On Performance of Iterative Prefactorization Methods for Solving Nonsymmetric Linear Systems

ZBIGNIEW I. WOŹNICKI and HENRYK A. JEDRZEJEC
Institute of Atomic Energy
05–400 Otwock–Świerk, POLAND
GRZEGORZ PUCEK
Warsaw University of Technology, POLAND

Abstract: The main subject of the paper is the study of performance of iterative prefactorization methods for solving linear systems of the type arising from the difference approximation of nonself-adjoint two-dimensional elliptic partial differential equations. The implementation of particular algorithms is demonstrated with solving the system of difference equations indexed by mesh points. Numerical experiments are performed for problems taken from the literature.

Key-Words: Linear equation systems, iterative methods, prefactorization algorithms, convergence analysis.

1 Introduction

Consider the iterative solution of the linear system

$$\boldsymbol{A\phi} = \boldsymbol{c}, \tag{1}$$

where it is assumed that $\boldsymbol{\phi}, \boldsymbol{c} \in \mathbb{R}^n$ and $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is a nonsingular matrix.

A large class of iterative methods for solving Eq.(1) can be formulated by means of the splitting

$$\boldsymbol{A} = \boldsymbol{M} - \boldsymbol{N} \quad \text{with} \quad \boldsymbol{M} \text{ – nonsingular}, \tag{2}$$

and the approximate solution $\boldsymbol{\phi}^{(t)}$ is generated as follows:

$$\boldsymbol{M\phi}^{(t+1)} = \boldsymbol{N\phi}^{(t)} + \boldsymbol{c}, \quad t \geq 0 \tag{3}$$

or equivalently

$$\boldsymbol{\phi}^{(t+1)} = \boldsymbol{\mathcal{V}\phi}^{(t)} + \boldsymbol{M}^{-1}\boldsymbol{c}, \quad t \geq 0, \tag{4}$$

where the starting vector $\boldsymbol{\phi}^{(0)}$ is given and $\boldsymbol{\mathcal{V}} = \boldsymbol{M}^{-1}\boldsymbol{N}$ is the ***iteration matrix***. The above iterative method is convergent to the unique solution

$$\boldsymbol{\phi} = \boldsymbol{A}^{-1}\boldsymbol{c} \tag{5}$$

for each $\boldsymbol{\phi}^{(0)}$ if and only if $\boldsymbol{\varrho(\mathcal{V})} < 1$. The convergence analysis of the above method is based on the spectral radius of the iteration matrix $\boldsymbol{\varrho(\mathcal{V})}$. For large values of t, the solution error decreases in magnitude approximately by factor of $\boldsymbol{\varrho(\mathcal{V})}$ at each iteration step; the smaller is $\boldsymbol{\varrho(\mathcal{V})}$, the quicker is the convergence. The convergence properties of different splittings of $\boldsymbol{A}$ are extensively analized in [5,10].

The evaluation of efficiency of different algorithms is usually performed by comparing the computational work of solutions obtained with the same stopping criterion and initial guess. In the analysis of the reliability of iterative solutions of $\boldsymbol{A\phi} = \boldsymbol{c}$, it is convenient to consider the (***true***) ***error vector***

$$\boldsymbol{e}^{(t)} = \boldsymbol{\phi} - \boldsymbol{\phi}^{(t)}, \tag{6}$$

the ***inner*** (or ***pseudo-residual***) ***error vector***

$$\boldsymbol{\delta}^{(t)} = \boldsymbol{\phi}^{(t+1)} - \boldsymbol{\phi}^{(t)} \tag{7}$$

and the ***residual vector***

$$\boldsymbol{r}^{(t)} = \boldsymbol{A\phi}^{(t)} - \boldsymbol{c}, \tag{8}$$

where $\boldsymbol{\phi} = \boldsymbol{A}^{-1}\boldsymbol{c}$ is assumed as the "exact" solution.

From the viewpoint of the solution reliability in iterative methods based on the splitting of $\boldsymbol{A} = \boldsymbol{M} - \boldsymbol{N}$ [4], it is desired to use the ***relative inner error vector*** $\bar{\boldsymbol{\delta}}^{(t)}$ whose components are defined, as follows

$$\bar{\boldsymbol{\delta}}_i^{(t)} = \frac{\boldsymbol{\phi}_i^{(t)} - \boldsymbol{\phi}_i^{(t-1)}}{\boldsymbol{\phi}_i^{(t)}} \tag{9}$$

and the ***relative*** (***true***) ***error vector*** $\bar{\boldsymbol{e}}^{(t)}$ with components

$$\bar{\boldsymbol{e}}_i^{(t)} = \frac{\boldsymbol{\phi}_i - \boldsymbol{\phi}_i^{(t)}}{\boldsymbol{\phi}_i^{(t)}}. \tag{10}$$

If for any i, $\boldsymbol{\phi}_i^{(t)} = 0$ (or very close to zero) appears in the iteration process, then such a component of $\bar{\boldsymbol{\delta}}^{(t)}$ and $\bar{\boldsymbol{e}}^{(t)}$ is ignored in computations.

In the numerical analysis of matrix splitting iterative methods the termination test

$$\| \bar{\boldsymbol{\delta}}^{(t)} \|_\infty \leq \boldsymbol{\varepsilon} \tag{11}$$

can be practically considered as the most useful stopping criterion independent on the initial guess $\boldsymbol{\phi}^{(0)}$.

