cdots + x^n$, gives such an approximation. Keeping the first two terms of a series, we obtain

$$(\bar{A}_b^S)^{-1} \simeq \bar{A}_b^{-1}(I + \bar{A}_{br}\bar{A}_r^{-1}\bar{A}_{rb}\bar{A}_b^{-1}) \equiv (\bar{A}_b^N)^{-1}.$$

[2] Contribution of Philippe Quandalle (IFP, France).

Thus, the truncated Neumann series preconditioner is

$$\bar{A}_{NS} = \begin{pmatrix} \bar{A}_r & O \\ \bar{A}_{br} & \bar{A}_b^N \end{pmatrix} \begin{pmatrix} \bar{I} & \bar{A}_r^{-1}\bar{A}_{rb} \\ O & I \end{pmatrix}.$$

The arithmetical work needed to evaluate $\bar{A}_{NS}^{-1}$ equals two LSOR iterations approximately. Numerical tests show that the condition number of $\bar{A}_{NS}^{-1}\bar{A}$ depend on the minimal mesh step. To enhance the preconditioner, one could use better approximation to the inverse of the Schur complement. We have found that further expansion of the Neumann series is more expensive than inner iterative loop resulting in an operator

$$\widetilde{A_b^N} \equiv A_b^S \left(I - \prod_j (I - \tau_j (\bar{A}_b^N)^{-1} \bar{A}_b^S) \right)^{-1}.$$

Parameters τ_j are not fixed, they are computed by a Krylov subspace method so that the relative accuracy of the iterative inversion be fixed (in our cases, the tolerance for the residual reduction is chosen to be 0.6). The inner iterative process may be either GMRES or BCGStab, or even CG method applied for symmetric modifications of A_b^S. In Table 10 we compare the performance of $\bar{A}_{NS}$ preconditioner and its GMRES acceleration $\bar{A}_{NS,GMRES}$. We note, that since the latter preconditioner is nonlinear, the flexible GMRES method is used in the solution of system (9). For the hydrology model, the speed-up due to iterative acceleration may be essential, and the CPU time is comparable with the block Gauss-Seidel/LSOR preconditioning. The advantage of the truncated Neumann series method is that the number of inner iterations is chosen automatically, basing on the convergence tolerance. In the case of black oil no speed-up is observed.

Case	#GMRES it.		#GMRES per Newton st.		CPU time	
	$\bar{A}_{NS}$	$\bar{A}_{NS,GMRES}$	$\bar{A}_{NS}$	$\bar{A}_{NS,GMRES}$	$\bar{A}_{NS}$	$\bar{A}_{NS,GMRES}$
Case 1	228	103	11.4	5.1	13.8	11.4
Case 2	558	146	23.2	6.3	321	232
Case 3	144	134	5.5	5.1	17.7	17.6
Case 4	436	429	11.2	10.7	444	493

Table 10: Truncated Neumann series without and with GMRES acceleration as preconditioners for the pressure block.

5.3 Separable preconditioner

The separable preconditioner is based upon an inversion of the following discrete operator

$$\bar{A}_{Sp} = A_1 \otimes M_2 \otimes M_3 + M_1 \otimes A_2 \otimes M_3 + M_1 \otimes M_2 \otimes A_3.$$

Here, A_i stands for symmetric tridiagonal matrix and M_i stands for a diagonal matrix. The method was proposed by Yu.Kuznetsov [12] and P.Vassilevski [25] about 15 years ago. Its numerical properties were reported in [1]. The method features insensitivity to anisotropy of the grid, arbitrary dimensions of matrices A_i, M_i, and suboptimal arithmetical complexity ($O(N \log N)$) [12].

For IPARS implementation, we chose the parallel and single processor versions of the separable preconditioner written by Tuomo Rossi and Jari Toivanen from Jyväskylä University, Finland. For details of the method implementation we refer to [17], [18], [19],[23]. Within IPARS, matrices M_i were taken to be identities and matrices A_i were chosen to be submatrices of the pressure block of the transport term of unmodified Jacobian A. The submatrices are associated with given "tracing grid lines" in directions x, y, z. Preconditioning a heterogeneous operator by the tensor product of "one dimensional" matrices may result in loosing high frequencies, and hence we augment the separable preconditioner by a LSOR(1) smoother:

$$\bar{A}_{Sp,2}^{-1} = \bar{A}_{Sp}^{-1} + \bar{A}_{LSOR(1)}^{-1}(I - \bar{A}\bar{A}_{Sp}^{-1}).$$

