
Journal of Mathematical Modelling and Algorithms **1:** 269–282, 2002.

Solving Symmetric Arrowhead and Special Tridiagonal Linear Systems by Fast Approximate Inverse Preconditioning ⋆

GEORGE A. GRAVVANIS
Department of Information and Communication Systems Engineering, University of the Aegean, GR 832 00 Karlovasi, Samos, Greece. e-mail: gag@aegean.gr

(Received: 1 May 2002; accepted in final form: 20 May 2002)

Abstract. A new class of approximate inverses for arrowhead and special tridiagonal linear systems, based on the concept of sparse approximate Choleski-type factorization procedures, are introduced for computing fast explicit approximate inverses. Explicit preconditioned iterative schemes in conjunction with approximate inverse matrix techniques are presented for the efficient solution of symmetric linear systems. A theorem on the rate of convergence of the explicit preconditioned conjugate gradient scheme is given and estimates of the computational complexity are presented. Applications of the proposed method on linear and nonlinear systems are discussed and numerical results are given.

Mathematics Subject Classifications (2000): 65F05, 65F10, 65F50, 65N06, 65Y05, 65Y20.

Key words: arrowhead matrix, bordered diagonal matrix, special tridiagonal systems, approximate factorization, approximate inverses, preconditioning, parallel iterative methods.

1. Introduction

In recent years, research efforts have been directed towards the production of numerical algorithms and software for solving sparse linear systems of algebraic equations on uniprocessor or multiprocessor systems. The sparse linear systems occur in many engineering and scientific problems.

Recently, explicit preconditioned iterative methods have been extensively used for solving efficiently sparse linear systems, resulting from the finite element, finite difference, or domain decomposition discretization of partial differential equations, on multiprocessor systems, cf. [5–7, 9–12]. The effectiveness of the explicit preconditioned iterative methods based on adaptive approximate inverse matrix techniques is related to the fact that the approximate inverse exhibits a similar 'fuzzy' structure as the coefficient matrix.

Let us consider a matrix X, which is an arrowhead matrix (or bordered diagonal), and x denotes the nonzero entries. If we apply the Gaussian elimination

⋆ A preliminary version of this paper has appeared in [19].

method on X without any re-ordering, we will have a full matrix by the end of this process. If we permute by matrix P, then there will be no 'fill-in' terms during the factorization process of the matrix Y.

$$X = \begin{bmatrix} x & x & x & x & x \\ x & x & & 0 & \\ x & & x & & \\ x & 0 & & x & \\ x & & & & x \end{bmatrix}, \qquad Y = PXP^T = \begin{bmatrix} x & & & & x \\ & x & & 0 & x \\ & & x & & x \\ & 0 & & x & x \\ x & x & x & x & x \end{bmatrix}.$$

Arrowhead matrices occur in practice (cf. [2, 15, 18], for example) when using the Lanczos method for solving the eigenvalue problem for large sparse matrices ([16]), in the eigenstructure problems of arrowhead matrices which arise from applications in molecular physics ([14]), in the application of the finite element or finite difference method over a region by removing part of the region, and in Markovian reliability modeling.

Discussions for solving linear systems where the coefficient matrix is an arrow-type matrix have been given by many researchers. Reduction of an arrowhead matrix to tridiagonal form using orthogonal similarity transformations combined with QR algorithms has been presented in [14]. Recently, it was proved that there is no QR-like algorithm for computing the eigenvalues of arrowhead matrices ([1]). Additional methods for reducing to tridiagonal forms have been presented in [4, 17, 18]. Ordering schemes for reducing arrow matrices to band form matrices have also been presented in [9, 18] and computational techniques on the solution of special tridiagonal linear systems have been presented in [1–4, 14–17].

The derivation of suitable parallel methods was the main objective for which several forms of an approximate inverse, based on Choleski-type factorization procedures, have been proposed ([5, 6, 10–12]). The main motive for the derivation of the approximate inverses lies in the fact that they can be used in conjunction with explicit preconditioned iterative schemes and are suitable for solving linear systems on parallel and vector processors.

In Section 2, we introduce approximate inverse matrix techniques, based on Choleski-type factorization procedures. In Section 3, explicit preconditioned conjugate gradient schemes, based on approximate inverse matrix techniques, are presented. A theorem on the rate of convergence of the explicit preconditioned conjugate gradient scheme is given and estimates of the computational complexity are presented. Finally in Section 4, the performance and applicability of the proposed explicit preconditioned schemes is discussed by solving linear and non-linear systems and numerical results are given.

