SOLVING INITIAL – VALUE PROBLEMS BY EXPLICIT DOMAIN
DECOMPOSITION APPROXIMATE INVERSES

George A. Gravvanis

Department of Mathematics
University of the Aegean
GR 832 00 Karlovasi, Samos, Greece
e-mail: gag@aegean.gr

Key words: Arrow-type matrix, domain decomposition method, approximate factorization procedures, approximate inverse matrix techniques, preconditioning, parallel iterative methods.

Abstract. A class of hybrid heterogeneous methods using time - implicit backward differences and Crank-Nicolson approximating schemes in conjunction with explicit domain decomposition approximate inverse matrix techniques is introduced for computing various families of approximate inverses based on approximate LU-type factorization techniques. Explicit preconditioned conjugate gradient - type schemes in conjunction with approximate inverse matrix techniques are presented for the efficient solution of linear systems. Application of the new proposed hybrid method on a 2D initial value problem is discussed and numerical results are given.
1 INTRODUCTION

Many engineering and scientific problems are described by sparse unsymmetric linear systems derived from the discretization of Elliptic and Parabolic Equations in two space variables. This category of equations represents a large class of commonly occurring problems in Mathematical Physics and Engineering (i.e. See-page Flow or irrotational Hydrodynamics Flow problems, Diffusion Theory and Plasma Physics problems, etc.).

An important achievement over the last decades is the appearance and use of Approximate Inverse Preconditioning methods for solving Partial Differential Equations. The effectiveness of the Preconditioning methods relies on the construction and use of efficient preconditioner factors in the sense that the preconditioners are close approximants to the coefficient matrix. Current research efforts are focused on the development of computational methods suitable for multiprocessor systems. The derivation of parallel numerical algorithms was the main objective for which several forms of approximate inverses of a finite difference or finite element matrix, based on adaptive approximate factorization procedures, have been proposed. The main motive for the derivation of the approximate inverses lies in the fact that they can be efficiently used in conjunction with explicit preconditioned conjugate gradient – type schemes which are appropriate for solving unsymmetric linear systems on parallel and vector processors.

The purpose of this work is the derivation of hybrid heterogeneous schemes based on time implicit approximating schemes, namely backward differences and Crank – Nicolson method, with explicit preconditioned schemata for the efficient iterative solution of unsymmetric linear systems, which arise from the domain decomposition discretization of Parabolic Partial Differential Equations in two space dimensions.

Our attention is now focused on domain decomposition techniques on regular domains and on multiprocessor systems. A domain is decomposed into smaller regular domains and the resulting sparse linear system of algebraic equations is of the following type:

\[
\begin{bmatrix}
A_{11} & 0 & A_{13} \\
0 & A_{22} & A_{23} \\
A_{13}^T & A_{23}^T & A_{33}
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
u_3
\end{bmatrix}
= 
\begin{bmatrix}s_1 \\
s_2 \\
s_3
\end{bmatrix}
\]  

Such linear systems, usually called arrow-type systems, occur in practice, and interesting discussions have been given by many researchers.

The derivation of suitable parallel methods was the main objective for which several forms of an approximate inverse of a given arrow-type matrix, based on adaptive approximate LU-type factorization procedures, where L and U are sparse lower and upper triangular matrices respectively of the same profile as the coefficient matrix, using either the “Location-Principle” or the “Magnitude-Principle”, have been proposed. The main motive for the derivation of the approximate inverse techniques of an arrow-type matrix, according to a “fish-bone” computational procedure by using the “Location-Principle” strategy, i.e. only δl (“retention” parameter) elements are retained, lies in the fact that they can be used in conjunction with
explicit preconditioned iterative schemes are suitable for solving linear systems on parallel or vector processors and systolic arrays.

Optimized forms, using a moving window shifted from bottom to top, of the pseudoinverse algorithm, in which both sparseness of the original matrix is relatively retained and storage requirements are substantially reduced, have been efficiently used for solving boundary value problems on multiprocessor systems.

It is known that the approximate factorization and pseudoinverse algorithms are in general complicated. However, as the demand for solving 2D initial value problems grows, the need to use efficient 2D linear equation solvers involving approximate factorization and corresponding pseudoinverse algorithms becomes of great importance.

Finally, the performance and applicability of the new proposed hybrid heterogeneous implicit-explicit preconditioned conjugate gradient - type schemes on Parabolic P.D.E's is discussed by solving characteristic time-dependent problems in two dimensions and numerical results are given.

2 DOMAIN DECOMPOSITION APPROXIMATE INVERSES

In this section we present domain decomposition approximate inverse algorithmic procedures, based on approximate LU-type factorization procedures.