Implementation of such an enhanced preconditioner resembles the two-level method [29] where $\bar{A}_{Sp}$ plays the role of a coarse mesh solver. The difference is that $\bar{A}_{Sp}$ is defined on the whole grid space. The motivation for the two-stage preconditioner is very simple. If we assume that heterogeneity of porous media affects mainly the high frequencies of the discrete heterogeneous operator preconditioned by a discrete homogeneous operator, then we may damp those frequencies by a cheap smoother (LSOR). Roughly speaking, we may try to correct the homogeneous preconditioner for improperly preconditioned high frequencies. The implementation steps of the two-stage separable preconditioner $y = \bar{A}_{Sp,2}^{-1}x$ are:

1. $y = \bar{A}_{Sp}^{-1}x$,
2. $r = (\bar{A}_{LSOR(1)})^{-1}(x - \bar{A}y)$,
3. $y := y + r$.

In Table 11 we compare the performance of the two preconditioners in the case of homogeneous (layered) media. It is seen that evaluation of the two-stage preconditioner is slightly more expensive but results in better convergence. The advantage of the two-stage modification will be seen in Subsection 5.5, where we consider media with variable and heterogeneous permeability. For the hydrology model with mildly variable or homogeneous permeability, the two-stage separable preconditioner turns out to be very competitive method. Another important observation is that the separable preconditioner works very bad in case of the black oil model. We postpone the discussion of this phenomenon to the end of the paper.

5.4 Algebraic multilevel preconditioners

We consider two versions of an algebraic multilevel method, the algebraic multigrid (AMG) [22] and the multilevel incomplete LU (MLILU) method [3]. They are different in the Setup phase and grid transfer operators only. To define any multigrid method,

Case	#GMRES it.		#GMRES per Newton st.		CPU time	
	$\bar{A}_{Sp}$	$\bar{A}_{Sp,2}$	$\bar{A}_{Sp}$	$\bar{A}_{Sp,2}$	$\bar{A}_{Sp}$	$\bar{A}_{Sp,2}$
Case 1	79	76	3.9	3.8	6.5	7.1
Case 2	102	88	4.9	4.2	70	70
Case 3	260	232	9.3	8.6	32	30.6

Table 11: Separable preconditioner and two-stage separable preconditioner as preconditioners for the pressure block.

several components are required. Using superscripts to indicate level number, where 1 denotes the finest level so that $A^1 = \bar{A}$ and the finest grid (set of degrees of freedom Ω^1) coincides with that of $\bar{A}$ (Ω), the components that AMG needs are as follows:

1. "Grids" $\Omega^M \subset \cdots \subset \Omega^2 \subset \Omega^1$.
2. Grid operators $A^M, \ldots, A^2, A^1$.
3. Grid transfer operators:
   "Interpolation" $I^k_{k+1}, k = 1, 2, \ldots, M-1$,
   "Restriction" $I^{k+1}_k, k = 1, 2, \ldots, M-1$.
4. Relaxation scheme for each level.

The recursively defined $(1,1)V$-cycle uses the above components as follows:

```
Algorithm: $MV^k(u^k, f^k)$.
    If $k = M$, set $u^M = (A^M)^{-1} f^M$.
    Otherwise:
        Relax on $A^k u^k = f^k$.
        Perform coarse grid correction:
            Set $u^{k+1} = 0, f^{k+1} = I^{k+1}_k (f^k - A^k u^k)$.
            "Solve" on level $k+1$ with $MV^{k+1}(u^{k+1}, f^{k+1})$.
            Correct the solution by $u^k := u^k + I^k_{k+1} u^{k+1}$.
        Relax on $A^k u^k = f^k$.
```

Operators on coarse grids are defined via Galerkin projection:

$$A^{k+1} = I^{k+1}_k A^k I^k_{k+1}.$$

We note that in the case of the algebraic multigrid the interpolation I^k_{k+1} and restriction I^{k+1}_k operators are transposed, $I^{k+1}_k = (I^k_{k+1})^T$. For the MLILU method, this is true only for symmetric matrices.

