PROJECT REPORT HG CP 94–00820–CICT–5.1

Circulant Block Factorization Preconditioning of 3D Elasticity Systems.

WP 3.3 Parallel Iterative Solvers.

Ivan Lirkov and Svetozar Margenov
BAS, Sofia, Bulgaria

Abstract

This report follows the research plan of the HIPERGEOS Project COP–94 00820. We analyze the parallel complexity of the circulant block factorization preconditioning for the numerical solution of large scale elasticity systems. A set of preliminary numerical tests is presented as a part of the report.

1 Introduction

In this report we are concerned with the numerical solution of 3D elasticity systems. After discretization, such problems are reduced to find the solution of linear systems of the form $Ax = b$ where A is sparse symmetric positive definite matrix. Such linear systems are often solved by the Preconditioned Conjugate Gradient method.

One of the most popular and the most successful class of preconditioners is the class of incomplete LU (ILU) factorizations. One potential problem with the ILU preconditioners is that they have limited degree of parallelism. Some attempts to modify the method and to devise more parallel variants often result in a deterioration of the convergence rate.

In [3], R. Chan and T. F. Chan proposed another class of preconditioners which is based on averaging coefficients of A to form a block-circulant approximation (see also [6, 7]). The block-circulant preconditioners are highly parallelizable but they are substantially sensitive with respect to a possible high variation of the coefficients of the elliptic operator.

The sensitivity of the block-circulant approximations with respect to a high variation of the problem coefficients was relaxed in the circulant block factorization (CBF) preconditioners ([9]).

Let us remind that a circulant matrix C has the form $(C_{k,j}) = \left(c_{(j-k) \bmod m}\right)$, where m is the size of C. Let us also denote for any given coefficients $(c_0, c_1, \ldots, c_{m-1})$ by $C = (c_0, c_1, \ldots, c_{m-1})$ the circulant matrix

$$\begin{bmatrix} c_0 & c_1 & c_2 & \ldots & c_{m-1} \\ c_{m-1} & c_0 & c_1 & \ldots & c_{m-2} \\ \vdots & \vdots & \vdots & & \vdots \\ c_1 & c_2 & \ldots & c_{m-1} & c_0 \end{bmatrix}.$$

Any circulant matrix can be factorized as

$$C = F\Lambda F^*,$$

where Λ is a diagonal matrix containing the eigenvalues of C, and F is the Fourier matrix

$$F = \frac{1}{\sqrt{m}} \left\{ e^{2\pi \frac{jk}{m} \mathbf{i}} \right\}_{0 \leq j,k \leq m-1}.$$

Here $\mathbf{i}$ stands for the imaginary unit.

The main goal of this study is analysis of the parallel complexity of the PCG method with considered circulant block-factorization preconditioner. We estimate only the parallel execution time for one PCG iteration. We analyze the parallel complexity, speed-up and parallel efficiency for the studied preconditioners on various standard architectures. Finally, we demonstrate some preliminary numerical results obtained on a symmetric multiprocessor system.

1.1 Circulant block factorization

The CBF preconditioning technique incorporates the circulant approximations into the framework of the *LU* block factorization. The computational efficiency and parallelization of the resulting algorithm are as high as of the block circulant one.

If we consider the 2D elliptic problem:

$$-(a(x_1, x_2)u_{x_1})_{x_1} - (b(x_1, x_2)u_{x_2})_{x_2} = f(x_1, x_2) \tag{1}$$

on the unit square $[0, 1] \times [0, 1]$ with Dirichlet boundary condition, where the domain is discretized by uniform grid with n_1 and n_2 grid points in the coordinate directions and a standard five-point FDM (FEM) approximation is used, then the stiffness matrix A admits a block-tridiagonal structure where the diagonal blocks are tridiagonal matrices and the off-diagonal blocks are diagonal matrices.

For this problem, the general form of the *circulant block factorization preconditioner* is defined by

$$M_{CBF} = tridiag(-C_{i,i-1}, C_{i,i}, -C_{i,i+1}) \qquad i = 1, 2, \ldots, n_1, \tag{2}$$

where $C_{i,j} = Circulant(A_{i,j})$ is a circulant approximation of the corresponding block $A_{i,j}$.

