Bull. Nov. Comp. Center, Num. Anal., 11 (2002), 51–59

Two explicit incomplete factorization methods*

V.P. Il'in, E.A. Itskovich

Two explicit incomplete factorization methods and their program implementation are presented for solution of linear algebraic systems with real symmetric positive definite (SPD) matrices. The algorithms are based on the efficient Eisenstat modification of preconditioning for the matrix row sparse format. The fast iterative convergence is provided by conjugate gradient approaches, relaxation, and compensation parameters.

1. Introduction

Iterative solution of linear algebraic systems of equations with very large sparse matrices is the topic of many books, papers, and codes, see for example [1–5]. The purpose of this paper is the description of algorithms and building blocks for solving the real systems with the SPD matrices, which arise in finite-difference, or finite-volume, or finite-element approximations of multi-dimensional boundary value problems (BVPs):

$$Au = f, \quad A = D - L - U. \tag{1}$$

Here $A = \{a_{i,j}\}$, $u = \{u_i\}$, $f = \{f_i\}$ is square matrix, unknown, and given vectors, $i, j = 1, \ldots, N$; D, L, and U are diagonal, strictly low, and upper triangular matrices, respectively. For the SPD matrices ($A = A^t$, $L^t = U$, $\lambda(A) > 0$) we denote the spectral condition number $\mathrm{cond}(A) = \lambda_{\max}(A)/\lambda_{\min}(A)$.

We also consider the positive type (PT) matrices which mean positive diagonal and non-positive off-diagonal entries, and diagonal dominance:

$$a_{ii} > 0, \quad a_{ij} \le 0 \text{ for } i \ne j, \quad a_{ii} \ge \sum_{j=1,\ldots,N,\ j \ne i} |a_{ij}|, \tag{2}$$

under existence of strictly inequality even for one i in the last relation (2). The PT-matrices are the particular class of M-matrices, i.e., monotone ($A^{-1} \ge 0$) ones with the non-positive off-diagonal entries. A particular

*Supported by the Russian Foundation for Basic Research under Grants 01-07-90367, 02-01-01176.

type of the symmetric PT-matrices is called the Stiltijes matrices which have both the SPD and monotonicity properties.

In this paper, two explicit incomplete factorization methods are presented. Its implementation is based on the efficient Eisentat modification of preconditioning and different types of conjugate gradient approaches, see [1].

2. Description of algorithms

Define the explicit (i.e., non-block, or point-wise) preconditioning matrix as

$$B = (G-L)G^{-1}(G-U), \quad G = \frac{1}{\omega}D - \theta S, \quad Se = \left(\frac{1-\omega}{\omega}D + LG^{-1}\right)e, \tag{3}$$

where ω, θ are the relaxation and the compensation parameters, e is the vector with unit entries, S and G are diagonal matrices. The preconditioner B provides for $\theta = 1$ the row sum criteria, or compensation principle, $Be = Ae$. For symmetric original matrix A and $\theta = 0$ from (2) and conjugate gradient approach, we obtain the SSOR-CG method [2].

If G is the SPD-matrix, system (1) is transformed to equivalent preconditioned system

$$\begin{aligned}
\tilde{A} &= G^{1/2}(G-L)^{-1}A(G-U)^{-1}G^{1/2} \\
&= (I-\tilde{L})^{-1} + (I-\tilde{U})^{-1} - (I-\tilde{L})^{-1}(2I-\tilde{D})(I-\tilde{U})^{-1}, \\
\tilde{L} &= G^{-1/2}L\,G^{-1/2}, \quad \tilde{U} = G^{-1/2}U\,G^{-1/2}, \quad \tilde{D} = G^{-1/2}D\,G^{-1/2}, \\
\tilde{u} &= (I-\tilde{U})G^{1/2}u, \quad \tilde{f} = (I-\tilde{L})^{-1}G^{-1/2}f, \quad \tilde{A}\tilde{u} = \tilde{f}.
\end{aligned} \tag{4}$$

