Applied Mathematics, 2016, 7, 1225-1240
Published Online July 2016 in SciRes. http://www.scirp.org/journal/am
http://dx.doi.org/10.4236/am.2016.711108

Scientific Research Publishing

A Class of Generalized Approximate Inverse Solvers for Unsymmetric Linear Systems of Irregular Structure Based on Adaptive Algorithmic Modelling for Solving Complex Computational Problems in Three Space Dimensions

Anastasia-Dimitra Lipitakis

Department of Informatics and Telematics, Harokopio University, Athens, Greece
Email: adlipita@hua.gr

Received 18 May 2016; accepted 9 July 2016; published 12 July 2016

Abstract

A class of general inverse matrix techniques based on adaptive algorithmic modelling methodologies is derived yielding iterative methods for solving unsymmetric linear systems of irregular structure arising in complex computational problems in three space dimensions. The proposed class of approximate inverse is chosen as the basis to yield systems on which classic and preconditioned iterative methods are explicitly applied. Optimized versions of the proposed approximate inverse are presented using special storage (k-sweep) techniques leading to economical forms of the approximate inverses. Application of the adaptive algorithmic methodologies on a characteristic nonlinear boundary value problem is discussed and numerical results are given.

Keywords

Adaptive Algorithms, Algorithmic Modelling, Approximate Inverse, Incomplete LU Factorization, Approximate Decomposition, Unsymmetric Linear Systems, Preconditioned Iterative Methods, Systems of Irregular Structure

1. Introduction

In recent years, extensive research work has been focused on the computation of exact and approximate inverse matrices for solving efficiently complex computational problems particularly on parallel computer systems [1]-[20]. In this article, a new class of Sparse Approximate Inverses matrices based on adaptive algorithmic modelling methods is presented. These adaptive algorithmic solution methods can be used for solving large sparse linear finite difference (FD) and finite element (FE) systems of irregular structures derived mainly from the discretization of parabolic and elliptic PDE's in both two and three space dimensions [21]-[31].

Let us consider a class of boundary value problems defined by the equation

$$-\varepsilon_p \sum_{i,j=1}^{N} \left\{ \partial \left[a_{i,j}(x) \partial u(x) / \partial x_j \right] / \partial x_i \right\} - \sum_{j=1}^{N} \left\{ b_j(x) \partial u(x) / \partial x_j \right\} + c(x) u(x) = f(x),\ x \in D < R^N \tag{1.1}$$

subject to the general boundary conditions

$$\alpha(x) u + \beta(x) \partial u / \partial \zeta = \gamma(x),\ \ x \in \partial D, \tag{1.1a}$$

where D is a closed bounded domain in R^N, with $N \le 3$ and ∂D the boundary of D, ε_p a predetermined singular perturbation parameter, $a_{i,j}(x) > 0, b_j(x) > 0, c(x) \ge 0$; and the coefficients $a_{i,j}, b_j, c$ are continuous and differentiable functions in D, $\alpha(x) > 0, \beta(x) > 0, a_{i,j}, b_j, c$; f are sufficiently smooth functions on D and $\partial \zeta$ is the direction of the outward normal derivative.

The discrete analogue of Equation (1.1) leads to the solution of the general linear system

$$Au = s\ , \tag{1.2}$$

where the coefficient matrix A is a large sparse real ($n \times n$) matrix of irregular structure. The structure of A is shown in the following **Figure 1**.

For solving the system (1.2), there is a choice between direct and iterative, assuming that there are no barriers due to memory requirements for the former or excessive runtimes (e.g. time dependent problems) for the latter. Note that for generality purposes the coefficient matrix is assumed to be unsymmetric (case occurring in the discretization of flow equations that arise in certain Hydrology studies [32] and of irregular non-zero structure (case resulting from the triangulation of irregular or regular domains into irregular elements in pipeline networks [33]. Algorithmic solution methods for the linear systems (1.2) applicable to both two and three space dimensions can be applied [22], where the unsymmetric coefficient matrix, in which all the off-centre terms are grouped into regular bands, can be factorized exactly to yield direct algorithmic procedures for the FD or FE solution.

It should be noted that in the case of very large sparse linear and nonlinear systems with coefficients of irregular structure, the memory requirements and the corresponding computational work are prohibitively high and the use of exact inverse solvers is usually not recommended. In such cases, preconditioned iterative techniques for solving numerically the FD or FE linear systems (1.2) can be used by deriving semi-direct solution methods following the principle [27] [34] that implicit procedures based on approximately decomposing discrete operators into easily invertible factors facilitating the solution of (1.2). Sparse factorization procedures yield efficient

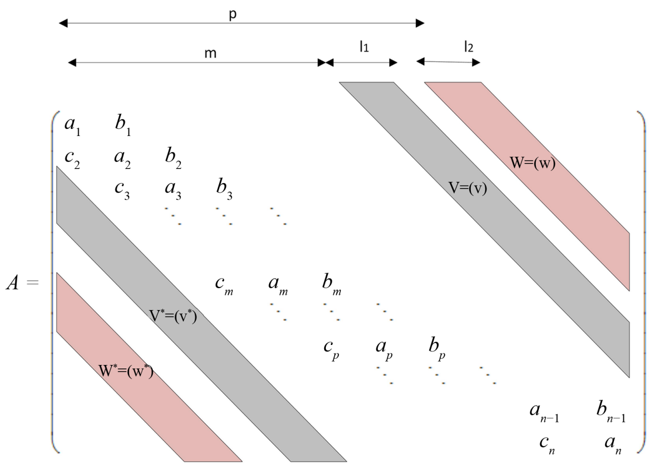

Figure 1. The structure of coefficient matrix *A*.

procedures for the FE or FD solution by manipulating the problem of the fill-in terms, which occur during the factorization [22] [35]. Note that simple compact storage schemes for the considered data can be used and preconditioned algorithmic solvers do not require any searching operations. An important feature of the proposed adaptive algorithmic methods is the provision of a class of iterative methods for solving large sparse unsymmetric systems of irregular non-zero structure, with additional computational facilities, *i.e.* the choice of fill-in parameters, rejection parameters, entropy-adaptivity-uncertainty (EAU) parameters [36], by which the best method for a given problem can be selected. The proposed methods have a universal scope of application for numerically solving of elliptic and parabolic boundary value problems by either FD or FE discretization methods in both two and three space dimensions with the only restriction being that the coefficient matrix should be diagonally dominant.

2. Approximate LU Decomposition and Approximate Inverse Methods

The approximate factorization techniques and approximate inverse methodologies have been widely used for solving a large class of linear and nonlinear systems resulting in complex computational problems [12] [29] [31] [37]-[49]. The LU factorization of a given matrix is characterized as a *high level* algebraic description of Gaussian elimination and by expressing the outcomes of matrix algorithms in the language of matrix factorizations facilitates generalization and certain connections between algorithms that may appear different at scalar level [47]. The solution of linear system can be computed by a two-step triangular solving process, *i.e.*

$$Ly = s, Ux = y \Rightarrow Ax = LUx = Ly = s \tag{2.1}$$

For solving the symmetric problem $Ax = s$, a variant of the LU factorization in which A is decomposed into three-matrix product, *i.e.* $A = LDM^{\mathrm{T}}$, where *D* is diagonal and L, M are lower unit lower triangular. In this case the solution can be obtained in $O(n^2)$ flops by solving $Ly = s$ (forward elimination), $Dz = y$ and $M^{\mathrm{T}}x = z$ (by substitution). Note that if $A = A^{\mathrm{T}}$ then *L* = *M* and the computational work for the factorization is half of that required by Gaussian elimination. The factorization takes the form $A = LDL^{\mathrm{T}}$ and can be used for solving symmetric problems, as well as the four-matrix product, *i.e.* $A = DTT^{\mathrm{T}}D$, where T^{T} is the transpose of T (Varga, 1962). In the case of symmetric positive definite systems the factorization $A = LDL^{\mathrm{T}}$ exists and is computationally stable. The factorization $A = LL^{\mathrm{T}}$ is known as *Cholesky factorization* and by solving the triangular system $Ly = s$ and $L^{\mathrm{T}}x = y$, then $s = Ly = (LL^{\mathrm{T}})x = Ax$. Note that the Gaxpy Cholesky version requires $n^3/3$ flops, where Gaxpy is a BLAS level-2 routine defined algebraically as $z = Ax + y$ requiring $O(mn)$ operations.

In the general case, such as the case of three space dimensions and the finite element discretization, the coefficient matrix has an irregular structure of the nonzero elements, where the non-diagonal elements can be grouped in regular bands of width l_1 and l_2 (width parameters) in distances *m* and *p* (semi-bandwidths) respectively.

