'ROCEEDINGS, Twenty-Second Workshop on Geothermal Reservoir Engineering
tanford University, Stanford, California, January 27-29, 1997
GP-TR-155

T2SOLV: AN ENHANCED PACKAGE OF SOLVERS FOR THE TOUGH2 FAMILY OF CODES

George 3. Moridis and Karsten Pruess

Lawrence Berkeley National Laboratory
Earth Sciences Division, 1 Cyclotron Road, 90-1116
Berkeley, California 94720

ABSTRACT

T2SOLV is an enhanced package of solvers for the TOUGH2 family of codes. T2SOLV includes all the 'reconditioned Conjugate Gradient (PCG) solvers ised in T2CG1, the current solver package, as well as LUBAND, a new direct solver, and DLUSTB, a PCG solver based on the BiCGSTAB method. Additionally, T2SOLV includes the D4 grid numbering scheme and two sets of preprocessors. Results From test problems indicate that LUBAND is faster, more reliable and requires less storage than MA28, the BiCGSTAB solver is superior to the other PCG methods in T2SOLV, and that the preprocessors improve the performance of the PCG solvers and allow the solution of previously intractable problems.

INTRODUCTION

Most of the computational work in the numerical simulations of fluid and heat flows in permeable media arises from the solution of large systems of linear equations **Ax = b,** where **A** is a banded matrix of order *N*, **x** is the vector of the unknowns, and **b** the right-hand side. These are solved using either direct or iterative methods. The most reliable solvers are based on direct methods. The robustness of direct solvers comes at the expense of large storage requirements and execution times. Iterative techniques exhibit problem-specific performance and lack the generality, predictability and reliability of direct solvers. These disadvantages are outweighed by their low memory requirements and their speed especially in the solution of very large matrices.

In the TOUGH2 general-purpose reservoir simulator ***[Pruess,*** 1991] the matrix **A** is a Jacobian with certain consistent characteristics. **A** has a known block structure with well defined sparsity patterns. Typically, **A** is non-symmetric, not positive definite, not diagonally dominant and ill-conditioned. Due to the fact that **A** is a Jacobian, the elements of **A** in a single row may vary by many orders of magnitude. TOUGH2 creates very challenging matrices with all the attributes that cause most iterative techniques to fail. In addition, the general-purpose nature of TOUGH2 means that different matrix characteristics may arise for different types of problems. This explains the past heavy reliance of TOUGH2 on the direct solver MA28 [*Duff*, 19771.

In the current TOUGH2 version, T2CG1, a package of preconditioned conjugate gradient solvers complement the MA28 direct solver and significantly increases the size of tractable problems. T2CG1 includes three solvers: (a) DSLUBC, a routine based on the Bi-Conjugate Gradient (BiCG) method, (b) DSLUCS, a Conjugate Gradient Squared (CGS) routine, and (c) DSLUGM, a Generalized Minimum Residual (GMRES) routine. Tests of T2CG1 ***[Moridis** and **Pruess,*** 1995] on a variety of computing platforms and from systems with up to 30,000 equations have shown that the PCG routines in T2CG1 are significantly (and invariably) faster than MA28 and require far less memory.

In this paper we discuss T2SOLV, an enhanced package of solvers for the TOUGH2 family of codes. It was developed as a replacement for T2CG1, the current solver package. T2SOLV includes all the Preconditioned Conjugate Gradient (PCG) solvers used in T2CG1 as well as a new routine, DLUSTB, based on the Bi-Conjugate Gradient Stabilized (BiCGSTAB) method. T2SOLV also replaces the MA28 by LUBAND, a general banded-matrix direct solver. Additionally, it includes an option for using the D4 ordering scheme and two sets of matrix preprocessors to enhance the PCG performance.

THE LUBAND SOLVER

LUBAND is a direct solver which replaces the **MA28** solver currently used in the TOUGH2 family of codes. It is derived from routines in the LAPACK [1993] package, which have been enhanced and extensively modified to conform to the TOUGH2 architecture and memory management approach. It is based on a LU decomposition with partial pivoting and row interchange, and allows the solution of systems with a large number of zeros on the main diagonal. Unlike MA28 (which is a general solver), LUBAND is a banded matrix solver, and as such it capitalizes on the significantly lower and well defined memory requirements of this class of solvers.

