INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 2, Supp , Pages 15–27

AN EFFICIENT AND EFFECTIVE NONLINEAR SOLVER IN A PARALLEL SOFTWARE FOR LARGE SCALE PETROLEUM RESERVOIR SIMULATION

JIANWEN CAO AND JIACHANG SUN

Abstract. We study a parallel Newton-Krylov-Schwarz (NKS) based algorithm for solving large sparse systems resulting from a fully implicit discretization of partial differential equations arising from petroleum reservoir simulations. Our NKS algorithm is designed by combining an inexact Newton method with a rank-2 updated quasi-Newton method. In order to improve the computational efficiency, both DDM and SPMD parallelism strategies are adopted. The effectiveness of the overall algorithm depends heavily on the performance of the linear preconditioner, which is made of a combination of several preconditioning components including AMG, relaxed ILU, up scaling, additive Schwarz, CRP-like(constraint residual preconditioning), Watts correction, Shur complement, among others. In the construction of the CRP-like preconditioner, a restarted GMRES is used to solve the projected linear systems. We have tested this algorithm and related parallel software using data from some real applications, and presented numerical results that show that this solver is robust and scalable for large scale calculations in petroleum reservoir simulations.

Key Words. Petroleum reservoir simulation, Nonlinear solver, Preconditioning, Inexact Newton, BFGS, Krylov subspace, Parallel performance.

1. Introduction

Petroleum reservoir simulation solves the multidimensional and multiphase equations of conservation of mass in porous media, subject to appropriate initial and boundary conditions. The processes occurring in petroleum reservoirs are basically fluid flow and mass transfer. Black Oil Model [1, 2] is regarded as the fundament of reservoir simulation work, where fluids of different phases are usually considered to be at constant temperature and in thermodynamic equilibrium throughout the reservoir. There are three distinct phases, namely oil, water and gas, in this model. Flow in a porous media is governed by three kinds of equations: PDEs describing material flow between blocks which are governed by Darcy's law, a phase-constraint equation describing a saturation relationship of three different phases, capillary pressure equations describing surface tension and the curvature of the interface between the two fluids within the small pores.

In last few years, the performance of parallel petroleum reservoir simulation has been significantly improved ([3]-[10]). However, only a few papers offer their results and effects of practical reservoir problems on MPP computers with more than 32 CPUs. We have developed a parallel black-oil simulator based on a sequential code ([11]), it works well on distributed-memory machines. This simulator uses a

Received by the editors October 1, 2004 and, in revised form, October 30, 2004.
2000 *Mathematics Subject Classification.* 65N22, 65H10, 65Y05, 65Y20, 76S05, 76T30.

fully implicit scheme to discretize the coupled PDEs. The resulting set of nonlinear equations is solved by using inexact Newton method with special choice the initial guess([12]). Efficiency, flexibility and portability are emphasized throughout processes of design and implementation. The solver package is designed and coded so that it is adapted to solving a variety of multi phase flow problems, not being limited to black-oil problems.

Newton method has attractive theoretical and practical properties. If the initial guess is close enough to the exact solution, then quadratic convergence can be obtained. In the nonlinear solver, choosing a good initial guess is one of our emphases. We use BFGS method to provide a good initial guess.

In Newton iteration the most expensive part is solving large sparse linear systems. Usually, each Newton step uses Krylov subspace method with a proper preconditioner. Numerical tests show that, comparing different Krylov subspace algorithms with their "proper" chosen preconditioners, no one algorithm is obviously better than the other ([15]). So the most important part is the choice of preconditioning strategy. Our parallel simulator uses a FGMRES method ([16])with an iterative preconditioning as a typical linear solver. The used preconditioner adopts a so-called multipurpose oblique projection correction strategy ([12]), which involves several preconditioning components such as AMG, relaxed ILU, up scaling, DDM, CRP ([17]) etc.

2. The Black Oil Model and Discretization

The three-phase flow conservation equations can be expressed as [18]

(1)
$$\begin{aligned} \nabla[T_w \nabla(P_w - \rho_w g D)] \ + \ q_w \ &= \ \frac{\partial(\phi b_w S_w)}{\partial t} \\ \nabla[T_o \nabla(P_o - \rho_o g D)] \ + \ q_o \ &= \ \frac{\partial(\phi b_o S_o)}{\partial t} \\ \nabla[T_g \nabla(P_g - \rho_g g D)] + \nabla[T_o R_s \nabla(P_o - \rho_o g D)] + q_g + R_s q_o &= \frac{\partial(\phi b_g S_g + \phi b_o S_o R_s)}{\partial t}, \end{aligned}$$

where $T_l := M_l b_l$ is the transmissibility of phase-l ($l = w, o, g$), $b_l := f_1(P_o)$ is the reciprocal of formation volume factor, D is the vertical depth, $R_s := f_2(P_o)$ is the gas-oil ratio, and $\phi := f_3(P_o)$ is the rock porosity. As a factor of T_l, the mobility $M_l := \dfrac{K f_4(S_w, S_g)}{\mu_l}$ gives a relationship between the flow rate $\vec{v}_l$ and the pressure gradient ∇P_l in each phase through Darcy's Law

$$\vec{v}_l = -M_l \nabla(P_l - \rho_l g D) \ .$$

As an empirical fact, the capillary pressure is a unique function of saturation which provides a relationship between different phase pressures

$$P_w = P_o - P_{cow}(S_w) \ , \ P_g = P_o + P_{cgo}(S_g) \ .$$

As a result, the three unknowns of the above PDEs are oil-phase pressure (P_o), water-phase and gas-phase saturation (S_w, S_g). More details of the variables and their physical properties can be found in many literatures, e.g. ([2]). This model is being used in the commercial reservoir simulation software packages such as VIP ([7]), ECLIPSE ([8]), IPARS([9]) and Simbest-II ([11]). The model represents mathematically a class of important industrial problems rather than simply being an idealized model for benchmark tests and uses realistic saturation coefficients, permeability, and transmissibility which are in-situ field data collected over a long period of time.

