The Block Stride Reduction (B.S.R.) Algorithm with application to domain decomposition methods

D. J. Evans

Parallel Algorithms and Architectures Research Centre, Department of Computer Studies, Loughborough University, Loughborough, Leicestershire, UK
Email: D. J. Evans@lboro.ac.uk

Abstract

In this paper we consider the application of the Block Stride Reduction algorithm for the solution of separable self-adjoint elliptic equations on rectangular domains. Non-overlapping domain decomposition techniques are applied to reduce the differential operator to block diagonally bordered linear systems which can then be solved by the Preconditioned Conjugate Gradient method. Finally, a new preconditioner for such systems is outlined.

1 Introduction

In the numerical solution of elliptic partial differential equations for certain cases, the structure of the coefficient matrix in the linear system, i.e. block tridiagonal, allows the use of methods of reduced complexity compared to the standard direct (Gaussian Elimination or LU) or iterative (SOR, ADI or PCG) methods. This is the case in particular for linear, separable, self-adjoint elliptic equations defined on rectangular domains, with appropriate Dirichlet boundary conditions which is an important problem in many fields of science and engineering. Recently techniques have been developed to solve these problems much faster than the traditional iterative methods the most commonly used is the Block Cyclic Reduction (BCR) based on matrix decomposition in conjunction with fast transform algorithms (e.g. FFT). The basic idea of the first step is to combine every block row of even number with the two adjacent block rows so as to eliminate the odd (block) variables. A backsubstitution is then applied to compute all the unknowns. Recently, Evans [4] has extended this algorithm to one involving a stride of 3 (BSR) with an equivalent stabilised Buneman variant.
4 High Performance Computing

With the growth of parallel computers efforts to exploit these extra computational resources to achieve the best possible performance is underway. The FFT-based methods are very suitable, primarily because of the immediate decoupling of the equations into multiple scalar tridiagonal systems, and also because of the amenability of the FFT to parallel computation. Efficient algorithms for these methods have been developed (Hockney [6]). A parallel version of the BSR (PBSR) algorithm, using a partial fraction expansion of the rational matrix equation is proposed here. Further the parallel FFT-based methods and PBSR have parallel complexity which is polynomial in logn.

Finally domain decomposition (DD) usually refers to a class of techniques for solving partial differential equations on a given domain by first decomposing the domain into smaller domains, followed by a process consisting of solving problems on these subdomains and combining the partial solutions, as in the Schwarz's alternating method. Intuitively, domain decomposition lends itself well to parallel processing as work on the subdomains is smaller and can be done in parallel.

Here we consider non-overlapping rectangular subdomains of T-shaped regions and concentrate on separable elliptic PDEs with Dirichlet boundary conditions; because our subdomains are rectangles, we can use PBSR to solve their associated systems.

The reduction of the differential operator on the whole domain to a Schur Complement operator on the interfaces between the subdomains has been considered previously. The equations for the interfaces are then solved by iterative methods, such as preconditioned conjugate gradient (PCG) methods. Typically, each iteration requires the solution of a problem on each of the subdomains, so for efficiency reasons it is very important to keep the iteration count low by using a good preconditioner. Alternatively, since the structure of the matrix is regular, i.e. diagonally bordered then a near optimal preconditioner for the complete system can be considered.

2 Block Stride Reduction

This section describes the block stride reduction algorithm including the Buneman extension which was applied to stabilise the sequential Block Cyclic Reduction (BCR) algorithm.

2.2 Block Stride Reduction (BSR)

Consider a block tridiagonal linear system of the form,
where each block is an $m \times m$ tridiagonal matrix and the sub-vectors $x_i$ and $y_i$ are all of size $m$ as derived from the finite difference discretisation of the separable self adjoint elliptic equation

$$a(x) \frac{\partial^2 U}{\partial x^2} + b(x) \frac{\partial U}{\partial x} + c(x) U + \frac{\partial^2 U}{\partial y^2} = f(x,y),$$  

(2.1b)

on a square grid of points in the unit square with Dirichlet boundary conditions.

We restrict our attention to $n$ of the form $3^\mu - 1$, for integers $\mu$. A detailed description of the algorithm is now presented.

