

A New Robust and Vectorized ILU-based Preconditioning Technique for Reservoir Simulation

RICOIS, Olivier, BEICIP-FRANLAB, FRANCE
SARDA, Sylvain, Institut Français du Pétrole, FRANCE
MEGE, Philippe, Institut Français du Pétrole, FRANCE

Paper presented at the 5th European Conference on the Mathematics of Oil Recovery, Leoben, Austria, 3-6 Sept. 1996

ABSTRACT

To date, the most efficient solver used in the petroleum industry for the resolution of linear systems in reservoir simulation is the generalized conjugate-gradient acceleration method called Bi-CGSTAB. However, difficulties still appear in matrix resolution when the Bi-CGSTAB method is used without an appropriate preconditioner.

Moreover, most effective preconditioners are very costly in terms of memory consumption and the high recursivity of their algorithm is often a problem on vector computers.

In this study, a variant of the Incomplete Block LU Factorization coupled with Bi-CGSTAB has been implemented in a multipurpose reservoir simulator. It provides an efficient solver with low memory cost well suited for difficult problems such as heterogeneous or dual permeability models with local grid refinement. The algorithm has been vectorized and is therefore highly efficient on supercomputers such as Cray or Fujitsu. The factorization is based on a special matrix ordering by blocks in which Incomplete LU factorization is performed. This sorting varies according to the problem.

This paper describes the definition of the matrix blocks in different cases (regular or refined grid blocks, fractured reservoir). Numerical results are presented for several real reservoir applications and show the robustness of this technique compared with conventional ILU preconditioners.

INTRODUCTION

In the last 10 years, important technology advances drastically increased the computing power available for reservoir simulation. At the same time, features of reservoir simulators became more and more complex. Nowadays, full field sophisticated EOR models such as

compositional, thermal, dual porosity and permeability models, coupled with fully implicit numerical schemes, are commonly used in reservoir simulation. As problem size increases, the linear solver often dominates the total execution time. Then, direct solvers cannot be used because of the huge memory and CPU requirements. Moreover, with use of local grid refinement around wells, the Jacobian matrices generated by simulators are no more block-banded matrices with constant bandwidth.

Acceleration methods based on Krylov algorithms were developed and adapted for reservoir simulations to overcome these difficulties. In the following section, we present a standard acceleration method commonly used in reservoir simulation. This method has been improved by an appropriate preconditioner for very ill-conditioned linear systems which is presented thereafter. Then, some simulation experiments carried out with different resolution methods are discussed.

BI-CGSTAB AND ILU PRECONDITIONER

1. The Bi-CGSTAB Algorithm

Krylov algorithms are among the fastest and most robust iterative solvers for a wide variety of applications.

The Bi-CGSTAB method developed by Van der Vorst^[1] as a variant of the Bi-CG algorithm^[2] is one of the most efficient and economic solver within the Krylov family of non-symmetric solvers.

The basic idea, as in all the Krylov algorithms, is the projection of the original set of equation $Ax = b$ on an increasing Krylov subspace $K^i(A, r_0)$, where r_0 is the initial residue for an initial guess x_0 .

$$r_0 = b - Ax_0 \quad (1)$$

$$K^i(A, r_0) = \{r_0, Ar_0, A^2r_0, \dots, A^i r_0\} \quad (2)$$

The resulting set of equations is then solved and the projected solution is used to correct the solution of the original problem. The projection operator is computed so that r_j satisfies the relation $r_j = P_j(A)Q_j(A)r_0$ where P and Q are polynomial functions of A . P is the Bi-CG polynomial that creates an A -orthogonal Krylov basis and $Q_i(x) = (I - \omega_1x)(I - \omega_2x)\dots(I - \omega_ix)$ where the ω_i are defined recursively so as to minimize $r_i = P_i(A)Q_i(A)r_0$. Thus, Bi-CGSTAB can be seen as the product of Bi-CG and GMRES(1) or GCR(1)^[6]. Due to the local minimization, Bi-CGSTAB displays a much smoother convergence than Bi-CG and does not use the transpose operator of A .

2. The ILU preconditioner

Overall performance of Krylov methods can be improved when one uses preconditioning. Instead of solving the system $Ax = b$, the system $M^{-1}AM^{-1}(M_2x) = M^{-1}b$ is solved where $M^{-1} = M_2^{-1}M_1^{-1}$ is an approximation to A^{-1} and is easier to compute. Since only matrix-vector products are needed for the Bi-CGSTAB method, it is not necessary to explicitly form AM_2^{-1} . By rewriting the steps of the algorithm, only two new computational steps 'solve u from $Mu = v$ ' are introduced^[1].