By means of considering special cases, we may know about their obscure characteristics. First, the PDEs behave mainly parabolic characteristics. A single-phase PDE has the same form of a heat conduction equation and maybe nonlinear. Two-phase PDEs superficially resemble heat conduction equations also. Second, the PDEs have some characters of elliptic equations. The effects of compressibility c_t usually don't dominate, especially for incompressible flow or slight compressible flow. Thus, as a practical matter, the pressure equation must also be treated as being elliptic or nearly elliptic

$$\nabla(M_o + M_w)\nabla(P_o + P_w) + 2 \times (\frac{q_o}{\rho_o} + \frac{q_w}{\rho_w}) \;\simeq\; \phi c_t \frac{\partial(P_o + P_w)}{\partial t} \;.$$

Third, the saturation equation can be regarded as a nonlinear variation of the diffusion-convection equation

$$\nabla(f_5(S_w)\nabla S_w) - f_6(S_w)\vec{v}_t \nabla S_w + \frac{q_w}{\rho_w} \;\simeq\; \phi\frac{\partial S_w}{\partial t} + \nabla(\frac{M_o M_w(\rho_w - \rho_o)g}{M_o + M_w}\nabla D) \;.$$

If the diffusion term dominates which means that the capillary pressure P_{cow} effect dominates ($f_5(S_w) := -\frac{M_o M_w}{M_o + M_w}\frac{dP_{cow}}{dS_w}$), this PDE behaves like a parabolic equation. However, if the capillary effects are small, when velocities $\vec{v}_t$ are large, the convection term dominates ($f_6(S_w) := \frac{d[M_w/(M_o + M_w)]}{dS_w}$), and this PDE approaches a first-order nonlinear hyperbolic equation. These characteristics require appropriate difference formulations and suitable preconditioned linear solvers in order to solve various applications efficiently. According to above analysis, we can draw the following conclusion: the pressure PDE is parabolic in nature, in many cases, it is nearly elliptic; the gas saturation PDE is a nonlinear diffusion-convection equation, whereas capillary pressure effects dominate; the oil saturation PDE behaves nearly hyperbolic, especially when capillary pressure effects dominate, and more important sometimes, when velocities are large.

Finite difference formulation of the component conservation equation adopts block-centered grid system in our simulator. Considering convection-dominated PDEs, the choice of a first-order difference scheme is crucial. Both up streaming and centered scheme in spatial direction can satisfy the requirement of unconditionally stable. Large time step requirement discards the choice of explicit scheme. Thus, there are four combinations of first-order schemes available, up streaming-in-distance with implicit-in-time, up streaming-in-distance with centered-in-time, centered-in-distance with implicit-in-time, and centered-in-distance with centered-in-time. All the four combinations may lead to numerical dispersion or oscillation (overshoot) phenomenon. Numerical results and theoretical analysis assure that we can't avoid the two phenomena at the same time ([1],[2]). By choosing different combinations, trade off between one and the other is available. In order to keep the scheme to be unconditional stable and avoid numerical oscillation, the choice strategy of first-order difference scheme in our simulator is: fully implicit scheme in the time direction and up steaming scheme in the distance direction. The simulator also adopts the so-called upstream weighting for the relative permeability in the discretization of the second-order diffusion term.

3. Nonlinear Solver and Linear Solver

Fully implicit formulation leads to nonlinear difference equations, thus Newtonian iteration method is required. Newton method has been the most popular choice to solve the nonlinear systems resulting from the fully implicit discretization of the fluid-flow PDE at each time step. It is noted that nonlinearity of the model equation leads to time step restriction also, though it is much less stringent than that for less-implicit difference scheme such as IMPES ([2]), etc. When a fully implicit scheme converts the coupled partial differential equations of black oil reservoir simulation to algebraic equations, usually a set of nonlinear algebraic equations of the form $F(u) = 0$, have to be solved at every time step. The following provides a general description of the nonlinear inexact Newton method.

```
**Algorithm IN** (Inexact Newton Method)
  Define $\delta u := u^{(n+1)} - u^{(n)}$
  (a) Give initial guess $u^{(0)}$
  (b) For $n = 0, 1, 2, \ldots$ until convergence, do
      *Using Taylor's formula, to discretize the nonlinear equation*
          $F(u^{(n+1)}) = F(u^{(n)} + \delta u) \approx F(u^{(n)}) + J(u^{(n)})\delta u = 0$
      *Get the following linear system*
(2)       $\|J(u^{(n)})\delta u + F(u^{(n)})\|_2 \leq \eta_n \|F(u^{(n)})\|_2$
      *Choose a proper forcing term $\eta_n$, which is a function of $n$ and $F(u^{(n)})$*
      *Solve the linear system and obtain its solution $\delta u^{(n)}$*
      *Choose a proper backtracking step length $\alpha_n$, which is a function of $n$, $\delta u^{(n)}$ and $\delta u^{[\text{max tolerance}]}$*
      *Compute the new approximate solution*
          $u^{(n+1)} = u^{(n)} + \alpha_n \delta u^{(n)}$
      *Check if $u^{(n+1)}$ satisfies the convergence tolerance of $F(u) = 0$*
```