Let us consider a class of time dependent problems defined by the following Parabolic Partial Differential Equation:

\[
\frac{\partial u}{\partial t} - \sum_{i,j=1}^{N=2} a_{i,j}(x) \frac{\partial u}{\partial x_i} + \sum_{i=1}^{N=2} b_i(x) \frac{\partial u}{\partial x_i} + c(x) u = g(x,t), \quad (x,t) \in \Omega = \mathbb{R} \times [0 \leq t \leq T],
\]

where \(\mathbb{R}\) is a two dimensional domain, subject to the initial conditions and boundary conditions respectively:

\[
u(x,0) = u_0(x), \quad x \in \partial \Omega \tag{2.a}
\]

\[
u(x,t) = 0, \quad (x,t) \in \partial \Omega \times [0 \leq t \leq T]. \tag{2.b}
\]

Let us assume that a uniform mesh-size \(h\) is used and the time-dependent functions were approximated by certain schemes, which can be written in the following parametric form:

\[
K_c^{(k+1)} - c^{(k)} = \frac{1}{h^2} F(0)c^{(k+1)} - (1-0)c^{(k)} = \theta b^{(k+1)} + (1-\theta)b^{(k)}, \quad \text{and} \quad Kc^{(0)} = e, \tag{3}
\]

where the value of the parameter \(\theta\), i.e. \(\theta \in [0, 1]\), denotes the time approximating scheme.

In the case of \(\theta = 1\), the above parametric form results in the "time-implicit" scheme, using backward time differences, i.e.

\[
(K + \rho F)c^{(k+1)} = Kc^{(k)} + \Delta t b^{(k+1)}, \quad \text{and} \quad Kc^{(0)} = e, \tag{4}
\]
and is unconditionally valid and independent of the mesh-ratio $\rho$. While, when $\theta = 1/2$ the "time-implicit" Crank-Nicolson scheme is obtained, using central time differences, i.e.,

$$
(K + \frac{1}{2} \rho F) c^{(k+1)} = (K - \frac{1}{2} \rho F) c^{(k)} + \frac{\Delta t}{2} (b^{(k+1)} + b^{(k)}), \quad \text{and} \quad Kc^{(0)} = e,
$$

which is unconditionally valid and independent of the mesh-ratio.

In matrix notation we obtain a linear system of the following compact form:

$$
A u = s \quad (6)
$$

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

$$
A = \begin{bmatrix}
    b_1 & c_{1,1} & \cdots & c_{1,l} & u_{1,1} & \cdots & u_{1,l_2} \\
    a_{1,1} & \vdots & \ddots & \vdots & \vdots & & \\
    a_{1,l} & \vdots & \ddots & u_{n-1,l-1,l_2} & \vdots & & \\
    v_{1,1} & \cdots & v_{n-1,l-1,l_2} & \vdots & \vdots & \ddots & \\
    v_{1,l_2} & \cdots & v_{n-1,l-1,l_2} & a_{n-1,l_2} & \cdots & a_{n-1,1} & b_n
\end{bmatrix}
$$

According to the structure of the coefficient matrix $A$, "fill-in" terms are required during the decomposition process.

Let us now assume the approximate factorization of the coefficient matrix $A$ such that, viz.

$$
A = LU \quad (8)
$$

retaining exactly the same number of non-zero entries, by applying the so-called “position-principle” in the factorization process, where $L$ and $U$ are sparse strictly lower and upper (with main diagonal unity elements) triangular matrices, cf. (9)-(10), of the same profile as the coefficient matrix $A$, cf. (7).

Then, the elements of the decomposition $L$ and $U$ factors can be computed by the so-called Domain Decomposition Approximate LU-type Factorization procedure (henceforth called the DODALUFA algorithm).

The memory requirement of the DODALUFA algorithm is $\approx O(2l_1 + 2l_2 + 1)h$ words and the computational work required by the factorization process is $\approx O(3l_1 + 3l_2 + 2)h$. 


multiplicative operations. The **DODALUFA** algorithm can be implemented on multiprocessor systems by following certain parallel decomposition techniques.

Let \( M^{\delta l} = (U^T)^{-1} \), \( i \in [1,n], j \in [\max(1,i-\delta l+1),\min(n,i+\delta l-1)] \), a \([n \times (2\delta l-1)]\) matrix, be the approximate inverse of the coefficient matrix \( A \). i.e.,

\[
M^{\delta l} = (U^T)^{-1}.
\]

The elements of the approximate inverse can be determined by retaining a certain number of elements of the inverse, i.e. only \( \delta l \) elements in the lower part and upper part of the inverse (by applying the so-called “position-principle”), next to the main diagonal, and its elements can be computed by solving recursively the following systems:

\[
M^{\delta l} L = U^{-1} \quad \text{and} \quad UM^{\delta l} = L^{-1} \quad \delta l \in [1,...,n]
\]
without inverting the L and U decomposition factors.