LUBAND can be applied without any problem in the current TOUGH2 version and is fully backward compatible with all older input data files. The MESHMAKER routine was also enhanced to minimize the bandwidth of matrix **A.** Defining work W as the number of multiplications and divisions necessary to convert the full matrix to an upper triangular form and to perform back substitution, *Price and Coats* [1974] showed that for direct solvers $W = NM^2$ and the minimum storage $S = NB$, where N is the order of the matrix and M its half-bandwidth, the full bandwidth being $B = 2M+1$.

For a given problem size N, work and storage are minimized when M is minimized. If I, J, K are the number of subdivisions in the x-, y- and z-directions respectively, the shortest half-bandwidth is $M=JK$ when $I>J>K$. This is called *standard* ordering [*Aziz and Settari*, 19791, and the resulting matrices are banded. As W increases with the square of M, it is obvious that the penalty for non-optimization of the ordering of equations may be substantial.

THE DLUSTB SOLVER

DLUSTB was developed based on the BiCGSTAB(m) algorithm [***Sleijpen** and Fokkema*, 19931, a recent extension of the more traditional BiCGSTAB algorithm of *van der Vorst* [1992] which is still an option in T2SOLV. It was developed to address the problem of irregular convergence behavior of the PCG solvers in situations where the iterations are started close to the solution (e.g. when approaching steady state). This is a weakness which afflicts most PCG solvers, and may lead to severe residual cancellation and errors. BiCGSTAB(m) alleviates the irregular (oscillatory) convergence common to the BiCG [*Fletcher*, 19761 and CGS [*Sonneveld*, 19891 methods, thus improving the speed of convergence. It also alleviates potential stagnation or even breakdown problems which may be encountered in traditional BiCGSTAB. According to ***Sleijpen** and Fokkema* [1993], BiCGSTAB(m) combines the speed of BiCG with the monotonic residual reduction in the Generalized Minimum Residual (GMRES) method, while being faster than both. Theoretical analysis indicates that the BiCGSTAB(m) algorithm is especially well-suited to the solution of very large (i.e. N>50,000) problems [*vander Vorst*, 19921.

DLUSTB uses the Boeing-Harwell matrix storage scheme of TOUGH2, and has the same architecture as the other routines in T2SOLV. As in all other PCG solvers in T2SOLV, it uses a modified LU decomposition for preconditioning. Its memory requirements increase linearly with the order m *of* the Minimal Residual polynomial. For m = 4, it requires twice the memory of BiCG or CGS. The optimum value of m is calculated internally in DLUSTB.

THE D4 SCHEME

The Alternating Diagonal Scheme (D4) for gridblock ordering was added as an option to T2SOLV. D4 is a technique belonging to the matrix-banding class of solvers, which derives its benefits from the numbering of the grid points. More details can be found in *Price and Coats* [1974].

D4 ordering partitions the matrix into four distinct entities. This structure allows forward elimination through the equations in the lower half of **A,** which zeroes all original entries in the lower left quadrant of **A** and transforms it into a null matrix, while creating non-zero entries in the submatrix $\mathbf{A}_{LU}$ in the lower right quadrant of **A.** The submatrix $\mathbf{A}_{LU}$ is of order $N/2$, and allows the calculation of the lower half of **x,** from which the upper half is obtained by simple substitution. The resulting reduced matrix $\mathbf{A}_{LU}$ can be solved using either direct or iterative methods.

D4 numbering reduces the order of the matrix by 50% while not increasing the bandwidth. Depending on the grid geometry, D4 makes possible execution speed improvement by a factor ranging between 2 and 5.85 [*Price and Coats*, 19741 over standard ordering. Moreover, it reduces storage requirements by a factor of 2. Compared to iterative solvers, D4 is competitive in 2-D systems and slower in 3-D systems, and offers the advantage of a robust solution. D4 with LUBAND makes possible the robust direct solution of large multi-dimensional problems. However, D4 can only be used with regular grids.