The linear system (1.2) can be solved by direct (explicitly) or iterative (implicitly) methods depending on the availability of memory requirements. Several factorization/decomposition techniques can be used for facilitating the numerical solution of linear system (1.2), *i.e.* two, three, four term factorization schemes of the coefficient matrix *A*. Following an explicit solution of system (1.2) this system can equivalently written as

$$MAx = Ms, \tag{2.2}$$

where *M* is the inverse matrix of *A*, *i.e.* $M = A^{-1}$. Since the computation of the exact inverse is a difficult computational problem particularly in the case of complex 3D problems, the approximate inverse matrix approach can be alternatively used.

Let us consider an approximate factorization of the coefficient matrix *A*,

$$A \approx L_S U_S \tag{2.3}$$

where L_S and U_S, are lower and upper sparse triangular matrices of irregular structures of semi-bandwidths *m* and p retaining r_1 and r_2 fill-in terms respectively. The decomposition factors L_S and U_S are banded matrices with l_1 and l_2 the numbers of diagonals retained in semi-bandwidths m and p respectively (**Figure 2** and **Figure 3**), of the following form.

The computation of the elements of the sparse decomposition factors has been presented in [23] [24] [27].

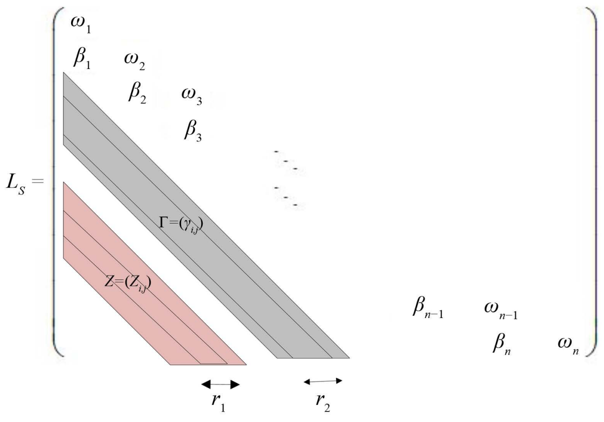

Figure 2. The structure of lower decomposition factor L_S.

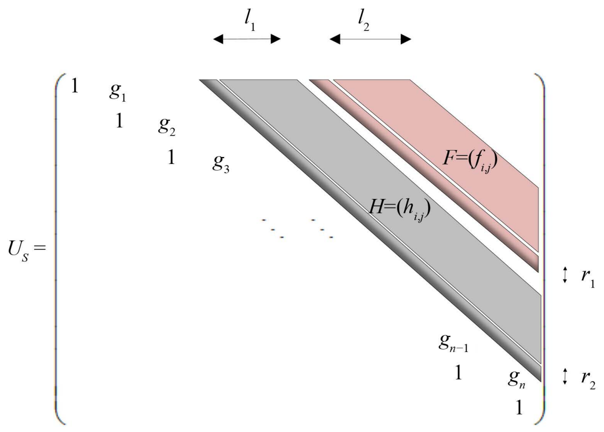

Figure 3. The structure of upper decomposition factor U_S.

The relationships of the elements of *V* matrix and the corresponding conventional (for $n-m+1=5$ and $l_1=3$) is shown in the following **Figure 4**.

An analogue scheme can be obtained for matrix *W*, while the relationships of the elements of *H* matrix and the corresponding conventional (for $n-m+1=5$ and $r_l+l_1-1=6$) is shown in the following **Figure 5** (the same holds for the matrix *F*).

The (near) optimum values of fill-in parameters are mainly depended on the nature of the problem and structure of the coefficient matrix *A* [22] [47]-[49].

3. Generalized Approximate Inverse Solvers for Unsymmetric Linear Systems of Irregular Structure

3.1. Introduction

An exact inverse algorithm based on adaptive algorithmic methodologies for solving linear unsymmetric systems of irregular structure arising in FD/FE discretization of boundary-value problems in three space dimensions has been recently presented [36]. This algorithm computes the elements of an exact inverse of a given unsymmetric matrix of irregular structure using an exact LU factorization [22]. The computational work of the EBAIM-1 algorithm is $\approx O\left[n\left(\delta l_1+\delta l_2\right)\left(m+p+l_1+l_2\right)\right]$ multiplications, while the memory requirements are $(n \times n)$ words. In the case of very large systems the memory requirements could be prohibitively high and the usage of approximate inverse iterative techniques is desirable.

It should be also noted that in the case that only $r_1<m-1$ and $r_2<p-1$ fill-in terms are retained in semi-bandwidths m and p respectively, then a class of approximate inverse matrix algorithms for solving large sparse unsymmetric linear systems of irregular structure [50] arising in the FD/FE discretization of elliptic and

$$\begin{array}{ccc} \multicolumn{3}{c}{\textit{Stored array}} \\ v_{1,1} & v_{1,2} & v_{1,3} \\ v_{2,1} & v_{2,2} & v_{2,3} \\ v_{3,1} & v_{3,2} & v_{3,3} \\ v_{4,1} & v_{4,2} & \\ v_{5,1} & & \end{array} \qquad \begin{array}{ccccc} \multicolumn{5}{c}{\textit{Conventional array}} \\ v_{1,1} & v_{1,2} & v_{1,3} & & \\ & v_{2,1} & v_{2,2} & v_{2,3} & \\ & & v_{3,1} & v_{3,2} & v_{3,3} \\ & & & v_{4,1} & v_{4,2} \\ & & & & v_{5,1} \end{array}$$

Figure 4. The relationship of matrix V in its banded stored form and the corres- ponding one in **Figure 1**.

$$\begin{array}{ccccc} h_{1,1} & h_{1,2} & h_{1,3} & h_{1,4} & h_{1,5} \\ h_{2,1} & h_{2,2} & h_{2,3} & h_{2,4} & h_{2,5} \\ h_{3,1} & h_{3,2} & h_{3,3} & h_{3,4} & h_{3,5} \\ h_{4,1} & h_{4,2} & h_{4,3} & h_{4,4} & h_{4,5} \\ h_{5,1} & h_{5,2} & h_{5,3} & h_{5,4} & \\ h_{6,1} & h_{6,2} & h_{6,3} & & \end{array} \qquad \begin{array}{ccccc} h_{1,1} & h_{1,2} & h_{1,3} & & \\ h_{2,1} & h_{2,2} & h_{2,3} & h_{1,4} & \\ h_{3,1} & h_{3,2} & h_{3,3} & h_{2,4} & h_{1,5} \\ h_{4,1} & h_{4,2} & h_{4,3} & h_{3,4} & h_{2,5} \\ & h_{5,1} & h_{5,2} & h_{4,4} & h_{3,5} \\ & & h_{6,1} & h_{5,3} & h_{4,5} \\ & & & h_{6,2} & h_{5,4} \\ & & & & h_{6,3} \end{array}$$

Figure 5. The relationship of matrix H in its stored form and the corres- ponding one in **Figure 3**.

parabolic boundary values, can be obtained. Such algorithms are described in the next sections.

A class of optimized approximate inverse variants can be obtained by considering a (near) optimized choice of the approximate inverse M depends on the selection of related parameters, *i.e.* fill-in parameters r_1, r_2, retention parameters δl_1, δl_2 and entropy-adaptivity-uncertainty (EAU) parameters [36] [51]. Note that the selection of retention parameter values as multiples of the corresponding semi-bandwidths of the original matrix leads to improved numerical results [22]. Then, the following sub-classes of approximate inverses, depending on the accuracy, storage and computational work requirements, can be derived as indicated in the following **Figure 6**. where $M^{\delta l_1,\delta l_2}_{r_1=m-1,\, r_2=p-1}$ of sub-class I is a banded form of the exact inverse retaining $\delta l_1, \delta l_2$ elements along each row and column respectively, while its elements are equal to the corresponding elements of the exact inverse. The term $M^{S\ \delta l_1,\delta l_2}_{r_1=m-1,\, r_2=p-1}$ of sub-class II is a banded form of M, retaining only $\delta l_1, \delta l_2$ elements along each row and column during the computational procedure of the approximate inverse and under certain hypotheses can be considered as a *good approximation* of the original inverse, while the entries of the approximate inverse in sub-class III have been retained after computing M^{*} $\left(r_1 < m-1 \text{ and } r_2 < p-1\right)$ and are less accurate than the corresponding entries of $M^{*\ \delta l_1,\delta l_2}_{r_1=m-1,\, r_2=p-1}$. Finally, in sub-class IV the elements of the approximate inverse can be computed.