Most recent iterative methods terminate when the residual vector $\boldsymbol{r}^{(t)}$ is sufficiently small and the termination test

$$\frac{\| \boldsymbol{r}^{(t)} \|_2}{\| \boldsymbol{r}^{(0)} \|_2} \leq \boldsymbol{\varepsilon} \tag{12}$$

is most commonly used criterion in Krylov subspace algorithms among which the generalized minimal residual algorithm GMRES [1] is considered as one of the most effective iterative methods for solving nonsymmetric linear systems. As is well known, this stopping criterion has the disadvantage of depending too strongly on the initial guess $\boldsymbol{\phi}^{(0)}$. If $\boldsymbol{\phi}^{(0)}$ is very large and very inaccurate, $\| \boldsymbol{r}^{(0)} \|_2$ will be very large and the criterion (12) may stop the iteration too soon. Moreover, as can be seen in numerical experiments for solving systems with nonsymmetric matrices, the stopping criterion (12) used in GMRES algorithms is accompanied by large values of both $\| \boldsymbol{r}^{(t)} \|_2$ and $\| \boldsymbol{e}^{(t)} \|_2$ in comparison to matrix splitting iterative solutions obtained by means of classical and prefactorization algorithms.

Usually the matrix $\boldsymbol{A}$ is defined by the following decomposition:

$$\boldsymbol{A} = \boldsymbol{K} - \boldsymbol{L} - \boldsymbol{U} \tag{13}$$

where $\boldsymbol{K}$, $\boldsymbol{L}$ and $\boldsymbol{U}$ are nonsingular diagonal, strictly lower triangular and strictly upper triangular parts of $\boldsymbol{A}$, respectively. For a large class of matrix problems, arising at the discretization of elliptic partial differential equations, coefficient matrices $\boldsymbol{A}$ are diagonally dominant, and their solutions are mainly obtained by solving a system of difference equations with unknowns oriented in the case of two-dimensional geometry by mesh points (x_n, y_m), where the associated unknown ϕ_n^m depends only on its neighbors coupled by a given difference formula and corresponding matrices $\boldsymbol{A}$ have a sparse and usually regular structure of nonzero entries. Thus, in actual practice solutions are obtained by solving not matrix equations but their difference analogue.

In principle this paper is an abbreviated version of [9] and since there is a limited space, the implementation of particular algorithms is demonstrated in rectangular geometry on the example of the discretized form of two-dimensional elliptic partial differential equations represented by the following five-point difference formula:

$$k_n^m \phi_n^m = c_n^m + e_n^m \phi_n^{m-1} + l_n^m \phi_{n-1}^m + w_n^m \phi_{n+1}^m + \phi_n^{m+1} \tag{14}$$

where the unknowns at mesh points coupled by Eq.(14) are visualized in Fig.1, with $1 \leq m \leq M$ and $1 \leq n \leq N$. The above equation is normalized in such a way that the coefficient at ϕ_n^{m+1} is equal to unity, which allows us to save one multiplication in the iteration process. At this notation the coefficients k are entries of the diagonal matrix $\boldsymbol{K}$, l and e are entries of the strictly lower matrix $\boldsymbol{L}$, and w and 1 are the entries of the strictly upper triangular matrix $\boldsymbol{U}$, forming a five diagonal structure of the matrix $\boldsymbol{A}$.

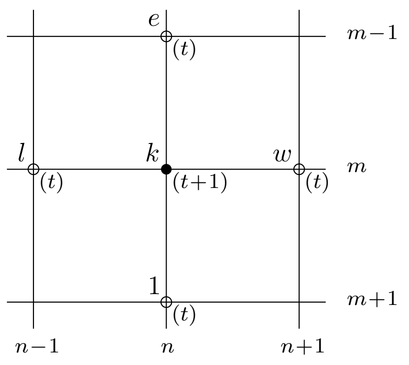

Fig.1

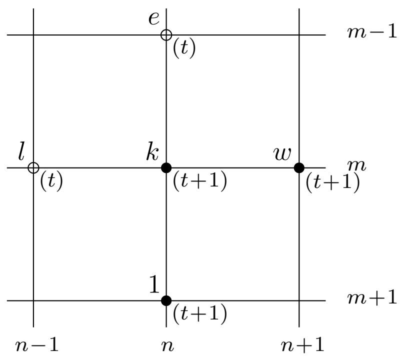

Fig.2

The point Jacobi algorithm is performed by the following formula:

$$(\phi_n^m)^{(t+1)} = \frac{1}{k_n^m}[c_n^m + e_n^m(\phi_n^{m-1})^{(t)} + l_n^m(\phi_{n-1}^m)^{(t)} + w_n^m(\phi_{n+1}^m)^{(t)} + (\phi_n^{m+1})^{(t)}], \tag{15}$$

and the point Gauss-Seidel (GS-II) algorithm by:

$$(\phi_n^m)^{(t+1)} = \frac{1}{k_n^m}[c_n^m + e_n^m(\phi_n^{m-1})^{(t)} + l_n^m(\phi_{n-1}^m)^{(t)} + w_n^m(\phi_{n+1}^m)^{(t+1)} + (\phi_n^{m+1})^{(t+1)}], \tag{16}$$

Both these algoritms are visualized in Figs.1 and 2, respectively, where empty circles correspond to unknows with the iteration index (t) and black circles correspond to unknowns with the iteration index $(t+1)$. The point SOR-II algorithm, derived from the point Gauss-Seidel algorithm is represented by the formula

$$(\phi_n^m)^{(t+1)} = \omega \frac{1}{k_n^m}[c_n^m + e_n^m(\phi_n^{m-1})^{(t)} + l_n^m(\phi_{n-1}^m)^{(t)}$$

$$+w_n^m(\phi_{n+1}^m)^{(t+1)} + (\phi_n^{m+1})^{(t+1)}] - (\omega-1)(\phi_n^m)^{(t)} \quad (17)$$

The notation GS-II or SOR-II means that these algorithms are derived for two-dimensional problems and in the case of three-dimensional problems these algorithms will be denoted by GS-III and SOR-III.

2 Explicit Prefactorization Methods

These methods represent a broad class of algorithms, called briefly as AGA algorithms, are well documented and their computational efficiency found some theoretical support with a convenient matrix notation in comparison theorems [2,3,5]. The implementation of AGA algorithms is especially convenient in the mesh structure of discrete problems as is demonstrated in the below subsections.