In most cases, the initial guess of Newton method is close enough to the exact solution. However, on few cases (usually less than 5%), the initial guess is far enough that Newton method is difficult to converge or even diverge. There are two approaches to overcome these non convergence phenomena. The first approach is to cut the length of current time step, another way is to try to find a better initial guess. Obviously, considering the solution efficiency, the latter is better. In our simulator, we use BFGS to find a proper initial guess, and obtain the following algorithm:

```
**Algorithm INNS**(Inexact Newton Nonlinear Solver)
  (a) Choose the initial vector $u^{(0)}$
  (b) Use Algorithm IN to get the approximation solution vector $u^{(k)}$
  (c) Do the convergence history evaluation. Determine that the nonlinear iteration process is satisfied or not.
      Case 1: If this process is satisfied, we continue to use Algorithm IN till convergence.
      Case 2: If this process isn't satisfied, which means that it is difficult to observe the IN's convergence behavior, or even the IN diverges. In this case, we use BFGS to obtain a better approximation $u^{(k*)}$ than that of $u^{(k)}$, then let $u^{(0)} := u^{(k*)}$, and go back to (b)in order to construct a new approximation vector. Here, we need to choose an initial approximation of $J(u^{(*)}) \equiv F'(u^{(*)})$ as $B_0$.
```

```
**Algorithm BFGS**
  Get the approximation vector $u^{(k-1)}$ and mark it with $v^{(0)}$
```

Get an initial approximation of $B_0 := J(v^{(0)}) \equiv F'(v^{(0)})$

Choose ILU(1) as a preconditioner of B_0

For $j = 0, 1, \ldots, m$ until the convergence criteria is satisfied, and mark the solution $v^{(m)}$ with $u^{(k*)}$.

(a) Let $g_j := F(v^{(j)})$, solve $B_0 z = -g_j$ using GMRES-ILU preconditioned iterative method

(b) Solve the matrix system : $B_j d_j = -F(v^{(j)})$

(c) Compute α_j so that $v^{(j+1)} = v^{(j)} + \alpha_j d_j$ can decrease the merit function $f(j) := 1/2\|F(v^{(j)})\|_2^2$ along the direction d_j

(d) Check if $v^{(j+1)}$ satisfies the tolerance $\|F(v^{(j+1)})\|_2 \le 0.1 \times \|F(v^{(0)})\|_2$

(e) Compute and get the following vector

$$\begin{aligned} g_{j+1} &:= F(v^{(j+1)}) \\ y_j &:= g_{j+1} - g_j \equiv F(v^{(j+1)}) - F(v^{(j)}) \\ s^{(j)} &:= \alpha_j d_j \equiv v^{(j+1)} - v^{(j)} \end{aligned}$$

(f) B_{j+1} is obtained from B_j by means of a rank-2 updates,

$$B_{j+1} = B_j + \frac{g_j\, g_j^T}{g_j^T\, d_j} + \frac{y_j\, y_j^T}{y_j^T\, s^{(j)}}.$$

One of the important parts of our nonlinear solver is choosing a good initial guess. The reason of adopting BFGS is that there is a fast implementation of BFGS algorithm. The j-th BFGS iteration needs to solve a linear system with a fixed matrix B_0. We may get an ILU decomposition of B_0 and repeatedly use it as a preconditioner during iteration process *(a)*. Another computation-sensitive process of BFGS is *(b)*, it only needs to solve a small dense matrix linear system of order $(j+1) \times (j+1)$ (usually less than 10×10) which can be solved easily and efficiently by calling BLAS3 mathematic library. In our nonlinear solver, considering the role of BFGS, its maximum iteration number m is set to be 9, and its stopping tolerance is set to be $\|F(v^{(j+1)})\|_2 / \|F(v^{(0)})\|_2 \le 0.1$. The computation process *(b)* is depicted as follows([12]):

(b1) Compute and store the following arrays

$$\begin{aligned} \mathtt{GD}(j-1) &:= g_{j-1}^T d_{j-1} \\ \mathtt{YS}(j-1) &:= (g_j - g_{j-1})^T (v^{(j)} - v^{(j-1)}) \\ \mathtt{BG}(j) &:= B_0^{-1} g_j \\ \mathtt{GBG}(i,k) &:= g_i^T B_0^{-1} g_k \ (i = 0, 1, \ldots, j;\ k = 0, 1, \ldots, j) \end{aligned}$$

(b2) Form the following $(j+1) \times (j+1)$ dense matrix linear system

$$\mathtt{C}(k) + \sum_{i=0}^{j} \left[\begin{array}{r} \Big(\frac{1}{\mathtt{GD}(i)} + \frac{1}{\mathtt{YS}(i)} + \frac{1}{\mathtt{YS}(i-1)}\Big)\mathtt{GBG}(k,i) \\ - \frac{1}{\mathtt{YS}(i)}\mathtt{GBG}(k,i+1) \\ - \frac{1}{\mathtt{YS}(i-1)}\mathtt{GBG}(k,i-1) \end{array} \right] \mathtt{C}(i) = -\mathtt{GBG}(k,j)$$

$k = 0, 1, 2, 3, \ldots, j,$ *where* $\mathtt{GD}(j) := \infty,\ \mathtt{YS}(-1) := \infty,\ \mathtt{YS}(j) := \infty$

(b3) Call BLAS3 to solve the above small system, and obtain the solution array $\mathtt{C}(0),\ldots,\mathtt{C}(j)$*, then the desired solution vector of BFGS method can be obtained as*

$$\begin{aligned} d_j \ = \ & -\textstyle\sum_{i=0}^{j-1}\left[\dfrac{\mathtt{C}(i)}{\mathtt{GD}(i)}-\dfrac{\mathtt{C}(i+1)-\mathtt{C}(i)}{\mathtt{YS}(i)}+\dfrac{\mathtt{C}(i)-\mathtt{C}(i-1)}{\mathtt{YS}(i-1)}\right]\ \mathtt{BG}(i) \\ & -\left[1+\dfrac{\mathtt{C}(j)-\mathtt{C}(j-1)}{\mathtt{YS}(j-1)}\right]\ \mathtt{BG}(j) \end{aligned}$$