Currently, a wide variety of preconditioners are available. Attention must be focused on the fact that changing the preconditioner changes the convergence criteria, which is based on the size of the residue. The preconditioner choice is computer (scalar or vector machines) dependent as well as problem dependent.

A broad class of preconditioners is based on incomplete factorization of the coefficient matrix. Such a preconditioner is then given in a factorized form $M = LU$ with L lower and U upper triangular matrices. The efficiency of the preconditioner depends how well M approximates A .

The point incomplete factorization ILU(0) preconditioner can be seen in two ways:

The first is to perform a Gaussian elimination of A and store only given elements, usually so that the preconditioner has the same sparsity pattern as A .

The second is to have a preconditioner P with same sparsity pattern as A , and to compute the elements of P by minimizing the norm of $(I - PA)$.

The ILU(0) factorization is a good candidate for reservoir simulation matrix preconditioning and is recommended for most problems, especially on scalar machines.

However, for difficult problems such as heterogeneous or dual permeability models, it may not converge. Moreover, in the absence of significant vectorization, this preconditioner is not suitable for vector computers.

BLOCK FACTORIZATION

Several authors proposed variants of ILU preconditioners^[9]. Well known examples are block SOR^{[3][8]} and the nested factorization algorithm^[4].

Recently, Meyerink^[5] has proposed an incomplete block factorization of the matrix for a three dimensional case with a regular geometry. The differential equation is approximated by a five point finite difference equation and the first-order convective term is discretized using the usual upstream formulation.

The Bi-CGSTAB algorithm is applied to the entire Jacobian matrix generated by discretization of both mass balance equations and well constraint equations (with the well coupling option). However, it is not necessary to have a preconditioner for the well terms, since they are not a problem for Bi-CGSTAB convergence. The preconditioned matrix A is then partitioned as follow:

$$A = D + L_1 + U_1 + L_2 + U_2 + L_3 + U_3 \quad (3)$$

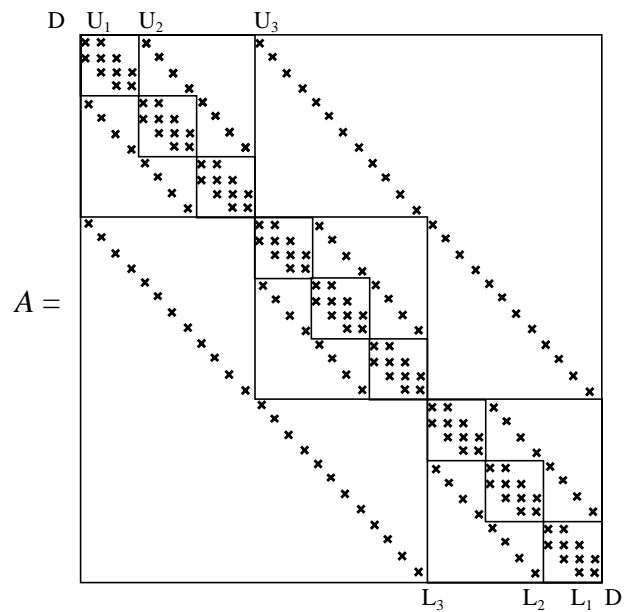


Fig. 1-Matrix coefficients for a 3x3x4 grid.

The algorithm of the block factorization can be described as follow:

A LU factorization is computed on the first diagonal block A_{11} . The factorization is exact because diagonal blocks are tridiagonal submatrices. Then, elements of L_2 and L_3 below the first diagonal block are eliminated. To keep the same sparsity pattern as matrix A , only the diagonal of A^{-1}_{11} and the two codiagonals above and below this one are used to eliminate these elements. This process is applied to the other diagonal blocks.

This block factorization gives an attractive alternative to standard incomplete LU(0) preconditioners for three reasons:

1. The algorithm can easily be **vectorized** because of the

defined block structure.

2. It requires **less memory space**. In fact, only the LU factorization of the diagonal blocks are stored (two diagonals for a normalized Jacobian matrix A).