2.1 The EWA-II algorithm

In this simplest algorithm of explicit prefactorization methods, we postulate the solution in the form [2,3]

$$\phi_n^m = \frac{\beta_n^m + W_n^m \phi_{n+1}^m + \phi_n^{m+1}}{D_n^m} \quad (18)$$

visualized in Fig.3. Writing the above equation at mesh points $(m-1, n)$ and $(m, n-1)$, i.e.,

$$\phi_n^{m-1} = \frac{\beta_n^{m-1} + W_n^{m-1}\phi_{n+1}^{m-1} + \phi_n^m}{D_n^{m-1}}$$

and

$$\phi_{n-1}^m = \frac{\beta_{n-1}^m + W_{n-1}^m \phi_n^m + \phi_{n-1}^{m+1}}{D_{n-1}^m}$$

and substituting it into Eq.(14), one obtains

$$(k_n^m - E_n^m - L_n^m W_{n-1}^m)\phi_n^m = c_n^m$$
$$+ E_n^m(\beta_n^{m-1} + W_n^{m-1}\phi_{n+1}^{m-1}) + L_n^m(\beta_{n-1}^m + \phi_{n-1}^{m+1})$$
$$+ w_n^m \phi_{n+1}^m + \phi_n^{m+1} \quad (19)$$

where

$$E_n^m = \frac{e_n^m}{D_n^{m-1}} \quad \text{and} \quad L_n^m = \frac{l_n^m}{D_{n-1}^m}. \quad (20)$$

After equating coefficients in Eqs.(18) and (19), we have

$$\beta_n^m = c_n^m + E_n^m(\beta_n^{m-1} + W_n^{m-1}\phi_{n+1}^{m-1}) + L_n^m(\beta_{n-1}^m + \phi_{n-1}^{m+1}) \quad (21)$$

and

$$W_n^m = w_n^m \quad \text{and} \quad D_n^m = k_n^m - E_n^m - L_n^m W_{n-1}^m. \quad (22)$$

Introducing iteration indices, we obtain

$$(\beta_n^m)^{(t+1)} = c_n^m + E_n^m[(\beta_n^{m-1})^{(t+1)} + W_n^{m-1}(\phi_{n+1}^{m-1})^{(t)}]$$
$$+ L_n^m[(\beta_{n-1}^m)^{(t+1)} + (\phi_{n-1}^{m+1})^{(t)}], \quad (23a)$$

$$(\phi_n^m)^{(t+1)} = \frac{(\beta_n^m)^{(t+1)} + W_n^m(\phi_{n+1}^m)^{(t+1)} + (\phi_n^{m+1})^{(t+1)}}{D_n^m} \quad (23b)$$

which are the executive equations in the **EWA-II** algorithm being a particular case of the AGA method. The values of $(\beta_n^m)^{(t+1)}$ are computed recursively for increasing indices (line by line from top to bottom) in the ***forward sweep*** represented by Eq.(23a), and the values of $(\phi_n^m)^{(t+1)}$ are computed recursively for decreasing indices (line by line from bottom to top) in the ***backward sweep*** represented by Eq.(23b).

The overrelaxation procedure can be used as follows

$$(\beta_n^m)^{(t+1)} = \omega_\beta\{c_n^m + E_n^m[(\beta_n^{m-1})^{(t+1)}$$
$$+W_n^{m-1}(\phi_{n+1}^{m-1})^{(t)}] + L_n^m[(\beta_{n-1}^m)^{(t+1)} + (\phi_{n-1}^{m+1})^{(t)}]\}$$
$$-(\omega_\beta - 1)(\beta_n^m)^{(t)}, \quad (24a)$$

$$(\phi_n^m)^{(t+1)} = \omega_\phi \frac{(\beta_n^m)^{(t+1)} + W_n^m(\phi_{n+1}^m)^{(t+1)} + (\phi_n^{m+1})^{(t+1)}}{D_n^m}$$
$$- (\omega_\phi - 1)(\phi_n^m)^{(t)}. \quad (24b)$$

Dependently on the values of the parameters ω_β and ω_ϕ, we have the following cases [2,3]:
– the **EWA-II *forward* SOR** where $\omega_\beta \neq 1$ and $\omega_\phi = 1$,
– the **EWA-II *backward* SOR** where $\omega_\beta = 1$ and $\omega_\phi \neq 1$,
– the **EWA-II *double* SOR** where $\omega_\beta \neq 1$ and $\omega_\phi \neq 1$.

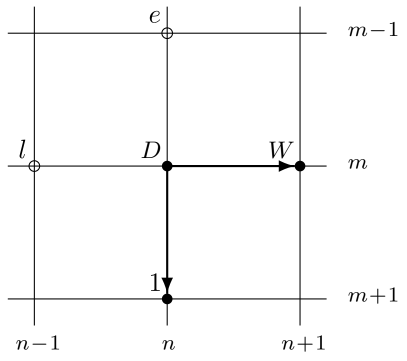

Fig.3

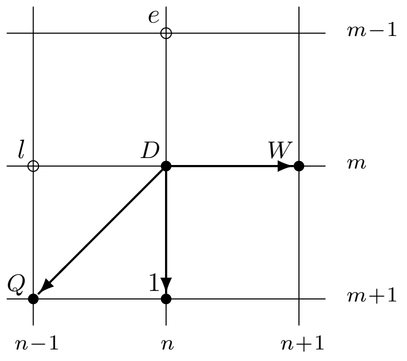

Fig.4

2.2 The AGA-II-A algorithm

In this algorithm opening a large class of algorithms the AGA algorithms, the equation of the backward sweep is the same as that for EWA-II given in Eq.(18) and it differs from EWA-II by the elimination of ϕ_{n+1}^{m-1} in Eq.(21). Rewriting Eq.(18) at the mesh point (m-1,n+1) and substituting it into Eq.(21), we obtain the equation which, after reordering and equating to Eq.(18), and introducing iteration indices, provides us