We now consider the case $n = 2 \times 3^2 - 1 = 17$ where the following systems of equations are to be solved,

$$Au = b,$$  

(2.2)

where $B$ is the tridiagonal matrix $(a_i, d_i, c_i)$, $i=1,2,\ldots,m$ with $a_1 = c_n = 0$ and $I$ is the $(17 \times 17)$ unit matrix when (2.1b) is the Laplace Equation, $B \equiv (1,-4,1)$.

We now consider the Stride of 3 Reduction (SR3) Algorithm which is given by the representative equation,

$$u_{i-3} + B^{[2]} u_i + u_{i+3} = b_i^{[2]}, \text{ for } i=3,6,9,12,15$$  

(2.3)

where $B^{[2]} = B(B^2 - I)$,  

(2.4)

and $b_i^{[2]} = b_{i-2} + b_{i+2} - B(b_{i-1} + b_{i+1}) - (I-B^2)b_i$, for $i=3,6,9,12,15$.  

(2.5)

These equations can be derived quite easily from 2 adjacent equations derived from the cyclic odd-even reduction algorithm, i.e.,
6 High Performance Computing

\[ u_{i-2} + B[2] u_{i} + u_{i+2} = b_i[2], \quad (2.6) \]

and

\[ u_{i} + B[2] u_{i+2} + u_{i+4} = b_i[2], \quad (2.7) \]

with

\[ \hat{B} = 2I - B^2 \quad \text{and} \quad \hat{b}_i = b_{i-1} - B b_i + b_{i+1}, \]

combined with,

\[ u_{i} + B u_{i+1} + u_{i+2} = b_{i+1}, \quad (2.8) \]

derived from (2.2).

By eliminating \( u_i \) and \( u_{i+2} \) from equations (2.6), (2.7) and (2.8) we obtain the result given in (2.3) and (2.5).

Now by carrying on the application of (2.3) to equation (2.2) we obtain the following reduced system,

\[
\begin{bmatrix}
B[2] & 0 & 0 & 0 \\
-1 & B[2] & 0 & 0 \\
0 & -1 & B[2] & 0 \\
0 & 0 & -1 & B[2]
\end{bmatrix}
\begin{bmatrix}
u_3 \\
u_6 \\
u_9 \\
u_{12} \\
u_{15}
\end{bmatrix}
=
\begin{bmatrix}
b_3[2] \\
b_6[2] \\
b_9[2] \\
b_{12}[2] \\
b_{15}[2]
\end{bmatrix},
\quad (2.9)
\]

where \( B[2] \) and \( b[2] \) are given by (2.4) and (2.5).

Now the solution to (2.9), can be obtained by applying the SR3 algorithm again to obtain the system, where \( B[2] \) and \( b[2] \) are given by (2.4) and (2.5).

Now the solution to (2.9), can be obtained by applying the SR3 algorithm again to obtain the system,

\[ B[3] u_9 = u_9[3], \quad (2.10) \]


and \( b_i^{[3]} = b_i^{[2]} + b_i^{[2]} - B[2](b_{i-1}^{[2]} + b_{i+1}^{[2]} - (I-B[2]^2)b_i^{[2]}), \) for \( i=9 \).

However the system \( B[3] u_9 = b_9^{[3]} \) etc. is not tridiagonal and appears to be not easily solvable but the solutions to these equations may be obtained by applying the solution to the tridiagonal problem repeatedly as follows.

In general, the matrices, \( B[n] \) satisfy the recursion relationships,

\[
B[1] = B \\
\vdots \\
\quad (2.11)
\]

\[...
\]
Further it can be seen that $B^{[n]}$ can be factorised as,

$$B^{[n-1]} (B^{[n-1]} + \sqrt{3}I) (B^{[n-1]} - \sqrt{3}I),$$

or

$$B^{[n]} = \prod_{k=1}^{3^{n-1}} G^{[n]}_3, \quad n > 1,$$

(2.12)

where $G^{[n]}_k$ is a tridiagonal matrix given by,

$$G^{[n]}_k = B - \lambda^{[n]}_k I,$$

(2.13)

and,

$$\lambda^{[n]}_k = 2 \cos \left[ \frac{(2k-1)\pi}{2 \cdot 3^{n-1}} \right].$$

(2.14)

Thus the equation (2.11) can be solved by applying the tridiagonal solver $G^{[n]}_k$ repeatedly, i.e. for $u_9$,

$$x_9 = \left[ \prod_{k=1}^{3^{n-1}} G^{[3]}_k \right]^{-1} y^{[3]}_9.$$

(2.15)

