LBL-35192
UC-250

Lawrence Berkeley Laboratory

UNIVERSITY OF CALIFORNIA

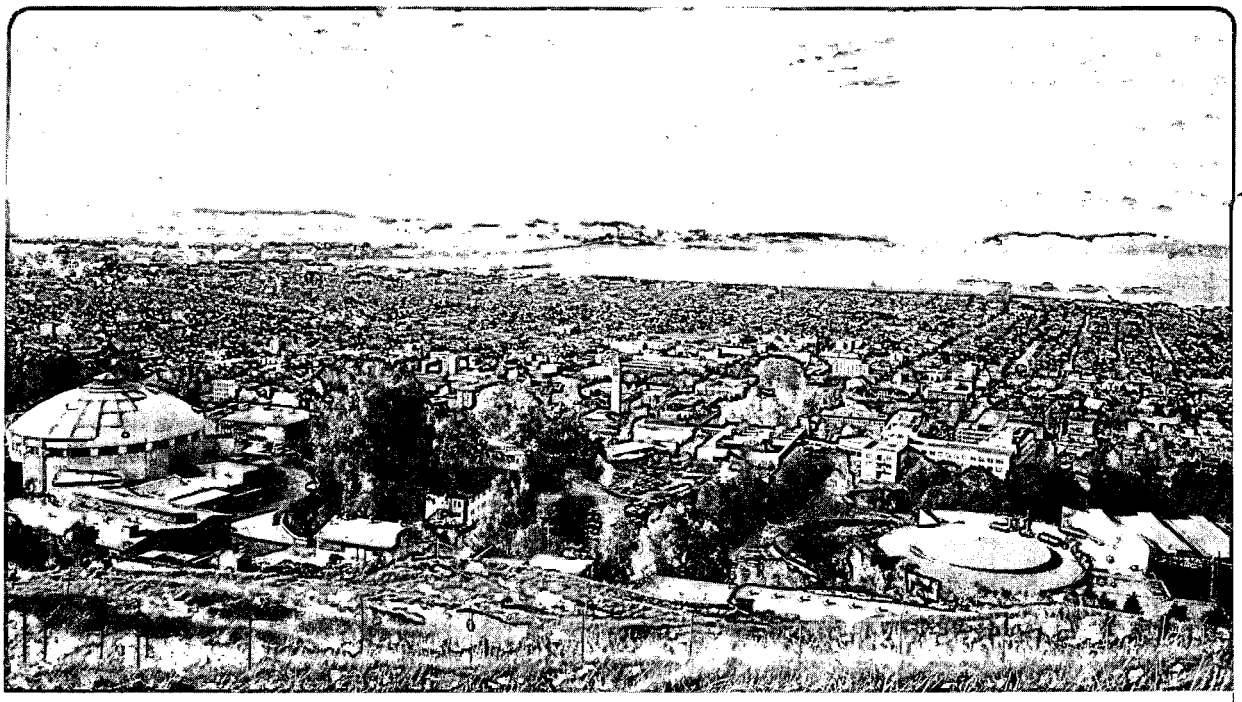
EARTH SCIENCES DIVISION

Presented at the Nineteenth Workshop on
Geothermal Reservoir Engineering at
Stanford, California, January 18–20, 1994,
and to be published in the Proceedings

Large-Scale Three-Dimensional Geothermal Reservoir Simulation on PCs

E. Atúnez, G. Moridis, and K. Pruess

January 1994



LOAN COPY
Circulates
for 4 weeks
Bldg. 50 Library.
Copy 2
LBL-35192

Prepared for the U.S. Department of Energy under Contract Number DE-AC03-76SF00098

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

LBL-35192
UC-250

Large-Scale Three-Dimensional Geothermal Reservoir Simulation on PCs

E. Antúnez, G. Moridis, and K. Pruess

Earth Sciences Division
Lawrence Berkeley Laboratory
University of California
Berkeley, California 94720

January 1994

This work was supported by the Assistant Secretary for Energy Efficiency and Renewable Energy, Geothermal Division, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

Neneteenth Workshop on Geothermal Reservoir Engineering
Stanford University, Stanford, CA, January 18-20, 1994

Large-Scale Three-Dimensional Geothermal Reservoir Simulation on PCs

Emilio Antúnez, George Moridis, and Karsten Pruess

Earth Sciences Division, Lawrence Berkeley Laboratory
University of California, Berkeley, CA 94720

ABSTRACT

TOUGH2, Lawrence Berkeley Laboratory's general purpose simulator for mass and heat flow and transport was enhanced with the addition of a set of preconditioned conjugate gradient solvers and ported to a PC. The code was applied to a number of large 3-D geothermal reservoir problems with up to 10,000 grid blocks. Four test problems were investigated. The first two involved a single-phase liquid system, and a two-phase system with regular Cartesian grids. The last two involved a two-phase field problem with irregular gridding with production from and injection into a single porosity reservoir, and a fractured reservoir.