An important sense of the above transformation consists of spectral condition number property (for the SPD matrices B, A)

$$\mathrm{cond}(\tilde{A}) = \mathrm{cond}(B^{-1}A) \tag{5}$$

and simple implementation of the matrix-vector multiplication

$$\tilde{A}p = q + (I-\tilde{L})^{-1}[p - (2\,I - \tilde{D})q], \qquad q = (I-\tilde{U})^{-1}p, \tag{6}$$

which approximately demands the same number of arithmetical operations as the matrix-vector product Ap.

The classical preconditioned conjugate gradient method for the SPD matrix $\tilde{A}$, in terms of system (4), can be written as follows (we call it the EXIFCG):

$$\begin{aligned}
&\tilde{r}^0 = \tilde{f} - \tilde{A}\tilde{u}^0, \quad p^0 = \tilde{r}^0, \\
&\tilde{u}^{n+1} = \tilde{u}^n + \alpha_n p^n, \quad \alpha_n = (\tilde{r}^n, \tilde{r}^n)/(\tilde{A}p^n, p^n), \\
&\tilde{r}^{n+1} = \tilde{r}^n - \alpha_n \tilde{A}p^n, \quad p^{n+1} = \tilde{r}^{n+1} + \beta_n p^n, \\
&\beta_n = (\tilde{r}^{n+1}, \tilde{r}^{n+1})/(\tilde{r}^n, \tilde{r}^n), \quad n = 0, 1, 2, \ldots .
\end{aligned} \tag{7}$$

This iterative process provides at each n the minimization of the functional $\varphi_n = (\tilde{A}^{-1}\tilde{r}^n, \tilde{r}^n)$. The necessary number of iteration for satisfying the condition $\varphi_n/\varphi_0 \leq \varepsilon < 1$ is estimated by the inequality

$$n(\varepsilon) \leq 1 + \frac{1}{2}\left|\ln\frac{\varepsilon}{2}\right|\sqrt{\mathrm{cond}\tilde{A}}, \tag{8}$$

where the condition number of preconditioned matrix $\tilde{A}$ is defined in (5). But, because the values φ_n are not known during implementation of formulas (7), the following stopping criteria in iterations is used:

$$[(\tilde{r}^n, \tilde{r}^n)/(\tilde{r}^0, \tilde{r}^0)]^{1/2} \leq \varepsilon. \tag{9}$$

The vectors $\tilde{r}^n$, p^n in (7) satisfy the following orthogonal conditions:

$$(\tilde{r}^n, \tilde{r}^k) = \hat{c}_{n,k}\delta_{n,k}, \quad (Ap^n, p^k) = \check{c}_{n,k}\delta_{n,k}, \tag{10}$$

where $\delta_{n,k}$ is the Kronecker symbol and $\hat{c}_{n,k}$, $\check{c}_{n,k}$ are some normalizing values.

Another algorithm is a variant of the MINRES which minimizes a functional $\psi_n = (\tilde{r}^n, \tilde{r}^n)$ and is described by the formulas which can be obtained from (7) by simple changing the computation of the parameters α_n, β_n to

$$\alpha_n = (\tilde{A}\tilde{r}^n, \tilde{r}^n)/(\tilde{A}p^n, \tilde{A}p^n), \quad \beta_n = (\tilde{A}\tilde{r}^{n+1}, \tilde{r}^{n+1})/(\tilde{A}\tilde{r}^n, \tilde{r}^n). \tag{11}$$