It should be noted that the computation of the elements of the approximate inverse, using a "fish-bone" computational procedure, can be successively determined as follows: From the equations of (12) for \( i=n, \ldots, 1 \) and \( j=\max(1, i-\delta_l+1), \ldots, \min(n, i+\delta_l-1) \) respectively, we can compute the elements of the various families of the approximate inverses.

Then, the elements of the approximate inverse can be computed by the so-obtained Domain Decomposition Generalized Approximate Inverse Matrix technique (henceforth called the DODGAIM algorithm).

In order to solve efficiently large linear systems the DODGAIM algorithm has to be redesigned, by using a moving window shifted from bottom to top, such that only \([n(2\delta_l-1)]\)-vectors are retained in storage. This Optimized form of the Domain Decomposition Generalized Approximate Inverse Matrix (henceforth called the ODODGAIM algorithm) is particularly effective for solving "narrow-banded" sparse systems of very large order, i.e. \( \delta_l << n/2 \).

The memory requirements of the ODODGAIM algorithm are \([n(2\delta_l-1)]\) words and the computational work involved is \( O([l_1 + l_2 + 1]6l)n \) multiplicative operations.

Hence, by considering an implicit approximating scheme \((\theta=1, \theta=1/2)\) in conjunction with explicit domain decomposition generalized approximate inverse matrix techniques (pseudo-inverses) yields a hybrid heterogeneous scheme. The new hybrid schemes have a universal scope of application for the numerical solution of parabolic P.D.E's by certain discretization methods in two space variables.

It should be noted that according to the proposed computational strategy this class of approximate inverses can be considered that includes 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.,

\[
\begin{align*}
\text{class I} & \quad \text{class II} & \quad \text{class III} \\
A^{-1} \equiv M & \quad \tilde{M}^{\delta_l} & \quad M^{\delta_l} & \quad M_1
\end{align*}
\]

where the entries of \( \tilde{M}^{\delta_l} \) have been retained after the computation of the exact inverse, while the entries of \( M^{\delta_l} \) have been computed and retained during the computational procedure of the approximate inverse. The diagonal inverse \( M_1 \) was computed based on the inversion of the diagonal elements of the L decomposition factor, cf. (9). This class of inverse can be implemented on parallel machines requiring \( O(k) \), i.e. a fast inverse algorithm.

It should be noted that if \( v_{i,j}=0 \) and \( u_{i,j}=0 \), cf. (7), then the DODALUFA and DODGAIM algorithms are reduced to BLUFA and OAIBM algorithms respectively for solving banded matrices. It should be also mentioned that if \( l_1 = 0 \) and \( l_2 = 1 \), cf. (7), then the DODALUFA and DODAIM algorithms are reduced to ALUFA and OAIAM algorithms respectively.
3 EXPLICIT PRECONDITIONED ITERATIVE METHODS

In this section we present a class of explicit preconditioned iterative schemes based on the ODDGAIM techniques of section 2 for solving the linear systems (6).

The Explicit Preconditioned Generalized Conjugate Gradient Square (EPGCGS) scheme can be expressed by the following compact scheme:

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

\[
\text{set } u_0 = 0 \quad \text{and} \quad e_0 = 0, \quad \text{solve } r_0 = M^{\delta l} (s - Au_0), \quad (14)
\]

\[
\text{set } \sigma_0 = r_0 \quad \text{and} \quad p_0 = (\sigma_0, r_0). \quad (15)
\]

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

\[
\text{form } q_i = A \sigma_i, \quad \text{calculate } \alpha_i = p_i / \left( (\sigma_0, M^{\delta l} q_i) \right), \quad (16)
\]

\[
\text{compute } e_{i+1} = r_i + \beta_i e_i - \alpha_i M^{\delta l} q_i, \quad d_i = r_i + \beta_i e_i + e_{i+1} \quad \text{and} \quad u_{i+1} = u_{i+1} + \alpha_i d_i. \quad (17)
\]

\[
\text{form } q_i = A d_i, \quad \text{compute } r_{i+1} = r_i - \alpha_i M^{\delta l} q_i, \quad (18)
\]

\[
\text{set } p_{i+1} = (\sigma_0, r_{i+1}), \quad \text{evaluate } \beta_{i+1} = p_{i+1} / p_i, \quad (19)
\]

\[
\text{compute } \sigma_{i+1} = \beta_{i+1}^2 - \beta_{i+1} + \alpha_i. \quad (20)
\]

The computational complexity of the EPGCGS method, assuming that the approximate inverse can be compactly stored in \( n \times (2\delta - 1) \) diagonal vectors is \( \mathcal{O}([4\delta + l_1 + 4 l_2 + 11] n \text{ mults} + 8 n \text{ adds}) \) operations, where \( \nu \) denotes the number of iterations required for convergence to a predetermined tolerance level.