The choice of components in the algebraic multilevel methods is done in a separate preprocessing step:

Setup phase:

1. Set $k = 1$.

2. Partition Ω^k into disjoint sets C^k and F^k.
    (a) Set $\Omega^{k+1} = C^k$.
    (b) Define "interpolation" I^k_{k+1} and "restriction" I^{k+1}_k.
3. Set $A^{k+1} = I^{k+1}_k A^k I^k_{k+1}$.
4. If Ω^{k+1} is small enough, set $M = k+1$ and stop. Otherwise, set $k = k+1$ and go to step 2.

The goal of the setup phase is to choose the set C^k of coarse grid points, for each fine grid point $i \in F^k \equiv \Omega^k \setminus C^k$, a small set $C^k_i \subset C^k$ of interpolating points and for each coarse grid point $j \in C^k \equiv \Omega^{k+1}$ a small set $F^k_j \subset F^k$ of restriction points. Interpolation and restriction are then defined in terms of weights by:

$$(I^k_{k+1}u^{k+1})_i = \left\{ \begin{array}{cl} u^{k+1}_i & \text{if } i \in C^k, \\ \sum\limits_{j \in C^k_i} \omega_{ij} u^{k+1}_j & \text{if } i \in F^k, \end{array} \right.$$

$$(I^{k+1}_k u^k)_j = u^k_j + \sum_{i \in F^k_j} \bar{\omega}_{ji} u^k_i .$$

We recall that in case of AMG $\bar{\omega}_{ji} = \omega_{ij}$.

Thus, the general structure of AMG and MLILU methods is the same. However, the techniques for generation of the inter-grid transfer operators and separation into fine and coarse nodes are totally different. First, it is important to note that AMG is applied to a positive definite operator whereas MLILU is appropriate for a negative definite operator. As a result, the definitions of sets of influencing points differ:

$$S_i = \left\{ j \neq i \,|\, -a_{ij} \geq \sigma \max_{k \neq i}(-a_{ik}) \text{ (AMG)} \right\},$$

$$S_i = \left\{ j \neq i \,|\, a_{ij} \geq \sigma \max_{k \neq i}(a_{ik}) \text{ (MLILU)} \right\}.$$

The use of these sets is important for the computation of the weights ω_{ij}, $\bar{\omega}_{ji}$. The weight evaluation is based on different ideas for AMG and MLILU. The AMG algorithm is focused on errors e satisfying $Ae \approx 0$ [22]. The MLILU method is designed to approximate the Schur complement K associated with a Gauss elimination, by a matrix $\bar{K}$ so that a filtering condition is satisfied $(K - \overline{K})t = 0$, for a given test vector t [3].

The other major difference stays in the separation step. The main tool of the separation is an oriented adjacency matrix graph whose ties are described by the formula:

$$e^{(0)}_{ij} = \left\{ \begin{array}{ll} 1, & \text{if } j \in S_i, \\ 0, & \text{otherwise.} \end{array} \right.$$

The coarse nodes are defined to be the nodes influencing the maximum number of fine nodes. By the influence of a node i we understand the summation of all the ties coming from other nodes, *i.e.*

$$\lambda_i = \sum_{j \in (S_i)^T} e_{ji},$$

where $S_i^T = \{k, i \in S_k\}$. In the MLILU method, the sets C^k and F^k are constructed to reduce a fill-in. Namely, for each node i, the MLILU separation step looks forward to find two weakly connected nodes (parent nodes) which provide a minimum fill-in in the next graph after elimination of the node i. Then, the ties for the parent nodes are reinforced compared to other nodes linked to the node i:

$$e_{ij}^{(1)} = \begin{cases} e_{ij}^{(0)}, & \text{if } j \text{ is not a parent node,} \\ \beta_{ij}(\nu_i, nb_i), & \text{otherwise,} \end{cases}$$

where ν_{ij} is the number of new edges in the next graph and nb_i is the number of neighbors of the node i. The ties are updated in MLILU whereas they stay the same in AMG. In other words, the separation step for AMG exploits all the nodes from the influencing sets while for MLILU selects just subsets of the influencing sets. After this step the separation is as follows:

1. Set $C^k = \{\}$, $F^k = \{\}$, $U^k = \Omega^k$.

   (a) Until U^k is empty select i with maximum λ_i.