2. Approximate Inverse Matrix Techniques

In this section we present algorithmic procedures for computing the elements of the approximate inverse, based on Root-Free Cholesski factorization procedures ([5, 6, 11, 12]).

2.1. ARROWHEAD INVERSES

Our attention is now focused on solving arrowhead symmetric linear systems, i.e.,

$$Au = s, \tag{1}$$

where A is a sparse arrow-type $(n \times n)$ matrix of the following form:

$$A = \begin{bmatrix} b_1 & & & & c_1 \\ & \ddots & & 0 & \vdots \\ & & \ddots & & \\ & \text{symmetric} & & \ddots & c_{n-1} \\ & & & & b_n \end{bmatrix}. \tag{2}$$

Let us now assume the root-free Choleski factorization of the coefficient matrix A, i.e.

$$A = LDL^T \tag{3}$$

retaining the same number of nonzero entries, by applying the 'position-principle' in the factorization process, where D is a nonzero diagonal matrix and L is a sparse strictly lower (with main diagonal unity elements) triangular matrix of the same profile as the coefficient matrix A, cf. (2), viz.,

$$D \equiv \text{diag}(d_1, \ldots, d_n), \tag{4}$$

$$L = \begin{bmatrix} 1 & & & & \\ & \ddots & & 0 & \\ & & \ddots & & \\ & 0 & & \ddots & \\ e_1 & & \cdots & & 1 \end{bmatrix}. \tag{5}$$

Then, an **A**rrow**h**ead **A**pproximate **R**oot-**F**ree **C**holeski-type **FA**ctorization procedure (**AHARFCFA** algorithm) for computing the elements of the L and D decomposition factors can be expressed algorithmically by the following compact form:

For $i = 1, \ldots, n-1$,

$$d_i = b_i; \qquad e_i = c_i/d_i, \tag{6}$$

$$d_n = b_n - \sum_{k=1}^{n-1} d_k e_k^2. \tag{7}$$

The memory requirements of the **AHARFCFA** algorithm are O($3n$) words and the computational work required by the factorization process is $\approx$ O($3n$) multiplicative operations.

Let

$$M^{\delta l} \equiv (\mu_{i,j}), \quad i \in [1, n],\ j \in [\max(1, i - \delta l + 1),\ \min(n, i + \delta l - 1)],$$

an $[n \times (2\delta l - 1)]$ matrix, be the approximate inverse of the coefficient matrix A. The elements of the approximate inverse can be determined by retaining a certain number of elements of the inverse, i.e. only δl elements in the lower part (by applying the so-called 'position-principle'), and can be computed by recursively solving the following systems:

$$M^{\delta l} L D = (L^T)^{-1} \quad \text{and} \quad L^T M^{\delta l} = (LD)^{-1}, \quad \delta l \in [1, \ldots, n) \tag{8}$$

without inverting the decomposition factor L (cf. [5, 6, 11]).

It should be noted that the elements of the approximate inverse can be successively computed, using a 'fish-bone' computational procedure ([10, 11]). Optimized forms of the **B**anded **A**pproximate **I**nverse **A**rrowhead **M**atrix technique, using a moving window shifted from bottom to top, such that only $[n \times (2\delta l - 1)]$-vectors are retained in storage ([10, 11]), is particularly effective for solving 'narrow-banded' sparse systems of very large order, i.e. $\delta l \ll n/2$. Then the so-obtained **O**ptimized **B**anded **A**pproximate **I**nverse **A**rrowhead **M**atrix technique (**OBAIAM** algorithm) for computing the elements of the approximate inverse, can be expressed by the following compact scheme:

The elements of the nth row and column of the optimized form of the approximate inverse can be determined as follows:

$$\mu_{1,1} = 1/d_n, \tag{9}$$

For $j = n-1, n-2, \ldots, n - \delta l + 1$,

$$\mu_{1,n-j+1} = -e_j \mu_{1,1}, \tag{10}$$

$$\mu_{1,\delta l+n-j} = \mu_{1,n-j+1}. \tag{11}$$

Then the elements of the remaining $n-1, \ldots, 1$ rows and columns of the optimized form of the approximate inverse can be determined as follows:

For $i = n-1, n-2, \ldots, 1$
  for $j = i, i-1, \ldots, \max(1, i-\delta l+1)$
    if $i = j$ **then**

$$\mu_{n-i+1,1} = 1/d_j - e_j \mu_{1,\delta l+n-i}, \tag{12}$$

    else

$$\mu_{n-i+1,i-j+1} = -e_j \mu_{1,\delta l+n-i}, \tag{13}$$

  for $j = i-1, i-2, \ldots, \max(1, i-\delta l+1)$

$$\mu_{n-i+1,\delta l+i-j} = \mu_{n+1-i,i-j+1}. \tag{14}$$

The memory requirements of the **OBAIAM** algorithm are $\approx [n \times (2\delta l - 1)]$ words and the computational work involved is $\approx O(\delta l + 1)n$ multiplicative operations.

It should be also noted that, according to the proposed computational strategy, this class of approximate inverses can be considered as including various families of approximate inverses having in mind the desired requirements of accuracy, storage, and computational work as can be seen by the following diagrammatic relationship, i.e.,

$$\begin{array}{ccccccccc} & & \underline{\text{class I}} & & \underline{\text{class II}} & & \underline{\text{class III}} & & \underline{\text{class IV}} \\ A^{-1} \equiv M & \leftarrow & \widetilde{M}^{\delta l} & \leftarrow & M^{\delta l} & \leftarrow & M_n^{\delta l} & \leftarrow & M_i, \end{array} \tag{15}$$

where the entries of $\widetilde{M}^{\delta l}$ have been computed based on the diagonal and subdiagonal elements of the D and L decomposition factors, respectively. The entries of $M^{\delta l}$ have been computed and retained during the computational procedure of the (approximate) inversion. The entries of $M_n^{\delta l}$ have been retained after the computation of the approximate inverse, by additionally retaining the nth row and column of the approximate inverse. The M_i class of approximate inverse is computed by inverting the elements of the diagonal matrix D, cf. (4), yielding a fast inverse algorithm.

2.2. APPROXIMATE INVERSES OF SPECIAL TYPE TRIADIAGONAL SYSTEMS

Let us consider a one-dimensional elliptic equation, i.e. $\partial^2 u/\partial x^2 = f(x)$, $x \in R$, subject to the periodic boundary conditions $u(x_0) = u(x_n)$, $x \in \partial R$. By applying the finite difference method, we obtain a linear system, viz.

$$Au = s, \tag{16}$$

where A is a special tridiagonal $(n \times n)$ matrix, viz.,

$$A = \begin{bmatrix} b_1 & a_1 & & & v_1 \\ & \ddots & \ddots & & \\ & & & 0 & \\ & & & \ddots & a_{n-1} \\ \text{symmetric} & & & & b_n \end{bmatrix}. \tag{17}$$

Let us now assume the approximate root-free Choleski factorization of the coefficient matrix A, retaining r 'fill-in' terms during the decomposition process, viz.,

$$A \approx L_r D_r L_r^T, \tag{18}$$

where D_r is a diagonal matrix, cf. (4), and L_r is a sparse, strictly lower (with main diagonal unity elements) triangular matrix, cf. (19), of the same profile as the matrix A, cf. (17).

Then, a **S**pecial **T**ridiagonal **A**pproximate **R**oot-**F**ree **C**holeski-type **FA**ctorization procedure (**STARFCFA** algorithm) for computing the elements of the decomposition factors can be expressed algorithmically by the following compact scheme:

$$L_r = \begin{bmatrix} 1 & & & & \\ g_1 & 1 & & 0 & \\ & \ddots & \ddots & & \\ & 0 & & \ddots & \\ e_1 & \cdots & e_r & g_{n-1} & 1 \end{bmatrix}, \tag{19}$$

$$d_1 = b_1, \qquad g_1 = a_1/d_1, \qquad e_1 = v_1/d_1 \tag{20}$$

for $i = 2, \ldots, n-2$

$$d_i = b_i - d_{i-1} g_{i-1}^2, \qquad g_i = a_i/d_i \tag{21}$$

if $i \leqslant r$ **then**

$$e_i = -d_{i-1} e_{i-1} g_{i-1}/d_i \tag{22}$$

endif

$$d_{n-1} = b_{n-1} - d_{n-2}g_{n-2}^2, \qquad g_{n-1} = (a_{n-1} - d_{n-2}e_{n-2}g_{n-2})/d_{n-1}, \tag{23}$$

$$d_n = b_n - d_{n-1}g_{n-1}^2 - \sum_{k=1}^{r} d_k e_k^2. \tag{24}$$

The memory requirements of the **STARFCFA** algorithm are O($5n$) words and the computational work required by the factorization process is $\simeq$ O($6n$) multiplicative operations.

