Efficient block preconditioners for elasticity problems

R. Beauwens
and
Y. Notay [1]

Service de Métrologie Nucléaire
Université Libre de Bruxelles (C.P. 165)
50, Av. F.D. Roosevelt, B-1050 Brussels, Belgium.
email : ynotay@ulb.ac.be

Report GANMN 97-04

November 1997

[1]Supported by the "Fonds National de la Recherche Scientifique", Chercheur qualifié.

1 Introduction

In this report, we make a synthetic comparison of the solvers for discrete linear elasticity problems that have been developed within the framework of the Workpackage 3 of the project HIPERGEOS (Copernicus Grant No CP 94-00820).

This comparison is made on the basis of the numerical results obtained for a set of benchmark problems that involved characteristic features of real-life applications met by the industrial partners.

The solvers are presented in Section 2, where we outline their main common points and differences. The benchmarks are briefly described in Section 3. Section 4 contains the numerical results, on which basis we develop our conclusions and final recommendations.

2 Solvers

Let

$$A\mathbf{u} = \mathbf{b} \tag{1}$$

be the linear system to solve. As a result of the collaborations between partners since the beginning of the project, the different solvers use similar underlying principles. They all consider a preconditioned iterative method in which the preconditioner is based on the 3×3 block structure of the stiffness matrix

$$A = \begin{pmatrix} A_{xx} & A_{xy} & A_{xz} \\ A_{yx} & A_{yy} & A_{yz} \\ A_{zz} & A_{zy} & A_{zz} \end{pmatrix} \tag{2}$$

induced by the separate ordering of the different components of the unknown displacements.

Further, the building block of the preconditioner is in each case an incomplete factorization of the block diagonal part

$$D = \begin{pmatrix} A_{xx} & & \\ & A_{yy} & \\ & & A_{zz} \end{pmatrix} , \tag{3}$$

previously reduced to an M-matrix by a diagonal compensation technique [1, 16] in cases the used discretization scheme does not directly lead to an M-matrix.

The different solvers are listed below with the peculiarities of the used numerical techniques.

PCG-1 (IGAS, Ostrava [3, 4, 6]) uses the Conjugate Gradient method with as preconditioner a pointwise incomplete factorization of D based on the techniques developed in [7].

CGLIB (CMD, Brussels [15, 16]) uses the same method as PCG-1, except that the incomplete factorization is based on the DRIC algorithm [13], which uses a dynamic relaxation rule to compute the diagonal pivots.

DRBILU (NM, Brussels [14]) uses also the Conjugate Gradient method, but the preconditioner is a *block* incomplete factorization of D, based on the DRBILU algorithm [8], which is a block extension of the DRIC method used in CGLIB.

D-MBILU (KUN, Nijmegen [11]) uses as building block a block incomplete factorization of D much similar to DRBILU [10]. However, this preconditioner is here used in *inner* Conjugate Gradient iterations to solve systems with D to a relative accuracy 10^{-3}, and this solution procedure is itself used as preconditioner for an *outer* iterative scheme to solve (1). For the latter, it is made use of the Generalized Conjugate Gradient Least Square method (GCGLS, with 1 search direction), which is well suited to cope with variable preconditioners [2].

Main implementation features

PCG-1 Single precision FORTRAN 77. The upper triangular part of the (symmetric) stiffness matrix is stored in disk memory using a regular 42 point stencil format.

CGLIB FORTRAN 77; results are available for both single and double precision versions. The upper triangular part of the matrix is stored in core using a general sparse row-wise format (indirect addressing scheme storing only the actual nonzeros).

DRBILU Double precision FORTRAN 77. The upper triangular part of the matrix is stored in core using the "diagonal" format, in which nonzero diagonals are stored in a rectangular array, and their offset with respect to the main diagonal in a companion integer vector. This scheme reproduces the performances of a direct (stencil based) addressing scheme, but is more versatile with respect to the use of different discretizations.

D-MBILU Double precision, FORTRAN 77 and C. The upper triangular part of the matrix is stored in core using a general sparse row-wise format, but a diagonal-wise storage is used for the preconditioner, assuming that the diagonal blocks are 7-diagonal matrices.

Operation count

Besides the computing time, it is interesting to compare the number of basic ($+$ or $*$) floating point operations (*flops*) needed by each software. As will be seen, this allows to reach separate conclusions for the numerical method on the one hand, and the implementation strategy on the other hand.