$$(\beta_n^m)^{(t+1)} = c_n^m + E_n^m[(\beta_n^{m-1})^{(t+1)} + G_n^m[(\beta_n^{m-1})^{(t+1)} + W_{n+1}^{m-1}(\phi_{n+2}^{m-1})^{(t)}]] + L_n^m[(\beta_{n-1}^m)^{(t+1)} + (\phi_{n-1}^{m+1})^{(t)}], \tag{25}$$

where

$$E_n^m = \frac{e_n^m}{D_n^{m-1}},\ L_n^m = \frac{l_n^m}{D_{n-1}^m},\ G_n^m = \frac{W_n^{m-1}}{D_{n+1}^{m-1}},$$
$$W_n^m = w_n^m + E_n^m G_n^m \text{ and } D_n^m = k_n^m - E_n^m - L_n^m W_{n-1}^m. \tag{26}$$

The equations (25) and (24a) are the executive equations in the **AGA-II-A** algorithm. Thus, this algorithm differs from EWA-II by the equation of the forward sweep and the increased value of the coefficients W_n^m. The overrelaxation procedure can be used in a similar way as in the case of EWA-II, providing the AGA-II-A forward SOR, AGA-II-A backward SOR and AGA-II-A double SOR algorithms.

2.3 The AGA-II-B algorithm

We can postulate the backward sweep in the form

$$\phi_n^m = \frac{\beta_n^m + W_n^m\phi_{n+1}^m + Q_n^m\phi_{n-1}^{m+1} + \phi_n^{m+1}}{D_n^m} \tag{27}$$

involving an additional coupling with mesh point $(m+1, n-1)$ as is visualized in Fig.4. Rewriting the above equation at mesh points $(m-1, n)$ and $(m, n-1)$ and substituting it into (14), eliminating firstly ϕ_n^{m-1} and next ϕ_{n-1}^m; after reordering and equating the coefficients of the obtained equation to those of (27), the following equations can be derived.

$$\beta_n^m = c_n^m + E_n^m(\beta_n^{m-1} + W_n^{m-1}\phi_{n+1}^{m-1}) + Q_n^m(\beta_{n-1}^m + Q_{n-1}^m\phi_{n-1}^{m+1}) \tag{28}$$

and

$$E_n^m = \frac{e_n^m}{D_n^{m-1}},\ Q_n^m = \frac{l_n^m + E_n^m Q_n^{m-1}}{D_{n-1}^m},$$
$$W_n^m = w_n^m \text{ and } D_n^m = k_n^m - E_n^m - Q_n^m W_{n-1}^m. \tag{29}$$

Introducing iteration indices, the equations

$$(\beta_n^m)^{(t+1)} = c_n^m + E_n^m[(\beta_n^{m-1})^{(t+1)} + W_n^{m-1}(\phi_{n+1}^{m-1})^{(t)}] + Q_n^m[(\beta_{n-1}^m)^{(t+1)} + Q_{n-1}^m(\phi_{n-2}^{m+1})^{(t)}], \tag{30a}$$

$$(\phi_n^m)^{(t+1)} = \tfrac{1}{D_n^m}[(\beta_n^m)^{(t+1)} + W_n^m(\phi_{n+1}^m)^{(t+1)} + Q_n^m(\phi_{n-1}^{m+1})^{(t+1)} + (\phi_n^{m+1})^{(t+1)}] \tag{30b}$$

are the executive equations in the **AGA-II-B** algorithm.

The eliminating ϕ_{n+1}^{m-1} in Eq.(28), provides us the version of this algorithm called the **AGA-II-B1**, differing from AGA-II-B only by the formula of the forward sweep.

2.4. Other AGA-II algorithms

As is demonstrated in [9] the subsequent algorithms of the AGA method are created by involving the successive mesh points on lines m and $m-1$ to the recurrence formula of the backward sweep. The application of the double SOR procedure turned out to be a most efficient technique for the convergence acceleration in AGA algorithms, where the value of optimum relaxation parameters can be obtained by means of the OMEST procedure described in detail in [7,8,9].

3 Semi-explicit Prefactorization Methods with an Implicit Forward Sweep

This type of prefactorization methods, introduced recently [9], is a new class of very efficient algorithms called OLA algorithms. Their creation in mesh structures is a simple task, allowing us to clearly present the mechanism of construction of these algorithms, as is demonstrated below in the example of the OLA-II-(1,1) algorithm implemented in rectangular geometry represented by the unnormalized five-point difference formula:

$$k_n^m\phi_n^m = c_n^m + e_n^m\phi_n^{m-1} + l_n^m\phi_{n-1}^m + w_n^m\phi_{n+1}^m + u_n^m\phi_n^{m+1} \tag{31}$$

In these algorithms, the formula of the backward sweep is coupling with some mesh points located only on the mesh line $m+1$.

3.1 The OLA-II-(1,1) algorithm

In this algorithm, we postulate the following formula for the backward sweep:

$$\phi_n^m = \frac{\beta_n^m + \psi_n^m + T_n^m\phi_{n-1}^{m+1} + U_n^m\phi_n^{m+1} + V_n^m\phi_{n+1}^{m+1}}{D_n^m} \tag{32}$$

which mesh point coupling is visualized in Fig.5.