The code modifications to TOUGH2 and its setup in the PC environment are described. Algorithms suitable for solving large matrices that are generally non-symmetric and non-positive definite are reviewed. Computational work per time step and CPU time requirements are reported as function of problem size.

The excessive execution time and storage requirements of the direct solver in TOUGH2 limits the size of manageable 3-D reservoir problems to a few hundred grid blocks. The conjugate gradient solvers significantly reduced the execution time and storage requirements making possible the execution of considerably larger problems (10,000+ grid blocks). It is concluded that the current PCs provide an economical platform for running large-scale geothermal field simulations that just a few years ago could only be executed on mainframe computers.

INTRODUCTION

The introduction of computers have allowed practicing engineers and scientists to go beyond oversimplified analytical solutions (that certainly have their merits) to approximated semi-analytical or fully numerical solution of large and complex problems. Nevertheless, programmers and modelers very often find themselves saturating the capabilities of their hardware, demanding more memory, storage and processing speed. At the same time, software developers are continuously trying to exploit the capabilities of the current hardware, writing more efficient computer codes.

Prior to the mid '80s, the cost of more powerful machines used to grow exponentially as more memory, storage and processing speed were required. Fortunately, that trend has eased up as computer components started being mass-produced (lowering production cost) to satisfy the strongly emerging workstation and personal computer (PC) market. As workstations and PCs became more capable, it became more difficult to draw the line between them.

In geothermal engineering, computers have facilitated the analysis of processes that range from the simple movement of fluids through porous media (in which mass and energy balances have to be simultaneously accounted for) to the more complex problems in which heat pipes, non-condensable gases, salinity and chemical processes have to be included in the analysis.

Considering that current PCs have the same or more computational power than mainframes and minicomputers of a few years ago, it is not surprising that software that was developed for mainframes and minicomputers had started migrating towards the more cost-effective PC platforms. This has been the case of Lawrence Berkeley Laboratory's general purpose reservoir simulator TOUGH2 [Pruess, 1991].

TOUGH2 is a numerical simulation program for non-isothermal flows of multicomponent, multiphase fluids in porous and fracture media. This code has been widely applied in geothermal reservoir engineering, nuclear waste disposal, environmental restoration, and unsaturated groundwater hydrology.

TOUGH2 with its standard direct matrix solver and a package of three different preconditioned conjugate gradient (CG) solvers[1], were ported to a 486-66 MHz PC. These memory efficient and fast CG algorithms are analyzed and compared with the direct matrix solution.

SOLVERS PACKAGE

VCG [Moridis, 1994], a package of preconditioned conjugate gradient solvers, has been added to TOUGH2 to complement its direct solver and

[1] Conjugate Gradient solvers are algorithms for the iterative solution of large sets of linear equations.

significantly increase the size of problems tractable on personal computers. This package significantly decreases the execution time and memory requirements, and thus makes possible the simulation of much larger (in terms of number of equations) systems.

VCG was derived from the Sparse Linear Algebra Package (SLAP) Version 2.0 [Seager, 1988] developed for the solution of large sparse linear $N \times N$ systems

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$$

where N is the order of the A matrix. SLAP is a collection of various conjugate gradient solvers, with two matrix preconditioning options: diagonal scaling (DS) and modified incomplete LU factorization (ILU).

In TOUGH2 the matrix A is a Jacobian with certain consistent characteristics. In systems with regular geometry, A has a known block structure with well defined sparsity patterns. In general, A matrices arising from TOUGH2 simulations (and geothermal reservoir engineering problems in particular) are non-symmetric with weak (or no) diagonal dominance. Although A can be positive definite in regular systems with homogeneous property distributions, in realistic heterogeneous large-scale simulations it usually is not, and ill-conditioning is expected . A being a Jacobian, its elements in a single row often vary by several orders of magnitude.

In TOUGH2 simulations of flow and transport through fractured media, the implementation of the "multiple interactive continua" (MINC) concept [Pruess, 1983] results in a large number of zeroes on the main diagonal of A, making pivoting impossible and resulting in very ill-conditioned matrices. It is evident that TOUGH2 simulations create matrices which are among the most challenging, with all the features that cause most iterative techniques to fail. This explains the heavy reliance of TOUGH2 on direct solvers in the past.

Extensive testing of the SLAP package in a variety of flow and transport problems identified the most promising conjugate gradient methods. The properties of the A matrix essentially precluded the use of DS preconditioning, a fact which was confirmed in the process of testing SLAP. Without exception, ILU preconditioning was far more effective and often the only possible option. Of the 15 methods available in SLAP, three were identified as the ones with the most potential. In terms of increasing robustness, these were the Bi-Conjugate Gradient (BCG) method, the Lanczos-type Bi-Conjugate Gradient Squared (BCGS) method, and the Generalized Minimum Residual (GMRES) method. In terms of the SLAP terminology, these methods corresponded to the subroutines DSLUBC, DSLUCS, and DSLUGM, respectively.