THE Z-PREPROCESSORS

Some of the most numerically challenging matrices arising in TOUGH2 simulations involve a large number of zero entries on the main diagonal of the Jacobian. Such matrices are quite common in non-isothermal two-component systems (such as modeling of two-water geothermal systems using the EOS 1 module) and result in at least $0.5N$ of non-zero entries on the main diagonal of the matrix.

Such matrices pose no problem for the LUBAND direct solver. The iterative solvers, however, are directly affected by the diagonal dominance of the matrix and the relative number of the zero entries on the main diagonals. Up to $0.1N$ zero elements have little discernible effect on the PCG solvers in T2SOLV. Matrices with as many as $0.3N$ (and occasionally up to $0.5N$) zero elements are tractable without any special treatment, but usually require a large number of iterations for convergence, i.e. exceeding $0.5N$.

'he four Z-preprocessors implemented in T2SOLV nhance the performance of the PCG solvers in matri-es with a large number of main-diagonal zeros. 'hese preprocessors are invoked only when (a) PCG olvers are used to solve (b) matrices with main diag-nals populated with a large number of zeros and (c) he number of the primary variables NEQ>1.

'he first option, Z1, replaces the zeros with a small lumber (typically 10^{-25}), and can substantially de-rease the number of iterations for convergence in ma-rices with as many as $0.5N$ zero main-diagonal ele-nents. The performance of the PCG solvers in Z1-rocessed matrices deteriorates rapidly when the main-liagonal zero elements exceed $0.5N$.

'he second pre-processing option, 22, is more com-utationally intensive and involves linear combina-ions of the flow equations in each gridblock. Z2 in-ludes a search algorithm which identifies the appro-riate equation to be added to the equation correspond-ng to the zero main-diagonal element. By adding the wo equations, the corresponding elements in the acobian are replaced with the non-zero sum of the riginal elements. The 22 option requires limited omputational effort and significantly improves the erformance of the PCG solvers.

While very effective, Z2-preprocessing can still suffer rom poor conditioning because of persisting lack of liagonal dominance and large differences in the mag-itude of the added elements. The problem can some-imes be alleviated by the *23* option, which precedes he linear combination with normalization with re-spect to the largest element in the corresponding row. Addition of the normalized elements leads to an im-roved PCG performance because the relative magni-ude of the elements and the corresponding roundoff rror can be reduced. The *23* option is more compu-tationally intensive than 22. The 22 and Z3 prepro-essors can easily handle up to $0.75N$ zero diagonal lements.

I'he 24 pre-processing option is somewhat more omputationally intensive than *23*. It creates unit main-diagonal submatrices through multiplication by the inverse matrix $\mathbf{A}_M^{-1}$, computed using the method of determinants. The PCG performance improvement delivered by Z4 can be affected by roundoff errors; un-der favorable conditions, it matches those of Z2 and Z3.

THE O-PREPROCESSORS

The O-preprocessors are applied to matrices with no zero entries on the main diagonal and aim to improve the **PCG** solver performance by improving the matrix conditioning. Four such preprocessors are available in T2SOLV. The first three options, 0 1 through *0 3*, are in essence steps in the replacement of the $\mathbf{A}_M$ submatrix by the unit matrix through a central pivot-ing process, and involve increasing levels of compu-tational effort.

The 0 1 option eliminates the lower half of the main-diagonal submatrix, and thus removes NEQ-1 subdi-agonals from the global matrix. This reduces the computational effort by reducing the number of non-zero matrix entries and can improve the PCG perfor-mance. Execution times are burdened by the addi-tional work for the elimination of the lower half of the matrix, but usually this is overcome by the sav-ings in the PCG computations.

In the 0 2 option, in addition to 0 1 the upper half of the main-diagonal submatrix is eliminated, resulting in a diagonal submatrix and eliminating an additional NEQ-1 superdiagonals from the global matrix. Compared to the original, the O2-preprocessed matrix is significantly sparser and better-conditioned and the performance of the PCG solvers can be enhanced. The increased computational effort for the 0 2 prepro-cessing is usually compensated by the reduction in the PCG iterations.

The *0 3* options involves normalization of the 0 2 matrix, resulting in a unity main diagonal. *0 3* does not further increase matrix sparsity, but may improve the matrix conditioning. Finally, the 0 4 option is identical to the Z4 option discussed previously.