In the following we present a modified form of the van der Vorst BI-CGSTAB method \(^{34}\), using the explicit preconditioner \( M^{\delta l} \). This modified method, henceforth called the Explicit Preconditioned Biconjugate Conjugate Gradient-STAB (EPBICG-STAB) method, can be expressed by the following compact scheme:

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

\[
\text{set } u_0 = 0 \quad \text{compute } r_0 = s - Au_0, \quad (21)
\]

\[
\text{set } r_0 = r_0, \quad p_0 = \alpha = \omega_0 = 1 \quad \text{and} \quad v_0 = p_0 = 0. \quad (22)
\]
Then, for \( i=0,1,\ldots \) (until convergence) compute the vectors \( \vec{r}_{i+1} \) and the scalar quantities \( \alpha, \beta, \omega_i \) as follows:

\[
\begin{align*}
\text{calculate} & \quad \rho_i = \begin{pmatrix} r_0^i, r_{i-1}^i \end{pmatrix} \quad \text{and} \quad \beta = \left( \frac{\rho_i}{\rho_{i-1}} \right) / \left( \frac{\alpha}{\omega_{i-1}} \right), \\
\text{compute} & \quad p_i = r_{i-1} + \beta \left( \rho_{i-1} - \omega_{i-1} v_{i-1} \right) \\
\text{form} & \quad y_i = M^{\delta l} p_i, \quad v_i = A y_i \quad \text{and} \quad \alpha = \rho_i / \left( r_0^i, v_i \right). \\
\text{compute} & \quad x_i = r_{i-1} - \alpha v_i, \quad \text{form} \quad z_i = M^{\delta l} x_i \quad \text{and} \quad t_i = A z_i, \\
\text{set} & \quad \omega_i = \begin{pmatrix} M^{\delta l} t_i, M^{\delta l} x_i \end{pmatrix} / \left( M^{\delta l} t_i, M^{\delta l} x_i \right) \\
\text{compute} & \quad u_i = u_{i-1} + \alpha y_i + \omega_i z_i \quad \text{and} \quad r_i = x_i - \omega_i t_i. 
\end{align*}
\]

The computational complexity of the EPBI-CGSTAB method is \( \mathcal{O}(\nu \left( 6 \delta l + 4 l_1 + 4 l_2 + 12 \right) \text{mults} + 6 \nu \text{adds}) \) \( \nu \) operations, where \( \nu \) denotes the number of iterations required for convergence to a predetermined tolerance level. The effectiveness of the explicit preconditioned iterative methods using the ODODGAIM algorithm is related to the fact that the approximate inverse of the original sparse coefficient matrix \( A \) exhibits a similar banded "fuzzy" structure as the coefficient matrix \( A \).

The convergence analysis of similar explicit approximate inverse preconditioning has been presented by Gravvanis [35].

Assuming a PRAM linear array model with \( n \)-processors is used the computation of the approximate inverse with \( \delta l = 1 \) can be performed in \( \mathcal{O}(1) \) while the inner product of the explicit preconditioned CG-type methods can be performed in \( \mathcal{O}(\log n) \), using the prefix computation model, leading to a fast explicit scheme.

### 4 NUMERICAL RESULTS

In this section we examine the applicability and effectiveness of the proposed schemes for solving characteristic problems in two dimensions.

Let us consider a 2D — initial value problem, i.e.,

\[
\begin{align*}
\frac{\partial u}{\partial t} &= u_{xx} + u_{yy}, \quad (x,y) \in \mathbb{R}, \\
\text{subject to boundary conditions and initial conditions} & \begin{align*}
u(x,y,t) &= 0, \quad (x,y,t) \in \partial \mathbb{R} \times [0 \leq t \leq T], \\
u(x,y,0) &= u_0(x,y), \quad x \in \partial \mathbb{R}.
\end{align*}
\end{align*}
\]
where $R$ is the unit square and $\partial R$ denotes the boundary of $R$.