Fletcher [1976] proposed BCG for the solution of linear, but not necessarily positive definite or symmetric systems. Theoretical analysis of the properties of BCG indicates that as long as the recurrences in the method do not break down, it must converge in $m < N$ iterations. Although there is no guarantee of reduction of the quadratic functionals (i.e. that the recurrences will not break down or become unstable), in practice this is rare. If a good preconditioner is used, BCG is an effective method [Seager, 1988].

The BCGS [Sonneveld, 1989] method is related to the BCG, but it does not involve adjoint matrix-vector multiplications, requires half the computational work, and the expected convergence rate is about twice that of BCG. For a $N \times N$ problem, BCGS was theoretically shown to converge in at most N steps. Seager [1988] reports that when BCG diverges, BCGS diverges twice as fast, and when BCG stagnates, BCGS is more likely to diverge. He also suggested using BCGS after first successfully applying BCG. However, in most TOUGH2 applications, this behavior was not observed. In addition, a non-monotonic reduction in the error of BCGS, with many local peaks (sometimes significant) in the convergence performance was seen. These local peaks are also observed in BCG, but they are usually smaller in magnitude.

The GMRES method of Saad and Schultz [1986] is a Lanczos-type extension of conjugate gradients for general non-symmetric systems which is expected to converge in $m < N$ steps for any non-singular matrix if truncation errors are not considered. It generates an orthonormal basis from the Krylov subspace

$$\mathbf{K}(m) = span\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \mathbf{A}^2\mathbf{r}_0, \mathbf{A}^3\mathbf{r}_0, \ldots\ldots, \mathbf{A}^{m-1}\mathbf{r}_0\},$$

where $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{r}_0$ is the initial residual. Since storage requirements increase with m and the number of multiplications with m^2, m has to be much smaller than N. If the convergence criterion is not met within m iterations, the iteration can be restarted using as an initial value of x the one obtained at the m-th iteration of the previous cycle. The GMRES used in the VCG package employs this approach. It was found that a m=20 to 30 is needed in most TOUGH2 simulations. Unsatisfactory performance is generally obtained for m<15, and it is usually pointless to use m>35 (since this probably indicates that GMRES may not be a good method for that particular problem). A unique feature of GMRES is that the residual norm is minimized at every iteration, i.e., the decrease in the error is monotonic.

In the VCG package the nomenclature of SLAP was maintained, but the structure and content of the subroutines was substantially modified . Most subroutines used in the SLAP structure were eliminated and large segments of the code were reprogrammed to take advantage of the well-defined sparsity pattern of matrix A. This resulted in a compact code optimized for

TOUGH2, which is substantially faster and lacks the modular structure of SLAP.

The standard TOUGH2 solver MA28 [Duff, 1977] uses a matrix storage scheme that is identical to the SLAP Triad Matrix Storage Format, remained unaltered in VCG. The ILU preconditioner was kept for use in simulations with irregular geometry. However, for simulations with regular geometry, using the known structure of the A matrix (determined by the integrated finite difference formulation of TOUGH2), an optimized Incomplete Block LU factorization (IBLU) preconditioner [Sonneveld, 1989] was developed. The IBLU preconditioner was based on an approach proposed by Meijerink [1983], and significantly sped up the convergence rate of the three methods compared to the ILU. Moreover, Sonneveld's [1989] observation that the IBLU factorization has the additional advantage of being less sensitive to special directions in the problem (e.g., the advection direction in the advection-diffusion equation, layering, etc.) was confirmed.

Storage requirements in VCG remained the same as in SLAP [Seager, 1988]. BCG and BCGS have the same requirements, while GMRES needs several times more memory. In terms of speed, our previous experience in a large number of TOUGH2 simulations indicates that BCGS is the fastest by a substantial margin, followed by BCG. GMRES is the slowest, but also the most robust, and managed to solve efficiently some of the most demanding problems. Contrary to Seager's [1988] observations, BCGS is the second most robust. Although one or two methods in the VCG package occasionally fails to converge successfully, no case where all three methods are unsuccessful in a TOUGH2 simulation has been encountered yet.

In the case of fractured systems (using the MINC approach) the large number of zeroes in the main diagonal resulted in a very poor convergence of the solution in all three CG solvers in VCG. The problems was alleviated by exchanging the zeroes in the main diagonal by a small number (10^{-30}). This approach resulted in no detectable effects on the accuracy of the solution and considerable improvement in processing speed.

PCs SETUP AND REQUIREMENTS

To run the standard TOUGH2 or the version with the solvers package on any 32 bit machine (i.e., machines with 386, 486 or higher processor), it is necessary to modify the code to declare all variables REAL*8 (or DOUBLE PRECISION), and to comply with the FORTRAN77 ANSI X-3.9-1978 standards, also all floating point constants must be converted from E##.# to D##.# format. The processing speed of the code will depend on the machine being used. The maximum size of computational grid will depend on the amount of extended memory (XMS) available on the machine[2]. A minimum configuration to run TOUGH2/PC would be a 386 PC equipped with 4 MB of RAM, an 80 MB hard drive and an optional (but recommended) numerical coprocessor (387). This configuration will allow to perform 3-D[3] simulations with grids of approximately 1,000 elements and 3,000 connections when using the VCG solvers; or approximately 500 elements and 1,500 connections using the standard version of TOUGH2/PC with the direct matrix solver (MA28).