TEST PROBLEMS

The solvers were tested in four test problems. Performance results are presented in Figures 1 through *3* and Tables 1 through 4.

Test Problem 1

Test problem 1 involves a study of non-isothermal flow in a "two-water" system. Such systems are known to be the most challenging for the solvers in TOUGH2, as they routinely create matrices with $0.67N$ zeros on the main diagonal. The PCG rou-tines in T2CG1 have in the past been unable to solve even the smallest of this class of problems. The problem discussed here involves injection of "water 2" at a temperature of *30* ^{o}C into a geothermal reser-voir of "water 1" at 280 ^{o}C. The EOS1 module is used. The *3-D* domain consists of $9\times8\times5 = 360$ grid-blocks in (x,y,z), with NK = 2 and NEQ = *3*, result-ing in a total of N = 1080 equations.

The fundamental weakness of MA28, i.e. its large (especially for *3-D* problems) and not well defined memory requirement, was obvious in the problem. Despite memory allocation which sufficed for the LUBAND solution of *3-D* problems 15 times larger, MA28 could not complete the LU decomposition due to insufficient memory.

Table 1 and Figure 1 show that DLUSTB with and without the Z-preprocessors has the best performance. It is the fastest and requires the least number of PCG iterations to convergence. DLUSTB seems to be the only solver that can proceed without Z-preprocessing. Note that the use of the Z-preprocessors makes possible the solution of a previously-intractable problem by all the PCG solvers in T2SOLV. The Z2 preprocessor seems to offer the best overall performance.

Test Problem 2

Test problem 2 involves a laboratory convection cell experiment. A porous medium consisting of glass beads fills the annular region between the two vertical concentric cylinders. Application of heat generates a thermal buoyancy force, giving rise to the development of convection cells. This problem has been discussed in detail by *Moridis and* ***Pruess*** [1992}. The EOSl module is used. The domain consists of $16\times26 = 416$ gridblocks in (r,z), with NK = 1 and NEQ = 2, resulting in a total of N = 832 equations.

Table **2** and Figures 2 and 3 show the performance of the various solvers in Problem **2,** which does not pose any significant challenges to the T2SOLV routines. DLUSTB is the fastest routine and requires the least number of iterations to convergence.

In this 2D problem LUBAND appears as a competitive alternative. The effect of the 0-preprocessors vary. With 01, it is pronounced in terms of PCG iterations and execution times in DSLUBC and DLUSTB, but seems to be limited in DSLUCS and DSLUGM. The evolution of residuals of DSLUCS and DSLUGM in the first Newtonian iteration of the first timestep is identical with and without 0 1 preprocessing (Figures 2 and 3), while the DSLUCS execution time with 0 1 increases. Conversely, the use of the 0 2 and 0 3 preprocessors seems to offer the greatest improvement in the performance of DSLUCS and DSLUGM.

Test Problem 3

Test problem **3** examines fluid and mass flow in a large three-dimensional model of a geothermal reservoir. The basic computational grid is composed of $15\times15\times20 = 4500$ grid blocks in (x,y,z). Cold water is injected through **4** wells, while hot water is withdrawn from 5 wells. EOSl is used with NK = 1, NEQ = 2, resulting in a total of $N = 9000$ equations.

This is a relatively large but well-behaved problem, the size of which precluded the use of a direct solver. The use of D4 allowed a direct solution by LUBAND, which is competitive with the PCG solutions. D4 with DLUSTB had a performance on a par with DLUSTB, the fastest PCG solver. In light of the overhaed needed to set up the D4 system, this result is very encouraging.

DLUSTB demonstrated its superiority by being the fastest and requiring the least number of PCG iterations to convergence. DSLUGM seems to be an inappropriate method for this type of problem. As expected, the benefits of 0-preprocessing in this well-behaved system are not evident in the execution times, although the PCG iterations are often reduced. It is noteworthy, however, that despite the increased computational load, the execution times for the O-preprocessed solutions are practically identical to those without any preprocessing.

