Robust iterative solvers for sparse linear systems

P.M. de Zeeuw
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands
E-mail: pauldz@cwi.nl

Abstract
Firstly we consider a general linear 2nd order elliptic PDE in two dimensions on a bounded domain. The diffusion coefficients and the righthandside function are allowed to be discontinuous across internal boundaries. We describe a multigrid method and a conjugate gradient method respectively, both meant to solve the discretized equation. Secondly we extend the scope of the described conjugate gradient method to a system of PDEs. Our first concern is robustness all along, our second concern efficiency.

1 Introduction

We consider a general linear 2nd order elliptic PDE in two dimensions

\[ - \nabla \cdot (D \nabla u) + b_1(x, y) \frac{\partial u}{\partial x} + b_2(x, y) \frac{\partial u}{\partial y} + c(x, y)u = f(x, y) \]  

(1)

on a bounded domain \( \Omega \). \( D(x, y) \) is a positive definite \( 2 \times 2 \) matrix function and \( c(x, y) \geq 0 \). \( D(x, y) \) and \( f(x, y) \) are allowed to be discontinuous across internal boundaries in \( \Omega \). Already for a long time various incomplete decompositions have been applied fruitfully both as smoother in multigrid methods and as preconditioner in conjugate gradient methods. We choose the incomplete line LU decomposition (ILLU) which has been originated by Underwood [10], and has also been proposed and elaborated upon by others [3,4,7,8]. We give an outline of ILLU in section (2), in [6,9] an extensive description of this method can be found. In section (3) the blackbox multigrid solver MGD9V is described. Within MGD9V we use ILLU as
smoother and also matrix-dependent grid transfer operators to ensure robustness. Recently Van der Vorst developed the Bi-CGSTAB method [11] which is descended from the method of Induced Dimension Reduction (IDR) as developed by Sonneveld [14]. A particular version of this method is described in section (4). In section (4.2) we generalize ILLU for the case of discretized systems of PDEs.

2 Incomplete line LU

Here we repeat the general outline of the method. We assume to have a discretization on a rectangular computational grid that may be curvilinear in the geometrical sense. Let $n_x$ denote the number of vertical lines in the case of a vertex-centered discretization. Likewise we define $n_y$, corresponding with the $y$-direction. Further we assume the common five point coupling (as with central differences) or nine point coupling (as with bilinear finite elements). With these assumptions we obtain after discretization a block tridiagonal linear system of the form

$$Ax = b$$

(2)

where

$$A = \begin{pmatrix} D_1 & U_1 & & \\ L_2 & D_2 & U_2 & \\ & L_3 & D_3 & \\ & & & \ddots & \ddots & \ddots \\ & & & & & D_{ny} \end{pmatrix}$$

(3)

The block $D_j$ has the tridiagonal form:

$$D_j = \begin{pmatrix} d_{1j} & u_{1j} \\ l_{2j} & d_{2j} & u_{2j} \\ l_{3j} & d_{3j} & \ddots & \ddots \\ & & & d_{nj} \end{pmatrix}$$

(4)

The blocks $L_j, D_j$ and $U_j$ are of dimension $n_x$. In case of five point stencils the blocks $L_j$ and $U_j$ are diagonal matrices, in case of nine point stencils these blocks are tridiagonal. The ILLU-decomposition is defined by
\[ L_j(j = 2, \ldots, n_y), D_j(j = 1, \ldots, n_y), U_j(j = 1, \ldots, n_y - 1), \text{ with} \]
\[ D_1 = D_1, \]
\[ D_j = D_j - \text{tridiag}(L_j D_{j-1}^{-1} U_{j-1}), \quad j = 2(1)n_y. \]  

The operator `tridiag()` forces a block (by clipping) into the sparsity pattern of the \( D_j \). Without this particular operator, the factorization of \( A \) would be a complete one.

### 3 A blackbox multigrid solver

The general concept of multigrid methods is assumed to be known [1,5]. We have a set of increasingly coarser grids:

\[ \Omega_i, \Omega_{i-1}, \ldots, \Omega_k, \ldots, \Omega_1. \]

The discretization on the finest grid \( \Omega_l \) evokes the linear system

\[ A_l u_l = f_l \]  

We have to define our specific choice for the prolongation operator \( P_k \), the restriction operator \( R_{k-1} \) and the coarse grid matrices \( A_{k-1} \) (\( k = 2, \ldots, l \)). For the restriction we choose

\[ R_{k-1} = P_k^T. \]

Further we choose the Galerkin approximation

\[ A_{k-1} = R_{k-1} A_k P_k. \]

Hence, once \( P_k \) has been chosen, \( R_{k-1} \) and \( A_{k-1} \) follow automatically. Definition (8) is an essential ingredient for a blackbox algorithm because now a user only needs to define his problem on the finest grid (for a discussion on the concept of multigrid blackbox solvers see [13]). A standard choice for the prolongation is bilinear interpolation. This works out fine for a certain class of problems, but the multigrid rate of convergence deteriorates severely at two different instances:

1. The diffusion coefficients in \( D(x,y) \) are discontinuous across certain interfaces between subdomains.
2. The convection term is dominating, roughly speaking $h\|b\| > \|D\|$ with $h$ the meshsize.