      i. If $\lambda_i > 0$ set $C^k = C^k \cup \{i\}$ and $U^k = U^k \setminus \{i\}$

      ii. For all nodes j depending on i, *i.e.*, $j \in S_i^T$
      If (AMG) select all j as fine nodes $F^k = F^k \cup \{j\}$
      If (MLILU) select j as a fine nodes only if two of his parent nodes are coarse nodes or if it is linked to an unique coarse node
      Update ties of the nodes k influencing all the new fine nodes and the coarse node regarding
      if (AMG) initial sets S_k^T in case of AMG
      if (MLILU) updated sets $\overline{S_k^T}$ corresponding to the new graph

      iii. If $\lambda_i = 0$
      If (AMG) always select this node as a coarse node, $C^k = C^k \cup \{i\}$
      If (MLILU) impose weaker F node conditions and check if i could be a fine node otherwise select i as a coarse node

2. If (AMG) check all F nodes to ensure there is no connections between them or dependency towards interpolating nodes
   If (MLILU) improve interpolation and restriction by possible extensions of the sets of parent nodes

The last step of MLILU does not change the separation into subsets F^k and C^k, it modifies slightly interpolation and restriction patterns in terms of F_j^k, C_i^k. In contrast to MLILU, the last step of AMG modifies the separation essentially.

It is important to notice the difference in the node labeling for both methods. The MLILU separation seems to be more advanced compared to the AMG counterpart, since it provides better approximation for a lower complexity. The compensation for better labeling is a higher initialization price (ten to forty MLILU V-cycles versus five to ten AMG V-cycles). For other details we refer to [6] and [3], respectively.

In Table 12 we present the performance data of both preconditioners. The AMG method is implemented using AMG1R5 code (K.Stuben *et al.*), and the MLILU method is implemented by S.Lacroix. As it is seen from the table, the execution time is rather large although the convergence of GMRES is very fast. The detailed analysis shows that the major part of time is spent to initialization of AMG and MLILU. The preconditioner for the pressure block is to be reinitialized every Newton step. Due to the fast convergence of the method, the initialization time is not compensated by the iteration time. In order to balance initialization and iterative solution, we take advantage of the two-stage method discussed above. We note that LSOR iteration is a very good smoother for errors having dominant frequencies in the vertical direction. Therefore, we can replace preconditioning the whole pressure block by preconditioning its Galerkin projection, $\bar{A}_c = I_x^T \bar{A} I_x$, where I_x stands for an interpolation operator in the vertical direction. We define the interpolation operator as a block diagonal matrix where each block is associated to a vertical grid line. In order to define the matrix blocks, we group the unknowns within a block by triples, except those associated to the top and the bottom of the grid line: $\{1\}, \{2,3,4\}, \{5,6,7\}, \dots, \{N_x\}$. The block of I_x contains ones in each column associated with each subset:

Case	#GMRES it.		#GMRES per Newton st.		CPU time	
	$\bar{A}_{AMG}$	$\bar{A}_{MLILU}$	$\bar{A}_{AMG}$	$\bar{A}_{MLILU}$	$\bar{A}_{AMG}$	$\bar{A}_{MLILU}$
Case 1	50	49	2.5	2.4	11.3	15.8
Case 2	75	69	3.6	3.3	169	276
Case 3	103	102	4.1	4.1	23.4	31.1
Case 4	344	343	8.4	8.4	641	911

Table 12: Algebraic multigrid (AMG) and multilevel incomplete LU method as preconditioners for the pressure block.

$$I_x = \begin{pmatrix} 1 & 0 & 0 & \dots \\ 0 & 1 & 0 & \dots \\ 0 & 1 & 0 & \dots \\ 0 & 1 & 0 & \dots \\ 0 & 0 & 1 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

The restriction operator, I_x^T, is nothing else but the averaging operator. The 7-point stencil of the Galerkin projection $\bar{A}_c$ remains the same, an the order of matrix $\bar{A}_c$ is almost as much as 3 times smaller than that of matrix $\bar{A}$. If we denote the multilevel preconditioner (AMG or MLILU) for $\bar{A}_c$ by B_c, the two-stage preconditioners are:

$$\bar{A}^{-1}_{AMG,2} = I_x B^{-1}_{c,AMG} I_x^T + \bar{A}^{-1}_{LSOR(1)}(I - \bar{A} I_x B^{-1}_{c,AMG} I_x^T),$$

$$\bar{A}^{-1}_{MLILU,2} = I_x B^{-1}_{c,MLILU} I_x^T + \bar{A}^{-1}_{LSOR(1)}(I - \bar{A} I_x B^{-1}_{c,MLILU} I_x^T).$$

It is pertinent to note here that the introduction of *geometrical* interpolation operator may result in deterioration of the convergence, if the interpolation operator is not stable in appropriate norm. It may happen in a case of severe heterogeneity of coefficients, as we shall see later. In Table 12 we exhibit the performance data of both two-stage preconditioners. In the case of layered media, the convergence remains the same while the execution time drops considerably.