The elements of the optimized form of the approximate inverse can be computed by recursively solving the following equations:

$$M_r^{\delta l} L_r D_r = (L_r^T)^{-1} \quad \text{and} \quad L_r^T M_r^{\delta l} = (L_r D_r)^{-1}, \quad \delta l \in [1, \ldots, n), \tag{25}$$

and the so-obtained **O**ptimized **B**anded **S**pecial **T**ridiagonal **A**pproximate **I**nverse **M**atrix technique (**OBSTAIM** algorithm), can be expressed by the following compact algorithmic scheme:

For $j = n, n-1, \ldots, n-\delta l+1$

  if $j = n$ **then**

    $\mu_{1,1} = 1/d_n,$ (26)

  elseif $j > r$ **then**

    $\mu_{1,n-j+1} = -g_j \mu_{1,n-j},$ (27)

  else

    $\mu_{1,n-j+1} = -g_j \mu_{1,n-j} - e_j \mu_{1,1},$ (28)

For $j = n-1, n-2, \ldots, n-\delta l+1$

  $\mu_{1,\delta l+n-j} = \mu_{1,n-j+1}.$ (29)

Then the elements of the remaining $n-1, \ldots, 1$ rows and columns of the optimized form of the approximate inverse can be determined as follows:

For $i = n-1, n-2, \ldots, 1$

  for $j = i, i-1, \ldots, \max(1, i-\delta l+1)$

    if $i = j$ **then**

      if $j > r$ **then**

        $\mu_{n-i+1,1} = 1/d_j - g_j \mu_{n-j,\delta l+1},$ (30)

      else

        $\mu_{n-i+1,1} = 1/d_j - g_j \mu_{n-j,\delta l+1} - e_j \mu_{1,\delta l+n-i},$ (31)

    else

      if $j > r$ **then**

        $\mu_{n-i+1,i-j+1} = -g_j \mu_{n+1-i,i-j},$ (32)

      else

$$\mu_{n-i+1,i-j+1} = -g_j \mu_{n+1-i,i-j} - e_j \mu_{1,\delta l+n-i}, \tag{33}$$

for $j = i-1, i-2, \ldots, \max(1, i-\delta l+1)$

$$\mu_{n-i+1,\delta l+i-j} = \mu_{n+1-i,i-j+1}. \tag{34}$$

The memory requirements of the **OBSTAIM** algorithm are $\approx [n \times (2\delta l - 1)]$ words and the computational work involved is $\approx \mathrm{O}(3\delta l n)$ multiplicative operations.

Similarly, this class of approximate inverses can be considered to include various families of approximate inverses in terms of the desired requirements of accuracy, storage and computational work, as can be seen by the following diagrammatic relationship, i.e.,

$$A^{-1} \equiv M \;\leftarrow\; \underline{\text{class I}} \;\; M^{\delta l} \;\leftarrow\; \underline{\text{class II}} \;\; M_r^{\delta l} \;\leftarrow\; \underline{\text{class III}} \;\; M_{r,n}^{\delta l} \;\leftarrow\; \underline{\text{class IV}} \;\; M_i, \tag{35}$$

where the entries of $M^{\delta l}$ have been computed based on the diagonal, and codiagonal elements of the decomposition factors. The entries of $M_r^{\delta l}$ have been computed and retained during the computational procedure of the (approximate) inversion. The entries of $M_{r,n}^{\delta l}$ have been retained after the computation of the approximate inverse, by additionally retaining the nth row and column of the approximate inverse. The entries of M_i are computed by inverting the entries of the diagonal matrix D_r, yielding a fast inverse algorithm.

3. Explicit Preconditioned Conjugate Gradient Methods

In this section, we present a class of explicit preconditioned conjugate gradient–type schemes based on the derived approximate inverses, i.e. as preconditioners, to the original symmetric positive definite coefficient matrix A.