The code testing presented here was conducted on a 486-DX2-66 MHz PC equipped with 32 MB of RAM and a 250 MB hard drive. This study was limited to grids with a maximum of 10,000 elements; However, this configuration can handle models with larger number of elements.

The code with a maximum grid size of 10,250 elements and 30,750 connections was compiled and linked using Version 5.0 of the Lahey Fortran Compiler for 32 bit machines. The resulting executable version required approximately 26 MB of disk space and the same amount of XMS RAM to run. Memory requirement for TOUGH2/PC with the CG solvers is approximately linear, therefore, the amount of RAM required by a 3-D grid can be interpolated or extrapolated using the memory requirements for the 1,000 and 10,000 element grids. Hard disk space will depend on the amount of printout generated by the simulation run, which is a parameter controllable by the user. The largest model (10,000 elements), with printout at only the final time step, produced files requiring a total of approximately 6 MB of disk space.

SOLVER'S TESTING PROCEDURE

The testing of the different solvers was conducted using two reservoir models, one with a regular Cartesian grid and the second with irregular gridding. Two cases were analyzed for each of the models, for a total of four simulation cases:

(1) a regular Cartesian grid with a single-phase liquid system,

(2) a regular Cartesian grid with a two-phase system,

(3) a two-phase field problem with irregular gridding with production from and injection into a single porosity reservoir, and

(4) a two-phase field problem with irregular gridding with production from and injection into a fractured reservoir.

---

[2] Extended memory (XMS) is additional memory beyond the first MByte (MB) of random access memory (RAM). The first MB of RAM is usually occupied by the Disk Operating System (DOS), the 640 KB of DOS conventional memory and the Terminate and Stay Resident applications (TSR).

[3] 3-D simulations are the most memory demanding. 1-D and 2-D problems result in arrays of smaller size.

The testing was based on the time it took each algorithm to complete a Newtonian iteration, which consists of:

(a) Recalculating the terms of the Jacobian matrix that results from applying the mass and energy conservation equations at each grid element,

(b) Preconditioning (except for the direct solver MA28) and solving the matrix using VCG. The matrix solution provides the changes of all primary variables (pressure, temperature) for single-phase elements or (pressure, vapor saturation) for elements in two-phases, and

(c) recalculating all the secondary variables (density, internal energy, viscosity, relative permeabilities, capillary pressure, phase saturation, mass fractions of each component) for all the elements of the grid.

As mentioned before, each solver routine have internal convergence (closure criterion) that involve several internal iterations of the CG algorithm[4] (CG iterations) to a maximum specified by the user (usually 10% of the number of elements times the number of equations per element). A closure criterion of 10^{-6} was used in all three CG solvers.

REGULAR CARTESIAN GRID MODELS

Five simulation models with different discretization were constructed, as shown in Table 1. The simulation models have an areal extent of 5 x 4 km (20 km^2) and a thickness of 1000 m, divided in ten layers of 100 m each (Fig. 1). All have a well producing at a constant rate of 30 kg/s in the sixth layer, an injection well operating at a rate of 30 kg/s in the third layer, and a 30 MW heat source at the bottom layer (layer 10). The wells are located at the node of the element closer to the points (500, 500, 550) for the producer and (4500, 3500, 250) for the injector. The heat source is distributed among the required elements to cover an area of 4×10^5 m^2 (1000 m in x and 400 m in y) at the center of bottom layer (Fig. 1). All of this models were used to perform simulations for single-phase and two-phase conditions.

For the single-phase cases the initial conditions are 40 MPa and 280°C in all blocks; for the two-phase cases, 10 MPa and S_g=0.20 in all blocks. No-flow boundaries to mass and heat are employed. Relative permeabilities correspond to Corey's curves with residual saturations of liquid and steam equal to 0.3 and 0.05, respectively. Capillary pressures are neglected. Other relevant parameters are given in Table 2.

IRREGULAR GRID - CERRO PRIETO MODEL

The Cerro Prieto geothermal field developed by the Comisión Federal de Electricidad (CFE), is located approximately 35 km south of Mexicali, Baja California, México. Since the beginning of the exploitation of Cerro Prieto in 1973, one of the most important operational problems that CFE has had to face was the handling of the waste brine [Hiriart and Gutiérrez Puente, 1992]. Up to date most of the brine is sent to evaporation ponds that presently cover an area of 18.6 km^2, Figure 2. An infiltration area west of the ponds is used during the winter, when the evaporation rate is significantly lower.

