Incomplete Line LU as smoother and as preconditioner

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

Results are reported for Incomplete line LU (ILLU) in two different roles. One role is the role of smoother in a multigrid method for the solution of linear systems resulting from the 9-point discretization of a general linear second-order elliptic PDE in two dimensions. Together with features like matrix-dependent gridtransferoperators we obtain a blackbox multigrid solver (MGD9V). Another role for ILLU is as preconditioner in a stabilised bi-cg method (Bi-CGSTAB), as recently developed by Van der Vorst. In this role the preconditioner can easily be generalized for a discretized system of PDEs. A comparison is made between the two different roles.

1 Introduction

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

$$-\nabla \cdot (D\nabla u) + b_1(x,y)\frac{\partial u}{\partial x} + b_2(x,y)\frac{\partial u}{\partial y} + c(x,y)u = f(x,y) \tag{1}$$

on a bounded domain. $D(x,y)$ is a positive definite 2×2 matrix function and $c(x,y) \geq 0$. The discretized equation can be solved both by conjugate gradient and multigrid methods. Already for a long time incomplete decompositions have been applied fruitfully within these type of methods. In section 2 a description is given of the specific incomplete decomposition that we consider in this paper. The use of ILLU as smoother in multigrid methods is considered in section 3. Mere application of ILLU is not sufficient to ensure robustness and matrix-dependent gridtransfer operators are needed as well. Recently Van der Vorst developed the Bi-CGSTAB method [11] which is a variant of the method of Induced Dimension Reduction (IDR) as developed by Sonneveld [14]. We consider the use of ILLU as preconditioner for this method in section 4 and make a comparison with multigrid. In section 5 we generalize ILLU for the case of discretized systems of PDEs and we consider the application within Bi-CGSTAB. We end up with concluding remarks in section 6.

2 Incomplete line LU

The incomplete line LU decomposition (ILLU) has been originated by Underwood [10], and has also been proposed and elaborated upon by Concus, Golub and Meurant [4], Axelsson [2, 3], Meijerink [8] and others. In [6, 9] an extensive description of the method can be found. 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 volumes in the x-direction in the case of a cell-centered discretization or 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

$$Ax = b \tag{2}$$

where

$$A = \begin{pmatrix} D_1 & U_1 & & & & \\ L_2 & D_2 & U_2 & & & \\ & L_3 & D_3 & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & & & & \cdot & D_{n_y} \end{pmatrix}. \tag{3}$$

The block D_j has the tridiagonal form:

$$D_j = \begin{pmatrix} d_{1j} & u_{1j} & & & & \\ l_{2j} & d_{2j} & u_{2j} & & & \\ & l_{3j} & d_{3j} & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & & & & \cdot & d_{n_x j} \end{pmatrix}. \tag{4}$$

The blocks L_j and U_j are of dimension n_x, just like D_j. 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 point of the ILLU-method is to make an incomplete factorization of A by the following formulae:

$$\overline{D}_1 = D_1, \tag{5}$$

$$\overline{D}_j = D_j - \mathbf{tridiag}(L_j \overline{D}_{j-1}^{-1} U_{j-1}), \quad j = 2(1)n_y. \tag{6}$$

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.

Performing one ILLU-relaxation sweep requires the following steps:

ILLU-sweep:

$r = b - Ax$;
$z_1 = r_1$;
$z_j = r_j - L_j \overline{D}_{j-1}^{-1} z_{j-1}, \quad j = 2(1)n_y$;
$c_{n_y} = \overline{D}_{n_y}^{-1} z_{n_y}$;
$c_j = \overline{D}_j^{-1}(z_j - U_j c_{j+1}), \quad j = n_y - 1(-1)1$;
$x_{new} = x + c$;

3 ILLU and multigrid

We adhere to the assumptions about the grid and the discretization as made in the previous section. The general concept of multigrid methods is assumed to be known. We have a set of increasingly coarser grids:

$$\Omega_l, \Omega_{l-1}, \ldots, \Omega_k, \ldots, \Omega_1.$$

The discretization on the finest grid Ω_l evokes the linear system

$$A_l u_l = f_l \ . \tag{7}$$

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

$$R_{k-1} = P_k^T. \tag{8}$$

Further we choose the Galerkin approximation

$$A_{k-1} = R_{k-1} A_k P_k. \tag{9}$$

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]). We may consider possible choices for the prolongation operator. A standard choice is bilinear interpolation. This amounts to taking an equal average of values of the solution at neighbouring coarse gridpoints (see Figure 1).