After the determination of $u_9$ the solution of the remaining elements can be determined by solving the subsystems,

$$\begin{bmatrix} B^{[2]} & I^{[2]} \\ I^{[2]} & B^{[2]} \end{bmatrix} \begin{bmatrix} u_3 \\ u_6 \end{bmatrix} = \begin{bmatrix} b_3 \\ b_6 \end{bmatrix},$$

(2.16a)

$$\begin{bmatrix} B^{[2]} & I^{[2]} \\ I^{[2]} & B^{[2]} \end{bmatrix} \begin{bmatrix} u_{12} \\ u_{15} \end{bmatrix} = \begin{bmatrix} b_{12} \\ b_{15} \end{bmatrix},$$

(2.16b)

in a similar manner as before, i.e.,

$$(I-B^{[2]}^2)u_6 = b_3 B^{[2]} b_6, \quad (I-B^{[2]}^2)u_3 = b_6 B^{[2]} b_3,$$

(2.17)

with similar results for $u_{12}$ and $u_{15}$.

Again the systems $(I-B^{[2]}^2)$ are not tridiagonal but can be made to appear so by the application of the recursive procedure (2.12)-(2.14).

Finally the remaining elements can be obtained by solving the subsystems,

$$\begin{bmatrix} B & I \\ I & B \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, \quad \begin{bmatrix} B & I \\ I & B \end{bmatrix} \begin{bmatrix} u_4 \\ u_5 \end{bmatrix} = \begin{bmatrix} b_4 \\ b_5 \end{bmatrix}, \quad \begin{bmatrix} B & I \\ I & B \end{bmatrix} \begin{bmatrix} u_7 \\ u_8 \end{bmatrix} = \begin{bmatrix} b_7 \\ b_8 \end{bmatrix},$$

$$\begin{bmatrix} B & I \\ I & B \end{bmatrix} \begin{bmatrix} u_{10} \\ u_{11} \end{bmatrix} = \begin{bmatrix} b_{10} \\ b_{11} \end{bmatrix}, \quad \begin{bmatrix} B & I \\ I & B \end{bmatrix} \begin{bmatrix} u_{13} \\ u_{14} \end{bmatrix} = \begin{bmatrix} b_{13} \\ b_{14} \end{bmatrix}, \quad \begin{bmatrix} B & I \\ I & B \end{bmatrix} \begin{bmatrix} u_{16} \\ u_{17} \end{bmatrix} = \begin{bmatrix} b_{16} \\ b_{17} \end{bmatrix},$$

in a similar manner as previously shown.
High Performance Computing

The solution stages are illustrated in Figure 1.

![Figure 1: Reduction stages in the SR3 algorithm.](image)

The above block algorithm requires the multiplication of a large number of matrices to determine the right-hand side terms which can lead to round-off errors, thus invalidating the algorithm. The multiplication of matrices can be avoided by the use of the following strategy introduced by Buneman [5] in the cyclic odd-even reduction method.

Suppose the reduced systems (2.6) are written in the form,

\[
\begin{bmatrix}
B^{[2]} & I \\
I & B^{[2]}
\end{bmatrix}
\begin{bmatrix}
u_3 \\
u_6
\end{bmatrix} =
\begin{bmatrix}
b_3^{[2]} \\
b_6^{[2]}
\end{bmatrix} =
\begin{bmatrix}
B^{[2]}p_3^{[2]} + q_3^{[2]} \\
B^{[2]}p_6^{[2]} + q_6^{[2]}
\end{bmatrix},
\]

(2.18)

Now if \(p_k^{[n]}, q_k^{[n]}\) can be calculated without performing any matrix multiplications, troublesome rounding errors would be avoided. Then the vectors \(p_3^{[2]}\) and \(q_3^{[2]}\) can be determined as,

\[
b_3^{[2]} = B^{[2]}p_k^{[2]} + q_k^{[3]}
\]

\[
= B(B^{-2}3I)p_3^{[2]} + q_3^{[3]},
\]

(2.19)

where \(q_k^{[1]} = b_k\).