Quasi-Newton matrix B_0 comes from a Jacobian approximation of the nonlinear equation. BFGS is used until $u^{(k)}$ is much closer to solution $u^{(*)}$, so that Newton method may show its quadratic convergence rate. During the computation process along the temporal axis, on most cases, only an inexact Newton method is used for solving the nonlinear equation, a merit function of evaluating the iteration history needs to be provided priory. Once the iteration history isn't satisfied, which means that the inexact Newton algorithm may not converge successfully ([19, 20]), maybe the initial approximation is pretty bad, at that time BFGS is used to find a better initial guess. If INNS doesn't work, we have to cut in half the length of this time step (In our solver, the maximum limitation of cut number is set to be 3). If INNS doesn't work also, we need set the length of time step to be minimum, which is provided from the written data file and usually equals to be 0.01 days. In the nonlinear solver, we use merit function $f(j)$ to do the evaluation of convergence history. If $f(j-3) < f(j-2) < f(j-1) < f(j)$ comes into existence, we consider that a divergence process may occur, and so the INNS's nonlinear iteration process isn't satisfied. The default maximum nonlinear iteration number is set to be 15, the backtracking step length satisfies $\alpha_j = \min\{1.0, \max\{\exp^{-0.5\times10^{-2j}}, \dfrac{\delta u^{[\text{max tolerance}]}}{\|\delta u^{(j)}\|_\infty}\}\}$, where $\delta u^{[\text{max tolerance}]}$ is an experience value which means the maximum tolerance of the variation of $u^{(j)} - u^{(j-1)}$, it is given from the written data file.

The most computationally expensive part is the solution of the sparse linear equations (2), which can be expressed algebraically as

$$\begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix} \Leftrightarrow A\,x \ = \ f \tag{3}$$

where $x_i = (x_{i,1}, x_{i,2}, \cdots, x_{i,N})^T$, $(i = 1, 2, 3)$, $x_{1,j} \doteq P_{o,j}$, $x_{2,j} \doteq S_{w,j}$, $x_{3,j} \doteq S_{g,j}$, $(j = 1, 2, \cdots, N)$, and $N = N_x \times N_y \times N_z$ is the total number of grid nodes. Two ways of nature ordering are used in the linear solver of PRIS in fact. The above ordering is suitable for analysis. Another natural ordering is based $N \times N$ blocks and each block has a 3×3 sub-block. For example, matrix-vector multiplication, CRP preconditioning and decoupling operator adopt the nature ordering as alike as formulation (2), ILU decomposition, DDM and AMG adopt the second pattern of nature ordering for their specific aims. The Jacobian matrix A is sparse, each entry A_{ij}, $i, j = 1, 2, 3$, is a heptadiagonal matrix, significantly nonsymmetric and highly indefinite. Furthermore, the coefficient blocks associated with a particular type of unknown have different natures (the pressure diagonal block is of elliptic type, the saturation diagonal blocks are of hyperbolic type). In this instance, the single incomplete LU factorization, which is an algebraic preconditioner and doesn't consider the PDE characteristics, doesn't work efficiently. The natural approach to precondition this coupled system is to precondition different blocks separately, taking full advantage of their different natures. Since the blocks of (3) are coupled through non-diagonal blocks, ways to decouple the whole system are to be found.

A so-called decoupled preconditioning process is adopted before we solve the whole linear system ([12, 21, 22]).

Newton-Krylov-Schwarz method is used in our solver, where Schwarz method is used to get the parallel solver. Usually, each Newton step uses a so-called Krylov subspace method with a proper preconditioner. Both GMRES ([24]) and BICGSTAB ([25]) are considered as one of the best choices to solve the linear systems. For GMRES and BICGSTAB, [15] gives out the pattern of its proper preconditioner which is named as PRE-ITER and PRE-ILU respectively. Numerical tests show that, different Krylov subspace methods with an appropriate preconditioner are able to achieve similar performance, in other words, the choice of iterative algorithms isn't the most important part of solving the linear systems efficiently. Instead, the more important part is the choice of the preconditioning strategy.

The default linear solver used in our simulator is preconditioned FGMRES, according to the conclusion of [15], it is typical. For this solver, the default number of orthogonal vectors is 10, and the maximum number of iterations is set to 88. GMRES-ILU preconditioned iterative method is used to solve the small system of PRE-ITER, considering its role of preconditioning, the l_2 norm of the relative residual as the stopping condition is less than 0.1, the maximum restart number of GMRES is limited to 3. The forcing term η_n of Formulation (2) in IN algorithm can be depicted as follows:

$$\eta_n := \max\{10^{-5}, \min\{10^{-6} \times \sqrt{3N}, \epsilon_0 \times \frac{f(n)}{f(n-1)}\}\}$$

At the same time, the stopping condition of linear system (3) also has to satisfy : $\|x_1^{(j)} - x_1^{(j-1)}\|_\infty \le \epsilon_1$, $\|x_2^{(j)} - x_2^{(j-1)}\|_\infty \le \epsilon_2$ and $\|x_3^{(j)} - x_3^{(j-1)}\|_\infty \le \epsilon_2$. The default values of ϵ_0,ϵ_1 and ϵ_2 are 0.01, 0.3 and 0.001 respectively. They can be given from the written data file.

4. Preconditioning of the Linear Solver

A proper preconditioner should be computed easily and be chosen in a way to suit parallel computation . ILU preconditioning is sequential in nature and leads to poor efficiency of the implementation on distributed memory computer platforms. DDM-based preconditioning and combined preconditioning for Krylov subspace methods have been developed for solving an important class of linear systems in large-scale simulation applications. As preconditioning components, they are coupled together by using a so-called multi step method

Algorithm MSM(Multi Step Method)

Assume three types of preconditioning are available and denoted as T_0, T_1 and T_2, A is the matrix of the linear system, and r is the residual vector, then the Multi Step Method gives the following method of constructing a preconditioner

```
a): $z = T_0 r$
b): $r^* = r - A z$
c): $z = z + T_1 r^*$
d): $r^* = r - A z$
e): $z = z + T_2 r^*$
```

This preconditioner ([22]) involves several preconditioning components such as AMG, relaxed ILU, up scaling, DDM, CRP-like (constraint residual preconditioning, [17]) etc.. CRP involves solving a small linear system $(P^TAP)z = r$ by using

GMRES(m) iterative method. There are three oblique projection correction operators. The first is an oblique projection correction process from the whole matrix system A to sub-matrix P^TAP. As a special case, $P^TAP \equiv A_{11}$ is used for Black-oil model. The reason is that A_{11} shows an elliptic feature, and many algebraic algorithms (e.g. ILU, AMG etc.) can be used to solve this sub matrix system.