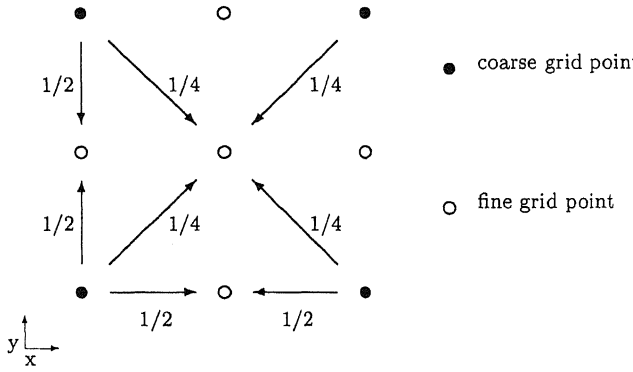

Figure 1: Bilinear prolongation.

At the gridpoints of the fine grid that coincide with the coarse grid we take identical values. One algorithm incorporating the foregoing prolongation is MGSYM [16]. It is a sawtooth multigrid correction scheme [12] which is furnished with ILLU as relaxation method. This algorithm works fine for a considerable class of problems. Yet 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 [1, 7] and also to [5, § 10.3] where a one dimensional interface problem is analysed. A first example that the second instance causes divergence can be found in [17]. A complete analysis for this instance and a remedy can be found in [16], here the results of the analysis are illustrated when applied to the following simple example.

3.1 Example of Galerkin approximation for a convection dominated problem

Consider the following linear operator:

$$Au \equiv -\epsilon \Delta u + \frac{\partial u}{\partial x} \quad . \tag{10}$$

For vanishing diffusion the following stencil is the result from a simple upwind discretization on a grid with meshsize equal to 1:

$$A_l \sim \begin{bmatrix} 0 & 0 & 0 \\ -1 & +1 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{11}$$

By repeatedly applying (8) and (9) for the standard choice we obtain on the n times coarsened grid Ω_{l-n} the stencil:

$$A_{l-n} \sim \begin{bmatrix} -\frac{1}{12} & \frac{1}{6} & -\frac{1}{12} \\ -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} \\ -\frac{1}{12} & \frac{1}{6} & -\frac{1}{12} \end{bmatrix} + 2^n \begin{bmatrix} -\frac{1}{12} & 0 & +\frac{1}{12} \\ -\frac{1}{3} & 0 & +\frac{1}{3} \\ -\frac{1}{12} & 0 & +\frac{1}{12} \end{bmatrix} + 2^{-n} \cdots \tag{12}$$

The two stencils are the same ones as evoked by discretization with bilinear finite elements of a diffusion and convection term in the $x-$direction. On the right hand side there is a remainder that decreases exponentially with n. What matters is the observation that the convection-stencil increases exponentially with n. The spurious solutions thus created on the coarse grids will affect severely the convergence of the multigrid algorithm as a whole.

A remedy for this particular example is to use an upwind prolongation instead, meaning here that only information from the left hand side is accepted. This corresponds to a prolongation with biassed weights, see Figure 2.