3. This preconditioner is **more efficient** than incomplete LU(0) because the factorization of the diagonal blocks is exact. For a diagonally dominant matrix, this improves the convergence behaviour of Bi-CGSTAB.

GRID REFINEMENT

Advanced reservoir simulators enable complex 3-D local grid refinement with several levels of subgridding so that the structure of the Jacobian matrix is no more regular.

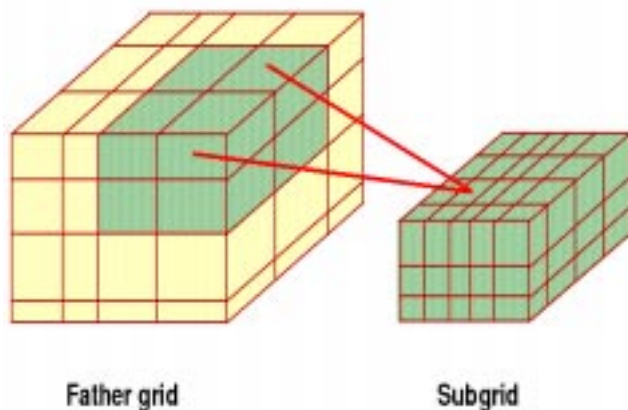


Fig. 2-Example of grid refinement.

This paper describes generalisation of Block factorization algorithm to an irregular matrix generated by a local grid refinement.

First, a privileged direction is computed. This is the direction for which the sum of transmissivities is the highest. Generally, the privileged direction is the Z direction because of the cell dimensions. Then, pseudoblocks are defined. A block contains all cells connected in the privileged direction. This leads to determine n blocks of different sizes without connection between them in the privileged direction. A ILU(0) factorization is then performed on each submatrix. Note that the factorization may be incomplete if grid refinement generate connections in the privileged direction. The chosen preconditioner is the block diagonal matrix M , with as blocks submatrices M_{ij} where:

M_{ii} is a $N_i \times N_i$ ILU(0) factorization of i^{eme} blocks.

M_{ij} , ($i \neq j$) is a zero matrix elsewhere.

DUAL PERMEABILITY MODEL

With the dual permeability option, the structure of the matrix is:

$$A = \begin{bmatrix} [MM] & [MF] \\ [FM] & [FF] \end{bmatrix} \quad (4)$$

$[MM]$ describes the matrix-matrix exchanges. It is a zero matrix in a dual porosity-single permeability model, a banded matrix when dual permeability and regular geometry options are used.

$[FF]$ describes the fracture-fracture exchanges. It is a banded matrix in case of regular geometry.

$[MF]$ and $[FM]$ describes the matrix-fracture exchanges. In case of a regular geometry, they are banded matrices and their coefficients cause the matrix A to be ill-conditioned because it is not a diagonally dominant matrix any more. Conventional ILU preconditioners are not good candidates for a dual permeability case because Bi-CGSTAB may have an irregular convergence behaviour. Moreover, dual permeability model generally includes a high number of cells because of the two media and vector aspects become prenominent.

The previous block LU factorization can be generalized for dual permeability cases. The principle does not change. The preconditioner is a diagonal blocks matrix. The diagonal blocks are built with both the connections of cells in the direction of the highest sum of transmissivities and the matrix-fracture and fracture-matrix connections.

CODE IMPLEMENTATION

The block LU factorization has been implemented in ATHOS^[10] multipurpose reservoir simulator for regular and refined geometry, black-oil, compositional, single and dual permeability models. For almost all tests run, a preconditioner built from the pressure equations is sufficient even with a fully implicit numerical scheme. But, for very difficult compositional models, it may be necessary to have a preconditioner for the entire Jacobian matrix A . Construction of the preconditioner and its application have both been vectorized which made easier the optimization of Fortran compilers even on scalar machines.

NUMERICAL RESULTS

To measure the efficiency of the block LU factorization, three test problems based on field study were carried out on different machines. For each problem, three resolution methods were compared: Bi-CGSTAB without preconditioner, with ILU(0) preconditioner and block LU preconditioner.

Firstly, a representative Jacobian matrix has been extracted on which the different method behaviours are highlighted. Figures show the decimal logarithm of the relative residual

norm versus the number of iterations. The relative residual norm at iteration i is defined as:

$$\epsilon = \|r_i\| / \|r_0\| \tag{5}$$

In this equation, r_i is the residue at iteration i and r_0 the residue of the arbitrary initial guess.