Test Problem 4

Test problem 4 describes a variation of the Thermal Enhanced Vapor Extraction System process, which is designed to extract solvents and chemicals contained in the Chemical Waste Landfill at Sandia National Laboratories. No NAPL is present in this system. In this process the ground is electrically heated, and boreholes at the center of the heated zone are maintained at a vacuum to draw air and vaporized contaminants into the borehole and to a subsequent treatment facility. The 3-D grid consists of 1300 gridblocks. EOS3 is used (NK = 2, NEQ = 3), and N = 3900 equations are solved. Additional information can be found in *Moridis and Pruess* [1995].

Without any preprocessing, the performance of DLUSTB in this rather well-behaved problem is practically identical to that of DSLUCS. These two are the fastest solvers, but DLUSTB requires the least number of PCG iterations. D4 with LUBAND appears as a competitive alternative. DLUSTB is the most responsive to 0 1 preprocessing, which results in the fastest solution with the least number of PCG iterations.

CONCLUSIONS

The following conclusions can be drawn:

(1) Without any matrix preprocessing, DLUSTB is shown to be a fast and efficient solver which outperforms the other PCG routines. It is the fastest and the most robust in T2SOLV and is shown to be practically free of stagnation, oscillation, and divergence problems.

(2) The use of the Z-preprocessors makes possible the solution of problems which were previously intractable to all the PCG solvers. The combination of the Z-preprocessors with the BiCGSTAB routine gives the best performance in such problems.

(3) In problems which are known to confound the other PCG solvers, DLUSTB converges smoothly but slowly to a solution without invoking the matrix-preprocessing facility.

(4) The 0-preprocessors are shown to improve the robustness and decrease the number of iterations to convergence, but their effect depends on the **PCG** solver in T2SOLV. DLUSTB appears **to** be the solver most consistently responsive to the 0-preprocessors. In well-behaved problems the effect of the O-preprocessors on the execution speed is not significant.
(5) LUBAND is shown to be consistently faster and more reliable than MA28, and can solve much larger problems.

(6) The gains in execution speed when the D4 scheme is used in regular grids are shown to be significant (especially compared to the direct solver). D4 with direct matrix solution seems to be competitive (in speed) to the **PCG** solvers in medium-sized problems, although this advantage is expected to disappear in large problems due to memory limitations.

Table 1. Solver Performance in Problem 1 (Macintosh PowerPC 9500/132)

SOLVER	PP	Δts	NI	Imx	Imn	IT	ET
MA28	Fails - insufficient memory						
LUBAND	-	8	25	-	-	-	63.9
DSLUBC	-	Fails					
	Z1	Fails					
	Z2	8	29	109	5	830	29.0
	Z3	8	28	109	7	877	29.6
DSLUCS	-	Fails					
	Z1	Fails					
	Z2	12	41	109	4	1002	38.6
	Z3	12	46	109	4	1253	46.5
DSLUGM	-	Fails					
	Z1	Fails					
	Z2	8	30	51	7	531	21.3
	Z3	8	28	27	2	496	19.3
DLUSTB	-	Fails					
	Z1	15	94	109	7	642	93.2
	Z2	8	25	20	4	289	16.9
	Z3	8	25	27	4	288	17.0

PP: Preprocessors
Ats: Number of timesteps
NI: Newtonian iterations
Imx: Maximum number of **PCG** iterations
Imn: Minimum number of **PCG** iterations
IT: Total **PCG** iterations
ET: Execution time (sec)