Let us consider now the similar 3D elliptic problem:

$$-(a(x_1,x_2,x_3)u_{x_1})_{x_1} - (b(x_1,x_2,x_3)u_{x_2})_{x_2} - (c(x_1,x_2,x_3)u_{x_3})_{x_3} = f(x_1,x_2,x_3) \qquad (3)$$

on the unit cube $[0,1] \times [0,1] \times [0,1]$ with Dirichlet boundary condition. If the domain is discretized by uniform grid with n_1, n_2 and n_3 grid points in the coordinate directions, and if a standard (for such a problem) seven-point FDM (FEM) approximation is used, then the stiffness matrix A admits again a block-tridiagonal structure. A can be written in the block-form

$$A = tridiag(-A_{i,i-1}, A_{i,i}, -A_{i,i+1}) \qquad i = 1,2,\ldots,n_1,$$

where $A_{i,i}$ are block-tridiagonal matrices which correspond to one x_1-plane and off-diagonal blocks are diagonal matrices.

Now, we apply the general CBF preconditioning approach and construct the preconditioner M_{CBF} in the form:

$$M_{CBF} = tridiag(-C_{i,i-1}, C_{i,i}, -C_{i,i+1}) \qquad i = 1,2,\ldots n_1, \qquad (4)$$

where $C_{i,j} = Block-Circulant(A_{i,j})$ is a block-circulant approximation of the corresponding block $A_{i,j}$. The relative condition number of the CBF preconditioner for the model (Laplace) 3D problem is studied in [8] for $n_1 = n_2 = n_3 = n$, where the following estimate is derived:

$$\kappa(M_0^{-1}A_0) \leq 4n. \qquad (5)$$

1.2 Elasticity problem and FEM discretization

We consider the 3D problem of linear elasticity under the usual general assumptions: small displacements and isotropic material properties. Then the stressed-strained state of the continuous media is described by a coupled elliptic system of three differential equations (the Navier equations).

Let us assume now, that the finite element method is applied to this problem. The corresponding FEM linear system can be written in the following block-form

$$\begin{pmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{pmatrix} \begin{pmatrix} \underline{u}_1 \\ \underline{u}_2 \\ \underline{u}_3 \end{pmatrix} = \begin{pmatrix} \underline{f}_1 \\ \underline{f}_2 \\ \underline{f}_3 \end{pmatrix} \qquad (6)$$

where vectors $\underline{u}_i$ correspond to the nodal unknowns of the displacements u_i.

2 Displacement decomposition circulant block factorization preconditioning

2.1 Displacement decomposition

There are a lot of works dealing with preconditioning iterative solution methods for the FEM elasticity systems. In an earlier paper, Axelsson and Gustafson [1] have implemented modified point-ILU factorization for such problem. As the coupled system does not lead to an M-matrix, they construct their preconditioners based on the point-ILU factorization of the displacement decoupled block-diagonal part of the original matrix. This approach is known as *displacement decomposition*. The displacement decomposition remains until now one of the most robust approaches (see also, e.g., [2]). A new block-ILU factorization preconditioner based on block-size reduction is studied in [4, 5]. An advantage of this approach is that the approximate factorization exists for symmetric and positive definite block-tridiagonal matrices that are not necessarily M-matrices.

To define the displacement decomposition preconditioner M_{DD} of the matrix K we consider the auxiliary Laplace equation $-u_{x_1x_1}-u_{x_2x_2}-u_{x_3x_3} = f$ with boundary conditions corresponding to the coupled elasticity problem. Let us assume primary, that this Laplace equation is discretized by the same finite elements as the original problem and let K_0 be the corresponding stiffness matrix.