Secondly, graphs were drawn to show the cumulative number of solver iterations versus the simulated time. Values were taken from the simulation numerical reports. Such an information is directly connected with the solver robustness tested during the simulation. A good method should lead to a smooth shaped curve which a low slope straight line fit.

Thirdly, CPU times are presented for the whole simulation using the three methods on IBM RS6000 and Fujitsu VP2400. These are the measured cpu times per successful time step and per cell. A time step is validated when no problem has occurred with the solver convergence (a maximum number of solver iteration is defined), the Newton's balance and the maximum variation of pressure and saturation. Thus, a successful time step can include several time step cuts and reruns which increase its measured cpu time.

Problem 1.

This is a dual permeability case dealing with black-oil 3-phase flow. It is a 25,476 cell father grid with 5 subgrids and 3 refinement levels.

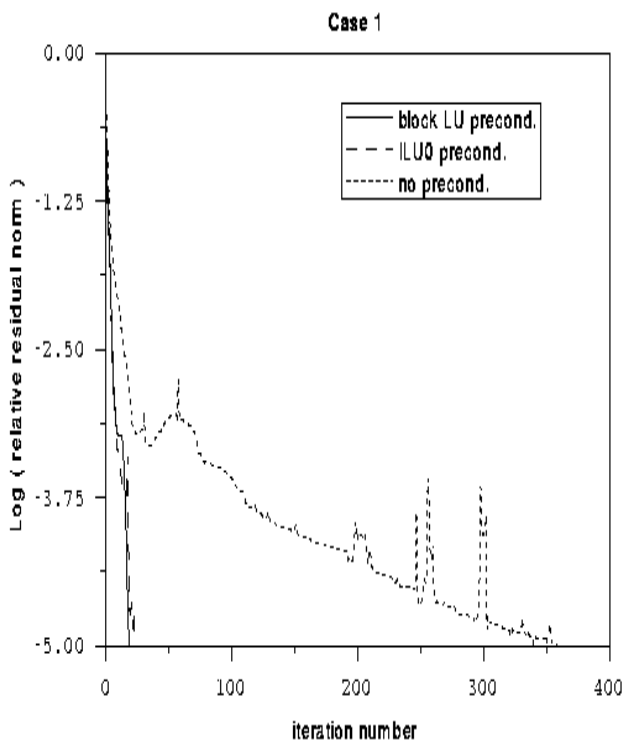


Fig. 3-Example 1: Convergence behaviours.

There are 9,053 active cells in the matrix and 2,170 active cells in the fracture (the others are dead cells). As explained above, the dual permeability features result in ill-conditioned Jacobian matrices.

Actually, as shown in Fig. 3, the non preconditioned solver has a slow convergence behaviour compared with others: the relative residual norm drops below 10^{-5} (stopping criterion) after 359 iterations.

The two preconditioners considerably decrease the number of iterations needed for convergence. Their behaviours are quite similar but the block LU preconditioner may iterate less than the other one (respectively 20 and 24).

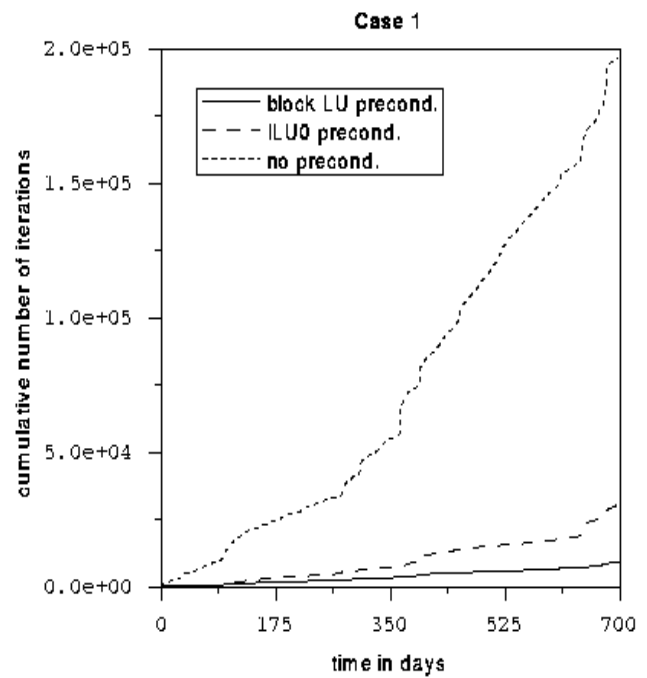


Fig. 4-Example 1: Resolution profile.