If we specify \(Bp_k^{[2]} = q_k^{[1]}\) then from (2.4) and (2.5) we have,

\[
p_k^{[2]} = B^{-1}q_k^{[1]},
\]

(2.20)

and,

\[
q_k^{[2]} = q_{k-2}^{[1]} + q_{k+2}^{[1]} + 2q_k^{[1]} - B^{-2}(p_{k-1}^{[2]} + p_{k+1}^{[2]}),
\]

(2.21)

where the operation \(B^{-2}\) is performed by repeatedly applying the tridiagonal solver.

The generation of equations (2.20) and (2.21) for \(n>2\) is then given by,

\[
p_k^{[n]} = B^{[n-1]}q_k^{[n-1]},
\]

(2.22)

and,

\[
q_k^{[n]} = q_{k-2}^{[n-1]} + q_{k+2}^{[n-1]} + 2q_k^{[n-1]} - \{B^{[n-2]}\}(p_{k-1}^{[n]} + p_{k+1}^{[n]}).
\]

(2.23)
3 Domain Decomposition Algorithms

In this section we discuss some non-overlapping domain decomposition (DD) techniques applied to the separable elliptic equation

$$\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = f(x,y),$$  \hspace{1cm} (3.1)

with Dirichlet boundary conditions on the T-shaped domain $\Omega$ pictured in Figure 2. The problem is discretized using the standard five-point stencil on a naturally ordered grid on $\Omega$.

![Figure 2: The domain $\Omega$.](image)

We use a 2-subdomain approach. Let the upper and lower rectangles of $\Omega$ be $\Omega_1$ and $\Omega_2$, respectively, and let $\Gamma$ be the interface between them. We discretize the problem with a 5-point finite difference method, choosing $n$ grid points in the vertical direction (at the tallest point of $\Omega$) and $m$ grid points in the horizontal direction (at the widest point of $\Omega$). If we use natural ordering to number the unknowns for the internal points of the subdomains first and those in the interface $\Gamma$ last, then the discrete solution vector $U = (u_1, u_2, u_3)^T$ satisfies the linear system,

$$Au = b,$$  \hspace{1cm} (3.2)

which can be expressed in block form as:

$$\begin{bmatrix}
A_{11} & A_{13} \\
A_{22} & A_{23} \\
A_{13}^T & A_{23}^T & A_{33}
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
u_3
\end{bmatrix} = 
\begin{bmatrix}
b_1 \\
b_2 \\
b_3
\end{bmatrix}.$$  \hspace{1cm} (3.3)

The system (3.3) can be solved by block Gaussian elimination as follows:

Compute

$$w_1 = A_{11}^{-1} b_1,$$  \hspace{1cm} (3.4)

and

$$w_2 = A_{22}^{-1} b_2,$$  \hspace{1cm} (3.5)

and solve

$$Cu_3 = b_3 - A_{13}^T w_1 - A_{23}^T w_2,$$  \hspace{1cm} (3.6)

where
10  High Performance Computing

\[ C = A_{33} - A_{13}^T A_{11}^{-1} A_{13} - A_{23}^T A_{22}^{-1} A_{23} . \]  

(3.7)

Then compute

\[ u_1 = w_1 - A_{11}^{-1} A_{13} u_3 , \]  

(3.8)

and

\[ u_2 = w_2 - A_{22}^{-1} A_{23} u_3 . \]  

(3.9)

The matrix \( C \) in (3.7) is the Schur complement of \( A_{33} \) in \( A \) and is also called the capacitance matrix. It corresponds to the reduction of the operator in \( \Omega \) to an operator on the interface \( \Gamma \). The order of \( C \) is \( m_\Gamma \), the number of grid points in \( \Gamma \).

Aside from (3.6), the linear systems requiring solution involve only one subdomain at a time; these can be done in parallel, with each subdomain solver assigned to a single processor.

The general strategy is to partition \( \Omega \) into subrectangles \( \Omega_i \), followed by a process consisting of solving problems on these \( \Omega_i \), assigning one processor per subdomain, and combining the partial solutions. Some (relatively minor) calculations remain, and this work is not distributed; instead it is left for the main processor. We partition \( \Omega \) into either horizontal strips, vertical strips, or 'blocks'. The important factors influencing the performance of approaches using these different decompositions are the shape of the subrectangles, the size of the interface between them, and, if the solution method is based on PCG, the type of preconditioner used. The subdomains must be rectangles to make use of the BSR method outlined in Section 2, so our 2-subdomain algorithm employs only the natural partition, into the lower and upper rectangles.