3.2. Approximate Inverse Algorithmic Methodologies

Algorithmic solution methods for the linear systems (1.2) applicable to both two and three space dimension can be applied [23], where the unsymmetric coefficient matrix, in which all the off-centre terms are grouped into regular bands, can be factorized exactly to yield direct algorithmic procedures for the FD or FE solution. Alternatively, preconditioned iterative techniques for solving numerically the FD or FE linear systems (1.2) can be used by deriving semi-direct solution methods following the principle [28] that implicit procedures based on approximately decomposing discrete operators into easily invertible factors facilitating the solution of (1.2). Sparse factorization procedures yield efficient procedures for the FE or FD solution by manipulating the problem of the fill-in terms, which occur during the factorization. Note that simple compact storage schemes for the considered data can be used and preconditioned algorithmic solvers do not require any searching operations. An important feature of the proposed adaptive algorithmic methods is the provision of both direct and iterative methods for solving large sparse unsymmetric systems of irregular non-zero structure, with additional computational facilities, *i.e.* the choice of fill-in parameters, rejection parameters, entropy-adaptivity-uncertainty (EAU) parameters [36] [51], by which the best method for a given problem can be selected. The proposed methods have a universal

$$\begin{array}{cccc} \underline{\textit{sub-class I}} & \underline{\textit{sub-class II}} & \underline{\textit{sub-class III}} & \underline{\textit{sub-class IV}} \end{array}$$

$$A^{-1} \equiv M \longrightarrow M^{\delta l_1,\delta l_2 ?}_{r_1=m-1,\, r_2=p-1} \longrightarrow M^{S\,\delta l1,\delta l2?}_{r_1=m-1,\, r_2=p-1} \longrightarrow M^{\delta l_1,\delta l_2}_{r_1,r_2} \longrightarrow M^{*\,\delta l_1,\delta l_2}_{r_1,r_2} \equiv M^*$$

Figure 6. Subclasses of approximate inverses.

scope of application for numerically solving of elliptic and parabolic boundary value problems by either FD or FE discretization methods in both two and three space dimensions with the only restriction being that the coefficient matrix be diagonally dominant.

Let us assume that M_{r_1,r_2}, a non-singular $(n \times n)$ matrix, is an approximate inverse of *A*, *i.e.* $M_{r_1,r_2} = \{M_{i,j}\}^{r_1,r_2}$, $i, j \in [1, n]$. Note that if $r_1 = m-1$ and $r_2 = p-1$ non-zero elements have been retained in the corresponding decomposition factors, then $M_{r_1,r_2} = M$, where M is the exact inverse of *A*. The elements of M can be determined by solving recursively the systems

$$ML = U^{-1} \text{ and } UM = L^{-1}, \tag{3.1}$$

having main disadvantages, *i.e.* high storage requirements and computational work involved particularly in the case of solving very large unsymmetric linear systems. A class of approximate inverses $M^{\delta l_1,\delta l_2}$ can be obtained by retaining only δl_1 and δl_2 diagonals in the lower and upper triangular parts of inverse respectively, the remaining elements being just not computed at all. Optimized forms of this algorithm are particularly effective for solving banded sparse FE systems of very large order, *i.e.* $[\delta l_1 + \delta l_2] > n/2$ or in the case of narrow-banded sparse FE systems of very large order, *i.e.* $[\delta l_1 + \delta l_2] \ll n/2$.

Let us consider now the approximate inverse of *A* with the form

$$M^{\delta l_1,\delta l_2}_{r_1,r_2} = \left(L_{r_1,r_2} U_{r_1,r_2}\right)^{-1}, \; r_1 \in [1, m-1], r_2 \in [1, p-1], \tag{3.2}$$

where r_1, r_2 are the fill-in parameters, *i.e.* the number of outermost off-diagonal entries retained in semi-bandwidths *m* and *p* respectively.

Then, by post-multiplying Equation (3.2) by L_{r_1,r_2} and pre-multiplying the same equation by U_{r_1,r_2}, we obtain

$$M^* L_{r_1,r_2} = \left(U_{r_1,r_2}\right)^{-1}; \; \left(U_{r_1,r_2}\right) M^* = \left(L_{r_1,r_2}\right)^{-1} \tag{3.3}$$

where $M^* \equiv M^{\delta l_1,\delta l_2}_{r_1,r_2}$. Note that in the 2D symmetric case, *i.e.* $M_r \approx \left(L_r D_r L_r^{\mathrm{T}}\right)^{-1}$, $r \in [1, m-1]$, where *r* is the fill-in parameter, *i.e.* the number of outer-most off diagonal entries retained in semi-bandwidth of the tridiagonal factor L_r, by considering the equations in the analytical form for i-row with $j = r+1$ and the j-column with $i = r+1$ respectively we can obtain

$$\mu_{i,j} + \beta_j \mu_{i,j+1} + \sum_{\lambda=0}^{n-m+r-j} \gamma_{r+\lambda,j-r+\lambda+1} \mu_{i,j+m-r+\lambda} = \delta_{i,j} \tag{3.4}$$

where $\delta_{i,j}$ is the Kronecker delta [21] [27] [52].

The elements of the approximate inverse for $i = n$ can be determined successively as $\mu_{n,n}, \mu_{n,n-1}, \cdots, \mu_{n,1}$ (*i.e.* elements of the n-th row of the inverse) and for $j = n$ we obtain $\mu_{n-1,n}, \mu_{n-2,n}, \cdots, \mu_{1,n}$ (*i.e.* the n-th column of the inverse). Proceeding in a similar manner we can explicitly determine for $i = n-1, n-2, \cdots, 1$ and $j = n-2, n-3, \cdots, 2$ respectively the remaining elements $\mu_{i,j}$. In the following **Figure 7** the form of an (8 × 8) approximate inverse matrix is indicatively demonstrated.

where $M^{\delta l_1,\delta l_2}_{r_1,r_2}$ is a (8 × 8) banded approximate inverse matrix with *retention parameters* $\delta l_1 = 5$ and $\delta l_2 = 4$. Note that for simplicity reasons the case $\delta l_1 = \delta l_2 = \delta l$ is considered.

3.3. Optimized Approximate Inverse Matrices and Storage Techniques

Let us consider the exact inverse *M* of the original coefficient matrix *A* in equation (2.1). Note that the computation of the inverse is indicated in the following characteristic diagram (**Figure 8**).

It should be noted that the diagonal elements (in bold) are firstly computed (starting from the last element of the inverse, *i.e.* $\mu_{n,n}$) and then computing upwards/column-wise and from right to left/row-wise. Note also that

$$
\begin{array}{c}
\xleftrightarrow{\quad\delta l_1\quad} \\
M_{r_1,r_2}^{\delta l_1,\delta l_2} = \begin{array}{cccccccc}
\mu_{1,1} & \mu_{1,2} & \mu_{1,3} & \mu_{1,4} & \mu_{1,5} & & & \\
\mu_{2,1} & \mu_{2,2} & \mu_{2,3} & \mu_{2,4} & \mu_{2,5} & \mu_{2,6} & & \\
\mu_{3,1} & \mu_{3,2} & \mu_{3,3} & \mu_{3,4} & \mu_{3,5} & \mu_{3,6} & \mu_{3,7} & \\
\mu_{4,1} & \mu_{4,2} & \mu_{4,3} & \mu_{4,4} & \mu_{4,5} & \mu_{4,6} & \mu_{4,7} & \mu_{4,8} \\
 & \mu_{5,2} & \mu_{5,3} & \mu_{5,4} & \mu_{5,5} & \mu_{5,6} & \mu_{5,7} & \mu_{5,8} \\
 & & \mu_{6,3} & \mu_{6,4} & \mu_{6,5} & \mu_{6,6} & \mu_{6,7} & \mu_{6,8} \\
 & & & \mu_{7,4} & \mu_{7,5} & \mu_{7,6} & \mu_{7,7} & \mu_{7,8} \\
 & & & & \mu_{8,5} & \mu_{8,6} & \mu_{8,7} & \mu_{8,8}
\end{array}
\end{array}
\quad \delta l_2 \updownarrow \text{ (rows 1–4)}
$$

Figure 7. The structure of an (8 × 8) banded approximate inverse.

$$
\begin{array}{ccccc}
\leftarrow\ \mu_{n-4,n-4} & \mu_{n-4,n-3} & \mu_{n-4,n-2} & \mu_{n-4,n-1} & \mu_{n-4,n} \\
 & \uparrow & & & \\
\leftarrow & \mu_{n-3,n-3} & \mu_{n-3,n-2} & \mu_{n-3,n-1} & \mu_{n-3,n} \\
 & & \uparrow & & \\
 & \mu_{n-2,n-1} \leftarrow & \mu_{n-2,n-2} & \mu_{n-2,n-1} & \mu_{n-2,n} \\
 & & & \uparrow & \\
 & \mu_{n-2,n-1} & \mu_{n-1,n-2} \leftarrow & \mu_{n-1,n-1} & \mu_{n-1,n} \\
 & & & & \uparrow \\
 & \mu_{n-2,n-1} & \mu_{n,n-2} & \mu_{n,n-1} \leftarrow & \mu_{n,n}
\end{array}
$$

Figure 8. Computing the elements of the approximate inverse M^* of sub-class IV following the KS technique ($K = 2$)*. (*) Note that this computational technique will be referred as *double-sweep* (DS) technique.

the last diagonal element $\mu_{n,n}$ is computed and then all the elements of n-row (from right to left) and n-column (upwards). Then, only the diagonal element $\mu_{n-1,n-1}$ is computed, and next the diagonal element $\mu_{n-2,n-2}$ is computed and all the elements of (n-2)-row (from right to left) and the elements of (n-2)-column (upwards). Continuing in this way the rest elements of this approximate inverse are computed. It should be noted that the computational work of the resulting inverse M^* of sub-class IV, is almost the half of that required by the approximate inverse of sub-class III. Note diagrammatically, as it is shown in **Figure 8**, only the underlined elements of the approximate inverse M^* are computed.