The Fig. 4 shows that the non preconditioned solver iterates nearly 200,000 times for a 700 day simulation. The periods when convergence problems occur correspond to the near-vertical parts of the curve. There are no such variation for the two other solver curves. So, the numbers of iterations for the ILU(0) and block LU preconditioners are much lower (respectively 35,000 and 10,000 at time 700 days).

On the IBM RS6000 scalar machine, these differences are confirmed at the whole simulation scale by the CPU times presented in Table 1. Nevertheless, block LU preconditioner appears to be faster than its compared behaviour should let us expect. On Fujitsu VP2400, the poor vector capabilities of the ILU(0) preconditioner decrease its performance. On the other hand, block LU preconditioner remains the most effective method.

| Resolution methods | IBM RS 6000 | FUJI VP2400 |
|--------------------|-------------|-------------|
| block LU | 2.9 ms | 0.570 ms |
| ILU(0) | 11.3 ms | 1.96 ms |
| No Preconditioner | 23.5 ms | 1.42 ms |

Table 1: CPU times per time step and per cell.

Problem 2.

This is the SPE9 Comparative Solution Project problem, An Expanded Three-Dimensional Problem with a Geostatistical Distribution of Permeability. This is an heterogeneous case dealing with black-oil 3-phase flow with a 9,000 cell regular grid.

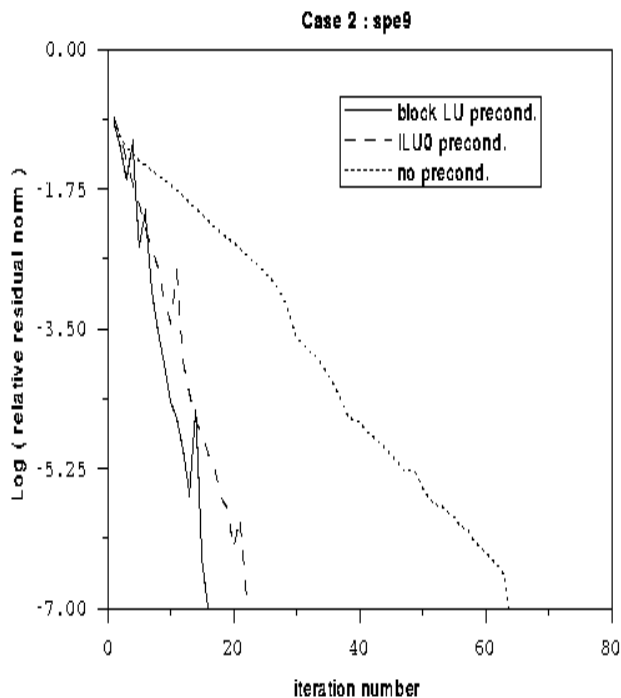


Fig. 5-Example 2: Convergence behaviours.

As shown in Fig. 5, the three methods have a quite regular convergence behaviour and the maximum number of iterations is only 64 for the non preconditioned solver with a 10^{-7} stopping criterion. Then, the use of preconditioners is not as effective as in the first example (16 and 23 iterations respectively for block LU and ILU(0) preconditioners).

In Fig. 6, the three curves have similar shapes. Thus, during simulation, the preconditioners reduce the number of solver iterations in a nearly constant way. These reduction factors are respectively 2.8 and 3.3 for the ILU(0) and block LU preconditioners.

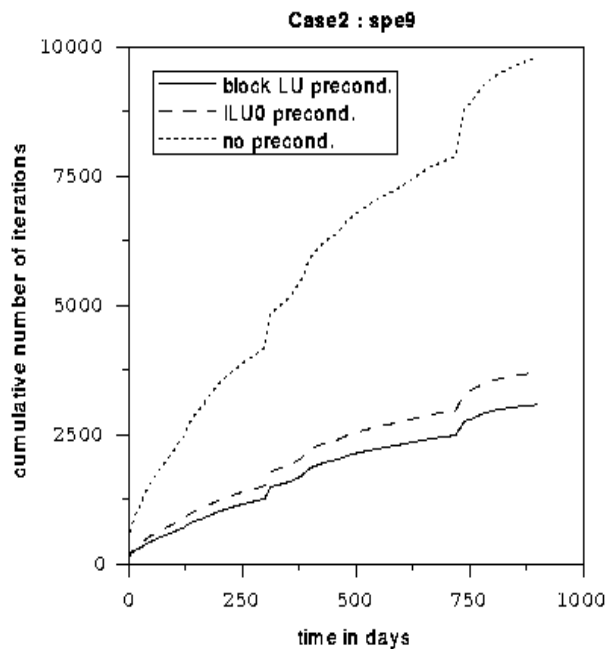


Fig. 6-Example 2: Resolution profile.