The following Korn's inequality is the theoretical background of the displacement decomposition methods:

$$\kappa(M_{DD}^{-1}K) = \mathcal{O}\left(\frac{1}{1-2\nu_{\max}}\right), \tag{7}$$

where $\nu_{\max} = \max_\Omega \nu$, $\nu \in (0, 1/2)$ is the Poisson ratio and

$$M_{DD} = \begin{pmatrix} K_0 & 0 & 0 \\ 0 & K_0 & 0 \\ 0 & 0 & K_0 \end{pmatrix}. \tag{8}$$

The next step in the construction used here is to substitute K_0 by A_0 in (8), where A_0 stands for the Laplace stiffness matrix corresponding to linear finite elements or seven point finite difference stencil discretization. This is motivated by the more simple structure of A_0 as well as by the spectral equivalence

$$\kappa(A_0^{-1}K_0) = \mathcal{O}(1). \tag{9}$$

2.2 Circulant block factorization

We use here the CBF preconditioner (4) for the matrix A_0. In this way we get the DD CBF preconditioner for the stiffness matrix K of the problem (6) defined by

$$M_{DD\ CBF} = \begin{pmatrix} M_0 & 0 & 0 \\ 0 & M_0 & 0 \\ 0 & 0 & M_0 \end{pmatrix}.$$

Now, we summarize (5), (7) and (9), and get the following estimate for the condition number of the DD CBF preconditioner

$$\kappa(M_{DD\ CBF}^{-1}K) = \mathcal{O}\left(\frac{n_{\max}}{1-2\nu_{\max}}\right). \tag{10}$$

Unfortunately, the constant in (10) increases when the coefficient of the differential problem are varying.

3 Numerical tests

In this section, we compare the convergence rate of the DD CBF preconditioner and the Block Size Reduction BILU preconditioner (see, e.g., [4]) for the 3D elasticity problems. The computational domain Ω is a parallelogram. We use an uniform grid for the discretization and trilinear finite elements. For CBF preconditioner we first symmetrically scale the stiffness matrix K by its diagonal to improve the convergence rate. The computations are done with double precision on a SUN Ultra-Enterprise Symmetric Multiprocessor. The iteration stopping criterion is $\|r^{N_{it}}\| < 10^{-3}$, where r^j stands for the residual at the jth iteration step of the preconditioned conjugate gradient method.

In these tests we consider the MOST XXI Benchmarks 1 and 2 as specified in the HIPERGEOS Project. The computational domain is the parallelogram $\Omega = (0, x_1^{\max}) \times (0, x_2^{\max}) \times (0, x_3^{\max})$. The boundary conditions are as follows: homogeneous Dirichlet boundary conditions on the bottom side $x_3 = 0$ (the displacements are equal to zero); homogeneous Neumann boundary conditions on the vertical sides of Ω (the stresses are equal to zero); Neumann boundary conditions on the top side $x_3 = x_3^{\max}$, where the load (V, H) from the upper construction is applied. The computational domain is determined by $x_1^{\max} = x_2^{\max} = 37.2m; x_3^{\max} = 31.0m$. The vertical cross section $x_2 = 18.6$ of Ω is presented in Figures 1 and 2. The cross section size of the piles is $D = 1.2m$. The length of the piles and the size of the active zone under the top of the piles are respectively $L_P = 15m$ and $H_{act} = 12m$. The mechanical characteristics of the piles are as follows: $E_P = 31500MPa$; $\nu_P = 0.2$. The mesh-sizes are $n_1 = n_2 = n_3 = 32$ and respectively the total size of the discrete problem is $N = 98304$. The number of iterations and the execution times for