By generalized this storage saving computational technique, we consider the above DS technique can be replaced by k-sweep (KS) technique, *i.e.* after the computation of the last diagonal element $\mu_{n,n}$, all the elements of n-row (from right to left) and n-column (upwards). Then, only the diagonal elements $\mu_{n-1,n-1}, \mu_{n-2,n-2}, \cdots, \mu_{n-k-1,n-k-1}, k \geq 2$ are computed, and next the diagonal element $\mu_{n-k-2,n-k-2}$ is computed and all the elements of (n-k-2)-row (from right to left) and the elements of (n-k-2)-column (upwards). Continuing in this way the rest elements of this approximate inverse are computed. It should be noted that the computational work of the resulting inverse M^* of this sub-class by using the KS-storage technique is considerably smaller than that required by the approximate inverse resulting from the application of the DS storage technique. In the case of $k = 2$ the KS-storage technique reduces to the example shown in **Figure 4**.

An optimized explicit banded approximate inverse by minimizing the memory requirements of EBAIM-1 *algorithm*

In order to minimize the memory requirements of EBAIM-1 algorithm, which in particular in the case of very large matrices of irregular structure can be prohibitively high, we consider the inverse M of equation (5.4) revolving its elements by 180˚ about the anti-diagonal removing the diagonal and the (δl-1) super diagonals in the first δl columns, while the rest δl sub-diagonals in the rest δl columns, then results the following form of the inverse (**Figure 9**).

4. The Optimized Approximate Inverse Algorithm

The application of this storage scheme on the approximate inverse leads to the following optimized approximate inverse algorithm. Note that the computation of the approximate inverse algorithm pre-assumes the approximate factorization of the coefficient matrix A, *i.e.* $A \approx L_s U_s$, where L_s and U_s are the lower and upper

$$
\overset{\xleftrightarrow{\qquad\qquad\qquad\qquad \delta l \qquad\qquad\qquad\qquad}}{}
$$

$$
M^{*}=\begin{array}{cccccccc}
\mu_{8,8} & \mu_{8,7} & \mu_{8,6} & \mu_{8,5} & \mu_{7,8} & \mu_{6,8} & \mu_{5,8} & \mu_{4,8} \\
\mu_{7,7} & \mu_{7,6} & \mu_{7,5} & \mu_{7,4} & \mu_{6,7} & \mu_{5,7} & \mu_{4,7} & \mu_{3,7} \\
\mu_{6,6} & \mu_{7,5} & \mu_{6,4} & \mu_{6,3} & \mu_{5,6} & \mu_{4,6} & \mu_{3,6} & \mu_{2,6} \\
\mu_{5,5} & \mu_{7,4} & \mu_{5,3} & \mu_{5,2} & \mu_{4,5} & \mu_{3,5} & \mu_{2,5} & \mu_{1,5} \\
\mu_{4,4} & \mu_{7,3} & \mu_{4,2} & \mu_{4,1} & \mu_{3,4} & \mu_{2,4} & \mu_{1,4} & \\
\mu_{3,3} & \mu_{7,2} & \mu_{3,1} & & \mu_{2,3} & \mu_{1,3} & & \\
\mu_{2,2} & \mu_{7,1} & & & \mu_{1,2} & & & \\
\mu_{1,1} & & & & & & &
\end{array}
$$

Figure 9. Transformed forms of optimized approximate inverse M^* (in banded storage).

sparse triangular decomposition factors [21] [23]-[25] [49].

Algorithm OBAIM-1 ($a, b, c, n, F, H, g, \Gamma, Z, \omega, \beta, r_1, r_2, m, p, l_1, l_2, \delta l, M^*$)

Purpose: This algorithm computes the elements of the approximate inverse of a given real ($n \times n$) matrix of irregular structure

Input: diagonal elements a of matrix A; superdiagonal elements b, subdiagonal elements c, n order of A; submatrices F, H, of upper triadiagonal decomposition factor U, superdiagonal elements g of L; submatrices Γ, Z, of lower tridiagonal matrix L; diagonal elements ω of L; subdiagonal elements β of L; fill-in parameters r_1, r_2; semi-bandwidths m, p; l_1 and l_2 numbers of diagonals retained in semi-bandwidths m and p respectively, δl_1 and δl_2 are the numbers of diagonal retained in approximate inverse M^*/for simplicity reasons $\delta l_1 = \delta l_2 = \delta l$ is chosen/