Rewriting the above equation at the mesh point $(m-1, n)$ and substituting it into (31), we obtain

$$D_n^m\phi_n^m = c_n^m + E_n^m(\beta_n^{m-1} + \psi_n^{m-1}) + (l_n^m + E_n^m T_n^{m-1})\phi_{n-1}^m + (w_n^m + E_n^m V_n^{m-1})\phi_{n+1}^m + u_n^m\phi_n^{m+1} \tag{33}$$

where

$$E_n^m = \frac{e_n^m}{D_n^{m-1}},\ \text{ and }\ D_n^m = k_n^m - E_n^m U_n^{m-1}.$$

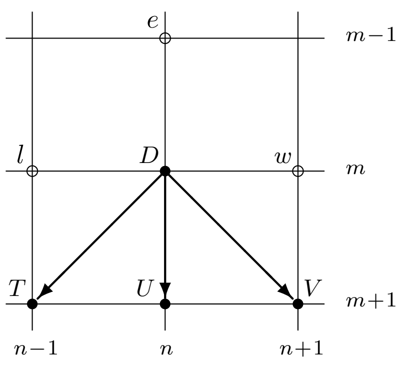

Fig.5

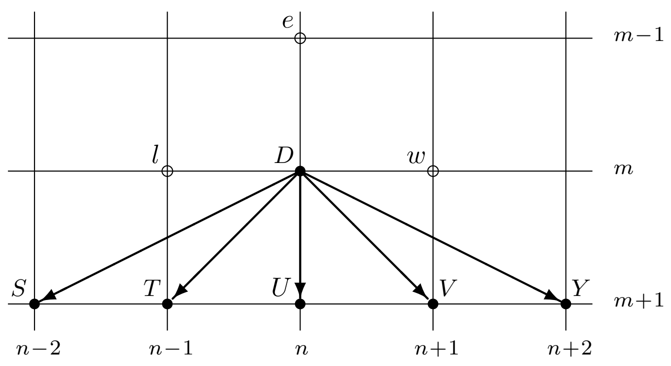

Fig.6

Rewriting Eq.(32) at the mesh points $(m, n-1)$ and $(m, n+1)$ and substituting it into (33) we obtain the equation which after reordering and equating to (32) provides us the coefficients

$$L_n^m = \frac{l_n^m + E_n^m T_n^{m-1}}{D_{n-1}^m}, \quad W_n^m = \frac{w_n^m + E_n^m V_n^{m-1}}{D_{n+1}^m},$$
$$U_n^m = u_n^m + L_n^m V_{n-1}^m + W_n^m T_{n+1}^m,$$
$$T_n^m = L_n^m U_{n-1}^m, \quad V_n^m = W_n^m U_{n+1}^m$$

and the recursive formulas for

$$\psi_n^m = L_n^m (T_{n-1}^m \phi_{n-2}^{m+1} + \psi_{n-1}^m) + W_n^m (V_{n+1}^m \phi_{n+2}^{m+1} + \psi_{n+1}^m) \tag{34}$$

and

$$\beta_n^m = c_n^m + E_n^m (\beta_n^{m-1} + \psi_n^{m-1}) + L_n^m \beta_{n-1}^m + W_n^m \beta_{n+1}^m. \tag{35}$$

As can be seen, when the coefficients are computed for increasing indices m and n, in formulas for W_n^m, U_n^m and V_n^m appear D_{n+1}^m, T_{n+1}^m and U_{n+1}^m, respectively, whose values are not determined yet. This difficulty can be omitted by computing these coefficients for the mesh point (m,n-1). For the coefficient U, we have

$$U_{n-1}^m = u_{n-1}^m + L_{n-1}^m V_{n-2}^m + W_{n-1}^m T_n^m$$

and after substituting V_{n-2}^m and T_n^m, we obtain

$$U_{n-1}^m = \frac{u_{n-1}^m}{1 - L_{n-1}^m W_{n-2}^m - W_{n-1}^m L_n^m}.$$

Thus, for whole iteration process coefficients are simultaneously computed, for each pair of indices (m, n) from top line to bottom line for increasing values $1 \leq n \leq N$, according to the following order:

$$\left.\begin{array}{l} E_n^m = \frac{e_n^m}{D_n^{m-1}}, \ D_n^m = k_n^m - E_n^m U_n^{m-1}, \\ L_n^m = \frac{l_n^m + E_n^m T_n^{m-1}}{D_{n-1}^m}, \\ W_{n-1}^m = \frac{w_{n-1}^m + E_{n-1}^m V_{n-1}^{m-1}}{D_n^m}, \\ T_{n-1}^m = L_{n-1}^m U_{n-2}^m, \\ U_{n-1}^m = \frac{u_{n-1}^m}{1 - L_{n-1}^m W_{n-2}^m - W_{n-1}^m L_n^m}, \\ V_{n-2}^m = W_{n-2}^m U_{n-1}^m. \end{array}\right\} \tag{36}$$

As can be noticed Eqs.(34) and (35) are three point formulas for line m, whose can be easily solved by postulating their backward solutions. In the case of Eq.(34), we can postulate the backward solution

$$\psi_n^m = \frac{\eta_n^m + W_n^m \psi_{n+1}^m}{P_n^m} \tag{37}$$

and rewriting the above equation at the mesh point $(m, n-1)$ and substituting it into (34), we have

$$\eta_n^m = L_n^m (T_{n-1}^m \phi_{n-2}^{m+1} + \frac{\eta_{n-1}^m}{P_n^m}) + W_n^m V_{n+1}^m \phi_{n+2}^{m+1} \tag{38}$$

and

$$P_n^m = 1 - \frac{L_n^m W_{n-1}^m}{P_{n-1}^m} \tag{39}$$

where $\eta_1^m = W_1^m V_2^m \phi_3^{m+1}$, $\psi_N^m = \frac{\eta_N^m}{P_N^m}$ and the values of P_n^m are computed with $P_1^m = 1$ for whole iteration process. In a similar way, Eq.(35) can be solved by postulating the backward solution

$$\beta_n^m = \frac{\gamma_n^m + W_n^m \beta_{n+1}^m}{P_n^m} \tag{40}$$

providing us

$$\gamma_n^m = c_n^m + E_n^m (\beta_n^{m-1} + \psi_n^{m-1}) + \frac{L_n^m \gamma_{n-1}^m}{P_n^m}. \tag{41}$$

<u>Iteration process</u>

In this algorithm, the overrelaxation process can be used in different ways but its use only to Eqs.(40) and (41), as the single or double SOR, provides best results.