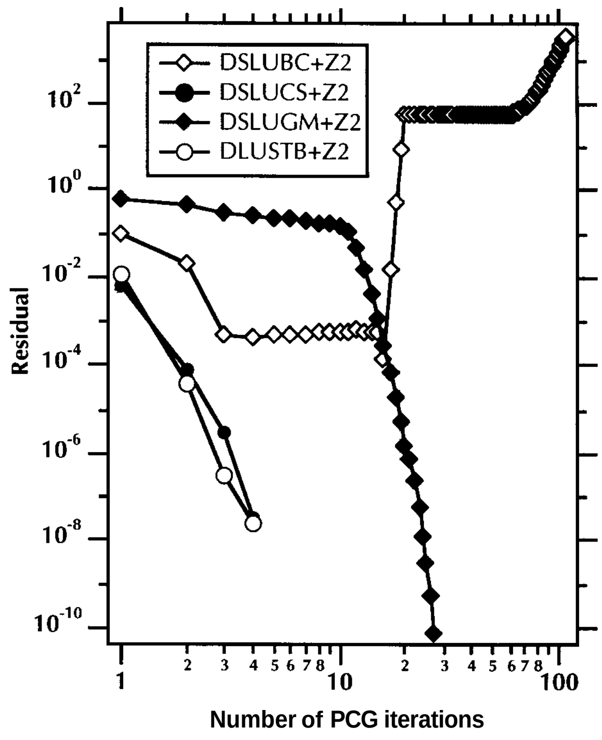

Fig. 1. PCG solvers with Z2 preprocessing in Test Problem I (1st NI of the 1st At).

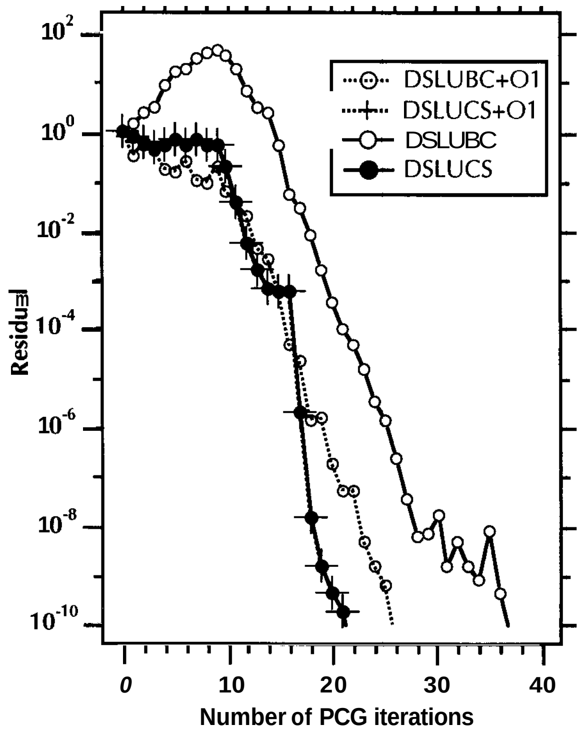

Fig. 2. DSLUBC and DSLUCS performance with and without 01 preprocessing in Test Problem 2 (1st NI of the 1st At).

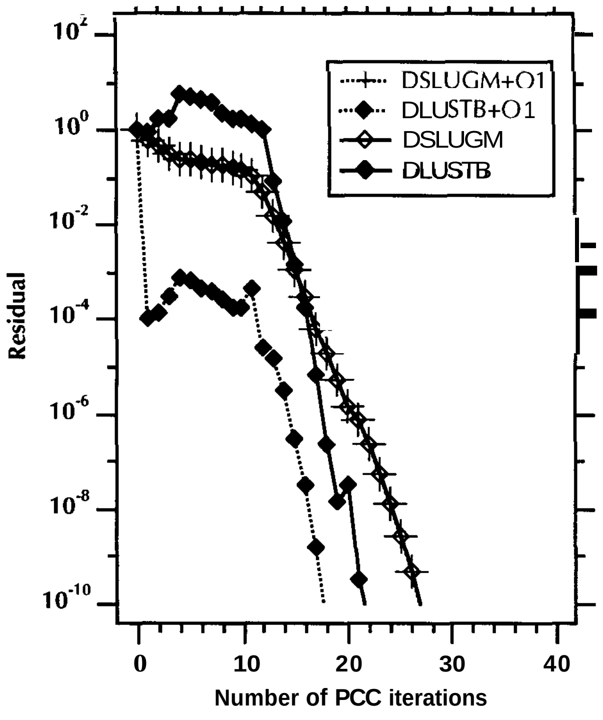

Fig. 3. ***DSLUGM*** *and DLUSTB performance with and without 0 1 preprocessing in Test Problem 2 (1st* ***NI*** *of the 1st* ***At).***