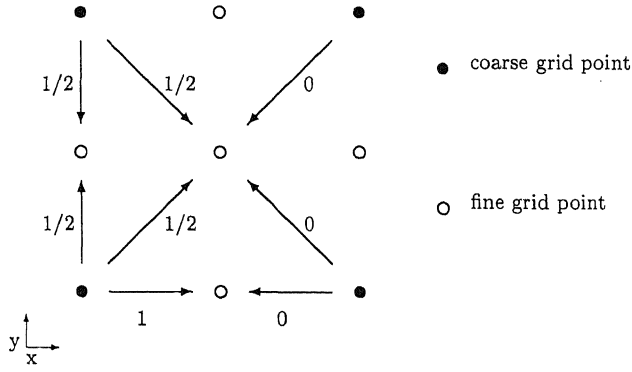

Figure 2: Example of upwind prolongation.

Now, by repeatedly applying (8) and (9) we obtain on the n times coarsened grid Ω_{l-n} the stencil:

$$A_{l-n} \sim 2^n \begin{bmatrix} 0 & 0 & 0 \\ -1 & +1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + 2^{-n} \cdots \tag{13}$$

and the multigrid algorithm as a whole converges.

3.2 Matrix-dependent prolongations and restrictions

In [16] a prolongation operator has been proposed, able to handle both the case of dominant convection (in general directions) and interface problems at the same time. Here we give only an outline of the method. The grid Ω_k is split into four disjunct subgrids as follows:

$\Omega_{k,(0,0)} \equiv \Omega_{k-1}$,

$\Omega_{k,(1,0)} \equiv \{(x + h_k, y) \in \Omega_k \mid (x, y) \in \Omega_{k-1}\}$,

$\Omega_{k,(0,1)} \equiv \{(x, y + h_k) \in \Omega_k \mid (x, y) \in \Omega_{k-1}\}$,

$\Omega_{k,(1,1)} \equiv \{(x + h_k, y + h_k) \in \Omega_k \mid (x, y) \in \Omega_{k-1}\}$,

where h_k is the meshsize of grid Ω_k.

1. Let $\xi \in \Omega_{k,(1,0)}$ or $\xi \in \Omega_{k,(0,1)}$ be a point where we have to interpolate a coarse g correction. Decompose the matrix A_k in its symmetric and antisymmetric part. T symmetric part S_k is supposed to correspond with diffusion and the zeroth order ter the antisymmetric part T_k with convection.

2. Reconstruct the diffusion and zeroth order coefficients at ξ from S_k, and the convect coefficients from T_k.

3. Use expressions that define an optimal choice for each sample of a set of degenerated ca for A_k.

4. At the fine grid points in $\Omega_{k,(0,0)}$ we adopt the values on Ω_{k-1}.

5. At the fine grid points in $\Omega_{k,(1,1)}$ we solve the homogeneous equation to obtain the corr tion.

Applying the derived formulae to the particular example of section 3.1 we obtain the upwi prolongation in Figure 2. Also here we adhere to (8) and (9), though the implementation the latter is far from trivial. The actual computation of the coarse grid matrices takes l work than the ILLU-decompositions. The above is employed in the code MGD9V (de Zeeu this code uses the sawtooth multigrid correction scheme [12] and ILLU for smoother. Fo detailed motivation of the prolongation and a description of the code, together with numeri experiments to illustrate its good behaviour, see [16].

4 ILLU and Bi-CGSTAB

In this section we consider the use of ILLU as preconditioner in Bi-CGSTAB (Van Vorst [11]). A numerical example and comparison with the multigrid method MGD9V is p sented at the end of this section. We have the linear system

$$Ax = b. \tag{1}$$

We consider two versions of Bi-CGSTAB: one with preconditioning from the right and one wi preconditioning from the left. The first version reads:

right-Bi-CGSTAB:

$x_0 = \underline{0}$;
to σ do $RELAX(A, x_0, b)$;
$r_0 = b - Ax_0$;
$\rho_0 = \alpha = \omega_0 = 1$;
$v_0 = p_0 = \underline{0}$;
for $i = 1, 2, 3, \ldots$
  $\rho_i = (r_0, r_{i-1})$; $\beta = (\rho_i/\rho_{i-1})(\alpha/\omega_{i-1})$;
  $p_i = r_{i-1} + \beta(p_{i-1} - \omega_{i-1} v_{i-1})$;
  $y = \underline{0}$; to σ do $RELAX(A, y, p_i)$