At the beginning of iteration process, $(\beta_n^m)^{(0)} = 0$ and $(\gamma_n^m)^{(0)} = 0$ are assumed for all mesh points, and starting values of $(\psi_n^m)^{(0)}$ are computed for a given initial guess $(\phi_n^m)^{(0)}$ by means of Eqs.(38) and (37).

The iterative algorithm is executed according to the following formulas.

– For all $m = 1, 2, \ldots, M$,

$$(\gamma_n^m)^{(t+1)} = \omega_\gamma [c_n^m + E_n^m ((\beta_n^{m-1})^{(t+1)} + (\psi_n^{m-1}))^{(t)}) + \frac{L_n^m (\gamma_{n-1}^m))^{(t+1)}}{P_n^m}] - (\omega_\gamma - 1)(\gamma_n^m)^{(t)} \tag{42}$$

and

$$(\beta_n^m)^{(t+1)} = \omega_\beta \frac{(\gamma_n^m)^{(t+1)} + W_n^m (\beta_{n+1}^m)^{(t+1)}}{P_n^m} - (\omega_\beta - 1)(\beta_n^m)^{(t)}. \tag{43}$$

– For $m = M-1, M-2, \ldots, 1$,

$$(\eta_n^m)^{(t+1)} = L_n^m [T_{n-1}^m (\phi_{n-2}^{m+1})^{(t+1)} + \frac{(\eta_{n-1}^m)^{(t+1)}}{P_n^m}] + W_n^m V_{n+1}^m (\phi_{n+2}^{m+1})^{(t+1)}, \quad (44)$$

$$(\psi_n^m)^{(t+1)} = \frac{(\eta_n^m)^{(t+1)} + W_n^m (\psi_{n+1}^m)^{(t+1)}}{P_n^m} \quad (45)$$

and

$$(\phi_n^m)^{(t+1)} = \frac{1}{D_n^m}[(\beta_n^m)^{(t+1)} + (\psi_n^m)^{(t+1)} + T_n^m (\phi_{n-1}^{m+1})^{(t+1)} + U_n^m (\phi_n^{m+1})^{(t+1)} + V_n^m (\phi_{n+1}^{m+1})^{(t+1)}]. \quad (46)$$

– If the convergence criterion

$$(\bar{\boldsymbol{\delta}}_n^m)^{(t+1)} = \left| \frac{(\phi_n^m)^{(t+1)} - (\phi_n^m)^{(t)}}{(\phi_n^m)^{(t+1)}} \right| \le \varepsilon \quad (47)$$

is not satisfied for all mesh points, the iteration process is continued.

A priori estimate of optimum values of ω_γ and ω_β can be obtained by means of the OMEST procedure [7,8,9].

3.2 Other OLA-II-(l, r) algorithms

Subsequent OLA-II-(l, r) algorithms can be derived in a similar way by involving other mesh points located on line $m+1$, where l denotes the number of mesh points at indices $n-1, n-2, \ldots, n-l$ and r denotes the number of mesh points at indices $n+1, n+2, \ldots, n+r$ coupled by the backward sweep formula.

In the simplest algorithm called OLA-II-(0,0), the backward sweep formula has the form

$$\phi_n^m = \frac{\beta_n^m + \psi_n^m + U_n^m \phi_n^{m+1}}{D_n^m}. \quad (48)$$

For OLA-II-(0,1) the term $T_n^m \phi_{n-1}^{m+1}$ does not exist in Eq.(32) and for OLA-II-(1,0) the term $V_n^m \phi_{n+1}^{m+1}$ does not exist in Eq.(32). The backward sweep formula in the **OLA-II-(2,2)** algorithm is visualized in Fig.6.

Finally, it should be mentioned that similar AGA and OLA algorithms can be derived in the rectangular geometry for nine-point difference formula of higher order of approximation or for reduced system represented also by (diamond) nine-point difference formula as well as in triangular or hexagonal two- and three-dimensional geometries.

4 Numerical Experiments

In this section we demonstrate the performance of discussed prefactorization algorithms for solving several noself-adjoint elliptic problems, used in the literature mainly for testing the efficiency of the GMRES algorithm, and descibed in detail in [7,8,9].

Example 4.1

This problem taken from [1] is the variable coefficient convection-diffusion equation

$$\left.\begin{array}{rcl} -\Delta\phi + (d\phi)_x + (e\phi)_y &=& f \text{ in } \Omega \\ \phi &=& 0 \text{ on } \partial\Omega \end{array}\right\} \quad (49)$$

in the unit square $\Omega = (0,1) \times (0,1)$ with boundary $\partial\Omega$ and f is a function defined on Ω, where

$$d(x,y) = \gamma(x+y) \text{ and } e(x,y) = \gamma(x-y). \quad (50)$$

It is assumed that $\gamma = 10$, the first derivative terms are approximated by means of centered differences [9] and the number of interior mesh points in each direction equal to $N = 32$, so that the order of five-diagonal nonsymmetric matrix is equal to $32 \times 32 = 1024$. The right-hand side of the matrix equation is generated as

$$\boldsymbol{c} = \boldsymbol{A}\boldsymbol{e}, \quad (51)$$

where $\boldsymbol{e}^T = [1, 1, \ldots, 1]$ which implies that the solution vector $\boldsymbol{\phi}$ is known in advance and all its components are equal to unity in the interior of Ω. The results of computations and computational work expressed by total numbers of flops, obtained in [8,9] for SLOR (line SOR) and several prefactorization algorithms, and shown in Table 1, are compared with the results of GMRES(s) with different preconditionings [1]. All results presented in the first row for each GMRES algorithm were obtained for the criterion $\| \boldsymbol{r}^{(t)} \|_2 / \| \boldsymbol{r}^{(0)} \|_2 < 10^{-07}$. GMRES(10) with ILUT(5,$10^{-4}$) provides the minimum number of iterations and flops but with greatest values of both errors and minimum values of both errors are obtained for GMRES(20) however, with six times greater number of flops. The SLOR-II, OLA-II-(1,1) and OLA-II-(2,2) algorithms provide similar results as those of GMRES(20) but with three times lesser number of flops.