An implementation of formulas (7), (11) demands two matrix-vector products $\tilde{A}p^n$ and $\tilde{A}\tilde{r}^n$. To avoid such disadvantage, we reformulate this method (we call it the EXIFMR) in the following way:

$$\begin{aligned}
&\tilde{r}^0 = \tilde{f}^0 - \tilde{A}\tilde{u}^0, \quad p^0 = \tilde{r}^0,\\
&\alpha_n = (\tilde{A}\tilde{r}^n, \tilde{r}^n)/(\tilde{A}p^n, \tilde{A}p^n), \quad \tilde{u}^{n+1} = \tilde{u}^n + \alpha_n p^n,\\
&\tilde{r}^{n+1} = \tilde{r}^n - \alpha_n \tilde{A}p^n, \quad \beta_n = (\tilde{A}\tilde{r}^{n+1}, \tilde{r}^{n+1})/(\tilde{A}\tilde{r}^n, \tilde{r}^n),\\
&p^{n+1} = r^{n+1} + \beta_n p_n, \quad \tilde{A}p^{n+1} = \tilde{A}\tilde{r}^{n+1} + \beta_n \tilde{A}p^n,\\
&n = 0, 1, 2, \ldots .
\end{aligned} \tag{12}$$

For this method, the number of iterations is also estimated by inequality (8), but ε is defined here from the condition $\psi_n/\psi_0 \leq \varepsilon$. So, the last method has an advantage that stopping criteria (9) is really optimal in this case, and the values $(\tilde{r}^n, \tilde{r}^n)$ decrease monotonically during iterations. Instead of (10), the vectors $\tilde{r}^n, p^n$ in (11) have the orthogonal properties

$$(\tilde{A}\tilde{r}^n, r^k) = \hat{c}_{n,k}\delta_{n,k}, \quad (\tilde{A}p^n, \tilde{A}p^k) = \check{c}_{n,k}\delta_{n,k}. \tag{13}$$

In both algorithms, the reconstruction of solution is performed after finishing iterations by means of the formula

$$u = G^{-1/2}(I - U)^{-1}\tilde{u}. \tag{14}$$

The diagonal entries of the matrix G in (3) are computed by the recurrent formula

$$g_i = [1 + \theta(\omega - 1)]a_{i,i}/\omega + w_i, \quad i = 1, 2, \ldots, N,$$
$$w_1 = 0, \quad w_i = \sum_{j=1}^{i-1} a_{i,j}t_j/g_j, \quad t_i = \sum_{j=i+1}^{N} a_{i,j}, \tag{15}$$

and the matrix-vector transformations (4) are implemented by simple relations

$$\tilde{D} = \{\tilde{a}_{i,i} = a_{i,i}/g_i\}, \quad \tilde{U} = \{\tilde{a}_{i,j} = a_{i,j}/\sqrt{g_i g_j}\},$$
$$\tilde{u}_N = u_N g_N^{1/2}, \quad \tilde{u}_i = u_i g_i^{1/2} - \sum_{j=i+1}^{N} \tilde{a}_{i,j}u_j g_j^{1/2}, \quad i = N-1, \ldots, 1, \tag{16}$$
$$\tilde{f}_1 = f_1 g_1^{1/2}, \quad \tilde{f}_i = f_i g_i^{-1/2} - \sum_{j=1}^{i-1} \tilde{a}_{i,j}\tilde{f}_j g_j^{-1/2}.$$

The transformation of the matrix L into $\tilde{L}$ can really be avoid because of symmetricity property, i.e., $a_{i,j}$ and $\tilde{a}_{i,j}$ for $i > j$ equal to $a_{j,i}$ and $\tilde{a}_{j,i}$, respectively.

If system (1) is obtained from the 5-point approximation of the two-dimensional boundary value problem for the Poisson equation at the rectangular quasi-uniform grid with characteristic mesh step $h = O(N^{-1/2})$, then for $\omega = \theta = 1$ the condition number of precondition matrix is

$$\mathrm{cond}(\tilde{A}) = O(h^{-1}),$$

and corresponding number of iterations equals to $n(\varepsilon) = O(h^{-1/2})$, see [2]. The same order of $\mathrm{cond}(\tilde{A})$ and $n(\varepsilon)$ is provided for the SSOR-CG method with optimal value of relaxation parameter ω and $\theta = 0$, see [1]. The similar results are valid for the 7-point grid systems, which approximate the 3D boundary value problems.