$v_i = Ay$
$\alpha = \rho_i/(r_0, v_i)$;
$s = r_{i-1} - \alpha v_i$;
$z = \underline{0}$; to σ do $RELAX(A, z, s)$
$t = Az$
$\omega_i = (t, s)/(t, t)$;
$x_i = x_{i-1} + \alpha y + \omega_i z$;
$r_i = s - \omega_i t$;
end

At the i-th sweep this scheme delivers some approximation x_i of the solution x of (14), and the corresponding residual r_i. This residual is called the *updated* residual for the way in which it is computed.

Note that compared with preconditioning from both sides, we have gained a degree of freedom, for we can choose $\sigma > 1$. When we apply ILLU for RELAX we have to compute $(2+2(\sigma-1))$ matrix times vector operations for each i. Therefore, choosing $\sigma = 2$ instead of $\sigma = 1$ roughly doubles the amount of work per sweep. Yet, various numerical experiments have indicated that $\sigma = 2$ provides a more efficient choice for this parameter because of the faster convergence. Still higher values of σ decrease the effiency.

Note further that the initial guess for x_0 is determined by performing σ ILLU-sweeps on the zero solution.

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

left-Bi-CGSTAB:

$x_0 = \underline{0}$;
to σ do $RELAX(A, x_0, b)$;
$r_0 = b - Ax_0$;
$\tilde{r}_0 = \underline{0}$; to σ do $RELAX(A, \tilde{r}_0, r_0)$;
$\rho_0 = \alpha = \omega_0 = 1$;
$v_0 = p_0 = \underline{0}$;
for $i = 1, 2, 3, ...$
$\rho_i = (\tilde{r}_0, \tilde{r}_{i-1})$; $\beta = (\rho_i/\rho_{i-1})(\alpha/\omega_{i-1})$;
$p_i = \tilde{r}_{i-1} + \beta(p_{i-1} - \omega_{i-1} v_{i-1})$;
$v_i = \underline{0}$; to σ do $RELAX(A, v_i, Ap_i)$
$\alpha = \rho_i/(\tilde{r}_0, v_i)$;
$s = \tilde{r}_{i-1} - \alpha v_i$;
$t = \underline{0}$; to σ do $RELAX(A, t, As)$
$\omega_i = (t, s)/(t, t)$;
$x_i = x_{i-1} + \alpha p_i + \omega_i s$;
$\tilde{r}_i = s - \omega_i t$;
end

At the i-th sweep this scheme delivers some approximation x_i of the solution x of (14). An important feature of this second scheme is that the gridfunction $\tilde{r}_i$ is *not* the updated residual belonging to system (14). Instead, it is the updated residual belonging to a left-preconditioned version of this system. The preconditioning consists of the approximate inverse of A corresponding to σ relaxation sweeps. An important advantage of this new version of Bi-CGSTAB is that $\tilde{r}_i$ is a properly *scaled* residual. In fact, when applying ILLU for RELAX, $\tilde{r}_i$ will be a close approximation of the *error* rather than the residual. This is of importance e.g. 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 (see [15]).

4.1 Problem 1

The following testproblem has been proposed by Van der Vorst [11]. It is a simplified aquifer-problem, the convection-diffusion equation reads

$$-\nabla \cdot (D\nabla u) + b(x,y)\frac{\partial u}{\partial x} = f(x,y)$$
$$\Omega = (0,1) \times (0,1)$$

where the diffusion coefficient function D can be read from Figure 2, and

$$b(x,y) = 2\exp(2(x^2+y^2)).$$

We have Dirichlet boundary conditions: $u = 1$ on $\partial\Omega$ except for $y = 1$ where $u = 0$. The function $f(x,y)$ equals zero everywhere, except for the small (dashed) square in the centre where $f(x,y) = 100$. We use meshsize $h = 1/130$, leading to a system with 129^2 unknowns.