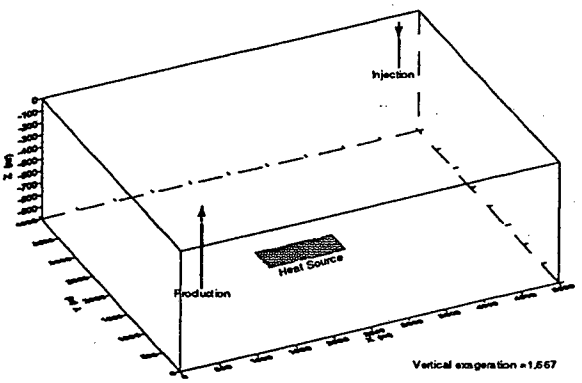


Fig. 1: Characteristics of the Cartesian grid models

Table 1: Discretization of the Cartesian grids

Grid No.	Elements in x	Elements in y	Elements in z	Total number of elements
1	5	10	10	500
2	10	10	10	1000
3	10	20	10	2000
4	25	20	10	5000
5	50	20	10	10000

Table 2: Reservoir parameters for the Cartesian models

rock density	2650	kg/m^3
porosity	3	%
saturated thermal conductivity	3.2	W/m °C
rock specific heat	1000	J/kg °C
permeability	200	md
initial steam saturation	0	%

Recently (1992-93), CFE started a series of cold brine (approximately at 20°C) injection tests, using brine from the evaporation ponds. The objective of these tests was to monitor the reservoir's response to the injection and to test

[4] Not to be confused with the Newtonian iterations which are external to the CG algorithm.

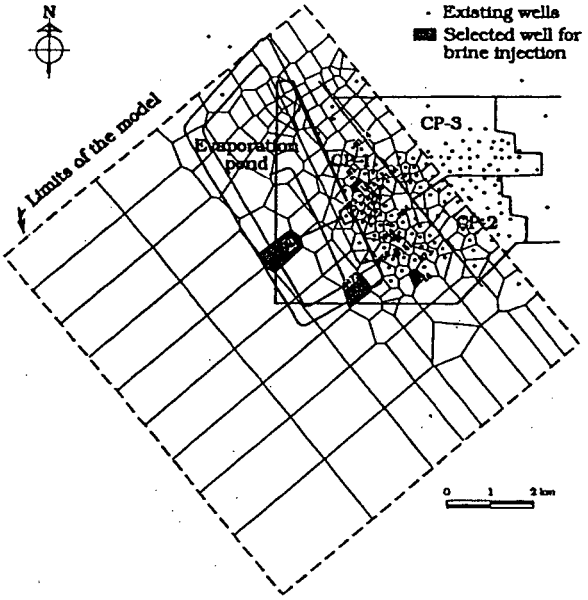


Fig. 2: Cerro Prieto model. Characteristics of the irregular computational grid.

the injectivity of different areas of CP1 in the western part of the field. CFE's final goal is to inject all the separated brine back into the system, to eliminate its surface disposal and, at the same time, provide pressure maintenance to the reservoir.

Under the DOE/CFE cooperative agreement on geothermal energy, a numerical model for CP1 was built, using data provided by CFE. The computational grid covering an area of 89 km^2, was defined based on the geological model of the field and the location and completion of the production and injection wells (Fig. 2).

In the vertical direction the model extends from the surface to 5,000 m depth, and is divided into six layers. All the layers have the same discretization and have 235 grid elements (Fig. 2), except layer five that has 47 additional blocks in the NE simulating the volume of the CP2, CP3 and CP4 areas. The numerical model has a total of 1457 elements and was developed as a single porosity model [Antúnez and Lippmann, 1992]. The model was calibrated with production and piezometric data, and was used to test several injection strategies.

For the Cerro Prieto model, the timing of the Newtonian iterations was conducted using the following scenario: Inject 3,500 t/h of 20°C brine evenly distributed between injection wells M-48, 101, 104, E-6, O-473 and M-6. Production wells will continue producing at a rate equal to that measured at the end of 1991 (for that year, the average field production was 5,459 t/h of steam and 6,394 t/h of separated brine). Injection well locations are shown in Figure 2. The reservoir parameters used on the Cerro Prieto model are given in Table 3.

Table 3: Reservoir parameters for the Cerro Prieto model

	Single-Porosity	Double-Porosity
Matrix properties		
density	2000-2800 kg/m^3	2000-2800 kg/m^3
porosity	5-20 %	15 %
saturated conductivity	0.5-1.3 W/m ^{o}C	0.5-1.3 W/m ^{o}C
specific heat	600-2200 J/kg ^{o}C	600-2200 J/kg ^{o}C
permeability	0.1-100 md	0.001-1 md
initial steam saturation	spatially variable	spatially variable
Fracture domain properties		
rock grain density		2000-2800 kg/m^3
porosity		1 %
rock specific heat		600-2200 J/kg ^{o}C
fracture spacing		50 m
permeability		0.1-6,000 md
initial steam saturation		spatially variable

DISCUSSION OF RESULTS

All models were run for 25 time steps. An average of 3 to 4 Newtonian iterations were required to reach convergence in each time step. An example of this procedure is presented in Fig. 3.