On IBM RS6000, the CPU times presented in Table 2 for the two preconditioners are close, more or less twice as small as in the third method.

On the other hand, the block LU preconditioner and the non preconditioned method realize nearly equal performances on the Fujitsu VP2400 machine. So, block LU preconditioner remains a good candidate for this case even on vector machine.

| Resolution methods | IBM RS 6000 | FUJI VP2400 |
|--------------------|-------------|-------------|
| block LU | 0.9 ms | 0.21 ms |
| ILU(0) | 1.1 ms | 0.31 ms |
| No Preconditioner | 2.0 ms | 0.20 ms |

Table 2: CPU times per time step and per cell.

Problem 3.

This is a multi component case using a 10,603 cell grid with local grid refinement (17 subgrids and 3 refinement levels).

In addition to grid refinement that damages the Jacobian matrix conditionement, vertical transmissivities appear to be much higher than the horizontal ones. That may explain the ILU(0) preconditioner ineffectiveness for that case.

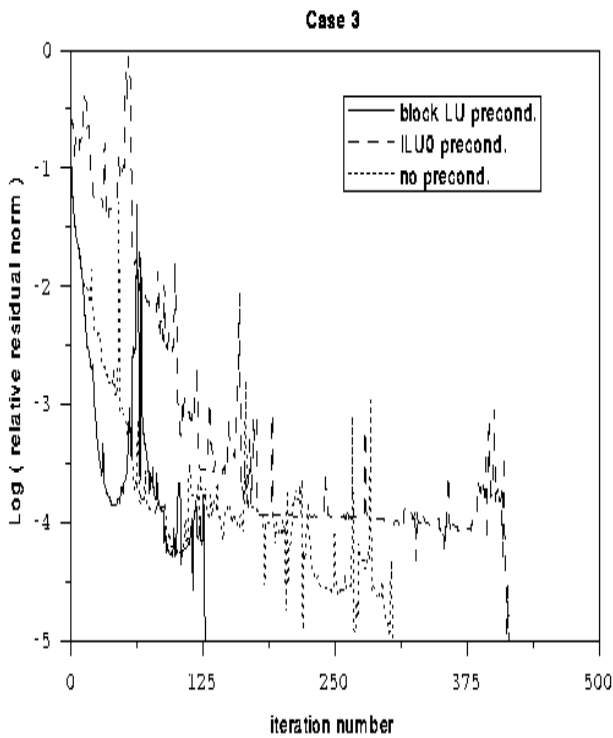


Fig. 7-Example 3: Convergence behaviours.

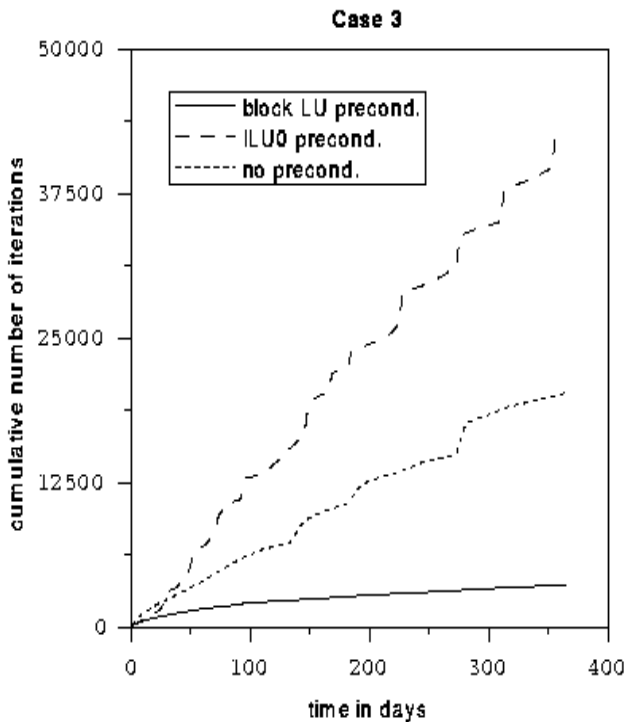


Fig. 8-Example 3: Resolution profile.