Since we want an implementation independent count, we did not use the evaluation made separately for each solver, but use the formulas listed below, that follow from a careful analysis of the numerical algorithms. These formulas give the number of *flops* per unknown and per (outer) iteration in function of the average number of nonzero entries per row in A (r_A) and in D (r_D). (If the sparsity is not fully exploited, these numbers should refer to the number of entries treated as nonzero).

PCG-1, CGLIB: $\mathit{flops} = 11 + 2(r_A + r_D)$.

DRBILU: $\mathit{flops} = 15 + 2(r_A + r_D)$.

D-MBILU: $\mathit{flops} = (15\,a_v + 17) + 2\,r_A + (4\,a_v + 2)\,r_D$, where a_v is the average number of inner iterations.

3 Benchmarks

3.1 Square footing problem

Description : Problem deals with the influence of flexible footing represented by localized pressure to the stress development in a soil medium. The computational domain is a parallelepiped 50x50x40 m which represents one quarter of the overall situation, see Figure below.

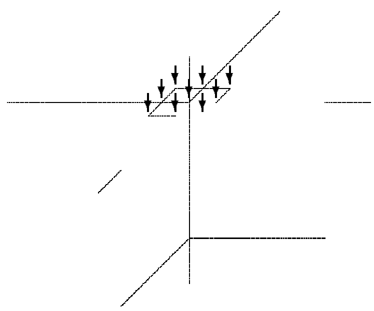


Figure: The square footing problem.

Equations used : equations of elasticity.

Material data : one isotropic material

- soil (E = 14 MPa, $\nu = 0.3$, $\rho = 2.5\,gcm^{-3}$)

Boundary conditions : zero normal displacements and zero tangential stresses on all sides except the top side. Stress on the top side.

Loading : pressure of 2.4 MPa on the square area 5x5 m at the left back corner of the top face (one quarter of footing).

Relevant user output : stresses.

Discretization : rectangular grid 40x40x40 nodes with a grid refinement under the footing. Maximal dimension of hexahedral bricks: 2.5x2.5x1.5 m, minimal dimension: 0.5x0.5x0.5 m.
The discretization is made by the linear tetrahedral finite elements and gives the linear system of 192000 unknowns. The discretization is made by TGRID preprocessor and the linear system is created by FEMSYS code, see [1].

References: [5] R. Blaheta and R. Kohut: *Benchmarks and programs for testing iterative solvers for 3D problems of geomechanics,* Report 9502 (Dept. of Applied Mathematics, Institute of Geonics, Ostrava 1995)

Contact: R. Kohut (kohut@ugn.cas.cz), A. Kolcun (kolcun@ugn.cas.cz)

3.2 Stope stability problem

Description : Problem concerns mining in the coal seam: the computational domain is a cube of 150x150x150 m which contains the characteristic stope mining situation.

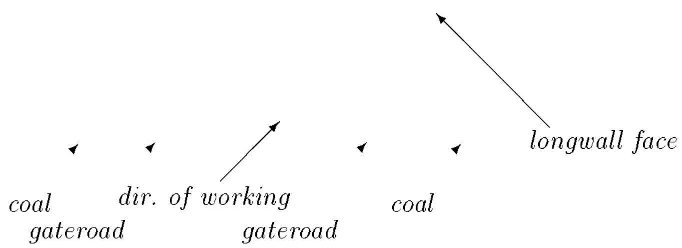


Figure: The stope mining problem.

Equations used : equations of elasticity.

Material data : three isotropic materials:

- rock (E = 8000 MPa, $\nu = 0.2$, $\rho = 2.5\, gcm^{-3}$)
- coal (E = 800 MPa, $\nu = 0.3$, $\rho = 1.3\, gcm^{-3}$)
- goaf (E = 200 MPa, $\nu = 0.4$, $\rho = 2.2\, gcm^{-3}$)

Boundary conditions : zero normal displacements and zero tangential stresses on all sides except the top side. Stress on the top side.

Loading : pressure of 12.5 MPa on the whole top side.

Relevant user output : stresses.

Discretization : rectangular grid of 40x40x40 nodes with the grid refinement in the area of mining. Maximal dimension of hexahedral bricks: 10x15x10 m, minimal dimension: 1x1.5x1.5 m.
The discretization is made by the linear tetrahedral finite elements and gives the linear system of 192000 unknowns. The discretization is made by TGRID preprocessor and the linear system is created by FEMSYS code, see [1].