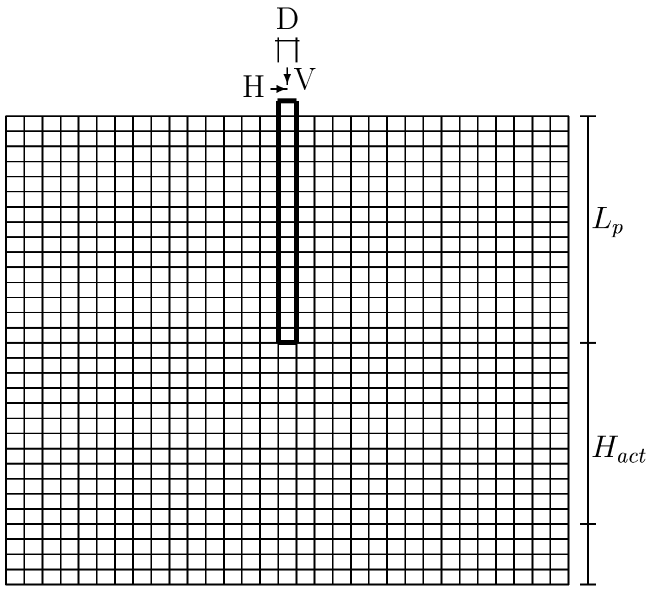

Figure 1: Benchmark 1; Cross section of the computational domain Ω. $E_{soil} = 10MPa$, $\nu_{soil} = 0.3$

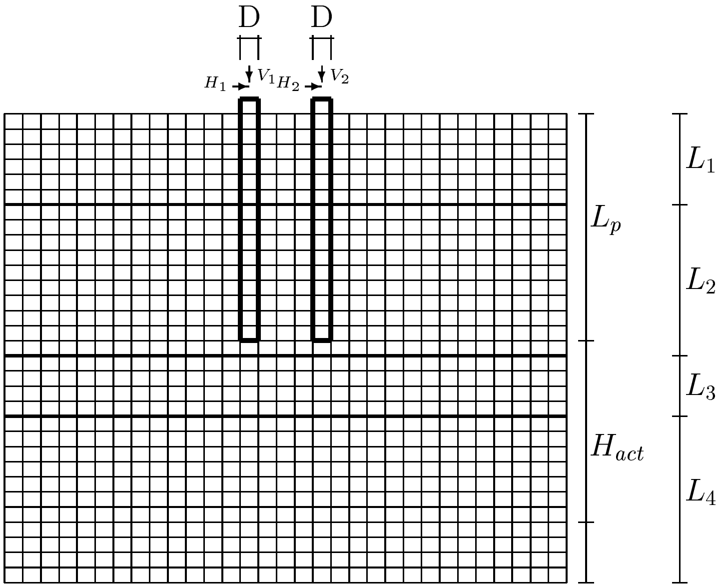

Figure 2: Benchmark 2; Cross section of the computational domain Ω. $E_{L_1} = 5.2MPa$, $E_{L_2} = 9.4MPa$, $E_{L_3} = 14.0MPa$, $E_{L_4} = 21.4MPa$; $\nu_{L_1} = 0.4$, $\nu_{L_2} = 0.35$, $\nu_{L_3} = 0.25$, $\nu_{L_1} = 0.2$.

Table 1: Number of iterations and execution times for BSR BILU and DD CBF preconditioners for Benchmark problems.

	Benchmark 1		Benchmark 2	
	Iterations	Time	Iterations	Time
BSR BILU	85	15:36.1	157	28:05.4
DD CBF	334	10:20.4	741	20:44.5

solving of linear algebraic system (6) with the two compared preconditioners are presented in the Table 1. Although the jump of the coefficients on the boundary between the piles and the soil the obtained results show that the proposed preconditioner is faster than BSR BILU preconditioner even for that problem. As it is shown bellow, there are very good parallel algorithms for the CBF preconditioner in contrast to the preconditioners based on the incomplete LU factorizations.

4 Model of computation and communication times

We assume that the computations and communications are not overlapped and therefore, the execution time of the parallel implementation is the sum of the computation time and the communication time.

We shall use in our analysis standard models for the arithmetic and communication times [10]. First, assuming no arithmetic vectorization, the execution of M arithmetic operations on one processor takes time

$$T_a = M * t_a, \tag{11}$$

where t_a is the average unit time to perform one arithmetic operation on one processor. Second, the local communication time of transfer of M data from one processor to its neighbor is approximated by

$$T_{local} = t_s + M * t_c, \tag{12}$$

where t_s is the start-up time and t_c is the incremental time necessary for each of all M words to be sent.

Finally, for the various architectures, we denote the following two communication times, respectively by:

- Broadcasting a number from one processor to all others: $b(p)$, where there are p processors in the computer system;
- Gathering p data packets, each packet with M/p words, in one processor from all others: $g(M, p)$.