Output: elements $\mu_{i,j}$ of the approximate inverse M^*

Computational Procedure:

```
step 1: let $rl1 = r_1 + l_1$; $rl2 = r_2 + l_2$; $rl11 = rl1 - 1$; $rl21 = rl2 - 1$; $mr1 = m - r_1$; $ml1 = m + l_1$;
$pr2 = p - r_2$; $pl2 = p + l_2$; $nmr1 = n - m + r_1$; $npr2 = n - p + r_2$
step 2: for $i = n$ to 1
step 3: for $j = i$ to max$(1, i - \delta l + 1)$
step 4: if $j > nmr1$ then
step 5: if $i = j$ then
step 6: if $i = n$ then
step 7: $\mu_{1,1} = 1$
step 8: else
step 9: $\mu_{n-i+1,1} = 1 - g_j \mu_{n-j,\delta l+1}$
step 10: $\mu_{n-i+1,1} = \omega_n - \beta_j \mu_{n-j,\delta l+1}$
step 11: else
$\mu_{n-i+1,i-j+1} = -g_j \mu_{n-i+1,i-j}$
step 12: $\mu_{n-i+1,i-j+1} = -\beta_j \mu_{n-j,\delta l+1}$
Step 13: else
if $j > npr2$ and $j \le nmr1$ then
step 14: if $i = j$ then
step 15: $\mu_{n-i+1,1} = 1 - g_j \mu_{n-j,\delta l+1} - \sum_{k=0}^{nmr1-j} h_{rl11-k,j+k+1-r_1} \mu_{x,y}$
step 16: call mw $(n, \delta l, i, j + mr1 + k, x, y)$
step 17: $\mu_{n-i+1,1} = \omega_1 - \beta_j \mu_{i,j+1} - \sum_{k=0}^{nmr1-j} \gamma_{rl11-k,j+k+1-r_1} \mu_{x,y}$
step 18: call mw $(n, \delta l, i, j + mr1 + k, x, y)$
step 19: else
if $j \ge rl1$ and $j \le npr2$ then
step 20: if $i = j$ then
```

```
    step 21: $\mu_{n-m+1,1} = 1 - g_j \mu_{n-j,\delta l+1} - \sum_{k=0}^{nmr1-j} h_{rl11-k,j+k+1-r_1} \cdot \mu_{x_2,y_2} - \sum_{k=0}^{npr2-j} f_{rl12-k,j+k+1-r_2} \cdot \mu_{x_1,y_1}$
    step 22: $call\ mw\ (n, \delta l, i, j+mr1+k, x_2, y_2)$
    step 23: $call\ mw\ (n, \delta l, i, j+pr2+k, x_1, y_1)$
    step 24: $\mu_{n-m+1,1} = 1 - g_j \mu_{n-j,\delta l+1} - \sum_{k=0}^{nmr1-j} h_{rl11-k,j+k+1-r_1} \cdot \mu_{x_2,y_2} - \sum_{k=0}^{npr2-j} z_{rl12-k,j+k+1-r_2} \cdot \mu_{x_1,y_1}$
    step 25: $call\ mw\ (n, \delta l, i, j+mr1+k, x_2, y_2)$
    step 26: $call\ mw\ (n, \delta l, i, j+pr2+k, x_1, y_1)$
    step 27: else $\mu_{n-i+1,i-j+1} = -g_j \mu_{in-i+1,i-j} - \sum_{k=0}^{nmr1-j} h_{rl11-k,j+k+1-r_1} \cdot \mu_{x_2,y_2} - \sum_{k=0}^{npr2-j} f_{rl12-k,j+k+1-r_2} \cdot \mu_{x_1,y_1}$
    step 28: $call\ mw\ (n, \delta l, i, j+mr1+k, x_2, y_2)$
    step 29: $call\ mw\ (n, \delta l, i, j+pr2+k, x_1, y_1)$
    step 30: $\mu_{n-i+1,i-j+1j} = -\beta_i \mu_{n-i+1,i-j} - \sum_{k=0}^{nmr1-j} \gamma_{rl11-k,j+k+1-r_1} \cdot \mu_{x_2,y_2} - \sum_{k=0}^{npr2-j} z_{rl12-k,j+k+1-r2} \cdot \mu_{x_1,y_1}$
    step 31: $call\ mw\ (n, \delta l, i, j+mr1+k, x_2, y_2)$
    step 32: $call\ mw\ (n, \delta l, i, j+pr2+k, x_1, y_1)$
    step 33: else
  if $i = j$ then
    step 34: if $i = 1$ then
    step 35: $\mu_{n,1} = 1 - g_1 \mu_{n-1,\ \delta l+1} - \sum_{k=1}^{l_1} h_{1,k} \mu_{x_2,y_2} \sum_{k=1}^{l_2} f_{1,k} \cdot \mu_{x_1,y_1}$
    step 36: $call\ mw\ (n, \delta l, l, p+k-1, x_1, y_1)$
    step 37: $call\ mw\ (n, \delta l, l, m+k-1, x_2, y_2)$
    step 38: $\mu_{n,1} = \omega_1 - \beta_1 \mu_{n-1,\ \delta l+1} - \sum_{k=1}^{l_1} \gamma_{1,k} \mu_{x_2,y_2} - \sum_{k=1}^{l_2} z_{1,k} \mu_{x_1,y_1}$
    step 39: $call\ mw\ (n, \delta l, l, p+k-1, x_1, y_1)$
    step 40: $call\ mw\ (n, \delta l, l, m+k-1, x_2, y_2)$
    step 41: else $\mu_{n-i+1,1} = 1 - g_j \mu_{n-j,\delta l+1} - \sum_{k=1}^{j-1} h_{j-k,11+k} \cdot \mu_{x_1,y_1} - \sum_{k=j+1-r_1}^{l_1} h_{j,k} \mu_{x_2,y_2} - \sum_{k=1}^{j-1} f_{j-k,l_2+k} \mu_{x_3,y_3} - \sum_{k=j+1-r_2}^{l_2} f_{j,k} \cdot \mu_{x_4,y_4}$
    step 42: $call\ mw\ (n, \delta l, i, ml1+k-1, x_1, y_1)$
    step 43: $call\ mw\ (n, \delta l, i, m+k-1, x_2, y_2)$
    step 44: $call\ mw\ (n, \delta l, i, pl2+k-1, x_3, y_3)$
    step 45: $call\ mw\ (n, \delta l, i, p+k-1, x_4, y_4)$
    step 46: $\mu_{n-i+1,1} = \omega_1 - \beta_j \mu_{n-j,\delta l+1} - \sum_{k=1}^{j-1} \gamma_{j-k,11+k} \mu_{x_1,y_1} - \sum_{k=j+1-r_1}^{l_1} \gamma_{j,k} \mu_{x_2,y_2} - \sum_{k=1}^{j-1} z_{j-k,l_2+k} \cdot \mu_{x_3,y_3} - \sum_{k=j+1-r_2}^{l_2} z_{j,k} \mu_{x_4,y_4}$
    step 47: $call\ mw\ (n, \delta l, i, ml1+k-1, x_1, y_1)$
    step 48: $call\ mw\ (n, \delta l, i, m+k-1, x_2, y_2)$
    step 49: $call\ mw\ (n, \delta l, i, pl2+k-1, x_3, y_3)$
    step 50: $call\ mw\ (n, \delta l, i, p+k-1, x_4, y_4)$
    step 51: else $\mu_{n-i+1,i-j+1} = -g_j \mu_{n-i+1,i-j} - \sum_{k=1}^{j-1} h_{j-k,11+k} \mu_{x_1,y_1} - \sum_{k=j+1-r_1}^{l_1} h_{j,k} \mu_{x_2,y_2} - \sum_{k=1}^{j-1} f_{j-k,l_2+k} \mu_{x_3,y_3} - \sum_{k=j+1-r_2}^{l_2} f_{j,k} \mu_{x_4,y_4}$
    step 52: $call\ mw\ (n, \delta l, i, ml1+k-1, x_1, y_1)$
    step 53: $call\ mw\ (n, \delta l, i, m+k-1, x_2, y_2)$
    step 54: $call\ mw\ (n, \delta l, i, pl2+k-1, x_3, y_3)$
    step 55: $call\ mw\ (n, \delta l, i, p+k-1, x_4, y_4)$
    step 56: $\mu_{n-i+1,i-j+1} = \omega_1 - \beta_j \mu_{n-i+1,i-j} - \sum_{k=1}^{j-1} \gamma_{j-k,11+k} \mu_{x_1,y_1} - \sum_{k=j+1-r1}^{l_1} \gamma_{j,k} \mu_{x_2,y_2} - \sum_{k=1}^{j-1} z_{j-k,l_2+k} \mu_{x_3,y_3} - \sum_{k=j+1-r_2}^{l_2} z_{j,k} \mu_{x_4,y_4}$
    step 57: $call\ mw\ (n, \delta l, i, ml1+k-1, x_1, y_1)$
```

```
step 58: call mw $(n,\delta l,i,m+k-1,x_2,y_2)$
step 59: call mw $(n,\delta l,i,pl2+k-1,x_3,y_3)$
step 60: call mw $(n,\delta l,i,p+k-1,x_4,y_4)$
step 61: for $j=(i-1)$ to $\max(1,i-\delta l+1)$
step 62: $\mu_{n-i+1,\delta l+i-j}=\mu_{n-i+1,i-j+1}$
step 63: form the approximate inverse matrix $M^*=\{\mu_{i,j}\}$
```

The subroutine mw (n, δl, s, q, x, y) performs the transformation in the indexes of the explicit approximate inverse matrix from its banded form to the optimized form. This routine has the following form:

```
Subroutine mw ($n$, $\delta l$, $s$, $q$, $x$, $y$)
If $s \ge q$
Then
    $x = n+1-s$
    $y = s-q+1$
else
    $x = n+1-q$
    $y = \delta l+q-s$
```

The computational work of the optimized OBAIM-1 algorithm is $\approx O\left[n\delta l\left(r_1+r_2+l_1+l_2+1\right)\right]$ multiplications, while the memory requirements have been reduced down to $\left[n\times(2\delta l-1)\right]$ words. It should be also noted that a class of approximate inverse matrix can be considered containing several sub-classes of approximate inverses according to memory requirements, computational work, accuracy, as indicated in the diagrammatic schemes (**Figure 3** and **Figure 7**).

5. Explicit Adaptive Iterative Methods

A class of Adaptive Iterative Schemes for solving large sparse linear systems includes the following adaptive preconditioned iterative methods:

$$u_{i+1}=u_i+\alpha M^* r_i,\ i\ge 0,\ \text{(Explicit preconditioned simultaneous displacement)} \tag{5.1}$$

$$u_{i+1}=u_i+\alpha_i M^* r_i,\ i\ge 0,\ \text{(Explicit Preconditioned first order Richardson)} \tag{5.2}$$

$$\delta u_{i+1}=\alpha M^* r_i+\beta\delta u_i,\ i\ge 0,\ \text{(Explicit Preconditioned second order Richardson)} \tag{5.3}$$

$$\delta u_{i+1}=\alpha_i\cdot M^* r_i+\beta_i\delta u_i,\ i\ge 0,\ \text{(Explicit Preconditioned Chebyshev)} \tag{5.4}$$

where $r_i=s-Au_i$, α and β are predetermined acceleration parameters, α_i and β_i are sequences of preconditioned acceleration parameters and $\delta u_{i+1}=u_{i+1}-u_i$, $i\ge 0$.