The second operator deals with an oblique projection correction from the whole solving region to local solving region which is represented by the so-called additive Schwarz preconditioning. From the view of parallelism, computational locality is important and be used to minimize communication frequency among processors. We partition vector x into p sub vectors and each of which is nonempty, possible overlapping, and the union of them is all of the elements of x. Let Boolean rectangular matrix R_i extracts the i_{th} subset of vector x which can be described as $x_i := R_i x$. Let $A_i := R_i A R_i^T$, and $M := \sum_{i=1}^{p} R_i^T A_i^{-1} R_i$. Obviously, M is an approximation of the inverse of Jacobian matrix A and named as additive Schwarz preconditioner.

Theoretical and numerical analysis show that single level additive Schwarz method is effective only for small number of subdomains ([23]). so M needs to be modified further similar to that of multilevel methods for PDEs, this modification process uses a coarse grid correction. With an addition of a coarse grid, we get a new preconditioner

$$M := R_0^T A_0^{-1} R_0 + \sum_{i=1}^{p} R_i^T A_i^{-1} R_i \ ,$$

which has been proved that is can be used as a "good" Schwarz preconditioning if the coefficient matrix A derives from an elliptic operator. Further more, solving a linear system also needs to choose a proper initial guess in order to decrease the computation cost. We use AMG algorithm to find a better initial guess so that the used Krylov method may converge more speedily. Considering the heterogeneous construction characteristics of some oil area, the so-called Watts correction is used which can also be considered as a coarse grid correction in some degree. As the third operator of oblique projection, coarse grid correction plays an important role in the linear solver of reservoir simulation.

If coarse grid correction is hoped to be used efficiently, matrix A should be elliptic. Due to the features of PDEs of (1), only the sub matrix A_{11} is elliptic, so we have to find a way to increase the effect of A_{11}, and decrease the effects of A_{22} and A_{33} in the whole coefficient matrix at the same time. CRP implements this goal in some degree, and we may find this effect from the following formula

$$AM_{\text{CRP}} \ := \ A(T_0 + T_1 - T_1 A T_0) = \begin{pmatrix} I & 0 & 0 \\ \varepsilon_{21} & \chi_{22} & \varepsilon_{23} \\ \varepsilon_{31} & \varepsilon_{32} & \chi_{33} \end{pmatrix},$$

where $T_1 = P_1(P_1^T A P_1)^{-1} P_1^T$ is a CRP, T_0 is a preconditioner of A such as M, P_1 is a projector such that $P_1^T A P_1 = A_{11}$, and the entries of ε_{ij} are much smaller than that of χ_{ii}.

Though CRP has decoupling effects in some degree, it isn't enough. We need to have a more powerful way to decouple the whole coefficient matrix, which is named as decoupling operator T_{Left} and is used as a left preconditioning such as

$$T_{\text{Left}} A x = T_{\text{Left}} f, \quad T_{\text{Left}}^{-1} = \begin{pmatrix} \text{diag}(A_{11}) & \text{diag}(A_{12}) & \text{diag}(A_{13}) \\ \text{diag}(A_{21}) & \text{diag}(A_{22}) & \text{diag}(A_{23}) \\ \text{diag}(A_{31}) & \text{diag}(A_{32}) & \text{diag}(A_{33}) \end{pmatrix}.$$