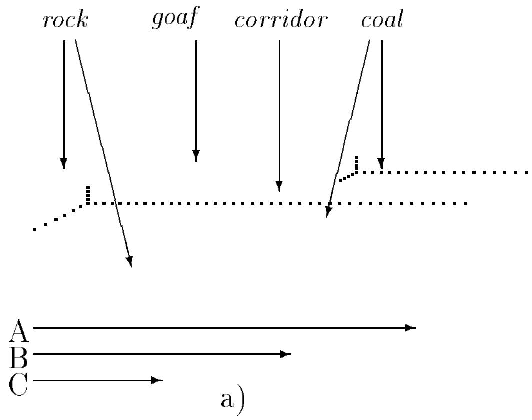
References: [5] R. Blaheta and R. Kohut: *Benchmarks and programs for testing iterative solvers for 3D problems of geomechanics,* Report 9502 (Dept. of Applied Mathematics, Institute of Geonics, Ostrava 1995)

Contact: R. Kohut (kohut@ugn.cas.cz), A. Kolcun (kolcun@ugn.cas.cz)

3.3 Headgate problems

Description : Problems concern the stability of coal pillars which are periodically distributed in the corridor passing throw a coal seam. Beside the corridor, the situation include a mined out area excavated by the longwall mining technology. Computational domain is a parallelepiped 1800x237x86 m.

We shall consider three variants of the described problem:
A – the whole situation,
B – restricted problem situated in 2/3 of the whole computational domain,
C – restricted problem situated in 1/3 of the whole computational domain.



a) b)

Figure: (a) Headgate problems with extent of the domain for the considered variants A, B and C, (b) top view on the detail of distribution of the coal pillars in the corridor.

Equations used : equations of elasticity.

Material data : three isotropic materials:

- rock (E = 8000 MPa, $\nu = 0.2$, $\rho = 2.5\,gcm^{-3}$)
- coal (E = 800 MPa, $\nu = 0.3$, $\rho = 1.3\,gcm^{-3}$)
- goaf (E = 80 MPa, $\nu = 0.4$, $\rho = 2.2\,gcm^{-3}$)

Boundary conditions : zero normal displacements and zero tangential stresses on all sides except the top side. Stress on the top side.

Loading : pressure of 12.5 MPa on the whole top side.

Relevant user output : stresses.

Discretization : rectangular grids of 302x26x33, 202x26x33, 102x26x33 nodes are used for the variants A, B and C, respectively. The grids are refined in the vicinity of the corridor. Maximal dimension of hexahedral bricks is: 6.1x16x6 m, minimal dimension is: 6.1x5.25x0.6 m.

The discretization is made by the linear tetrahedral finite elements and gives the linear system of 777348, 519948 and 262548 unknowns for the variants A, B and C, respectively. The discretization is made by TGRID preprocessor and the linear system is created by FEMSYS code, see [1].

References: [5] R. Blaheta and R. Kohut: *Benchmarks and programs for testing iterative solvers for 3D problems of geomechanics,* Report 9502 (Dept. of Applied Mathematics, Institute of Geonics, Ostrava 1995)
[2] R. Blaheta: *HIPERGEOS related benchmarks, software and testing,* HIPERGEOS Workshop, Rožnov p. R. 1996

Contact: R. Kohut (kohut@ugn.cas.cz), A. Kolcun (kolcun@ugn.cas.cz)

4 Numerical results and conclusions

The machines considered for the tests were an IBM RS6000 (67 MHz, 128 Mb RAM) (PCG-1), a SUN SPARC20 (50 MHz, 128 Mb RAM) (CGLIB, DRBILU) and a SUN Ultra-Enterprise (170 MHz, 1 Gb RAM) (D-MBILU). Timing results refer to the CPU time (user + system) needed by the solution process. To obtain comparable quantities, they are given in number of machine cycles (that is, times in seconds have been multiplied by the clock frequency). Ideally, one should have run all softwares on the same machine,

solver	# it. (a_v)	*flops* $\cdot 10^{-2}$ (per unknown)	CPU (cycles $\cdot 10^{-9}$)
PCG-1	98	$11. + 2.0\,(r_A{+}r_D) \;=\; 97.0$	28.3
CGLIB (s.p.)	76	$8. + 1.5\,(r_A{+}r_D) \;=\; 75.2$	70.9
CGLIB (d.p.)	70	$8. + 1.4\,(r_A{+}r_D) \;=\; 69.3$	72.6
DRBILU	61	$9. + 1.2\,(r_A{+}r_D) \;=\; 62.8$	10.6
D-MBILU	12 (3.972)	$9. + 0.2\,r_A + 3.8\,r_D \;=\; 44.8$	7.4