As can be seen in Table 1, the continuation of iteration processes to about two times greater number of iterations in SLOR and prefactorization algorithms provides solutions with $\| \boldsymbol{r}^{(t)} \|_2 \approx 10^{-13}$ and $\| \boldsymbol{e}^{(t)} \|_2 \approx 10^{-12}$ but the total number of flops increases about 40% for SLOR and 20% for prefactorization algorithms.

Example 4.2

This problem, taken from [8], is represented by the following partial differential equation

$$-(b\phi_x)_x - (c\phi_y)_y + d\phi_x + (d\phi)_x + e\phi_y + (e\phi)_y + f\phi = g \quad (52)$$

on the unit square $\Omega = (0,1) \times (0,1)$, with Dirichlet boundary conditions $\phi = 0$ on $\partial\Omega$, where $b(x,y) = e^{-xy}$, $c(x,y) = e^{xy}$, $d(x,y) = \beta(x+y)$, $e(x,y) = \gamma(x+y)$ and $f(x,y) = \frac{1}{1+x+y}$.

Table 1. *The results for Example 4.1.*

Algorithm	Preconditioning or relax. paramets.	Iters	Kflops	$\Vert \boldsymbol{r}^{(t)} \Vert_2$	$\Vert \boldsymbol{e}^{(t)} \Vert_2$	Quotation
GMRES(10)	ILUT(1,10^{-4})	18	964	0.47×10^{-03}	0.41×10^{-04}	p.293 in [1]
GMRES(10)	ILUT(5,10^{-4})	7	478	0.13×10^{-02}	0.90×10^{-04}	p.294 in [1]
GMRES	ILUTP(1)	18	964	0.47×10^{-03}	0.41×10^{-04}	p.295 in [1]
GMRES(20) with polynomial preconditioning	ILU(0)	56	2774	0.22×10^{-05}	0.51×10^{-06}	p.364 in [1]
SLOR-II	$\omega = 1.6709$	57	931	0.93×10^{-06}	0.26×10^{-05}	[8]
	(44iters, 360Kflops)	62	995	0.19×10^{-07}	0.44×10^{-06}	
		100	1327	0.92×10^{-13}	0.28×10^{-12}	
AGA-II-B1	$\omega_\beta = \omega_\phi = 1.092$	22	1106	0.11×10^{-05}	0.13×10^{-04}	—
	(68iters, 836Kflops)	27	1167	0.53×10^{-07}	0.27×10^{-06}	
		49	1438	0.94×10^{-13}	0.10×10^{-12}	
OLA-II-(1,1)	$\omega_\beta = 1$, $\omega_\gamma = 1.156$	17	1382	0.10×10^{-05}	0.17×10^{-04}	
	(58iters, 1069Kflops)	20	1401	0.51×10^{-07}	0.64×10^{-06}	
		33	1677	0.99×10^{-13}	0.41×10^{-12}	
	$\omega_\beta = \omega_\gamma = 1$	59	967	0.10×10^{-05}	0.26×10^{-04}	
OLA-II-(2,2)	$\omega_\beta = 1$, $\omega_\gamma = 1.090$	14	1677	0.44×10^{-06}	0.13×10^{-05}	
	(45iters, 1278Kflops)	28	2050	0.34×10^{-13}	0.69×10^{-13}	
	$\omega_\beta = \omega_\gamma = 1$	33	811	0.11×10^{-05}	0.26×10^{-04}	
		42	1032	0.19×10^{-07}	0.46×10^{-06}	
		70	1720	0.70×10^{-13}	0.16×10^{-11}	

Table 2. *The results for Example 4.2 with 324 mesh points.*

Algorithm	Iters	Kflops	$\Vert \bar{\boldsymbol{\delta}}^{(t)} \Vert_\infty$	$\Vert \boldsymbol{r}^{(t)} \Vert_2 / \Vert \boldsymbol{r}^{(0)} \Vert_2$	$\Vert \boldsymbol{r}^{(t)} \Vert_2$	$\Vert \boldsymbol{e}^{(t)} \Vert_2$
SLOR-II	57	577	0.31×10^{-03}	0.73×10^{-06}	0.22×10^{-02}	0.12×10^{-05}
$\omega = 0.65$	74	627	0.81×10^{-06}	0.40×10^{-08}	0.12×10^{-04}	0.65×10^{-08}
(141it., 411Kfl.)	121	764	0.56×10^{-12}	0.34×10^{-14}	0.10×10^{-10}	0.59×10^{-14}
AGA-II-B1	7	38	0.48×10^{-01}	0.62×10^{-06}	0.18×10^{-02}	0.19×10^{-05}
$\omega_\beta = \omega_\phi = 1$	10	47	0.30×10^{-06}	0.29×10^{-11}	0.85×10^{-08}	0.73×10^{-11}
(6it., 17Kfl.)	13	55	0.55×10^{-12}	0.55×10^{-15}	0.16×10^{-11}	0.18×10^{-14}
OLA-II-(1,1)	6	61	0.26×10^{-01}	0.37×10^{-06}	0.11×10^{-02}	0.13×10^{-05}
$\omega_\beta = \omega_\gamma = 1$	9	77	0.19×10^{-06}	0.27×10^{-11}	0.79×10^{-08}	0.59×10^{-11}
(5it., 28Kfl.)	12	94	0.53×10^{-12}	0.80×10^{-15}	0.24×10^{-11}	0.24×10^{-14}
OLA-II-(2,2)	4	57	0.38×10^{-01}	0.10×10^{-06}	0.31×10^{-03}	0.76×10^{-06}
$\omega_\beta = \omega_\gamma = 1$	6	73	0.61×10^{-07}	0.19×10^{-12}	0.57×10^{-09}	0.58×10^{-12}
(3it., 24Kfl.)	8	89	0.60×10^{-12}	0.84×10^{-15}	0.25×10^{-11}	0.27×10^{-14}

The equation (52) is discretized by using the five-point approximation, where the first derivative terms are approximated by the scheme of centered differences [9], in the square mesh with the mesh size $h = 1/(N+1)$. It is assumed that $\beta = -20$, $\gamma = 50$ and $N = 18$ which provides the matrix of order $s = 324$. The right-hand side g is chosen so that the solution is known to be $xe^{xy}\sin(\pi x)\sin(\pi y)$.