Case	#GMRES it.		#GMRES per Newton st.		CPU time	
	$\bar{A}_{AMG,2}$	$\bar{A}_{MLILU,2}$	$\bar{A}_{AMG,2}$	$\bar{A}_{MLILU,2}$	$\bar{A}_{AMG,2}$	$\bar{A}_{MLILU,2}$
Case 1	52	53	2.6	2.6	6.2	8.5
Case 2	72	75	3.4	3.6	73	117
Case 3	108	108	4.1	4.1	16.4	20.7
Case 4	343	345	8.4	8.4	422	489

Table 13: Two-stage algebraic multigrid (AMG) and multilevel incomplete LU methods as preconditioners for the pressure block.

5.5 Heterogeneous permeability

We compare the performance of all the above specified preconditioners for the hydrology and black oil models in heterogeneous porous media with permeability variations of order 10 (moderate heterogeneity) and 1000 (severe heterogeneity) (Figs.2,3). We consider two-well simulation of the flow during the first 18 days (10 time steps) on the mesh $(20 \times 40 \times 40)$, with different structure of permeability. The comparative data are shown in Tables 14, 15.

The results shown in the tables prove that the most robust to the degree of heterogeneity are the algebraic multilevel methods. The cost of evaluation and initialization of the AMG preconditioner is surprisingly lower in the case of heterogeneous media. The two-stage algebraic multilevel method is the fastest. Higher cost of initialization of the MLILU method makes it less competitive than the algebraic multigrid. Separable preconditioners, being very efficient for moderate heterogeneity in the hydrology models, are not suitable to the cases of severe heterogeneity and the black oil model. Conventional methods (block Gauss-Seidel, LSOR(6), truncated Neumann series) exhibit approximately the same performance but different convergence.

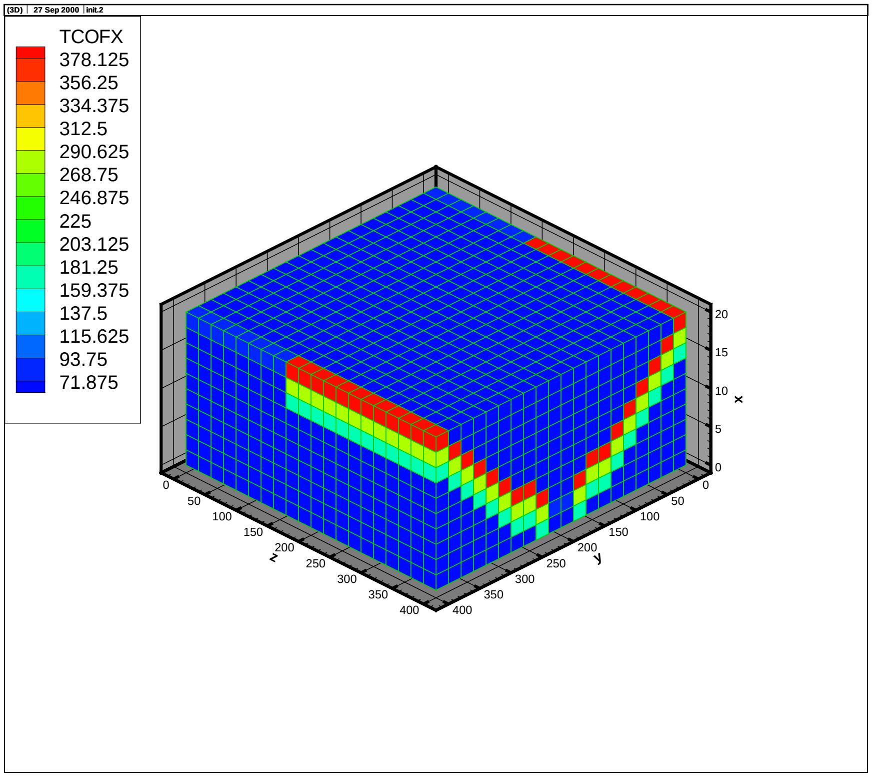

Figure 2: Permeability field with moderate heterogeneity.