Table 1: results for the Square footing problem

solver	# it. (a_v)	*flops* $\cdot 10^{-2}$ (per unknown)	CPU (cycles $\cdot 10^{-9}$)
PCG-1	130	$14. + 2.6\,(r_A{+}r_D) \;=\; 129.$	37.4
CGLIB (s.p.)	97	$11. + 1.9\,(r_A{+}r_D) \;=\; 96.0$	73.3
CGLIB (d.p.)	88	$10. + 1.8\,(r_A{+}r_D) \;=\; 87.1$	73.0
DRBILU	81	$12. + 1.6\,(r_A{+}r_D) \;=\; 83.4$	14.3
D-MBILU	15 (4.666)	$13. + 0.3\,r_A + 3.1\,r_D \;=\; 45.8$	10.1

Table 2: results for the Stope stability problem

but it clearly turned out that the computing time is mainly influenced by the effects of some implementation features, making a finer analysis of the CPU times relatively useless.

The numerical results for the different benchmark problems are listed in Tables 1–3. For CGLIB, s.p. refers to the single precision version and d.p. to the double precision one. In all cases, the zero vector was used as initial approximation, and the stopping criterion for the (outer) iterative process was a decrease of the 2-norm of the relative residual error by 10^{-8}. If an inner-outer scheme is used, # it. is the number of outer iterations, the average number of inner iterations a_v being indicated under brackets. The used values for r_A and r_D are respectively 37 and 7 , which corresponds to the diagonal storage scheme of DRBILU (the stencil scheme of PCG-1 leads to a somewhat larger value for r_A ($r_A \approx 45$), whereas general sparse row-wise formats lead to slightly smaller values, the difference being however not significant).

The following comments turn out.

1) Clearly, the inner-outer scheme of D-MBILU performs better, essentially because this approach involves less work with A and more work with D, whereas A is much less sparse than D (here, $\frac{r_A}{r_D} \approx 5$; other discretization schemes will often result in $\frac{r_A}{r_D} \approx 3$).

2) It pays off to consider a blockwise incomplete factorization instead of a pointwise scheme, especially if the number of unknowns is not nearly the same in all directions. This confirms the results in [9]. Also, the "static" method of [7] is outperformed by methods based on a dynamic relaxation rule [8, 10, 13].

3) General sparse row-wise storage formats lead to inefficient code. This is very clear with CGLIB, somewhat less with D-MBILU, because the indirect addressing concerns

solver	# it. (a_v)	*flops* $\cdot 10^{-2}$ (per unknown)	CPU (cycles $\cdot 10^{-9}$)
variant C ($102 \times 26 \times 33$ grid)			
PCG-1	166	$18. + 3.3\,(r_A + r_D) \;=\; 164.$	65.6
DRBILU	78	$12. + 1.6\,(r_A + r_D) \;=\; 80.3$	18.4
D-MBILU	14 (5.048)	$13. + 0.3\,r_A + 3.1\,r_D \;=\; 45.1$	13.7
variant B ($202 \times 26 \times 33$ grid)			
PCG-1	202	$22. + 4.0\,(r_A + r_D) \;=\; 200.$	158.
DRBILU	97	$14. + 1.9\,(r_A + r_D) \;=\; 99.9$	46.4
D-MBILU	15 (5.333)	$14. + 0.3\,r_A + 3.5\,r_D \;=\; 50.1$	29.4
variant A ($302 \times 26 \times 33$ grid)			
PCG-1	242	$27. + 4.8\,(r_A + r_D) \;=\; 240.$	363.
DRBILU	113	$17. + 2.3\,(r_A + r_D) \;=\; 116.$	81.5
D-MBILU	15 (6.111)	$16. + 0.3\,r_A + 4.0\,r_D \;=\; 55.2$	48.6

Table 3: results for the Headgate problem

there only the multiplication by A, which has to be done only once per outer iteration. However, comparing D-MBILU and DRBILU, the latter is improved by a factor about 2 in number of *flops*, but much less significantly in CPU time [1].