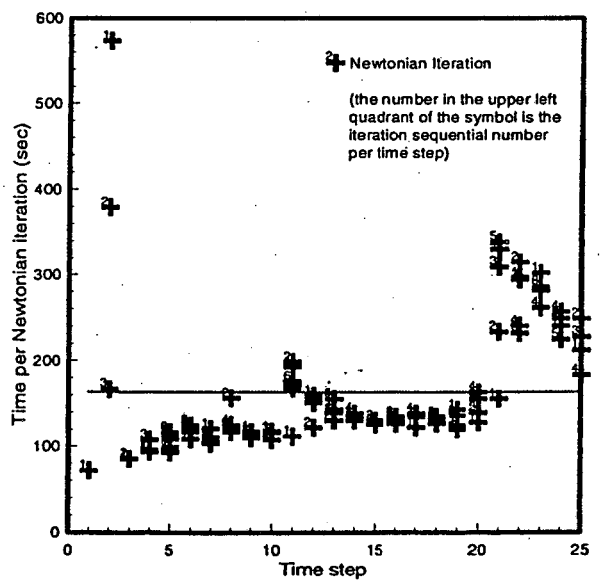


Fig. 3: Timing of Newtonian iterations for a 10,000 element Cartesian grid using the Lanczos-type Bi-Conjugate Gradient Squared solver (Two-phase).

This figure shows the number of Newtonian iteration per time step. Each Newtonian iteration for a given time step is identified with a number on the upper left quadrant of the symbol. The straight line corresponds to the arithmetic average of all iterations. To compare the performance of the different solvers, the average timing of all Newtonian iterations per run was plotted against the number of elements in the grid. Timing data of the various solvers are presented in Figs. 4 and 5.

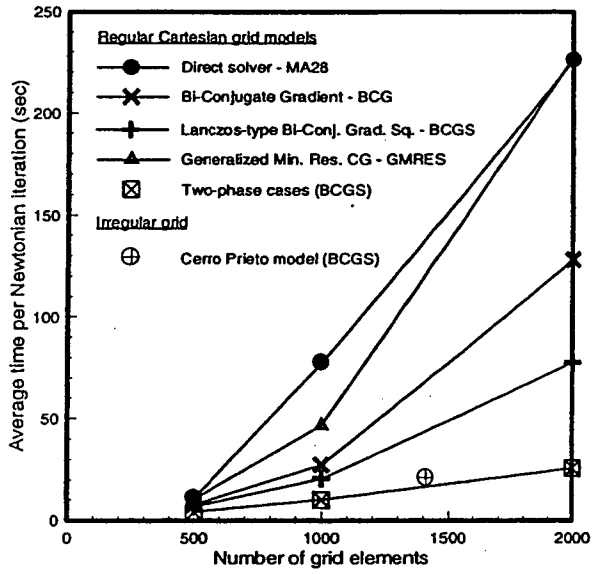


Fig. 4: Timing of Newtonian iterations for each of the analyzed solvers as function of problem size. (Medium size grids)

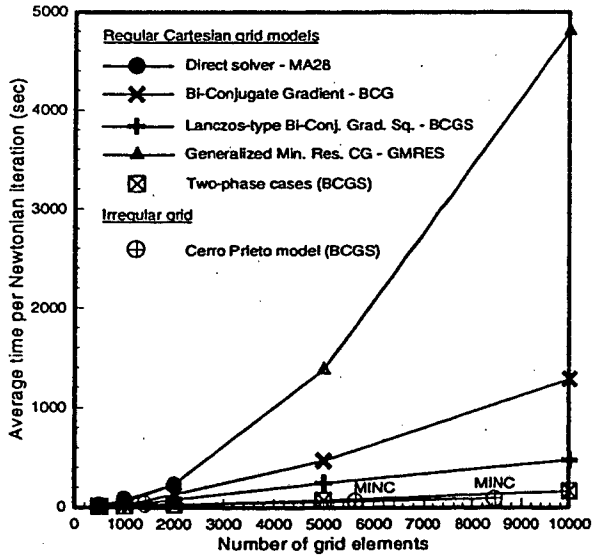


Fig. 5: Timing of Newtonian iterations for each of the analyzed matrix solvers as function of problem size.

The upper four curves on Figs. 4 and 5 correspond to the single-phase Cartesian models matrix solvers. The top curve is for the direct matrix solver MA28 [Duff, 1977]. which showed to be the slowest but most robust of all the tested solvers. The mayor disadvantage of this solver is that it is not optimized to handle sparse matrices, therefore, its operation requires considerable amount of RAM. The 2000 element 3-D grid required approximately 21 MB of RAM compared to 4 MB for the 500 element case. An additional problem of the MA28 solver is that the dimensioning of the arrays for 3-D problems is rather obscure and most of the time trial and error is required to accommodate grids larger than 500 elements.

The second curve from the top in Figs. 4 and 5 corresponds to the GMRES CG solver with *incomplete Block LU* (IBLU) factorization preconditioning of the matrix [Meijerink, 1983; Sonneveld, 1989]. This algorithm, is slower than the other two CG algorithms in the package. For the current problem settings it is as slow as MA28. Its main advantage, in geothermal problems, is that it needs significantly less memory than MA28, making possible 3-D simulations that the direct solver can not handle. If a solver is not specified by the user, this version of TOUGH2 with the CG package defaults to the GMRES solver. This solver has the advantage of monotonic decline in the residual error from one CG iteration to the next.