5.1. The Explicit Preconditioned Iterative Method

During the last decades extensive research work has been focused in the preconditioned approach and preconditioned iterative methods for solving large linear and nonlinear problems in sequential and parallel environments [5] [8] [11] [12] [14] [26] [29] [30] [38] [39] [41] [47] [53]-[58]. A predominant role in the usage of the preconditioned iterative schemes possess the explicit preconditioned Conjugate Gradient (EPCG) method and its variants using the sparse approximate inverse M^* due to its superior convergence rate for solving very large complex computational problems [47]. A characteristic explicit solver of this sub-class is the Explicit Preconditioned Generalized Conjugate Gradient (EPGCG) method [58]. This basic EPCG method can be expressed in the following compact form:

Algorithm EPCG-1 (*A*, *n*, *s*, *u*₀, *u*, *r*)

Purpose: This algorithm computes the solution vector of the linear system $Au=s$ by using the explicit preconditioned generalized Conjugate Gradient method.

```
Input: $A$ given matrix, $n$ order of $A$, $s$ known rhs vector, $u_0$ initial guess
Output: solution vector $u$, residual $r$
Computational Procedure:
Step 1: let $u_0$ be an arbitrary initial approximation to the solution u
Step 2: set $r_0 = s - Au_0$, form $r_0^* = M^* r_0$ and set $\sigma_0 = r_0^*$
Step 3: for $i = 1, 2, \cdots$ (until convergence)
compute $u_{i+1}, r_{i+1}, \sigma_{i+1}$
//compute scalar quantities $\alpha_i, \beta_{i+1}$ as follows://
Step 4: form $q_i = A\sigma_i$ and set $p_i = (r_i, r_i^*)$ (only for $i = 0$)
Step 5: evaluate $\alpha_i = p_i / (\sigma_i, q_i)$ and compute $u_{i+1} = u_i + \alpha_i \sigma_i$
Step 6: compute $r_{i+1} = r_i - a_i q_i$ and form $r_{i+1}^* = M^* r_{i+1}$
Step 7: compute $p_{i+1} = (r_{i+1}, r_{i+1}^*)$ and evaluate $\beta_{i+1} = p_{i+1} / p_i$
Step 8: compute $\sigma_{i+1} = r_{i+1}^* + \beta_{i+1}\sigma_i$
Step 9: if there is no convergence go to step 3,
Step 10: else
print the approximate solution $u_0$ and corresponding residual $r_0$.
```

Note that a good approximant M^* leads obviously to an improved EPCG method. The effectiveness of the explicit preconditioned iterative methods for solving certain classes of elliptic boundary value problems on regular domains is related to the fact that the exact inverse of A (although is full) exhibits a similar *fuzzy* structure around the principal diagonal and m-diagonals [22].

5.2. The Symmetric Case

In the case of symmetric coefficient matrix by using the four-matrix decomposition [27] [41] the corresponding inverse subclasses can be enlarged as follows in **Figure 10**.
where 1) the elements of exact inverse of subclass I are obtained after the exact decomposition $(r_1 = m-1, r_2 = p-1)$ of M^+, with excessive memory and computational requirements, 2) the elements of the inverse $D^{-1}M^{S1}D^{-1}$ of subclass II have been computed after the application of the exact inverse algorithm $(r_1 = m-1, r_2 = p-1)$, while only δl_1 and δl_2 diagonals have been retained, 3) the elements of inverse M^{S2} of subclass III have been computed from the approximate inverse, while the exact decomposition $(r_1 = m-1, r_2 = p-1)$ has been applied, 4) the elements of the inverse of subclass IV have been computed from the approximate factorization and the banded approximate inverse algorithm has been used for computing the elements of the inverse $(r_1 \le m-1, r_2 \le p-1)$, 5) the elements of inverse of subclass V have been retained only on the diagonal elements of the inverse, *i.e.* $\delta l_1 = \delta l_2 = 1$, that is $M_{r_1,r_2}^{S3} \equiv I$, leading to a fast algorithm for computing of approximate inverse.

Note that the largest elements of inverse matrix are mainly gathered around the main diagonal in distances $p_m * m$ and $p_p * p$ in a *recurring wave* like pattern (Lipitakis, 1984), where m and p are respectively the semi-bandwidths of the coefficient matrix A, and $p_m = 1, 2, \cdots, (m-1)$ and $p_p = 1, 2, \cdots, (p-1)$. Based on this observation the selection of retention parameters δl_1, δl_2 is recommended to be multiples of values of m and p, leading to preconditioners with better performance [22].

An indication of the sparsity and memory requirements of optimized versions of approximate inverses is given in the following **Table 1**.

It should be noted that in the case that $\delta l_2 = 0$ the approximate inverse algorithm is reduced to an algorithm for solving FE systems in two space dimensions of semi-bandwidth m, while if $\delta l_1 = \delta l_2 = 1$ then the algorithm reduces to one for solving FD linear systems in three space dimensions of semi-bandwidths m and p [23]. In the case of $\delta l_1 = 1$ and $\delta l_2 = 0$, then the approximate inverse reduces to one for solving linear FD systems in two space dimensions of semi-bandwidth m [25], while if $\delta l_1 = \delta l_2 = 0$ then the approximate inverse reduces to the one for solving tridiagonal linear systems (Thomas algorithm) [59].

subclass I	subclass II	subclass III	subclass IV	subclass V
A^{-1}	$\longrightarrow D^{-1}M^{+}D^{-1}$	$\longrightarrow D^{-1}M^{S1}D^{-1}$	$\longrightarrow D_r^{-1}M^{S2}D_r^{-1}$	$\longrightarrow D_{r_1,r_2}^{-1}M_{r_1,r_2}^{S3}D_{r_1,r_2}^{-1} \longrightarrow D_{r_1,r_2}^{-2}$

Figure 10. Subclasses of inverses for the symmetric case.

Table 1. Memory and sparsity requirements of the approximate inverse matrix ($n = 8000$, $m = 21$, $p = 401$), where δl denotes here the retention parameters $\delta l_1 = \delta l_2 = \delta l$.

	$\delta l = 2$	$\delta l = m$	$\delta l = 2m$	$\delta l = p$	$\delta l = 2p$	$\delta l = 4p$
Diagonal vectors	3	41	83	801	1603	3207
Spasity	99.9	99.5	99	90	80	59.9

6. Numerical Experiments

In this section a nonlinear case study by using approximate inverse preconditioned methods are presented.

The nonlinear case

Let us consider the nonlinear elliptic PDE

$$\partial^2 U/\partial x^2 + \partial^2 U/\partial y^2 = e^U, (x, y) \in R, \tag{6.1}$$

where $R \equiv \{(x, y) : 0 \le x \le x_{\max}, 0 \le y \le y_{\max}\}$ subject to the Dirichlet boundary conditions

$$U(x, y) = \gamma(x, y), \ (x, y) \in \partial R, \tag{6.1a}$$

where ∂R is the exterior boundary of the domain R.

Equation (6.1) arises in magnetohydrodynamics (diffusion-reaction, vortex problems, electric space charge considerations) with its existence and uniqueness assured by the classical theory [32] [33]. The solution of Equation (6.1) can be obtained by the linearized Picard and quasi-linearized Newton iterative schemes as outer iterative schemes of the form:

$$\left(\delta_x^2 + \delta_y^2\right) u^{(k+1)} = e^{u(k)}, \left(x_i, y_j\right) \equiv \left(ih_x, jh_y\right) \in R \quad \text{and} \tag{6.2}$$

$$\left(\delta_x^2 + \delta_y^2\right) u^{(k+1)} - e^{u(k)} u^{(k+1)} = \left[1 - u^{(k)}\right] e^{u(k)}, \left(x_i, y_j\right) \equiv \left(ih_x, jh_y\right) \in R, \tag{6.3}$$

where δ denotes here the usual central difference operator.

The resulting large sparse nonlinear system is of the form

$$\Omega u^{(k+1)} = s\left(u^{(k)}\right), \tag{6.4}$$

where Ω is a block tridiagonal matrix [23].

Then, composite iterative schemes can be used, where Picard/Newton iterations are the outer iteration, while the inner iteration can be carried out either directly by an exact algorithm or by an approximate algorithm in conjunction with an explicit iterative method (6.3). The latter method can be written as

$$\delta u_{i+1}^{(l+1)} = \alpha M^* r_i^{(l+1)}, \ i \ge 0, l \ge 0, \tag{6.5}$$

where the superscript l denotes the outer iteration index, the subscript I denotes the inner iteration and $r_i^{(l+1)} = s\left(u_i^{(l)} - A_l u_i^{(l+1)}\right)$.

The outer iteration was terminated when the following criterion was satisfied

$$\max_j \left| \left(u_{i,j}^{(l+1)?} - u_{i,j}^{(l)} \right) \Big/ u_{i,j}^{(l+)} \right| < \varepsilon_0 = 10^{-6}, \tag{6.6}$$

while the termination criterion of the inner iteration was

$$\left|u_{i+1} - u_i\right| \Big/ \left(\left|u_{i+1}\right| + 1\right) < \varepsilon_1, \ i \ge 0, \tag{6.7}$$

where ε_1 was taken initially as $\varepsilon_1 = 10^{-2}$ and then was decreased at each iterative step by 1/10 to 10^{-6}, where it remained constant during the next iterative steps. Numerical experiments were carried out for nonlinear problem (6.4) with $h = 1/20, x_{\max} = y_{\max} = 1$ and the initial guesses when u was on the boundary 0.0, 5.0, 10.0 were chosen as 0.0, 4.0, 6.0 respectively. The performance of the composite schemes Newton-Explicit Preconditioned Simultaneous Displacement (EPSD) and Picard/Newton-EPCG are given in the following Table 2 and Table 3.

7. Conclusions

A class of exact and approximate inverse adaptive algorithmic procedures has been presented for solving numerically initial/boundary value problems. Several subclasses of optimized variants of these algorithms have been also proposed for solving economically highly nonlinear systems of irregular structure. It should be stated that the proposed explicit preconditioned iterative methods and their related variants can be efficiently used for solving large sparse nonlinear systems of irregular structure of complex computational problems and for the numerical solution of highly nonlinear initial/ boundary value problems in two and three space dimensions.