The domain $R$ was decomposed into a number of sub-domains and was covered by a non-overlapping regular triangular network. The five-point finite difference discretization scheme with a row-wise ordering was used such that the length $l_1$ of the band was kept to low values, i.e. $l_1=3$.

The conjugate gradient-type iterative process was terminated when $\|r_1\|_\infty < 10^{-5}$, while the tolerance $\text{epss}$ for the steady state solution (s.s.s.) was set to $\text{epss}=10^{-5}$, with the initial guess chosen as $u_0 = 0.05$.

Numerical results are presented in Table 1 for the time-implicit backward difference scheme in conjunction with the EPGCGS and the EPBI-CGSTAB methods for several values of the time-step $\Delta t$ and the “retention” parameter $\delta l$ of the approximate inverse with $n=361$, $l_1=3$ and $l_2=76$.

<table>
<thead>
<tr>
<th>Method</th>
<th>$\Delta t$</th>
<th>outer iter. (s.s.s.)</th>
<th>$\delta l=1$</th>
<th>$\delta l=3$</th>
<th>$\delta l=6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>EPGCGS</td>
<td>0.0500</td>
<td>4</td>
<td>26</td>
<td>20</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>0.0100</td>
<td>6</td>
<td>40</td>
<td>34</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>0.0050</td>
<td>8</td>
<td>47</td>
<td>41</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>0.0010</td>
<td>13</td>
<td>70</td>
<td>61</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>0.0005</td>
<td>17</td>
<td>76</td>
<td>71</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>0.0001</td>
<td>35</td>
<td>100</td>
<td>97</td>
<td>80</td>
</tr>
<tr>
<td>EPBICG-STAB</td>
<td>0.0500</td>
<td>5</td>
<td>40</td>
<td>32</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>0.0100</td>
<td>7</td>
<td>48</td>
<td>38</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>0.0050</td>
<td>8</td>
<td>51</td>
<td>42</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>0.0010</td>
<td>14</td>
<td>63</td>
<td>57</td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>0.0005</td>
<td>18</td>
<td>66</td>
<td>66</td>
<td>59</td>
</tr>
<tr>
<td></td>
<td>0.0001</td>
<td>35</td>
<td>92</td>
<td>88</td>
<td>86</td>
</tr>
</tbody>
</table>

Table 1: The convergence behavior of the EPGCGS and EPBI-CGSTAB method for the time-implicit backward difference ($\theta=1$), with $n=361$, $m=20$, $l_1=3$ and $l_2=76$.

<table>
<thead>
<tr>
<th>Method</th>
<th>$\Delta t$</th>
<th>outer iter. (s.s.s.)</th>
<th>$\delta l=1$</th>
<th>$\delta l=3$</th>
<th>$\delta l=6$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.0005</td>
<td>15</td>
<td>51</td>
<td>48</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>0.0001</td>
<td>33</td>
<td>73</td>
<td>63</td>
<td>57</td>
</tr>
<tr>
<td>EPBICG-STAB</td>
<td>0.0005</td>
<td>15</td>
<td>47</td>
<td>46</td>
<td>46</td>
</tr>
<tr>
<td></td>
<td>0.0001</td>
<td>32</td>
<td>55</td>
<td>53</td>
<td>52</td>
</tr>
</tbody>
</table>

Table 2: The convergence behavior of the EPGCGS and EPBI-CGSTAB method for the Crank-Nicolson scheme ($\theta=1/2$), with $n=361$, $m=20$, $l_1=3$ and $l_2=76$.

In Table 2, numerical results are given for the Crank-Nicolson scheme in conjunction with
the EPGCGS and the EPBI-CGSTAB methods for several values of the time-step $\Delta t$ and the "retention" parameter $\delta l$ of the approximate inverse with $n=361, l_1=3$ and $l_2=76$.

It should be mentioned that the iterative GMRES scheme\textsuperscript{[6]}\textsuperscript{,}\textsuperscript{[12]} although it has good stability, requires storage of all the basis vectors of the Krylov space and its performance is depending on the restart vectors used, thus making this method problem dependent\textsuperscript{[13]}.

It should be mentioned that the convergence behavior of the EPGCGS and EPBI-CGSTAB methods, in conjunction with the DODALUFA and ODODGAIM algorithms, is much better when the domain is subdivided into many sub-domains.

Finally, we state that the explicit preconditioned domain decomposition methods, using the DODALUFA and ODODGAIM algorithms, can be efficiently used for solving non-linear initially/boundary value problems.

REFERENCES


[31] G.A. Gravvanis, Explicit preconditioned methods for solving 3D boundary value problems by approximate inverse finite element matrix techniques, I. J. Comp. Math. 56,