In [2], it is also proved that for grid approximation of the elliptic self-adjoint PDEs with strongly variable coefficients or mesh steps, it is possible to construct an adaptive ordering of mesh points (or unknowns), which provides the increasing (!) of convergence rate of iterations, comparing to the model problem with constant coefficients and uniform grid.

3. Description of the code

The presented algorithms are implemented into two subroutines `EXIFCG` and `EXIFMR`. The programming language is FORTRAN-77. All arithmetic operations are implemented in double precision only. These subroutines use the

same special row sparse format for saving non-zero entries of the matrices D and U only, because the matrix A is symmetric. So, for each i-th row of the matrix U the number `ne(i)` of nonzero entries $a_{i,j}$ for $j > i$, their corresponding column numbers j and own real values $a_{i,j}$ are given in the arrays `NEIB(NU)` and `AU(NU)`, where the `NU` is the total number of nonzero entries in the matrix U.

The iterations are going until the convergence criteria (9) for the given tolerance ε or the condition $n = n_{\max}$ will be held, where $n_{\max}$ is the given number. The call of the subroutines is identical for both cases, except its name, and has the form

```
call EXIFCG(D, U, F, NE, N, NEIB, AU, NU, EPS, NMAX, TETA, OM)
```

Here the arguments are:

`N` is an order of the system $Au = f$,

`NU` is the number of nonzero entries of U – the upper triangular part of the matrix A,

`EPS` is an accuracy of iterative solution (tolerance),

`NMAX` is a maximal number of iterations (input) and resulting number of iterations (output),

`U(N)` is an initial value of solution (input) and resulting solution (output),

`F(N)` is the right-hand side (input) and preconditioned residual $\tilde{r}^n$ (output),

`D(N)` is an array of diagonal entries of the matrix A (input), entries of the matrix $2I - \tilde{D}$ (output),

`NE(N)` is the numbers of nonzero entries into the rows of the matrix U,

`NEIB(NU)` contains the column numbers of nonzero entries of U,

`AU(NU)` contains the values of nonzero entries of U (input), corresponding entries of the matrix $\tilde{U}$ (output),

`TETA` is the compensation parameter θ,

`OM` is the relaxation parameter ω.

The subroutine `EXIFCG` uses, in addition, an auxiliary real arrays `u1(n)`, `ap(n)`, `g(n)`, `p(n)`.

The general structure and building blocks of subroutines can be described by the following scheme:

- computing the matrix G by formulas (15) (call the auxiliary subroutine `PRECO`),
- implementation of the Eisenstat matrix-vector transformations (4),

- calculations of initial residual $\tilde{r}^0$ by means of (7) and $(\tilde{r}^0, \tilde{r}^0)$, p^0 (for **EXIFMR** – the vector $\tilde{A}\tilde{r}^0$, also),
- realization of iterative process, by formulas (7),
- reconstruction of solution by formula (14).

The computational resources of considered subroutines are defined (per grid node) by the number Q of multiplications at each iteration and the volume of necessary operative memory P: for **EXIFCG**, $P = 11$ and $Q = 12$; for **EXIFMR**, $P = 12$ and $Q = 14$.

The robustness of subroutines is provided by the full control of possible division by zero and positive definiteness of the matrix.