In this report we discuss three distributed memory architecture models: ring, square grid and hypercube, and shared memory model. We model the shared memory architecture by assuming that all the processors in the system can be considered as neighbors. For these architectures the above communication times are respectively:

	$b(p)$	$g(M, p)$
ring	$(t_s + t_c)\frac{p}{2}$	$p\, t_s + M\, t_c$
grid	$(t_s + t_c)\sqrt{p}$	$2\sqrt{p}\, t_s + M(1 + \frac{1}{\sqrt{p}})\, t_c$
hypercube	$(t_s + t_c)\log p$	$\log p\, t_s + M(1 - \frac{1}{p})\, t_c$
shared memory	$\lceil \log p \rceil (t_s + t_c)$	$(p - 1)(t_s + \frac{M}{p} t_c)$

To achieve a low cost of the processor synchronization and communication, we partition the computational domain into p parallelepipeds so that each parallelepiped contains n_1/p x_1-planes. We map all grid points in one parallelepiped onto a processor so that the values at the grid points in neighbor parallelepiped are computed on the neighbor processor. In this way, the CG method requires a communication of only values from one grid plane with neighbor processors.

5 Analysis of the parallel complexity

We can now estimate the total execution time T_{PCG} for one PCG iteration for the considered circulant block-factorization preconditioner on a studied parallel system with p processors. Each iteration consists of one matrix vector multiplication with the matrix K, one multiplication with the inverse of the preconditioner $M_{DD\ CBF}$ (solving a system of equations with matrix M), two inner products and three linked triads (a vector updated by a vector multiplied by a scalar). Consequently

$$T_{PCG}(p) = T_{mult} + T_{prec} + 2T_{inn\_prod} + 3T_{triads}.$$

For simplicity we assume that the mesh sizes are equal and they are equal to an exact power of two, i.e., $n_1 = n_2 = n_3 = n = 2^l$. Then

$$T_{mult} = 243\frac{n^3}{p}t_a + 4(t_s + 3n^2 t_c),$$

$$T_{inn\_prod} = 6\frac{n^3}{p}t_a + g(p,p) + b(p),$$

$$T_{triads} = 6\frac{n^3}{p}t_a,$$

$$T_{prec} = 6\frac{n^2}{p}T_{FFT}(n) + 36\frac{n^3}{p}t_a + 2g(3\frac{n^3}{p}, p),$$

and where $T_{FFT}(n)$ is the time for execution of FFT on a given n-vector on one processor. If we use 2-radix algorithm, then $T_{FFT}(n) = 5n \log n t_a$.

Combining these results we obtain the following estimates of the execution times for the studied architectures

$$T_{PCG}(p) \;=\; 3\,(103 + 10\log n)\,\frac{n^3}{p}t_a + 4\left(t_s + 3n^2 t_c\right) + 2g(3\frac{n^3}{p}, p) + 2g(p,p) + 2b(p).$$

For example, for shared memory model we obtain

$$\begin{aligned} T_{PCG}(p) \;&=\; 2\,(p + \lceil \log p \rceil + 1)\,t_s + 2\left[3(p-1)\frac{n^3}{p^2} + 6n^2 + \lceil \log p \rceil\right] t_c \\ &+\; 3\,(103 + 10\log n)\,\frac{n^3}{p}t_a \end{aligned}$$

Then, the leading terms of the parallel time complexity functions are:

$$T_{PCG}(p) \approx 2pt_s + 6(1 - \frac{1}{p})\frac{n^3}{p}t_c + 3(103 + 10\log n)\frac{n^3}{p}t_a. \qquad (13)$$

Our next goal is to analyze the relative speed-up S_p and the relative efficiency E_p, where

$$S_p = \frac{T(1)}{T(p)} \le p \quad \text{and} \quad E_p = \frac{S_p}{p} \le 1.$$

We apply now (13) and obtain:

$$S_p \approx \frac{3(103 + 10\log n)}{2\frac{p^2}{n^3}\frac{t_s}{t_a} + 6(1 - \frac{1}{p})\frac{t_c}{t_a} + 3(103 + 10\log n)}p. \qquad (14)$$

Obviously, for our preconditioner,

$$\lim_{n\to\infty} S_p = p \quad \text{and} \quad \lim_{n\to\infty} E_p = 1,$$

i.e., the algorithms are asymptotically optimal. More precisely, if

$$\log n \gg \frac{p^2}{n^3}\frac{t_s}{t_a} + \frac{t_c}{t_a},$$

then E_p is near to 1. Unfortunately, the start-up time t_s is usually much larger than t_a, and for relatively small n the first term of the denominators in (14) is significant, in this case the efficiency is much smaller than 1.