SOLVER	PP	Δt	NI	lmx	lmn	IT	ET
MA28	-	26	108	-	-	-	78.1
LUBAND	-	27	111	-	-	-	55.8
DSLUBC	-	28	119	45	27	3275	56.2
	O1	26	105	41	23	2798	49.1
	O2	28	110	42	23	2868	50.9
	O3	26	106	53	21	2867	50.4
DSLUCS	-	26	112	37	18	2437	49.7
	O1	29	131	33	18	2851	59.0
	01	26	98	41	1	1685	38.9
	02	26	102	39	1	1684	40.6
	03	26	98	30	14	1728	39.7

SOLVER	PP	Δt	NI	lmx	lmn	IT	ET
MA28	Insufficient Memory						
LUBAND	Insufficient Memory						
D4t LUBAND	-	10	46	-		-	786
D4+ DLUSTB	-	10	46	57	43	1736	426
DSLUBC	-	10	46	106	63	2623	579
	01	10	46	75	57	2477	565
	O2	10	46	75	52	2475	563
DSLUCS	-	10	46	95	50	2051	488
	01	10	46	98	50	2034	485
	O2	10	46	94	50	2033	487
)SLUGM	-	10	46	930	95	14842	2087
	01	10	46	930	95	14994	2178
	02	10	46	930	95	15113	2189
DLUSTB	-	10	46	58	41	1736	423
	01	10	46	60	37	1695	421
	O2	10	46	58	37	1719	429

Table 4. Solver Performance in Problem 4 (DEC AlphaStation 200/233)

SOLVER	PP	Δt	NI	lmx	lmn	IT	ET
MA28	Insufficient Memory						
LUBAND	-	50	249	-	-	-	1227
D4 + LUBAND	-	50	249	-	-	-	678
DSLUBC	-	50	249	29	7	4756	712
	O1	50	250	28	7	4712	701
DSLUCS	-	50	249	29	4	3681	639
	O1	50	249	26	4	3669	649
DSLUGM	-	50	249	27	7	3986	506
	O1	50	249	26	7	3984	508
DLUSTB	-	50	249	15	4	2544	509
	01	50	249	15	4	2232	484

ACKNOWLEDGMENT

This work was supported by the Assistant Secretary for Energy Efficiency and Renewable Energy, Office of Geothermal technologies, of the U.S. Department of Energy, under contract No. DE-AC03-76SF00098. Drs. Curt Oldenburg and Stefan Finsterle are thanked for their helpful review comments.

REFERENCES

Aziz, **K.** and A. Settari (1979), Petroleum Reservoir Simulation, Elsevier, London and New York.

Duff, I.S. (1977) MA28 - A set of Fortran Subroutines for Sparse Unsymmetric Linear Equations, AERE Harwell Report R 8730.

Fletcher, R. (1976), Conjugate gradient methods for indefinite systems. Numerical Analysis, Lecture Notes in Mathematics **506,** Springer-Verlag, New York.

LAPACK (1993), Univ. of Tennessee, Univ. of California at Berkeley, NAG ltd., Courant Institute, Argonne National Lab., and Rice University, Version 1.1.

Moridis, G.J. and K. Pruess (1992), TOUGH simulations of Updegraffs set of fluid and heat flow problem, Lawrence Berkeley Laboratory report LBL-32611, Berkeley, CA.

Moridis, G.J. and K. Pruess (1995), T2CG1: A package of preconditioned conjugate gradient solvers for the TOUGH2 family of codes, Lawrence Berkeley Laboratory report LBL-36235, Berkeley, CA.

Price, H. **S.** and K. H. Coats (1974), Direct methods in reservoir simulation, Trans. SPE of AIME (SPEJ), **257,** 295-308.

Pruess, K (1991), TOUGH2 - A general-purpose numerical simulator for multiphase fluid and heat flow, Lawrence Berkeley Laboratory report LBL-29400, Berkeley, CA.

Sleijpen, G.L.G. and D. Fokkema (1993), BiCGSTAB(l) for linear equations involving unsymmetric matrices with complex spectrum, Electronic Transactions on Numerical Analysis, **1,** 11-32.

Sonneveld, P. (1989), CGS, A fast Lanczos-type solver for nonsymmetric linear systems. SIAM J. Sci. Stat. Comput., **10**(1), 36-52.

van der Vorst, H.A. (1992), Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG in the presence of rounding errors, SIAM J. Sci. Statist. Comput., **13,** 631-644.