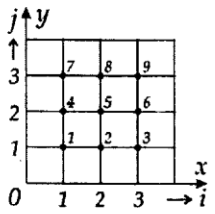
4. Examples

We present the input and output data for algebraic system which arise in finite difference approximation of the Dirichlet boundary problem for the Poisson equation in the rectangular computational domain at the uniform grid:

$$
\begin{aligned}
&-\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = f(x,y), \quad (x,y) \in \Omega = [0,(I+1)h] \times [0,(J+1)h], \\
&u(x,y)|_\Gamma = g(x,y), \\
&x_i = i\,h, \;\; y_j = j\,h, \;\; i = 0,1,\ldots,I+1, \;\; j = 0,1,\ldots,J+1, \\
&(A\,v)_{ij} \equiv -a_{ij}v_{i-1,j} - b_{ij}v_{i,j-1} - c_{ij}v_{i+1,j} - d_{ij}v_{i,j+1} + e_{ij}v_{i,j} = f_{ij}, \\
&i = 1,\ldots,I, \quad j = 1,\ldots,J.
\end{aligned}
\tag{17}
$$

Here the coefficients and f_{ij} are computed taking into account the exact solution and boundary conditions in (17):

$$
\begin{aligned}
&u(x,y) = g(x,y) = 1, \quad f(x,y) = 0, \\
&a_{ij} = 1 - \delta_{i,0}, \;\; b_{ij} = 1 - \delta_{j,0}, \;\; c_{ij} = 1 - \delta_{i,I}, \;\; d_{ij} = 1 - \delta_{j,J}, \;\; e_{ij} = 4, \\
&f_{ij} = \begin{cases} 0, & 1 < i < I, \;\; 1 < j < J, \\ 1, & i = 1, I \;\text{ or }\; j = 1, J, \\ 2, & i = j = 1; \;\; i = 1, j = J; \;\; i = I, j = 1; \;\; i = I, j = J. \end{cases}
\end{aligned}
\tag{18}
$$



Under natural row-by-row ordering of grid nodes and respective numbering of vector components by the index $k = i + (j-1)I$, for the presented in the figure grid example ($I = J = 3$, $N = 9$) we have $NU = 12$ and the structures of input arrays are shown in Tables 1 and 2.

Table 1. Input arrays F and NE for $I = J = 3$

k	1	2	3	4	5	6	7	8	9
F(k)	2	1	2	1	0	1	2	1	2
NE(k)	2	2	1	2	2	1	1	1	0

Table 2. Input arrays NEIB and AU for $I = J = 3$

l	1	2	3	4	5	6	7	8	9	10	11	12
NEIB(l)	2	4	3	5	6	5	7	6	8	9	8	9
AU(l)	1	1	1	1	1	1	1	1	1	1	1	1

Table 3. The results for $\varepsilon = 10^{-7}$, $\omega = \theta = 1$

$I+1$	16	32	64	128	256	512
EXIFCG	13 $1.7 \cdot 10^{-6}$	19 $2.1 \cdot 10^{-6}$	29 $8.0 \cdot 10^{-7}$	42 $1.2. \cdot 10^{-6}$	63 $9.0 \cdot 10^{-7}$	92 $8.6 \cdot 10^{-7}$
EXIFMR	13 $1.5 \cdot 10^{-6}$	19 $2.0 \cdot 10^{-6}$	28 $2.1 \cdot 10^{-6}$	42 $1.6 \cdot 10^{-6}$	62 $1.9 \cdot 10^{-6}$	90 $2.4 \cdot 10^{-6}$

Table 4. The numbers of iterations for EXIFCG, $\varepsilon = 10^{-7}$, $I + 1 = J + 1 = 256$

$\omega \setminus \theta$	0.0	0.2	0.4	0.6	0.8	0.9	0.97	0.98	0.99	1.0
1.0	187	177	166	151	143	123	93	86	74	63
1.2	154	148	155	145	126	109	84	78	67	63
1.4	140	137	131	123	109	95	76	70	61	63
1.6	111	108	105	101	96	86	68	63	55	63
1.8	79	81	80	78	73	67	57	53	50	63
1.9	61	61	61	60	58	54	50	50	50	63
1.91	58	58	59	58	56	53	50	50	50	63
1.92	57	56	56	55	54	51	50	50	51	63
1.93	56	55	54	54	52	51	50	50	51	63
1.94	55	54	53	53	51	50	50	50	51	63
1.95	54	53	53	52	51	50	50	51	52	63
1.96	55	53	53	52	51	51	51	51	52	63
1.97	55	54	53	53	52	52	52	52	52	63
1.98	58	55	54	53	53	54	56	57	58	63
1.99	74	68	66	65	64	63	63	64	64	63
2.0	112	103	95	92	86	86	80	78	72	63