Method	#GMRES it.		#GMRES per Newton st.		CPU time	
	moderate	severe	moderate	severe	moderate	severe
$A_{GS}/A_{LSOR(6)}$	541/211	946/290	27/10.5	39.4/12.1	236/145	407/198
$A_{NS}/A_{NS,GMRES}$	414/110	617/141	20.7/5.5	25.7/5.9	239/174	351/229
$A_{Sp}/A_{Sp,2}$	105/98	>1000	5.2/4.9	>100	71/76	>1000
$A_{AMG}/A_{AMG,2}$	72/73	58/179	3.6/3.6	2.8/7.4	78.2/59	98/126
$A_{MLILU}/A_{MLILU,2}$	73/76	71/170	3.6/3.8	3.1/7.4	234/94	333/184

Table 14: Performance of the pressure preconditioners for the hydrology model with heterogeneous media.

6 Conclusions

We considered several issues related to the solution of the systems of nonlinear partial differential equations. The systems appear in the fully implicit simulation of multi-phase flow in porous media. Two models have been examined, the hydrology (oil and water), and the black oil model (oil, water and gas). The comparison study has been done in order to give certain recommendations to IPARS users. The recommendations are based

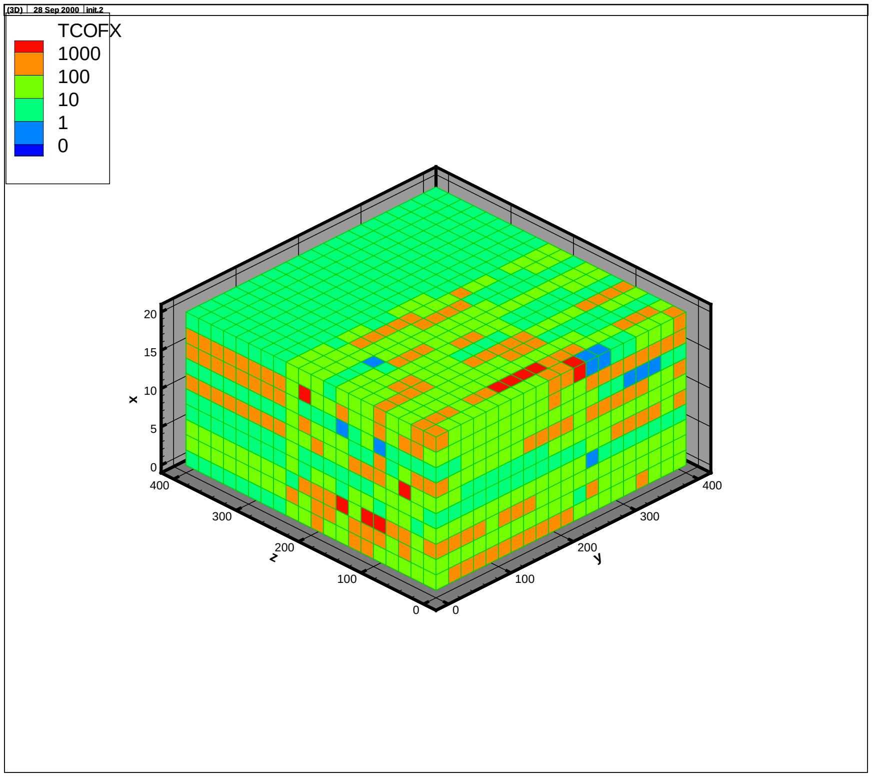

Figure 3: Permeability field with severe heterogeneity.

Method	#GMRES it.		#GMRES per Newton st.		CPU time	
	moderate	severe	moderate	severe	moderate	severe
$A_{GS}/A_{LSOR(6)}$	357/294	619/301	7.1/5.9	17.7/8.6	326/353	539/344
$A_{NS}/A_{NS,GMRES}$	342/340	512/443	6.7/6.5	14.6/12.6	366/393	515/545
$A_{AMG}/A_{AMG,2}$	304/292	251/261	6.3/5.8	7.2/7.5	416/347	379/301
$A_{MLILU}/A_{MLILU,2}$	300/277	261/263	5.9/5.8	7.2/7.5	863/404	728/394

Table 15: Performance of the pressure preconditioners for the black oil model with heterogeneous media.

mainly on the following conclusions derived from the numerical experiments.