The **E**xplicit **P**reconditioning **C**onjugate **G**radient (**EPCG**) method can be expressed by the following compact scheme:

Let u_0 be an arbitrary initial approximation to the solution vector u. Then,

$$\text{form} \quad r_0 = s - Au_0, \qquad \text{compute} \quad r_0^* = M^{\delta l} r_0, \tag{36}$$

$$\text{set} \quad \sigma_0 = r_0^*. \tag{37}$$

Then, for $i = 0, 1, \ldots,$ (until convergence) compute the vectors $u_{i+1}, r_{i+1}, \sigma_{i+1}$ and the scalar quantities α_i, β_{i+1} as follows:

$$\text{form} \quad q_i = A\sigma_i, \qquad \text{calculate} \quad p_i = (r_i, r_i^*) \quad \text{when } i = 0 \text{ only}, \tag{38}$$

$$\text{evaluate} \quad \alpha_i = p_i/(\sigma_i, q_i), \tag{39}$$

$$\text{compute} \quad u_{i+1} = u_i + \alpha_i \sigma_i \quad \text{and} \quad r_{i+1} = r_i - \alpha_i q_i. \tag{40}$$

$$\text{Then, form} \quad r_{i+1}^* = M^{\delta l} r_{i+1} \qquad \text{set} \quad p_{i+1} = (r_{i+1}, r_{i+1}^*), \tag{41}$$

$$\text{evaluate } \beta_{i+1} = p_{i+1}/p_i, \qquad \text{compute} \quad \sigma_{i+1} = r_{i+1}^* + \beta_{i+1}\sigma_i. \tag{42}$$

The computational complexity of the **EPCG** method is $\approx \mathrm{O}[(2\delta l + 7)n$ mults $+$ $3n$ adds$]\nu$, where ν is the number of iterations for convergence to a certain level of accuracy.

The **E**xplicit **P**reconditioned **C**onjugate **G**radient **S**quare (**EPCGS**) method has been presented in [6, 10]. The computational complexity of the **EPCGS** method is $\approx$ O[$(4\delta l + 14)n$ mults + $8n$ adds]ν operations.

The **E**xplicit **P**reconditioned **Bi**conjugate **C**onjugate **G**radient-**STAB** (**EPBI-CGSTAB**) method can be expressed by the following compact scheme:

Let u_0 be an arbitrary initial approximation to the solution vector u. Then,

$$\text{set} \quad u_0 = 0, \qquad \text{compute} \quad r_0 = s - Au_0, \tag{43}$$

$$\text{set} \quad r_0' = r_0, \qquad \rho_0 = \alpha = \omega_0 = 1 \quad \text{and} \quad v_0 = p_0 = 0. \tag{44}$$

Then, for $i = 0, 1, \ldots,$ (until convergence) compute the vectors u_i, r_i and the scalar quantities α, β, ω_i as follows:

$$\text{calculate} \quad p_i = (r_0', r_{i-1}) \quad \text{and} \quad \beta = (\rho_i/\rho_{i-1})/(\alpha/\omega_{i-1}), \tag{45}$$

$$\text{compute} \quad p_i = r_{i-1} + \beta(p_{i-1} - \omega_{i-1}v_{i-1}), \tag{46}$$

$$\text{form} \quad y_i = M^{\delta l}p_i \quad \text{and} \quad v_i = Ay_i, \tag{47}$$

$$\text{and} \quad \alpha = \rho_i/(r_0', v_i), \qquad \text{compute} \quad x_i = r_{i-1} - \alpha v_i, \tag{48}$$

$$\text{form} \quad z_i = M^{\delta l}x_i \quad \text{and} \quad t_i = Az_i, \tag{49}$$

$$\text{set} \quad \omega_i = (M^{\delta l}t_i, M^{\delta l}x_i)/(M^{\delta l}t_i, M^{\delta l}t_i), \tag{50}$$

$$\text{compute} \quad u_i = u_{i-1} + ay_i + \omega_i z_i \quad \text{and} \quad r_i = x_i - \omega_i t_i. \tag{51}$$

The computational complexity of the **EPBI-CGSTAB** method is $\approx$ O[$(6\delta l + 14)n$mults + $6n$adds]ν operations, where ν denotes the number of iterations required for convergence to a predetermined tolerance level.