For an elaboration on the first instance we refer to [2]. A first example that the second instance causes divergence, can be found in [17]. A new prolongation operator has been proposed in [16], able to handle both the case of dominant convection (in general directions) and interface problems at the same time. The prolongation weights are determined by decomposing the matrix $A_k$ in its symmetric and antisymmetric part. The symmetric part is supposed to correspond with diffusion and the zeroth order term, the antisymmetric part with convection. Thereupon we reconstruct the diffusion and zeroth order coefficients, and the convection coefficients. From that the prolongation weights are calculated. We adhere to (8) and (9), though the implementation of the latter is far from trivial. The actual computation of the coarse grid matrices takes less work than the ILLU-decompositions. The above is employed in the code MGD9V (de Zeeuw), this code uses the sawtooth multigrid correction scheme [12] and ILLU for smoother. For a detailed motivation of the prolongation and a description of the code, together with numerical experiments to illustrate its good behaviour, see [16]. The code has been written in standard FORTRAN 77 and has the outer appearance of a NAG-routine. Under (soft) conditions, the code is available from the author.

4 Versions of Bi-CGSTAB

In this section we report the use of an application of Bi-CGSTAB (Van der Vorst [11]). We have the linear system

$$Ax = b.$$  \hspace{1cm} (10)

Firstly we consider an application for the same type of problems as described in the previous section. Secondly we enhance the algorithm for systems of PDEs as they arise e.g. within the context of semiconductor equations.

4.1 Bi-CGSTAB preconditioned from the left

We introduce a variant of Bi-CGSTAB, based on preconditioning from the left.

left-Bi-CGSTAB:
\[ x_0 = \frac{b}{A}; \]

to \( \sigma \) do \( ILLU(A, x_0, b) \);
\[ r_0 = b - Ax_0; \]
\[ \tilde{r}_0 = 0; \] to \( \sigma \) do \( ILLU(A, \tilde{r}_0, r_0) \);
\[ \rho_0 = \alpha = \omega_0 = 1; \]
\[ v_0 = p_0 = q; \]

for \( i = 1, 2, 3, \ldots \)

\[ \rho_i = (\tilde{r}_0, \tilde{r}_i-1); \beta = (\rho_i/\rho_{i-1})(\alpha/\omega_{i-1}); \]
\[ p_i = \tilde{r}_{i-1} + \beta(\rho_{i-1} - \omega_{i-1}v_{i-1}); \]
\[ v_i = 0; \] to \( \sigma \) do \( ILLU(A, v_i, Ap_i) \)
\[ \alpha = \rho_i/(\tilde{r}_0, v_i); \]
\[ s = \tilde{r}_{i-1} - \alpha v_i; \]
\[ t = \tilde{r}_0; \] to \( \sigma \) do \( ILLU(A, t, As) \)
\[ \omega_i = (t, s)/(t, t); \]
\[ x_i = x_{i-1} + \alpha p_i + \omega_is; \]
\[ \tilde{r}_i = s - \omega_it; \]

end

At the \( i \)-th sweep this scheme delivers some approximation \( x_i \) of the solution \( x \) of (10). An important advantage of this version of Bi-CGSTAB is that \( \tilde{r}_i \) is a properly scaled residual. The \( \tilde{r}_i \) will be a close approximation of the error rather than the residual. This is of importance within the context of semiconductor problems. Jacobians originating from this problems depict entries that differ in orders of magnitude. This make it hard to decide whether the residual is small or not. For making this kind of decisions (criterions etc.) the \( \tilde{r}_i \) as approximation of the error, is a more convenient tool. For a detailed discussion and numerical results see ([15]).

4.2 ILLU for a system of PDEs

Suppose we have a system of \( n \) coupled PDEs. For \( n = 1 \) we obtain the matrix \( A \) as described in section (2). For \( n > 1 \) the entries of the matrix \( A \) become blocks of dimension \( n \) instead of scalars. For the construction of the incomplete line decomposition we have to replace operations on scalars \( x \) and \( y \) by operations on matrices \( X \) and \( Y \) of dimension \( n \) as follows:

\[ x \pm y \rightarrow X \pm Y \]
\[ xy \rightarrow XY \]
\[ x/y \rightarrow XY^{-1} \]
However, an important difference is that multiplication is no longer commutative. The H.I.U-decomposition thus obtained, is used as preconditioner in Bi-CGSTAB. For applications on semiconductor problems see [15].

References


ERCIM WORKSHOPS
on
Numerical Linear Algebra
Software Quality Principles and Techniques
Theoretical and Experimental Aspects of Knowledge Representation

Pisa - Palazzo dei Congressi, May 21-22, 1992

ORGANIZED BY:
- Area di Ricerca di Pisa - CNR
- Istituto di Elaborazione della Informazione - CNR
- Istituto CNUCE - CNR

SPONSORED BY:
- Area di Ricerca di Pisa
- Comitato Nazionale Scienze Matematiche
- Comitato Nazionale Scienze Fisiche
- Comitato Nazionale per la Ricerca Tecnologica
- Comitato Nazionale per la Scienza e le Tecnologie dell'Informazione