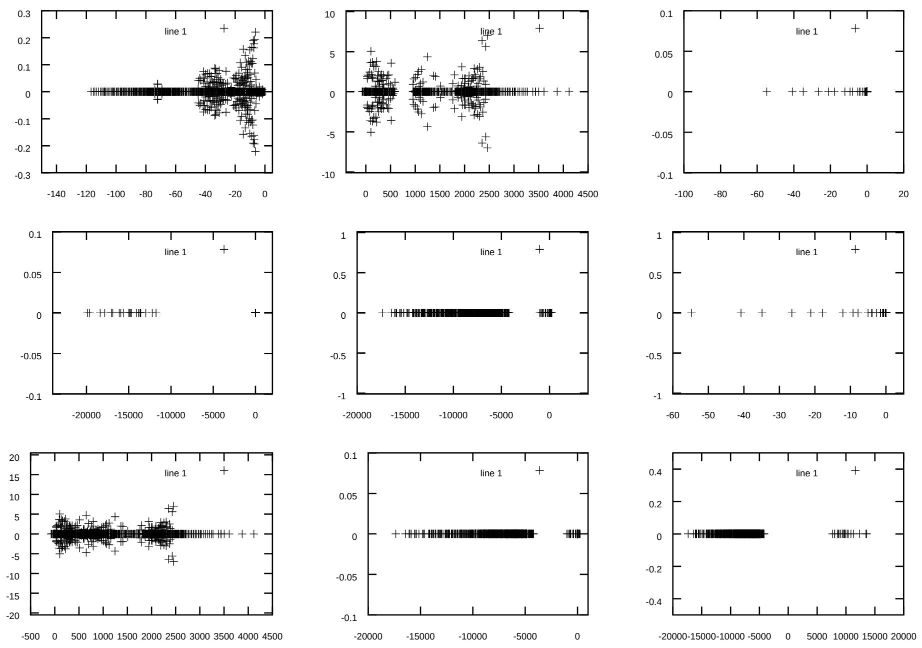

FIGURE 1. Spectral distribution of the original A_{ij}

The idea of decoupling operator is proposed as a way to weaken the coupling of drift-diffusion equations that occur in semiconductor device modelling. Experiments show that the decoupling operator leads to a significant clustering of eigenvalues associated with Jacobian matrices during the simulation process. Considering our simulator of black oil modelling, let $A^D := T_{\text{left}}A$, and $A^D_{ij} := (T_{\text{left}}A)_{ij}$ $(i = 1, 2, 3,\ j = 1, 2, 3)$, figures 1 and 2 give the spectral distribution of the nine sub matrices A_{ij} and A^D_{ij} respectively, and figure 3 gives the spectral distribution of matrices A and A^D. We observe through the figures 1 and 2 that, before decoupling process, all the nine sub matrices A_{ij} have obvious effects to the whole coefficient matrix A. Their spectral distributions show that their effect can not be neglected. However, after decoupling process, the effects of some sub matrix such as A_{12}, A_{13} and A_{23} is so little that they can even be neglected. Their eigenvalues are so small that they maybe considered as zero matrix without too much sacrifice of accuracy. Comparing the spectral distributions of A with A_{11}, and comparing the spectral distributions of A^D with A^D_{11}, we can see that matrix A is not similar to A_{11}, however, matrix A^D is very much similar to matrix A^D_{11}. These figures show significant effects of the decoupling preconditioning.

In summaries, by using Algorithm MSM, we construct a final preconditioner B for linear system (3), which consists of T_{left} and T_{right}, and satisfies

$$T_{\text{left}}AT_{\text{right}}T^{-1}_{\text{right}}u = T_{\text{left}}f\ . \tag{4}$$

Further more, T_{left} is a decoupling operator which deals with the coupled PDE and scaling, T_{right} satisfies

$$(I - AT_{\text{right}}) = (I - AT_c)(I - AT_2)(I - AT_1)(I - AT_0) \tag{5}$$

where, T_c consists of AMG preconditioning and Watts correction method, $T_2 = P_2(P_2^TAP_2)^{-1}P_2^T$ is CRPe for compositional model, $T_1 = P_1(P_1^TAP_1)^{-1}P_1^T$ is CRP

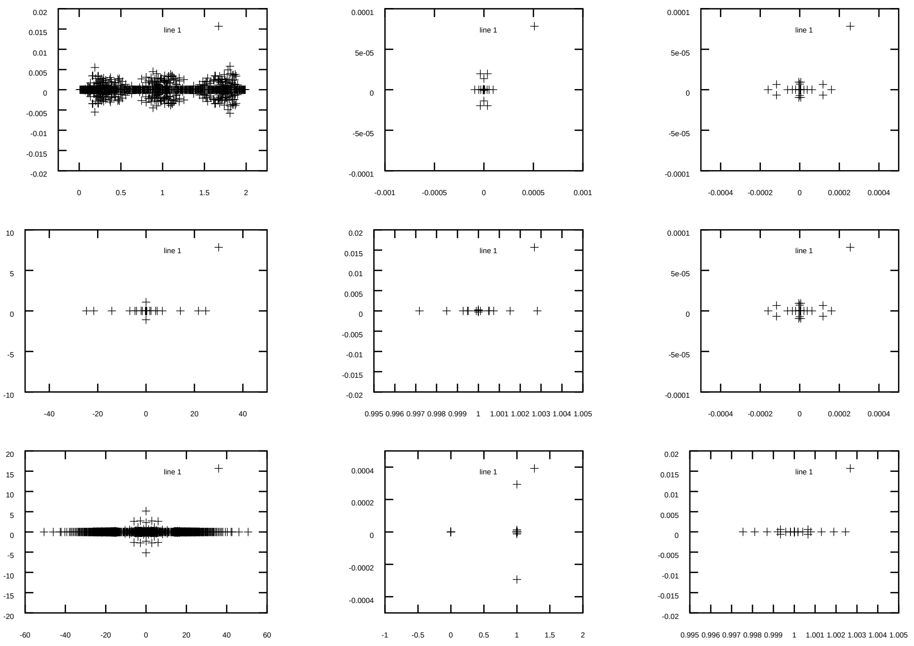

FIGURE 2. Spectral distribution of the preconditioned $(T_{\mathrm{Left}}A)_{ij}$

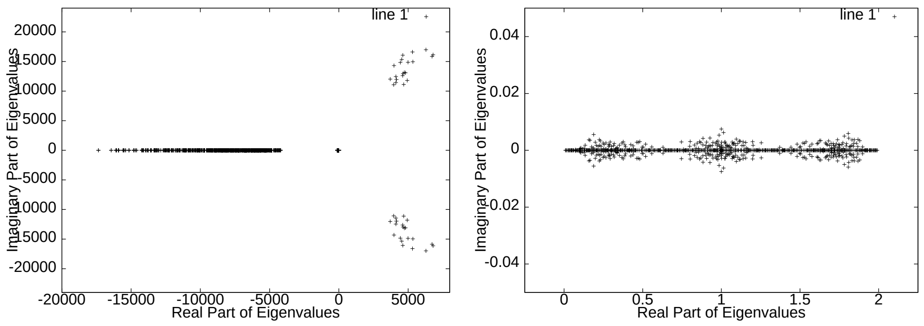

FIGURE 3. Spectral distribution of the original A and preconditioned $T_{\mathrm{Left}}A$

for black oil model which increases the effect of the pressure term in the whole matrix, T_0 is DDM preconditioning which deals with grid partition of the solving region, P_1 and P_2 are the two projection matrix. In both T_1 and T_2, relaxed ILU (.e.g. relaxedILU $:= 0.9 \times \mathrm{ILU}(\ell) + 0.1 \times \mathrm{MILU}$) is used in solving the sub region in its processor locally ([26]). Obviously, B has a similar form of multiplicative Schwarz algorithm. In fact, multiplicative Schwarz idea is used here for taking full advantage of "good" properties of A_{ij}, and Shur complement operation is used to do works related to block elimination processes.