The convergence analysis of similar explicit approximate inverse preconditioning has been presented in [10]. Then, the following theorem on the rate of convergence and computational complexity can be proposed:

THEOREM 3.1. *Let $\Omega = M^{\delta l}A$ be the preconditioning matrix of the **E**xplicit **P**reconditioned **C**onjugate **G**radient* (**EPCG**) *iterative scheme. Suppose there exists positive numbers ξ_1, ξ_2 such that $[\xi_1, \xi_2] \supset [\lambda_{\min}, \lambda_{\max}]$ where ξ_1 is independent of the mesh size and it can be shown that $\xi_2 = \mathrm{O}(n\delta l^{-1})$* (cf. [10]). *Then the number of iterations ν of the **EPCG** method required to reduce the L_∞-norm of the error by a factor $\varepsilon > 0$ is given by*

$$\nu = \mathrm{O}(n^{1/2}\delta l^{-1/2}\log \varepsilon^{-1}) \tag{52}$$

and the computational complexity for the computation of the solution is given by

$$\mathrm{O}(n^{3/2}\delta l^{1/2}\log \varepsilon^{-1}). \tag{53}$$

Assuming a PRAM linear array model with n processors, the computation of the elements of the class IV approximate inverse M_i can be done in O(1), *i.e. constant time, since the inversion of the diagonal elements of D is required. Additionally, in the implementation of the **EPCG** scheme, the inner product can be performed*

Table I. The convergence behavior and the computational work per iteration required for the **EPCG**, **EPCGS** and **EPBICG-STAB** method

Method	**EPCG** with $\delta l = 1$		**EPCGS** with $\delta l = 1$		**EPBI-CGSTAB** with $\delta l = 1$	
n	No. of iter.	Comput. work (mults)	No. of iter.	Comput. work (mults)	No. of iter.	Comput. work (mults)
200	5	1800	5	3600	4	4000
600	5	5400	5	10 800	4	12 000
1000	5	9000	5	18 000	4	20 000
2000	5	18 000	5	36 000	4	40 000
5000	5	45 000	5	90 000	4	100 000
10 000	7	90 000	7	180 000	4	200 000

in $\mathrm{O}(\log n)$*, using the prefix computation. Thus the computational complexity for obtaining the solution is given by*

$$\mathrm{O}(n^{1/2}\delta l^{1/2}\log n \log \varepsilon^{-1}). \tag{54}$$

4. Numerical Results

In this final section, the applicability and effectiveness of the derived explicit preconditioned conjugate gradient schemes is discussed by considering the following test problem:

Model Problem I. The nonzero elements of A, cf. (2), are set equal to

$$b_i = 1.0, \quad i = 1, \ldots, n; \qquad c_i = 4.0, \quad i = 1, \ldots, n-1.$$

The right-hand side vector of the liner system was chosen as the product of the coefficient matrix A by the solution vector, with its components equal to unity. The **EPCG**, **EPCGS** and **EPBI-CGSTAB** methods were terminated when $|u_{i+1} - u_i| < 10^{-7}$.

Numerical results are presented in Table I for the **EPCG**, **EPCGS** and the **EPBI-CGSTAB** methods respectively, for several values of order n for the M_i class of approximate inverse.

It should be mentioned that the convergence behavior of the **EPBI-CGSTAB** method is generally better than the **EPCG** and **EPCGS** methods in conjunction with the M_i class of approximate inverse, while the convergence behavior of the **EPCG** and **EPCGS** methods are equivalent. It should be also stated that the **EPCG** method is better in performance than the **EPCGS** and **EPBI-CGSTAB** methods,

using the M_i class of approximate inverse, in terms of the overall computational complexity for convergence.

Model Problem II. Let us consider the following test problems:

$$u_{xx} = f(x) \tag{55}$$

with periodic boundary conditions $u(0) = u(1)$. It should be noted that using the finite difference discretization scheme, the resulting linear system is singular. Our approach is to use isomorphic iterative schemes which are expected to improve the rate of convergence of the explicit iterative schemes, according to the following definitions, cf. [7, 13]:

DEFINITION 4.1. The operator Ω_h^* is said to be δ_0-isomorphous to an operator Ω_h if Ω_h^* is identical to Ω_h, i.e., $\Omega_h^* \equiv \Omega_h$.