The third and fourth curves from the top in Figs. 4 and 5, correspond to the BCG, and to the BCGS, respectively. The BCG solver is the least robust of the three, but is faster than the GMRES solver. The BCGS solver was consistently the fastest of all solvers. From past experience it has been noticed that the decline of the residual error is not uniform and monotonic, and may exhibit strong oscillations. In previous tests with complex problems this method has not performed as well as the GMRES algorithm. However, in the cases tested here it showed a solid performance in all cases. In the remaining cases the BCGS solver was used. This solver was chosen on the basis of its performance efficiency.

The two-phase cases are presented in Figs. 4 and 5 by the curves with crossed out-squares. Execution times in these runs for the different grid sizes showed better performance than the similar single-phase cases.

The additional two test cases were conducted using the Cerro Prieto irregular grid model. Irregular grids may considerably increase the number of connections between contiguous elements; in this case, some elements have up to nine connections. The higher the number of connections, the denser the population of non-zero elements in the matrix. This specific case was not possible to solve it using the MA28 solver since some of the connection-related arrays could not handle the Cerro Prieto mesh. However, this posed no problem to the conjugate gradient solvers with ILU preconditioner.

The timing per Newtonian iteration for the single-porosity, two-phase Cerro Prieto model using the BCGS solver is shown in Figs. 4 and 5. It is interesting to observe that the timing for this model almost falls on top of the corresponding line for the regular Cartesian grid models in two-phases.

The Cerro Prieto model was also used to compare execution times of a double-porosity formulation based on the MINC method [Pruess, 1983]. This method subdivides the elements in concentric shells. The external shell represents the fracture and is fully connected to fractures of neighboring elements. The internal shells represent the rock matrix and are connected to the fracture by means of single linear connections. For computational purposes, each of these shells or subdivisions of an element, becomes a new element of the Jacobian matrix to be inverted by the solvers. The linearity of the connection between matrix shells and between matrix and fracture add a large number of non-zero elements to the Jacobian matrix. The effectiveness of the ILU preconditioning is evident as only a very small increment in time is required for its solution when compared to the single-porosity case, as shown on Fig. 5 by the points labeled MINC.

Table 4 presents a summary of the results of testing the different solvers. Case 1 and 2 correspond to the Cartesian models for single and two-phase conditions. Cases 3 and 4 are for the two-phase conditions using an irregular grid with single and double porosity. The reported total number of iterations are the sum of: a) the Newtonian iterations (external iterations); b) the repeated

Table 4: Timing of the test runs for TOUGH2/PC with the solvers package

Newtonian iteration tolerance = 1×10^{-5}
Closure in CG solvers = 1×10^{-6}

Case	Grid size	Solver	Number of iterations			Time (sec)					Observations
			Total[5]	Newtonian	Repeated due to convergence failure	per Newtonian iteration	Input	CPU	Total execution	Simulated	
1	500	MA28	96	71	0	10.84	4.07	791.31	795.38	7.1677E9	Standard vers.
		GMRES	108	82	1	10.35	3.63	870.73	874.36	4.5053E9	
		BCG	104	78	1	7.56	3.46	611.21	614.67	4.7101E9	
		BCGS	97	71	1	6.79	3.46	518.06	521.52	4.7101E9	
	1,000	MA28	96	71	0	77.57	9.12	5551.59	5560.71	19.660E9	Standard vers.
		GMRES	126	97	4	46.78	8.08	4593.14	4601.22	4.0957E9	
		BCG	103	75	3	27.45	7.74	2246.84	2254.58	6.9629E9	
		BCGS	98	70	3	20.60	7.97	1726.42	1734.39	4.3005E9	
	2,000	MA28	100	75	0	226.01	21.64	17041.46	17063.10	4.3005E9	Standard vers.
		GMRES	144	113	6	227.88	19.55	25878.42	25897.97	1.2285E9	
		BCG	112	82	5	127.92	19.06	11947.56	11966.62	1.3821E9	
		BCGS	99	70	4	77.68	18.95	6655.65	6674.60	1.9453E9	
	5,000	GMRES	101	68	8	1381.20	80.57	94437.19	94517.76	1.7250E8	
		BCG	106	75	6	471.38	79.20	48964.26	49043.46	5.5501E8	
		BCGS	102	74	3	244.97	80.91	31499.60	31580.51	7.1650E8	
	10,000	GMRES	57	39	3	4803.00	261.83	187876.22	188138.1	4.1110E7	15 time steps
		BCG	109	77	7	1287.75	268.80	163326.51	163595.3	2.1730E8	
		BCGS	97	64	8	472.05	269.85	106601.76	106871.6	3.3250E8	
2	500	BCGS	140	113	2	4.02	3.79	485.38	489.17	9.0900E7	
	1,000	"	122	97	0	10.00	7.85	1017.44	1025.29	9.0900E7	
	2,000	"	134	108	1	25.75	18.73	2910.39	2929.12	9.0900E7	
	5,000	"	132	107	0	69.95	82.77	7732.52	7815.29	3.8100E7	
	10,000	"	134	109	0	162.41	274.29	18203.14	18477.43	2.5300E7	
3	1,411	BCGS	145	117	3	21.20	33.51	2595.45	2628.96	4.8061E8	
4	5644	BCGS	194	154	15	55.56	114.85	10347.09	10461.94	4.1001E8	3 MINC shells
	8466	"	179	142	12	85.53	169.94	16808.09	16978.13	4.1003E8	5 MINC shells