5. Parallelism and Parallel Test Cases

The used parallel simulator is designed based on strategies of both domain decomposition and SPMD parallelism. After discretization process, the reservoir area is split across a number of processors by means of load balance. Currently, grid cells

	Case 1	Case 2	Case 3
number of discrete time steps	126	166	166
number of nonlinear systems	451	718	1326
number of linear systems	2669	5023	9662
number of FGMRES iterations	20744	24831	59345
number of ILU-GMRES iterations	237383	128590	342190
elapsed hours on 16 node/32 CPU	2.99	2.76	24.62
elapsed hours on 32 node/64 CPU	1.45	1.51	12.50

TABLE 1. Statistics of the three industrial test cases

in z-direction need to remain intact. The entire computational grid is partitioned and distributed to a logical 2-D mesh network of processors.

In this paper, the used hardware platform is a Beowulf cluster LSSC-II [27], which has 256 computational nodes. Each computational node has two Intel 2GHz Xeon CPU and 1GB physical memory. Both a fast Ethernet and a Myrinet 2000 are installed for every computational node. The used compilers include both GNU C/C++ and Intel Fortran V6. MPICH–GM 1.2.5 is used as a parallel communication library.

Industrial cases are tested to evaluate efficiency and effectiveness of our parallel simulator with solver INNS. The first case is a three-phase black oil model, with a $199 \times 87 \times 67$ grid system, 6 rock types, 291 wells, the simulated period is the 31.5 years exploitation history of a DaQing oil section of China. The second case is also a three-phase black oil model from the Chinese ShengLi Oilfield, the grid dimensions are $160 \times 320 \times 27$, or 1382400 grid blocks and 4147200 unknowns, there are 326 wells in the simulation region, and the matching history is 14 years. The third case is a finery of the same reservoir block as Case 2, with a $320 \times 640 \times 27$ grid system, or 5.5296 million grid blocks and 16.5888 million unknowns.

Table 1 gives some statistics of the three industrial cases simulated. In fact, the elapsed simulation time of the first two cases is roughly the same, the cost of Case 3 is about 9 times larger than that of Case 2.

The average time step length of Case 1 is $31.5 \times 365 \div 126 \simeq 91$ days, the same datum of both Case 2 and Case 3 is 31 days. For Case 1, each time step consists of 3.58 Newton steps in average, each Newton step averagely needs to solve 5.92 number of linear syetems, each linear system averagely needs 7.77 FGMRES iterations, and each FGMRES step needs 11.44 ILU-GMRES iterations in order to get an iterative preconditioning. For Case 2, the corresponding data are 4.33, 6.99, 4.94 and 5.18 respectively. For Case 3, the corresponding data are 7.99, 7.29, 6.14 and 5.76 respectively.

Comparing correlative data of the first two cases, we see that larger time step length may lead to more number of FGMRES iterations and more accurate preconditioning (which is in direct proportion to the average number of ILU-GMRES iterations for each FGMRES iteration step); the nonlinear feature of Case 2 is stronger than that of Case 1, so Case 2 needs more number of both Newton steps and linear systems in average for each time step; comparing with Case 1, the formed nonlinear equations of Case 2 are easier to solve.

Comparing correlative data of the last two cases, we observe that if the unknowns increases 4 times, the totally simulation cost will increase about 9 times, where the

	$CPU = 8$	$CPU = 16$	$CPU = 32$	$CPU = 64$	$CPU = 128$
Elapsed Time	8.71	5.59	2.99	1.45	0.87
Relative Speedup	1	1.56	2.91	6.01	10.01

TABLE 2. Elapsed times and relative speedups on LSSC-II

number of Newton step improves approximately 2 times, the computation workload of linear iteration improves 4 times, and the frequency of global reduction communication improves about 3 times.

The first test case, i.e., the DaQing black oil model has been simulated using up to 128 processors on LSSC–II with our parallel simulator. Table 2 gives elapsed wall-clock times and relative speedups with variable CPUs ranging from 4 to 128. The elapsed time is given in hours, and the relative speedup is computed with respect to the case of 8 processors.

The relative parallel efficiencies on 16, 32, 64, and 128 processors with respect to 8 processors are 78%, 73%, 75%, and 63%, respectively. The parallel efficiencies are quite satisfactory considering the communication complexity of the parallel nonlinear solver. The communication / computation ratio is almost 1:1 in the case of 128 processors, indicating that 8 to 128 processors are suitable for one million-grid cell problems of black oil model on this kind of machines.

For the past five years, the simulation time has reduced dramatically from two months to an hour for this real data. It means the total simulation capability speed up to 1600 times than before. After a detailed analysis, if we exclude the factors 40 of hardware contributions (which consist of fivefold CPU frequency increasing and at most eightfold potential concurrence process of the 64-CPU hardware system), the left 40 times speedup belongs to the improvement of our preconditioned nonlinear algorithm (at least speed fivefold) and elaborate parallel implementation.

Acknowledgments

This research was supported by the Major Basic Project of China (No.G19990328), the National High Technology Research and Development Program of China (863 Program, 2002AA104540), and the Information Construction of Knowledge Innovation Projects of the Chinese Academy of Sciences “Super computing environment construction and applications” (INF105-SCE).

References

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

[2] Mattax, C.C., Dalton, R.L., Reservoir Simulation, H.L. Doherty Memorial Fund of AIME,1990,Richardson, TX:SPE.