DEFINITION 4.2. The operator $\Omega_h^* \equiv \{\omega_{i,j}^*\}, i, j \in [1, n]$, is said to be δ_i-isomorphous to an operator $\Omega_h \equiv \{\omega_{i,j}\}, i, j \in [1, n]$ if the following relations hold:

$$\omega_{i,i}^* \equiv \omega_{i,i} + \sum_{k=i+1}^{n} \omega_{i,k} + \sum_{k=i+1}^{n} \omega_{k,i}, \quad i \in [1, n], \tag{56}$$

$$\begin{aligned} \omega_{i,j}^* &\equiv \sum_{k=1}^{n-j+1} \omega_{i,j+k-1}, \quad i \in [1, n-1],\ j \in [2, \mu], \\ \omega_{i,j}^* &\equiv \sum_{k=1}^{n-i+1} \omega_{i+k-1,j}, \quad i \in [2, \mu],\ j \in [1, n-1], \end{aligned} \tag{57}$$

and

$$\Delta_k^*(\Omega_h^*) = \Delta_k(\Omega_h), \quad k \in [\mu + 1, n], \tag{58}$$

where Δ^* and Δ are the corresponding diagonal of operators Ω_h^* and Ω_h, respectively, while k is the number of diagonals retained next to the main diagonal of Ω_h^*.

The operator Ω_h^* is δ_3-isomorphous to the operator Ω_h iff the operator Ω_h^* retains only three modified diagonals, i.e. the main diagonal and the diagonals next to the main diagonal in the lower and upper part of Ω_h^* with the corresponding elements derived from the relations (56)–(57) and the remaining elements identical to the elements of Ω_h.

DEFINITION 4.3. An explicit iterative method is said to be an isomorphic iterative method iff it involves δ_i-isomorphous operators.

Table II. The convergence behavior of the **EPCG**, **EPCGS** and **EPBI-CGSTAB** schemes

Method	δ_3-isomorphous **EPCG**			δ_3-isomorphous **EPCGS**			δ_3-isomorphous **EPBI-CGSTAB**		
n	$\delta l = 1$	$\delta l = 2$	$\delta l = 5$	$\delta l = 1$	$\delta l = 2$	$\delta l = 5$	$\delta l = 1$	$\delta l = 2$	$\delta l = 5$
100	15	9	5	8	5	3	8	5	4
500	15	9	6	8	5	3	8	5	4
1000	15	9	6	8	5	3	9	5	4
2000	15	9	7	8	5	3	8	5	4
5000	15	10	7	8	5	5	8	5	4
10 000	15	9	7	8	5	5	8	5	5

An efficient solution of the linear system $Au = s$ can be obtained by considering a δ_i-isomorphic iterative scheme. Let us consider $\Omega_h^* u^* = s$, where Ω_h^* is a δ_i-isomorphous operator to A. Then the approximate solution u^* has to be proved that it is an acceptable approximate solution to the original system $Au = s$, i.e. the norm $\|u - u^*\|$ has very small values. Further details on isomorphic iterative schemes can be found in [7, 13].

The **EPCG**, **EPCGS** and **EPBI-CGSTAB** methods, were terminated when $\|r_i\|_\infty < 10^{-7}$.

Numerical results are presented in Table II for the **EPCG**, **EPCGS** and the **EPBI-CGSTAB** methods, using an δ_3-isomorphous operator, for several values of the order n and the 'retention' parameter δl of the approximate inverse.

It should be mentioned that the performance of the **EPCG** isomorphic iterative scheme, in terms of the overall computational complexity, is in general better than the **EPCGS** or the **EPBI-CGSTAB** isomorphic iterative scheme for large values of the order n. Additionally, the lowest complexity occurs for the choice of the 'retention' parameter $\delta l = 1$ for either the **EPCG**, **EPCGS** or the **EPBI-CGSTAB** method.

Model Problem III. Let us consider a 1D-nonlinear boundary value problem, i.e.,

$$u_{xx} = e^u, \quad x \in R, \tag{59}$$

subject to periodic boundary conditions

$$u(0) = u(1), \quad x \in \partial R. \tag{59a}$$

The linearized Picard and quasi-linearized Newton iterations are outer iterative schemes, i.e.

$$\begin{aligned} &L_h u^{(k+1)} = e^{u^{(k)}}, \quad k > 0, \quad \text{and} \\ &L_h u^{(k+1)} - e^{u^{(k)}} u^{(k+1)} = (1 - u^{(k)}) e^{u^{(k)}}, \ k > 0, \end{aligned} \tag{60}$$

Table III. The convergence behavior of the **EPCG**, **EPCGS** and **EPBI-CGSTAB** schemes

δ_3-isomorphous iterative scheme						
Method	b.c.	δl	Picard method		Newton method	
			Outer iter	Inner iter	Outer iter	Inner iter
EPCG	10.0	1	9	37	9	37
		2	6	22	6	22
		3	6	19	5	17
EPCGS	10.0	1	7	24	5	21
		2	5	16	4	14
		3	5	15	5	15
EPBICG-STAB	10.0	1	6	22	6	22
		2	4	14	4	14
		3	4	13	4	13

where L_h denotes the central difference operator. The initial guesses $u_0 = 5.0$ was chosen for the boundary conditions $U = 10$. The **EPCG**, **EPCGS** and the **EPBI-CGSTAB** method, was terminated when $\max |u_{i+1} - u_i| < 10^{-6}$.