Although we apply the algorithms on irregular domains, such as that in Figure 2, the ideas here could also be applied to problems on a rectangular region; in this case the DD algorithm can exploit the available knowledge about the spectrum of the interface operator.

4 The Preconditioned Conjugate Gradient Method

The selected method for solving the linear system (3.3) must take advantage of the sparseness and symmetry of the problem as well as minimizing the computational effort that is needed for solving the system of equations.

The preconditioned conjugate gradient method is specifically formulated to efficiently solve the algebraic system \( Au=b \) resulting from the above discretization. The PCG method is an iterative algorithm in which the following steps are computed at each iteration:

for \( k \geq 1 \), solve

\[ Mz_k = r_k , \]  

(4.1)

\[ \beta_k = (Mz_k, z_k)/(Mz_{k-1}, z_{k-1}) , \]  

\[ p_k = z_k + \beta_k p_{k-1} , \]  

(4.2)
where \( x_0 \) is given, \( r_0 = b - Ax_0 \), \( p_0 \) is arbitrary, and \( \beta_0 = 0 \). The convergence of the PCG method is determined both by the clustering of the eigenvalues and the condition number of \( M^{-1}A \), and thus critically depends on the selection of \( M \). Good preconditioners \( M \) are: 1) symmetric and positive definite and significantly reduce the condition number of the system, 2) are less expensive in solving \( Mz_k = r_k \) rather than \( Au = b \), and 3) do not significantly increase the amount of storage relative to the storage needed to solve \( Au = b \). By appropriately preconditioning the system we greatly reduce the amount of work expended in the computation of the solution \( u \).

Our goal is to develop a good preconditioner by domain decomposition, or substructuring, methods. The basic approach in domain decomposition techniques is to break up the domain of integration into many parts, solve the corresponding equation on each part, then construct the global solution from these local solutions.

### 5 A New Domain Decomposition (DD) Preconditioner

Many authors (Dryja [1], Golub and Mayers [2], Concus, Golub and Meurant [3]) have proposed preconditioners for the block diagonal bordered linear system (3.3) by considering only the Schur complement system. Here we introduce a new preconditioner for the complete system (3.3) based on ILU factorisation techniques which if it is optimal may be advantageous.

From a simplified normalised form of the D.D. matrix given by eqn. (3.3), i.e.,

\[
\begin{bmatrix}
    I & I & I \\
    T & T & T \\
    T & T & T
\end{bmatrix}
\begin{bmatrix}
    u_1 \\
    u_2 \\
    u_3
\end{bmatrix} = \begin{bmatrix}
    b_1 \\
    b_2 \\
    b_3
\end{bmatrix},
\]

a direct block ILU factorisation of \( A \) form can be obtained, i.e.,

\[
\begin{bmatrix}
    I & I & I \\
    T & T & T \\
    T & T & T
\end{bmatrix}
\begin{bmatrix}
    I & I & I \\
    I & I & I \\
    I & I & I
\end{bmatrix}
\]

from which a block preconditioner matrix \( M \) can be developed which is quite easy to implement.

Thus, if we consider the following matrix \( M \) to be composed of the triangular factors,
12 High Performance Computing

\[
M = \begin{bmatrix}
  A_{11} & A_{22} & A_{13} \\
  A_{22} & A_{23} & A_{33} \\
  A_{13}^T & A_{23}^T & A_{33}
\end{bmatrix}
\]

\[
\begin{bmatrix}
  I & A_{11}^{-1} A_{13} \\
  I & A_{22}^{-1} A_{23} \\
  I
\end{bmatrix}
\]

(5.3)

where \( C = A_{13}^T A_{11}^{-1} A_{13} + A_{23}^T A_{22}^{-1} A_{23} \). Thus \( M \) can be considered to satisfy the essential criteria for a good preconditioner for it is closely related to \( A \). The matrix elements \( A_{11}^{-1} A_{13} \) and \( A_{22}^{-1} A_{23} \) can be evaluated by use of the tridiagonal solvers \( A_{11} \) and \( A_{22} \) on the columns of \( A_{13} \) and \( A_{23} \) respectively. In addition, the matrix \( C \) can be evaluated in a similar manner. However in view of the excessive amount of computation involved, then an alternative strategy would be to use an arbitrary banded matrix for \( C \) derived from the corresponding terms of \( A_{11}^{-1} \) and \( A_{22}^{-1} \).

References