As we can see in Fig. 7, it takes 415 iterations to converge with a 10^{-5} stopping criterion, much less than the non preconditioned solver (305 iterations). Due to the block LU algorithm, the high vertical transmissivities (in that case, Z is the privileged direction) are no more a problem and the convergence is obtained after 128 iterations. Note that no method has a smooth convergence behaviour, which is the sign of a difficult case anyway.

For this case, Fig. 8 is particularly interesting. The ILU(0) preconditioner corresponds to the winding curve, meaning that a lot of convergence problems have been encountered (42,800 iterations for one year). The non preconditioned solver has three critical periods near times 140, 190 and 280 days (20,250 iterations). But the block LU preconditioner results in a perfect curve (3,650 iterations). Therefore, the way it takes into account geometry and physical aspects of this problem clearly improves the quality of the simulation.

| Resolution methods | IBMRS 6000 | FUJI VP2400 |
|--------------------|------------|-------------|
| block LU | 8.7 ms | 0.85 ms |
| ILU(0) | 24.9ms | 4.06 ms |
| No Preconditioner | 23.2 ms | 1.66 ms |

Table 3: CPU times per time step and per cell.

CPU times presented in Table 3 complete the precedent observations. ILU(0) preconditioner is the worst method on scalar and vector machines. Concurrently, the robustness of the block LU preconditioner is proved since it is the fastest method on both machines again.

CONCLUSIONS

1. A new robust and vectorized preconditioner is proposed for reservoir simulation. It is based on a LU blocks factorization. The construction of the blocks depends on geometry and physics simulated by the model.
2. The block LU factorization is cheaper than conventional preconditioner in terms of storage requirements and CPU consumptions.
3. It can be easily implemented in an industrial reservoir simulator and vectorized without memory or computing time overhead.
4. The block structure of this preconditioner is well suited for matrix structures of different physical problems and so, it considerably improves the convergence behaviour of Bi-CGSTAB.

ACKNOWLEDGEMENTS

The authors are grateful to the BEICIP-FRANLAB and Institut Français du Pétrole companies for permission to publish this paper. They also wish to thank Luce Weill from Institut Français du Pétrole for her advice and cooperation in evaluation of preconditioning methods.

REFERENCES

- [1] Van der Vorst, H. A., *Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems*. SIAM, J. Sci. Statist. Comput., 13, 1992, pp. 631-644.
- [2] Lanczos, C., *Solution of systems of linear equations by minimized iterations*. J. Res. Natl. Bur. Stand 49, 1952, pp 33-53.
- [3] Quandalle, P., *Vectorization and Parallel Processing of Models With Local Grid Refinement*. SPE 21210, Anaheim 1991.
- [4] Appleyard, J.R. and Cheshire, I. (1983), *Nested factorization*. Paper SPE 12264, Seventh SPE Symposium on Reservoir Simulation, San Francisco, California.
- [5] Meyerink, J.A., *Iterative methods for the Solution of Linear Equations based on Incomplete Block Factorization of the Matrix*. SPE 1262 San Francisco, 1983.
- [6] Chan, T. F., Gallopoulos, E., Simoncini, V., Szeto, T., Tong, C. H. *A quasi-minimal residual variant of the Bi-CGSTAB Algorithm for nonsymmetric systems*. SIAM, J. Sci. Statist. Comput., 15, 1994, pp 338-347.
- [7] Saad, Y and Schultz, H, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM, J. Sci. Statist. Comput., Vol 7 1986, pp 856-869.
- [8] Ashcraft, C. and Grimes, R., *On vectorizing incomplete factorizations and SSOR preconditioners*. SIAM J. Sci. Statist. Comput./ 9 (1988), pp 1542-1568.
- [9] Axelsson, O., *Incomplete block matrix factorization preconditioning methods. The ultimate answer?* J.Comput. Appl. Math., 12&13 (1985), pp.3-18.
- [10] Lefevre, D., Pellissier, G., Sabathier J.C., *A new Reservoir Simulation System for a Better Reservoir Management*, SPE 25604, 1993.