The results of computations are summarized in Table 2. This test problem was originally used by Saad and Schultz (see Reference 11 in [8] or Reference 186 in [1]) for comparing the performance of GMRES with other conjugate-like methods, with using the stopping criterion $\Vert \boldsymbol{r}^{(t)} \Vert_2 / \Vert \boldsymbol{r}^{(0)} \Vert_2 < 10^{-06}$, and it was recognized by the authors as an example more difficult to treat. As is demonstrated by Saad and Schultz, for finding the GMRES(20) solution with 300Kflops, ten or more times greater arithmetical effort was consumed whereas, a similar solution can be obtained by means of AGA-II-B1 with only 38Kflops.

In both above examples, an arithmetical effort required for computing optimum relaxation parameters is shown separately in Tables 1 and 2, and it is included to the total computational work expessed by Kflops.

Example 4.3
This problem, represented by the following partial differential equation

$$-\Delta\phi + \sigma(1-2x)\phi_x + \tau(1-2y)\phi_y = 0 \qquad (53)$$

on the unit square $\Omega = (0,1) \times (0,1)$, with Dirichlet boundary conditions $\phi = 0$ on $\partial\Omega$, was used among other problems for examining the performance of the modified line Gauss-Seidel algorithms in [7] and the SLOR algorithm in [8] with using $h = 1/32$ and several choices of σ and τ. For the approximation of the first derivative terms, the upwind difference scheme was used [9].

Comparison of computed spectral radii, with relaxation parameters equal to unity is shown in Table 3. All algorithms are convergent for $\sigma = \tau = 20$ and $\sigma = \tau = 40$. For $\sigma = \tau = 60$, the principal eigenvalue of the iteration matrix $\bar{\mathcal{L}}_1$, $\lambda_1 = -7.222$ in the line Gauss-Seidel algorithm but the remaining algorithms are convergent. However, as is shown in [8] the SLOR algorithm is convergent for some values of $\omega < 1$ and its spectral radius achieves the minimal value equal to 0.929 when $\omega_{best} = 0.537$ but its solution (without including computational work requested for determining ω_{best}) can be obtained with the number of flops nine times greater in comparison to the OLA-II-(2,2) solution with $\omega_\gamma = \omega_\beta = 1$.

For $\sigma = \tau = 80$, all algorithms presented in Table 3 are divergent and only the spectral radius of OLA-II-(2,2) achieves its minimum equal to about 0.52 for $\omega_\beta = \omega_\gamma \approx 0.94$. Since the solution of (53) is the null vector, then the stopping test $\| \boldsymbol{\phi}^{(t)} \|_\infty \leq \varepsilon$ can be considered as the most reliable measure of the error vector. Assuming that all components of starting vector $\boldsymbol{\phi}^{(0)}$ are equal to unity, OLA-II-(2,2) with $\omega_\beta = \omega_\gamma = 0.94$ provides the solution after 23 iterations (equivalent to 659Kflops) for $\varepsilon \leq 10^{-6}$ and after 44 iterations (equivalent to 1262Kflops) for $\varepsilon \leq 10^{-12}$.

References:

[1] Y. Saad, *Iterative methods for sparse linear systems*, PWS Pub. Co., Boston, MA, 1996.

[2] Z.I.Woźnicki, AGA two–sweep iterative method and their application in critical reactor calculations, *Nukleonika*, Vol.9, 1978, pp.941–968.

[3] Z.I.Woźnicki, Estimation of the optimum relaxation factors in the partial factorization iterative methods, *SIAM J. Matrix Anal. Appl.*, 14, 1993, pp.59–73.

[4] Z.I.Woźnicki, On numerical analysis of conjugate gradient method, *Japan J. Industr. Appl. Math.*, 10, 1993, pp.487–519.

[5] Z.I.Woźnicki, Nonnegative splitting theory, *Japan J. Industr. Appl. Math.*, 11, 1994, pp. 289–342.

[6] Z.I.Woźnicki, The Sigma-SOR algorithm and the optimal strategy for utilization of the SOR iterative method, *Mathematics of Computation*, 62(206), 1994, pp.619–644.

[7] Z.I.Woźnicki and H.A.Jedrzejec, A new class of modified line-SOR algorithms, *J. Comput. Appl. Math.*, 131, 2001, 89–142.

[8] Z.I.Woźnicki, On performance of SOR method for solving nonsymmetric linear systems, *J. Comput. Appl. Math.*, 137, 2001, pp.145–176.

[9] Z.I.Woźnicki, Matrix Splitting Iterative Methods and their Implementation in Mesh Structures, *Proc. Summer School on Iterative Methods and Matrix Computations.*, June 2–9, 2002, Rostov–on–Don, Russia. (http://conf.rsu.ru/immc02)

[10] Z.I.Woźnicki, Matrix splitting principles, *Inter.J.Math.Math.Sci.*, 28, 251–284, 2002.

Table 3. *Spectral radii of iteration matrices in Example 4.3 with relaxation parameters equal to unity.*

$\sigma = \tau$	SLOR-II	AGA-II-B1	OLA-II-(1,1)	OLA-II-(2,2)
20	0.925	0.718	0.619	0.438
40	0.864	0.550	0.427	0.247
60	7.222	0.433	0.309	0.154
80	314.	1.2	1.4	1.1