Table 2. The performance of composite iterative schemes Picard/Newton for the nonlinear system (6.4) using the EPSD method ($r = 4$), with $n = 361$, $m = 20$, for several values of acceleration parameter α and retention parameters δl.

	Picard-EPSD						**Newton-EPSD**					
δl	40		100		180		40		100		180	
						B.C. $U \equiv 0.0$						
a	Inner Iterat	Outer Iterat	Inner Iterat	Outer Iterat	Inner Iterat	Outer Iterat	Inner Iterat	Outer Iterat	Inner Iterat	Outer Iterat	Inner Iterat	Outer Iterat
1.20	72	27	56	23	55	23	79	30	61	25	59	24
1.30	73	31	52	22	51	22	74	29	56	23	55	23
1.40	>200	-	50	21	47	20	>200	-	52	22	50	21
1.50	-	-	45	20	44	20			48	21	47	21
1.60	-	-	42	19	41	19			45	20	45	21
						B.C. $U \equiv 2.0$						
1.20	188	9	142	9	131	9	58	7	48	7	37	6
1.30	173	9	130	9	122	9	>200	-	44	7	38	7
1.40	>200		123	9	114	9			41	7	42	6
1.50		-	118	9	117	9			42	7	38	6
1.60			132	10	130	10			>200	-	44	6

Table 3. The performance of composite iterative schemes Picard/Newton for the nonlinear system (6.4) using the EPCG method, with $n = 361$, $m = 20$, for several values of retention parameters δl and fill-in parameters r.

	Picard -EPCG				**Newton-EPCG**			
	Overall iterations			Outer iterations	Overall iterations			Outer iterations
δl	$r = 1$	$r = 2$	$r = 4$		$r = 1$	$r = 2$	$r = 4$	
			B.C.	**$U \equiv 0.0$**				
20	103	96	>150	6	85	94	161	6
40	75	70	64	6	68	54	55	6
60	71	53	57	6	65	49	49	6
			B.C.	**$U \equiv 10.0$**				
20	*	*	*	-	138	127	131	6
40	*	*	*	-	115	112	92	6
60	*	*	*	-	114	85	84	6

Future research work includes the parallelization of the proposed class of exact and approximate inverse matrices of irregular structure. These adaptive exact and approximate inverse algorithmic techniques can be used for solving efficiently highly nonlinear large sparse systems arising in the numerical solution of complex computational problems in parallel computer environments.

References

[1] Guillaume, P., Huard, A. and Le Calvez, C. (2002) A Block Constant Approximate Inverse for Preconditioning Large Linear Systems. *SIAM Journal on Matrix Analysis and Applications*, **24**, 822-851. http://dx.doi.org/10.1137/S0895479802401515

[2] He, B. and Teixeira, F.L. (2007) Differential Forms, Galerkin Duality, and Sparse Inverse Approximations in Finite Element Solutions of Maxwell Equations. *IEEE Transactions on Antennas and Propagation*, **55**, 1359-1368. http://dx.doi.org/10.1109/TAP.2007.895619

[3] Helsing, J. (2006) Approximate Inverse Preconditioners for Some Large Dense Random Electrostatic Interaction Matrices. *BIT Numerical Mathematics*, **46**, 307-323. http://dx.doi.org/10.1007/s10543-006-0057-0

[4] Cerdan, J., Faraj, T., Malla, N., Marin, J. and Mas, J. (2010) Block Approximate Inverse Preconditioners for Sparse Nonsymmetric Linear Systems. *Electronic Transactions on Numerical Analysis*, **37**, 23-40.

[5] Cui, X. and Havami, K. (2009) Generalized Approximate Inverse Preconditioners for Least Squares Problems. *Japan Journal of Industrial and Applied Mathematics*, **26**, 1-14. http://dx.doi.org/10.1007/BF03167543

[6] Gravvanis, G.A., Filelis-Papadopoulos, C.K., Giannoutakis, K.M. and Lipitakis, E.A. (2013) A Note on Parallel Finite Difference Approximate Inverse Preconditioning on Multicore Systems Using POSIX Threads. *International Journal of Computational Methods*, **10**, Article ID: 1350032. http://dx.doi.org/10.1142/S0219876213500321

[7] Giannoutakis, K., Gravvanis, G., Clayton, B., Patil, A., Enright, T. and Morrison, J. (2007).Matching High Performance Approximate Inverse Preconditioning to Architectural Platforms. *The Journal of Supercomputing*, **42**, 145-163. http://dx.doi.org/10.1007/s11227-007-0129-1

[8] Gravvanis, G.A., Epitropou, V.N. and Giannoutakis, K.M. (2007) On the Performance of Parallel Approximate Inverse Preconditioning Using Java Multithreading Techniques. *Applied Mathematics and Computation*, **190**, 255-270. http://dx.doi.org/10.1016/j.amc.2007.01.024

[9] Ngo, C., Yashtini, M. and Zhang, H.C. (2015) An Alternating Direction Approximate Newton Algorithm for Ill-Conditioned Inverse Problems with Application to Parallel MRI. *Journal of the Operations Research Society of China*, **3**, 139-162. http://dx.doi.org/10.1007/s40305-015-0078-y

[10] Wang, H., Li, E. and Li, G. (2013) A Parallel Reanalysis Method Based on Approximate Inverse Matrix for Complex Engineering Problems. *Journal of Mechanical Design*, **135**, Article ID: 081001. http://dx.doi.org/10.1115/1.4024368

[11] Xu, S., Xue, W., Wang, K. and Lin, H. (2011) Generating Approximate Inverse Preconditioners for Sparse Matrices Using CUDA and GPGPU. *Journal of Algorithms and Computational Technology*, **5**, 475-500. http://dx.doi.org/10.1260/1748-3018.5.3.475

[12] Alleon, G., Benzi, M. and Giraud, L. (1997) Sparse Approximate Inverse Preconditioning for Dense Linear Systems Arising in Computational Electromagnetics. *Numerical Algorithms*, **16.** http://dx.doi.org/10.1023/A:1019170609950

[13] Dubois, P.F., Greenbaum, A. and Rodrigue, G.H. (1979) Approximating the Inverse of a Matrix for Use in Iterative Algorithms on Vector Processors. *Computing*, **22**, 257-268. http://dx.doi.org/10.1007/BF02243566

[14] Grote, M.J. and Huckle, T. (1997) Parallel Preconditioning with Sparse Approximate Inverses. *SIAM Journal on Scientific Computing*, **18**, 838-853. http://dx.doi.org/10.1137/S1064827594276552

[15] Van der Vorst, H.A. (1989) High Performance Preconditioning. *SIAM Journal on Scientific Computing*, **10**, 1174-1185. http://dx.doi.org/10.1137/0910071

[16] Broker, O., Grote, J., Mayer, C. and Reusken, A. (2001) Robust Parallel Smoothing for Multigrid via Sparse Approximate Inverses. *SIAM Journal on Scientific Computing*, **23**, 1396-1417. http://dx.doi.org/10.1137/S1064827500380623

[17] Chow, E. (2000) A Priori Sparsity Patterns for Parallel Sparse Approximate Inverse Preconditioners. *SIAM Journal on Scientific Computing*, **21**, 1804-1822. http://dx.doi.org/10.1137/S106482759833913X

[18] Chow, E. (2001) Parallel Implementation and Practical Use of Sparse Approximate Inverse Preconditioners with a Priori Sparsity Patterns. *International Journal of High Performance Computing Applications*, **15**, 56-74. http://dx.doi.org/10.1177/109434200101500106

[19] Kallischko, A. (2008) Modified Sparse Approximate Inverses (MSPAI) for Parallel Preconditioning. Doctoral Dissertation, Technische Universitat Munchen Zentrum Mathematik, Germany.