Models

The current black oil model (water pressure is chosen to be the primary variable) suffers by the lack of computational efficiency. The main reason is that the pressure diagonal block of the Jacobian is not stiff enough, and the “ellipticity is spread” over the system. Indeed, we have seen that an improvement of the pressure preconditioner have not resulted in significant reduction of GMRES iterations. On the other hand, the feedback in pressure-saturation interaction accelerates the convergence. Moreover, the separable preconditioner for the transport part of the Jacobian pressure block is a poor preconditioner to the Jacobian pressure block, in contrast to the hydrology model. Since the main source of "ellipticity" is in the transport part, we conclude that it is smeared out the pressure diagonal block of the Jacobian.

The hydrology model does not have such bad properties. The alternative black oil model with the oil pressure primary variable is expected to improve the performance of linear solvers.

Nonlinear solver

Inexact Newton method is affected by the initial guess and the accuracy of the linear solver. Linear extrapolation from the previous two time steps yields better initial guess for the Newton method, compared to the constant extrapolation. The low rank correction of the final Newton iterate improves the mass balance for the solution. The forcing term technique combined with some oversolving is very convenient way to reduce the number of GMRES iterations to be performed at each time step.

Linear solver

The preconditioned GMRES method is a robust algorithm for solving sparse linear systems appearing in the porous media flow simulations. In case of nonlinear preconditioning, the flexible GMRES is the relevant substitution for the GMRES method. The set of two-stage preconditioners has been examined. The goal of the first stage is to decouple a pressure equation from saturation ones. This approach seems to be very promising in compositional models. Four approaches to decoupling have been tested for the black oil model. Three of them have exhibited the same convergence properties. Besides, the combinative technique may accelerate the convergence and reduce the overall CPU time. The goal of the second stage is to precondition the pressure equation. Five pairs of pressure preconditioners have been considered. Each pair is based on a particular computational technique and has two versions of a method, conventional and improved. The improved counterpart reduces the number of GMRES iterations but increases the computational cost of one iteration (with exception of algebraic multilevel methods). The numerical experiments have been targeted to the trade off between the conventional and enhanced counterparts as well as to the comparison of computational methods. These methods are: the block Gauss-Seidel method, the truncated Neumann series preconditioner, the separable preconditioner, and the algebraic multilevel preconditioners. For the hydrology model in almost homogeneous media the separable preconditioner is one of the fastest methods. In heterogeneous media, the most robust is the algebraic multigrid method. The two stage preconditioner based on the algebraic multigrid is the most efficient method. For the black oil model, almost all the methods (except the separable preconditioner) are similar in the performance. The reasons were discussed above. The two-stage algebraic multigrid remains one of the most efficient methods.

Acknowledgements

The authors are very grateful to J.Wheeler, R.Dean, M.Peszynska, J.Eaton, Q.Lu, Y.Achdou and R.Masson for fruitful discussions and valuable comments and to K.Stuben, T.Rossi and J.Toivanen for the permission to use their software.

References

[1] Abakumov A., Yeremin A. and Kuznetsov Yu., *Efficient Fast Direct Method of Solving Poisson's Equation on a Parallelepiped and its Implementation in an Array Processor.* Sov.J.Numer.Anal.Math.Modelling, V.3, 1–20, 1988.

[2] Aziz K. and Settari A., *Petroleum Reservoir Simulation.* Applied Science Publishers ltd., London, 1979.

[3] Bank R. and Wagner C., *Multilevel ILU decomposition.* Numer. Math. V.82, No.4, 543–576, 1999.

[4] Behie G. and Vinsome P., *Block iterative methods for fully implicit reservoir simulation.* Soc. of Pet. Eng. J., 658–558, 1982.

[5] Bell J.B., Trangenstein J.A. and Shubin G.R., *Conservation Laws of Mixed Type Describing Three-Phase Flow in Porous Media.* SIAM J.Appl.Math., V.46, 1000–1017, 1986.