Numerical results are presented in Table III for the Picard and Newton methods, as outer iterative schemes, in conjunction with the **EPCG**, **EPCGS** and **EPBI-CGSTAB** methods, as inner iterative schemes, for several values of the 'retention' parameter δl with $n = 1000$.

Finally, it should be stated that the proposed explicit preconditioned isomorphic iterative schemes based on the derived classes of the approximate inverse are very efficient for the solution of highly nonlinear initial/boundary value problem.

References

1. Arbenz, P. and Golub, G.: QR-like algorithms for symmetric arrow matrices, *SIAM J. Matrix Anal. Appl.* **13** (1992), 655–658.
2. Fisher, D., Golub, G., Hald, O., Leiva, C. and Widlund, O.: On Fourier–Toeplitz methods for separable elliptic problems, *Math. Comp.* **28** (1974), 349–368.
3. Golub, G.: Some modified matrix eigenvalue problems, *SIAM Rev.* **15** (1973), 318–334.
4. Gragg, B. and Harrod, W.: The numerically stable reconstruction of Jacobi matrix from spectral data, *Numer. Math.* **44** (1984), 317–355.
5. Gravvanis, G. A.: A fast explicit preconditioned symmetric finite element scheme, *Neural Parallel Sci. Comput.* **9**(1) (2001), 59–66.
6. Gravvanis, G. A.: Fast explicit approximate inverses for solving linear and non-linear finite difference equations, *Internat. J. Differential Equations Appl.* **1**(4) (2000), 451–473.
7. Gravvanis, G. A.: Generalized approximate inverse preconditioning for solving non-linear elliptic boundary-value problems, *Internat. J. Appl. Math.* **2**(11) (2000), 1363–1378.

8. Gravvanis, G. A.: Parallel preconditioned algorithms for solving special tridiagonal systems, In: *Proc. Third Internat. Conf. on Dynamical Systems and Applications*, Vol. III, 1999, pp. 241–248.
9. Gravvanis, G. A.: Approximate inverse banded matrix techniques, *Engrg. Comput.* **16**(3) (1999), 337–346.
10. Gravvanis, G. A.: The rate of convergence of explicit approximate inverse preconditioning, *Internat. J. Comput. Math.* **60** (1996), 77–89.
11. Gravvanis, G. A.: Explicit preconditioned methods for solving 3D boundary value problems by approximate inverse finite element matrix techniques, *Internat. J. Comput. Math.* **56** (1995), 77–93.
12. Gravvanis, G. A.: A three dimensional symmetric linear equation solver, *Comm. Numer. Methods Engrg.* **10**(9) (1994), 717–730.
13. Lipitakis, E. A. and Evans, D. J.: Numerical solution of non-linear elliptic boundary-value problems by isomorphic iterative methods, *Interact. J. Comput. Math.* **20** (1986), 261–282.
14. O'Leary, D. and Stewart, G.: Computing the eigenproblem and eigenvectors of arrowhead matrices, *J. Comput. Phys.* **90** (1990), 497–505.
15. Parlett, B.: *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cilffs, 1980.
16. Parlett, B. and Nour-Omid, B.: The use of refined error bound when updating eigenvalues of tridiagonals, *Linear Algebra Appl.* **68** (1985), 179–219.
17. Rutishauser, H.: On Jacobi rotation patterns, In: *Experimental Arithmetic, High Speed Computing and Mathematics*, Proc. Sympos. Appl. Math. 15, Amer. Math. Soc., Providence, 1963, pp. 219–239.
18. Zha, H.: A two-way chasing scheme for reducing a symmetric arrowhead matrix to tridiagonal form, *J. Numer. Algebra Appl.* **1** (1992), 49–57.
19. Gravvanis, G. A.: Solving symmetric arrowhead linear systems by fast approximate inverses, In: *Proc. Internat. Conf. on Parallel and Distributed Processing Techniques and Applications*, *Vol. IV*, CSREA Press, 2001, pp. 1762–1768.