6 Experimental results

In this section we report the results of the experiments executed on SUN Ultra-Enterprise Symmetric Multiprocessor. The code has been implemented using the MPI (Message-Passing Interface) library. The obtained parallel time, relative parallel speed-up and relative efficiency are presented in Table 2. One can see the increase of the influence of the communication time with the number of processors on the speed-up and on the parallel efficiency. The general behavior is in a good agreement with the theoretical estimates. We point your attention that these times are obtained in multi-user mode on the system with eight processors and this fact explains the lower than the expected efficiency on eight processors.

Table 2: Parallel time, speed-up and efficiency for the CBF preconditioner on a SUN Ultra-Enterprise symmetric multiprocessor.

n	p	$T(p)$	S_p	E_p
32	1	3.403		
	2	1.740	1.96	0.98
	4	0.893	3.81	0.95
	8	0.559	6.09	0.76
48	1	14.976		
	2	7.578	1.98	0.99
	3	5.138	2.91	0.97
	4	3.879	3.86	0.97
	6	2.673	5.60	0.93
	8	2.131	7.03	0.88

7 Summary

We are concerned with the numerical solution of elasticity problems. After discretization, such problems reduce to the solution of linear systems. We use a preconditioner based on a block-circulant approximation of the blocks of the matrix obtained by displacement decomposition method. We exploit the fast inversion of block-circulant matrices. The computation and the inversion of these circulant block factorization preconditioners are highly parallelizable on a wide variety of architectures.

Our next goal is to implement the assembling of the stiffness matrix K in parallel and to obtain real parallel times for the total solution of the Benchmark problems.

References

[1] O. Axelsson and I. Gustafsson. Iterative methods for the solution of the navier equations of elasticity. *Comp.Meth.Appl.Mech.Eng.*, 15:241–258, 1978.

[2] R. Blaheta. Displacement decomposition–incomplete factorization preconditioning techniques for linear elasticity problems. *Num. Lin. Alg. Appl.*, 1:107–128, 1994.

[3] R.H. Chan and T.F. Chan. Circulant preconditioners for elliptic problems. *J. Num. Lin. Alg. Appl.*, 1:77–101, Mar. 1992.

[4] T.F. Chan, S.D. Margenov, and P.S. Vassilevski. Performance of block–ILU factorization preconditioners based on block–size reduction for 2d elasticity systems. *SIAM J.Sci.Comput.*, to appear.

[5] A. Georgiev, S. Margenov, P.S. Vassilevski, and L. Zikatanov. Application of the block–size reduction bilu algorithm for 3d simulation of pile foundations. In *Proceedings of the conference Geomechanics'96, Roznov, CR*. Bolkema, September 1996.

[6] S. Holmgren and K. Otto. Iterative solution methods for block-tridiagonal systems of equations. *SIAM J. Matr. Anal. Appl.*, 13:863–886, 1992.

[7] T. Huckle. Some aspects of circulant preconditioners. *SIAM J. Sci. Comput.*, 14:531–541, 1993.

[8] I. Lirkov and S. Margenov. Conjugate gradient method with circulant block-factorization preconditioners for 3d elliptic problems. In *Proceedings of the Workshop # 3 Copernicus 94-0820 "HIPERGEOS" project meeting, Rožnov pod Radhoštěm*, 1993.

[9] I. Lirkov, S. Margenov, and P.S. Vassilevski. Circulant block-factorization preconditioners for elliptic problems. *Computing*, 53(1):59–74, 1994.

[10] Y. Saad and M.H. Schultz. Data communication in parallel architectures. *Parallel Comput.*, 11:131–150, 1989.