Table 5. The numbers of iterations for EXIFMR, $\varepsilon = 10^{-7}$, $I+1 = J+1 = 256$

$\omega \setminus \theta$	0.0	0.2	0.4	0.6	0.8	0.9	0.97	0.98	0.99	1.0
1.0	178	170	161	147	126	109	86	85	73	62
1.2	151	145	137	128	120	107	77	76	66	62
1.4	123	121	124	120	107	94	75	69	60	62
1.6	107	107	104	99	91	82	66	61	54	62
1.8	78	78	78	77	72	66	55	52	50	62
1.9	59	59	60	59	57	53	50	49	49	62
1.91	58	57	57	57	55	52	49	49	49	62
1.92	56	56	55	55	53	51	49	49	50	62
1.93	55	54	54	53	52	50	49	49	50	62
1.94	54	53	53	52	51	50	49	50	50	62
1.95	53	52	52	51	50	49	50	50	51	62
1.96	54	52	52	51	50	50	50	50	51	62
1.97	54	52	52	52	51	51	51	51	51	62
1.98	57	54	53	52	52	53	55	55	57	62
1.99	72	66	64	63	62	62	62	62	62	62
2.0	108	100	92	87	84	81	77	75	71	62

In Table 3, we present the output results for the subroutines EXIFCG and EXIFMR for different number of mesh steps $(I+1) \times (J+1) = 16^2$, 32^2, 64^2, 128^2, 256^2, and 512^2. In each cell of this table there are the numbers of iterations n for $\varepsilon = 10^{-7}$, $\omega = \theta = 1$, and resulting errors $\delta = \max_{ij}\{|1 - u_{i,j}^n|\}$.

Here and in the following the computations were done under initial value

$$u^0(x,y) = \left(10 \sin \frac{\pi x}{I+1} \sin \frac{\pi y}{J+1}\right)^2 + 2.$$

In Tables 4 and 5, there are the values of iterations for EXIFCG and EXIFMR under conditions $\varepsilon = 10^{-2}$, $I+1 = J+1 = 256$.

From these results and from similar numerous experiments, the following conclusions can be done:

1. The number of iterations in EXIFMR is slightly less than in EXIFCG and the end error is slightly bigger, but the differences are very small. The computational complexity of both algorithms is approximately the same.
2. The incomplete factorization method for $\omega = \theta = 1$ has approximately the same convergence rate as the SSOR for optimal ω_0 (and $\theta = 0$).
3. The considered methods have high iterative convergence and optimal values of iterative parameters $1 < \omega_0 < 2$, $0.9 < \theta_0 < 1$. But the number of iterations for $\omega = \theta = 1$ is close to optimal one, and these values can be recommended in practice.

References

[1] Axelson O. Iterative Solution Methods. – Cambridge: Univ. Press, 1994.

[2] Il'in V.P. Incomplete Factorization Method for Solution of Algebraic Systems. – Moscow: Nauka, 1995 (in Russian).

[3] Golub G., Loan C. Matrix Computations. – The Johne Hopkins Univ. Press, 1996.

[4] Barrett R., Berry M., et al. Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods. – Philadelphia: SIAM Pub., 1994.

[5] Gololobova S.P., Il'in V.P., Itskovich E.A. FACTOR: the program library for solution of grid equations. – Novosibirsk, 1996. – (Preprint / RAN Sib. Branch of Computing Center; 1079) (in Russian).