[20] Tanaka, T. and Nodera, T. (2002) Effectiveness of Approximate Inverse Preconditioning by Using the MR Algorithm

on an Origin 2400. In: Topping, B.H.V., Bittnar, Z., Topping, B.H.V. and Bittnar, Z., Eds., *Proceedings of the* 3rd *International Conference on Engineering Computational Technology*, Paper 44, 115-116. http://dx.doi.org/10.4203/ccp.76.44

[21] Lipitakis, E.A. (1983) A Normalized Sparse Linear Equations Solver. *Journal of Computational and Applied Mathematics*, **9**, 287-298. http://dx.doi.org/10.1016/0377-0427(83)90021-3

[22] Lipitakis, E.A. (1984) Generalized Extended to the Limit Sparse Factorization Techniques for Solving Unsymmetric Finite Element Systems. *Computing*, **32**, 255-270. http://dx.doi.org/10.1007/BF02243576

[23] Lipitakis, E.A. (1989) Numerical Solution of 3D Boundary Value Problems by Generalized Approximate Inverse Matrix Techniques. *International Journal of Computer Mathematics*, **31**, 69-89. http://dx.doi.org/10.1080/00207168908803789

[24] Lipitakis, E.A. and Evans, D.J. (1979) Normalized Factorization Procedures for Solving Self-Adjoint Elliptic PDE's in 3D Space Dimensions. *Mathematics and Computers in Simulation*, **21**, 189-196. http://dx.doi.org/10.1016/0378-4754(79)90133-2

[25] Lipitakis, E.A. and Evans, D.J. (1980) Solving Non-Linear Elliptic Difference Equations by Extendable Sparse Factorization Procedures. *Computing*, **24**, 325-339. http://dx.doi.org/10.1007/BF02237818

[26] Lipitakis, E.A. and Evans, D.J. (1981) The Rate of Convergence of an Approximate Matrix Factorization Semi-Direct Method. *Numerische Mathematik*, **36**, 237-251. http://dx.doi.org/10.1007/BF01396653

[27] Lipitakis, E.A. and Evans, D.J. (1984) Solving Linear FE Systems by Normalized Approximate Matrix Factorization Semi-Direct Methods. *Computer Methods in Applied Mechanics and Engineering*, **43**, 1-19. http://dx.doi.org/10.1016/0045-7825(84)90091-4

[28] Lipitakis, E.A. and Evans, D.J. (1987) Explicit Semi-Direct Methods Based on Approximate Inverse Matrix Techniques for Solving BV Problems on Parallel Processors. *Mathematics and Computers in Simulation*, **29**, 1-17. http://dx.doi.org/10.1016/0378-4754(87)90062-0

[29] Gravvanis, G.A. (2000) Fast Explicit Approximate Inverses for Solving Linear and Non-Linear Finite Difference Equations. *International Journal of Differential Equations and Applications*, **1**, 451-473.

[30] Gravvanis, G.A. and Giannoutakis, K.M. (2005) Normalized Explicit Preconditioned Methods for Solving 3D Boundary Value Problems on Uniprocessor and Distributed Systems. *International Journal for Numerical Methods in Engineering*, **65**, 84-110. http://dx.doi.org/10.1002/nme.1494

[31] Gravvanis, G.A. and Gianoutakis, K.M. (2003) Normalized Explicit FE Approximate Inverses. *International Journal of Differential Equations and Applications*, **6**, 253-267.

[32] Pinter, G.F. and Gray, W.G. (1977) Finite Element Simulation in Surface Hydrology. Academic Press, New York.

[33] Porsching, T.A. (1976) On the Origins and Numerical Solution of Some Sparse Non-Linear Systems. In: Bunch, J.R. and Rose, D.J., Eds., *Sparse Matrix Computations*, Academic Press, New York, 439.

[34] Gravvanis, G.A. and Lipitakis, E.A. (1996) An Explicit Sparse Unsymmetric FE Solver. *Communications in Numerical Methods in Engineering*, **12**, 21-29. http://dx.doi.org/10.1002/(SICI)1099-0887(199601)12:1<21::AID-CNM948>3.0.CO;2-K

[35] Gravvanis, G.A. (2000) Generalized Approximate Inverse Preconditioning for Solving Non-Linear Elliptic BV Problems. *International Journal of Applied Mathematics*, **2**, 1363-1378.

[36] Lipitakis, A.D. and Lipitakis, E.A.E.C. (2012) On the E-Valuation of Certain E-Business Strategies on Firm Performance by Adaptive Algorithmic Modelling: An Alternative Strategic Managerial Approach. *Computer Technology and Application*, **3**, 38-46.

[37] Bollhofer, M. (2003) A Robust and Efficient ILU that Incorporates the Growth of the Inverse Triangular Factors. *SIAM Journal on Scientific Computing*, **25**, 86-103. http://dx.doi.org/10.1137/S1064827502403411

[38] Cosgrove, J., Dias, J. and Griewank, A. (1992) Approximate Inverse Preconditioning for Sparse Linear Systems. *International Journal of Computer Mathematics*, **44**, 91-110. http://dx.doi.org/10.1080/00207169208804097

[39] Evans, D.J. (1967) The Use of Preconditioning in Iterative Methods for Solving Linear Equations with Symmetric Positive Definite Matrices. *IMA Journal of Applied Mathematics*, **4**, 295-314.

[40] Il'lin, V.P. (1972) Iterative Incomplete Factorization Methods. World Scientific Press, Singapore.

[41] Varga, R.S. (1962) Matrix Iterative Analysis. Prentice-Hall, Upper Saddle River.

[42] Gravvanis, G.A. and Giannoutakis, K.M. (2003) Normalized FE Approximate Inverse Preconditioning for Solving Nonlinear BV Problems. In: Bathe, K.J., Ed., *Computational Fluid and Solid Mechanics* 2003, *Proceedings of* 2nd *MIT Conference on Computational Fluid and Solid Mechanics*, **2**, 1958-1962.

[43] Huckle, T. (1998) Efficient Computation of Sparse Approximate Inverses. *Numerical Linear Algebra with Applica-*

tions, **5**, 57-71. http://dx.doi.org/10.1002/(SICI)1099-1506(199801/02)5:1<57::AID-NLA129>3.0.CO;2-C

[44] Jia, Z. and Zhu, B. (2009) A Power Sparse Approximate Inverse Preconditioning Procedure for Large Sparse Linear Systems. *Numerical Linear Algebra with Applications*, **16**, 259-299. http://dx.doi.org/10.1002/nla.614

[45] Kolotilina, L.Y. and Yeremin, A.Y. (1993) Factorized Sparse Approximate Inverse Preconditioning. *SIAM Journal on Matrix Analysis and Applications*, **14**, 45-58. http://dx.doi.org/10.1137/0614004

[46] Tang, W.-P. and Wan, W.L. (2000) Sparse Approximate Inverse Smoother for Multigrid. *SIAM Journal on Matrix Analysis and Applications*, **21**, 1236-1252. http://dx.doi.org/10.1137/S0895479899339342

[47] Golub, G.H. and Van Loan, C. (1989) Matrix Computations. John Hopkins University Press, Baltimore.

[48] Evans, D.J. and Lipitakis, E.A. (1980) A Normalized Implicit Conjugate Gradient Method for the Solution of Large Sparse Systems of Linear Equations. *Computer Methods in Applied Mechanics and Engineering*, **23**, 1-19. http://dx.doi.org/10.1016/0045-7825(80)90075-4

[49] Giannoutakis, C.M. (2008) Study of Advanced Computational Methods of High Performance: Preconditioned Methods and Techniques of Matrix Inversion. Doctoral Thesis, Department of Electrical and Computer Engineering, Democritus University of Thrace, Hellas.

[50] Tewarson, R.P. (1966) On the Product Form of Inverses of Sparse Matrices. *SIAM Review*, **8**, 336-342. http://dx.doi.org/10.1137/1008066

[51] Lipitakis, A.D. (2016) A Generalized Exact Inverse Solver Based on Adaptive Algorithmic Methodologies for Solving Complex Computational Problems in Three Space Dimensions. HU-DP-TR15-2016, Technical Report.

[52] Axelsson, O. (1985) A Survey of Preconditioned Iterative Methods of Linear Systems of Algebraic Equations. *BIT Numerical Mathematics*, **25**, 166-187. http://dx.doi.org/10.1007/BF01934996

[53] Saad, Y. and Van der Vorst, H.A. (2000) Iterative Solution of Linear Systems in the 20th Century. *Journal of Computational and Applied Mathematics*, **123**, 1-33. http://dx.doi.org/10.1016/S0377-0427(00)00412-X

[54] Chen, K. (2005) Matrix Preconditioning Techniques and Applications. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, Cambridge. http://dx.doi.org/10.1017/CBO9780511543258

[55] Axelsson, O., Carey, G. and Lindskog, G. (1985) On a Class of Preconditioned Iterative Methods for Parallel Computers. *International Journal for Numerical Methods in Engineering*, **27**, 637-654. http://dx.doi.org/10.1002/nme.1620270314

[56] Benji, M. (2002) Preconditioning Techniques for Large Linear Systems: A Survey. *Journal of Computational Physics*, **182**, 418-477. http://dx.doi.org/10.1006/jcph.2002.7176

[57] Evans, D.J. (1972) The Analysis and Applications of Sparse Matrix Algorithms in FE Applications. In: Whiteman, J.R., Ed., *Proceedings of Brunel University Conference*, Academic Press, London and New York, 427-447.

[58] Gravvanis, G.A. (2002) Explicit Approximate Inverse Preconditioning Techniques. *Archives of Computational Methods in Engineering*, **9**, 371-402. http://dx.doi.org/10.1007/BF03041466

[59] Thomas, L.H. (1949) Elliptic Problems in Linear Difference Equations over a Network. Watson Sc. Comp. Lab. Rep., Columbia University, New York.