4) The double precision version of CGLIB requires less iteration, on account of the influence of the rounding errors on the convergence of the Conjugate Gradient method [12]. On the other hand, it is not faster, although floating point arithmetic is as fast in double precision on modern computers. We explain this by the data movements (from disk to main memory, main memory to caches, and caches to registers). They tend to slow down sparse matrix computation[2], and are less intensive if single precision storage is used.

5) Floating point operations are slower with PCG-1 than with DRBILU, although both solvers use a direct addressing scheme, whereas the former uses single precision storage and the latter double precision storage. We explain this by the explicit use of disk storage in PCG-1. If the whole problem can fit in RAM memory, this is clearly not a clever approach. For large problems like variant A of Headgate, the use of disk memory is unavoidable, but virtual memory capabilities of modern computers manage this automatically, and this apparently results in more efficient codes.

Our final recommendations are therefore the following.

- Use the inner-outer scheme of D-MBILU with a block incomplete factorization of D as basic preconditioner.

[1] This is not true for variant A of Headgate, but the machine on which D-MBILU was tested has a larger RAM memory, allowing a relatively faster execution for such a large problem.

[2] Typically, one has one data transfer per floating point operation, which is much less than in most other types of applications.

- Store both stiffness matrix and preconditioner in core using a direct addressing scheme. Especially for the stiffness matrix, we recommend the diagonal storage scheme of DRBILU, as it is more versatile and better exploits the sparsity than a stencil based scheme.

- Make the computation in double precision, but consider single precision for the storage of the stiffness matrix and possibly of the preconditioner (assuming that the end users do not need to much accuracy on the computed displacements, as it is the case in the kind of applications targeted by the HIPERGEOS project).

References

[1] O. Axelsson: *Iterative Solution Methods*, University Press, Cambridge, 1994.

[2] O. Axelsson and P. Vassilevski: A Black-Box Generalized Conjugate Gradient Solver with Inner Iterations and Variable-Step Preconditioning, *SIAM J. Matrix Anal. Appl*, 12 (1991), 625–644.

[3] R. Blaheta: Displacement decomposition – incomplete factorization preconditioning techniques for linear elasticity problems, *J. Numer. Lin. Alg. Appl.*, 1 (1994), 107–126

[4] R. Blaheta, R. Kohut: PCG–1 iterative solver, TR – DAM – 9501, IGAS Ostrava, 1995.

[5] R. Blaheta, R. Kohut: Benchmarks and programs for testing iterative solvers for 3D problems of geomechanics, TR – DAM – 9502, IGAS Ostrava, 1995.

[6] R. Blaheta, O. Jakl, R. Kohut, A. Kolcun and J. Starý: IGAS collection of benchmarks, solvers and test results, TR – IGAS-DAM, Ostrava, 1997.

[7] I. Gustafsson: Modified Incomplete Cholesky (MIC) methods, in *Preconditioning Methods. Theory and Applications*, D. Evans, ed., New York-London-Paris, 1983, Gordon and Breach, pp. 265–293.

[8] Magolu Monga-made and Y. Notay: Dynamically relaxed block incomplete factorizations, TR – 985, Department of Mathematics, University of Utrecht, 1996.

[9] Magolu Monga-made and B. Polman: Theoretical comparison of pointwise, linewise and planewise preconditioners for solving three-dimensional problems, TR – 9609, Department of Mathematics, University of Nijmegen, 1996.

[10] Magolu Monga-made and B. Polman: Efficient planewise like preconditioners to cope with 3D problems, TR – 987, Department of Mathematics, University of Utrecht, 1996.

[11] M. Neytcheva and B. Polman: HIPERGEOS benchmark results for the 3D linear elasticity problem, TR, Department of Mathematics, University of Nijmegen, 1997.

[12] Y. Notay: On the convergence rate of the conjugate gradients in presence of rounding errors, *Numer. Math.*, 65 (1993), 301–317.

[13] Y. Notay: DRIC: a dynamic version of the RIC method, *J. Num. Lin. Alg. with Appl.*, 1 (1994), 511–532.

[14] Y. Notay: Benchmarking for HIPERGEOS, TR – GANMN 9703, University of Brussels, 1997.

[15] P. Saint Georges, R. Valkenberg : Benchmarking for HIPERGEOS – Part 1, TR – SMC 96/04, University of Brussels, 1996.

[16] P. Saint-Georges, G. Warzee, R. Beauwens, and Y. Notay: High performance PCG solvers for FEM structural analyses, *Int. J. Numer. Meth. Eng.*, 39 (1996), 1313–1340.