[6] Cleary A., Falgout R., Henson V., Jones J., Manteuffel T., McCormick S., Miranda G. and Ruge J., *Robustness and scalability of algebraic multigrid*, SIAM J.Sci.Comput., 1999.

[7] Coats K.H., *A Note on Impes and Some Impes-Based Simulation Models.* SPE 49774, presented at the SPE 1999 Resrv.Simul.Symp., Houston, Texas, February 14-17, 1999.

[8] Dean R., *Private Communication.*

[9] Edwards H.C., *A Parallel Multilevel-Preconditioned GMRES Solver for Multi-phase Flow Models in the Implicit Parallel Accurate Reservoir Simulator (IPARS).* TICAM Report 98-04, The University of Texas at Austin, 1998.

[10] Kelley C., *Iterative Methods for Linear and Nonlinear Equations.* SIAM, Philadelphia, 1995.

[11] Klie H., *Krylov-Secant Methods for Solving Large Scale Systems of Coupled Nonlinear Parabolic Equations.* PhD Thesis, Rice University, Houston, 1996.

[12] Kuznetsov Yu., *Matrix Computational Processes in Subspaces.* Comp.Math. Appl.Sci.Eng., V.6, North-Holland, Amsterdam, 15–31, 1984.

[13] Kuznetsov Yu., *Some problems in the theory and applications of iterative methods*, PhD thesis, Novosibirsk, Russia, 1969.

[14] Lu Q., *A Parallel Multi-block/Multi-physics Approach for Multi-phase Flow in Porous Media.* PhD Thesis, TICAM, The University of Texas at Austin, 2000.

[15] Peaceman D., *Fundamentals of Numerical Reservoir Simulation.* Elsevier Scientific Publishing Company, 1977.

[16] Proceedings of the 15th Symposium on Reservoir Simulation, Society of Petroleum Engineers Inc., Richardson TX, 1999.

[17] Rossi T., *Fictitious Domain Methods with Separable Preconditioners.* PhD Thesis, University of Jyväskylä, Finland, 1995.

[18] Rossi T. and Toivanen J., *A Parallel Fast Direct Solver for the Discrete Solution of Separable Elliptic Equations.* In "Proceedings of the Eighth SIAM Conference on Parallel Processing for Scientific Computing", Philadelphia, SIAM, 1997.

[19] Rossi T. and Toivanen J., *A Parallel Fast Direct Solver for the Discrete Solver for Block Tridiagonal Systems with Separable Matrices of Arbitrary Dimension.* SIAM J.Sci.Comp., 20, 1778–1793, 1999.

[20] Saad Y., *Iterative Methods for Sparse Linear Systems.* PWS Publishing Co., Boston, (1996).

[21] Saad Y., *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J.Sci.Comput., V.14, 461–469, 1993.

[22] Stuben K., *Algebraic multigrid (AMG): experiences and comparisons*, Appl.Math.Comput., V.13, 419-452, 1983.

[23] Toivanen J., *Fictitious Domain Method Applied to Shape Optimization.* PhD Thesis, University of Jyväskylä, Finland, 1997.

[24] Trangenstein J.A. and Bell J.B., *Mathematical Structure of the Black-Oil Model for Petroleum Reservoir Simulation*, SIAM J.Appl.Math., V.49, 749–783, 1989.

[25] Vassilevski P., *Fast Algorithm for Solving a Linear Algebraic Problem with Separable Variables.* C.r.Acad.Bulg.Sci., V.37, 305–308, 1984.

[26] Wallis J.R., Kendall R.P., and Little T.E., *Constrained Residual Acceleration of Conjugate Residual Methods.* SPE 13536, presented at the SPE 1985 Resrv.Simul.Symp., Dallas, Texas, February 10-13, 1985.

[27] Watts J.W., *A compositional formulation of the pressure and saturation equations*, SPE Reservoir Engineering, 1, 243–252, 1986.

[28] Watts J.W., *A Total-Velocity Sequential Preconditioner for Solving Implicit Reservoir Simulation Matrix Equations.* SPE 51909, presented at the SPE 1999 Resrv.Simul.Symp., Houston, Texas, February 14-17, 1999.

[29] Xu J., *A new class of iterative methods for nonselfadjoint or indefenite problems.* SIAM J.Numer.Anal., 29, 303–319, 1992.