[5] The total number of iterations includes one additional convergence iteration per prescribed time step, 25 in total. At each iteration convergence is checked and if convergence is satisfied a new time step is started.

external iterations due to convergence failure (after nine Newtonian iterations without reaching convergence, the incremental time used in the current time step is divided by five and the iteration procedure for that time step is repeated); and c) the convergence iterations (iterations that do not need to call the solver since convergence has been attained) one per prescribed time step. The average timing per Newtonian iteration only includes the completed Newtonian iterations; convergence iterations are not considered in this column. The CPU time corresponds to execution time for all iterations Newtonian and non-Newtonian, plus the time to write the output files.

Time comparisons for the different cases indicates that from the three CG tested, the BCGS solver showed the best performance for the runs conducted with the models used in this study, followed BCG and MA28 and GMRES. However, it is important to emphasize that iterative methods are problem specific. A solver that showed to be adequate for a given problem is not guaranteed that it will work with all problems. The GMRES solver showed to be the slowest of the three tested conjugate gradient solvers but in previous testing of some highly heterogeneous non-geothermal related problems this solver was the only one that could converged. Testing of the solvers with an specific problem is strongly recommended to define which is the best for the task

CONCLUSIONS

- Lawrence Berkeley Laboratory's general purpose simulator TOUGH2 together with a set of three preconditioned conjugate gradient solvers has been transported to PC platforms and successfully tested.
- The tested conjugate gradient solvers significantly reduced the execution time and storage requirements making possible the execution of considerably larger problems (10,000+ grid blocks) on PCs.
- The Lanczos-type Bi-Conjugate Gradient Squared demonstrated to be the fastest of all tested solvers. It is the best choice on the basis of its performance efficiency. It showed that its computation time and memory requirements increased with problem size only slightly faster than linear.
- Memory requirement for TOUGH2/PC with the conjugate gradient solvers is approximately linear, therefore, the amount of random access memory required by a grid can be easily interpolated or extrapolated.
- This study demonstrates that the combination of the analyzed preconditioned conjugate gradient solvers and the current PCs (386 and higher) are a feasible, economical and efficient combination to conduct large-scale three-dimensional geothermal reservoir simulations that just a few years ago could only be executed on mainframe computers.

ACKNOWLEDGMENTS

The authors want to express their gratitude to Marcelo Lippmann and Donald Mangold of LBL for reviewing this manuscript and for all their helpful suggestions. This work was supported by the Assistant Secretary for Energy Efficiency and Renewable Energy, Geothermal Division, of the U.S. Department of Energy under contract No. DE-AC03-76SF00098.

REFERENCES

Antúnez, E., and Lippmann, M. (1993) Numerical Study of the Effects of Brine Injection on the CP1 Production Area of Cerro Prieto. Earth Sciences Division Annual Report 1992, Lawrence Berkeley Laboratory report LBID-33000, pp. 97-99

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. pp. 73-89.

Hiriart, G.L. and Gutiérrez, H.P. (1992) An Update of Cerro Prieto Geothermal Field, Twenty Years of Commercial Power. Geothermal Resources Council Bull. Sept/Oct, pp. 289-294.

Meijerink, J.A. (1983) Iterative methods for the solution of linear equations based on incomplete block factorization of the matrix. Paper SPE 12262, Proc. 7th SPE Symposium on Reservoir Simulation, San Francisco, CA, November 15-18, pp. 297-304.

Moridis, G. (1994) Personal communication.

Pruess, K. (1983) GMINC- A mesh generator for flow simulations in fractured reservoirs, Lawrence Berkeley Laboratory report LBID-15227.

Pruess, K. (1991) *TOUGH2* - A General-Purpose Numerical Simulator for Multiphase Fluid and Heat Flow, Lawrence Berkeley Laboratory report LBL-29400.

Saad, Y. and Schultz, M.H. (1986) GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J. Sci. Stat. Comput., Vol. 7, No. 3, pp. 856-869.

Seager, M.K. (1988) A SLAP for the Masses. Paper presented at "Methods & Algorithms for PDE's on Advanced Processors", Austin, TX, Oct 17-18, 1988, Lawrence Livermore National Laboratory report UCRL-100195.

Sonneveld, P. (1989) CGS, A fast Lanczos-type solver for nonsymmetric linear systems. SIAM J. Sci. Stat. Comput., Vol. 10, No. 1, pp. 36-52.

LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
TECHNICAL INFORMATION DEPARTMENT
BERKELEY, CALIFORNIA 94720

AAT084
LBL Libraries