[3] Wallis J.R., et al., A New Parallel Iterative Linear Solution Method for Large-scale Reservoir Simulation, SPE 21209, presented at the SPE Symposium on Reservoir Simulation, Anaheim, California, Feberary 17-20, 1991, Society of Petroleum Engineers of AIME, 1991.

[4] Shiralkar G.S., et al., Falcon: A Production Quality Distributed Memory Reservoir Simulator, SPE Res. Eval. Eng., Oct. 1998

[5] Collins, D.A., Grabenstetter, J.E.,Sammon, P.H., A Shared-Memory Parallel Black-Oil Simulator with a Parallel ILU Linear Solver, SPE 79713 presented at the SPE Symposium on Reservoir Simulation held in Houston, Texas, Feberary 03C05, 2003,Richardson,TX:SPE,2003.

[6] Dogru A.H., et al., A Massively Parallel Reservoir Simulator for Large Scale Reservoir Simulation, SPE Paper 51886 presented at the SPE Symposium on Reservoir Simulation, Houston, Feberary 14-17,1999,Richardson,TX:SPE,1999.

[7] Killough J E, Commander D E, Scalable Parallel Reservoir Simulation on a Windows-NT Workstations Cluster, SPE 51883, presented at the Fifteenth SPE Symposium on Reservoir Simulation, Houston, February 14-17,1999, Richardson,TX:SPE,1999.

[8] Verdire S., et al., Applications of a Parallel Simulator to Industrial Test Cases, SPE Paper 51887 presented at the SPE Symposium on Reservoir Simulation, Houston, February 14-17,1999, Richardson,TX:SPE,1999.

[9] Abate J. et al., Parallel Compositional Reservoir Simulation on a Cluster of PCs, International Journal of High Performance Computing Applications, 2001, 15:13-21.

[10] Vassilevski, Y.V., Iterative Solvers for the Implicit Parallel Accurate Reservoir Simulator (IPARS), II: Parallelization Issues, TICAM Report 00-33, University of Texas at Austin, 2000.

[11] Wei Liu, Jianwen Cao, Mezzatesta A. et al., Parallel Reservoir Simulation on Shared and Distributed Memory System, SPE 64797, presented at the International Oil and Gas Conference and Exhibition, Beijing,China, November 7-10,2000,Richardson,TX:SPE,2000.

[12] Jianwen Cao, Efficient and effective solvers with preconditions in the parallel software of large-scale petroleum reservoir simulation, Ph.D thesis (in chinese), Institute of Software, the Chinese Academy of Sciences,2002.

[13] Friedlander A., Gomes-Ruggiero M.A., et al., Solving nonlinear systems of equations by means of quasi-Newton methods with a nonmonotone strategy, Optimization methods and Software, 1997,8:25-51.

[14] Martinez J. M., Practical Quasi-Newton methods for solving nonlinear systems, Journal of Computational and Applied Mathematics,2000, 124:97-122

[15] Jianwen Cao, Choi-Hong Lai, Numerical experiments of some Krylov subspace methods for black oil model, an International Journal of Computers and Mathematics with Applications, Elsevier, 2002, 44:125-141.

[16] Saad Y., Iterative Methods for Sparse Linear Systems, PWS Publishing Company, 1995.

[17] Wallis J.R., Incomplete Guassian Elimination as a Preconditioning for Generalized Conjugate Gradient Acceleration, SPE 12265, presented at the Reservoir Simulation Symposium , San Francisco, November 15-18, 1983, Society of Petroleum Engineers of AIME, ,1983.

[18] Jianwen Cao, Feng Pan, Jiachang Sun et al., Large-Scale Parallel Reservoir Simulation on Distributed Memory Systems, Proceedings of 2001 International Symposium on Distributed Computing and Applications to Business,Engineering and Science, Wuhan:Hubei Science and Technology Press, China, 2001.

[19] Bergamaschi, L., Moret, I., Giovanni Zilli, Inexact Quasi-Newton methods for sparse systems of nonlinear equations, Future Generation Computer Systems, 2001,18(1):41-53

[20] Kim Jong Gyun, Deo, M.D., Inexact Newton-krylov Methods for the Solution of Implicit Reservoir Simulation, SPE 51908 presented at the SPE Symposium on Reservoir Simulation, Houston, Feberary 14-17,1999,Richardson,TX:SPE,1999.

[21] Lacroix,S., Vassilevski, V.V., Wheeler, M.F., Iterative Solvers of the Implicit Parallel Accurate Reservoir Simulator (IPARS), I: Single Processor Case, TICAM Report 00-28, University of Texas at Austin, 2000.

[22] Jiachang Sun,Jianwen Cao, Large Scale Petroleum Reservoir Simulation and Parallel Preconditioning Algorithms Research, Science in China Ser.A (Mathematics), 2004, 47:32-40

[23] Smith B. F., Bjorstad P.E. and Gropp W.D., Domain Decomposition: Parallel multilevel methods for elliptic partial differential equations, Cambridge University Press, 1996

[24] Saad Y., Schultz M.H., GMRES: a generalized minimal residual algorithm for solving nonsymmetrical linear systems, SIAM Journal on Scientific and Statistical Computing,1986, 7:856-869.

[25] Van der Vorst H.A., Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems, SIAM Journal on Scientific and Statistical Computing,1992,12:631-644.

[26] Golub G.H. and Van der Vorst H.A., Closer to the Solution: Iterative Linear Solvers, Universiteit Utrecht, Preprint No. 982, October, 1996.

[27] Linbo Zhang, et al., Teracluster LSSC-II: Its Designing Principles and Applications in Large Scale Numerical Simulations, Science in China Ser.A (Mathematics), 2004, 47:53-68

Laboratory of Parallel Computing, Institute of Software, Chinese Academy of Sciences, Beijing 100080, China

E-mail: `cao@rdcps.ac.cn`

URL: `http://www.rdcps.ac.cn/∼cao/`