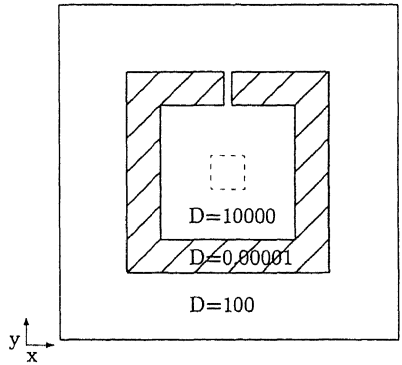

Figure 3: Geometry of Van der Vorst's aquifer-problem.

Here we use the same discretization as chosen by Van der Vorst, i.e. central differences. We observe that the diffusion coefficients are discontinuous across several interfaces and that at the shaded subdomain the convection is dominant.

In Figure 4 we see the 10-logarithm of the Euclidean norm of the iteration vectors r_i versus consumed workunits. One Bi-CGSTAB sweep ($\sigma = 2$) takes 2 workunits, one MGD9V cycle takes 1 workunit. For MGD9V and **right-Bi-CGSTAB** the r_i are residuals, as for **left-Bi-CGSTAB** the r_i are *scaled* residuals. We observe that from the viewpoint of efficiency, MGD9V has a clear advantage over the Bi-CGSTAB algorithms. Along with ILLU as relaxation method this is due to advanced features like matrix-dependent gridtransfers and an automatic Galerkin approximation of coarse grid matrices. However, it is far from trivial to generalize these features from the case of a scalar equation to the case of a system of coupled equations. As for Bi-CGSTAB, we show in the next section that, with a proper approach, the said generalization is rather straightforward.

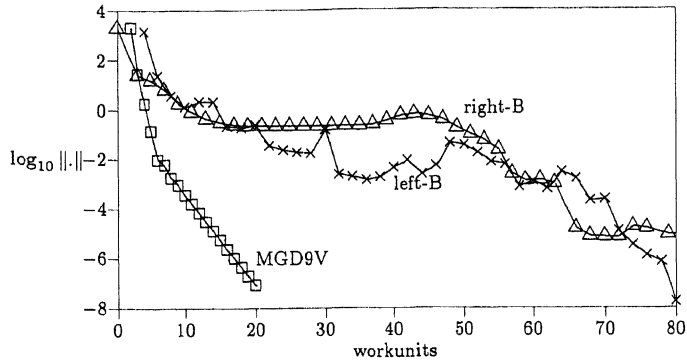

Figure 4: Convergence history of MGD9V and Bi-CGSTAB for the aquifer problem.

5 ILLU for discretized systems 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. At the actual working-out of the formulae for the decomposition, as known and described (e.g. [6]) for the scalar case, we have to replace operations on scalars x and y by operations on matrices X and Y of dimension n as follows:

$$\begin{aligned} x \pm y &\longrightarrow X \pm Y \\ xy &\longrightarrow XY \\ x/y &\longrightarrow XY^{-1}. \end{aligned}$$

However, an important difference is that multiplication is no longer commutative and therefore the working-out of the formulae needs careful overhauling.

In section 4 we mentioned the important advantage that in **left-Bi-CGSTAB** a properly scaled residual was computed for each i. Note that with the preconditioner for discretized coupled systems of PDEs, as developed in this section, we manage to do the scaling also over this coupling. For applications on semiconductor problems see [15].

6 Concluding remarks

We have given a description of a robust blackbox multigrid solver (MGD9V). Apart from ILLU as robust smoother, it features matrix-dependent gridtransferoperators in order to tackle problems with discontinuous diffusion and/or dominant convection. These features are hard to generalize to the case of a system of coupled PDEs. The solver (written in standard FORTRAN 77) is available from the author. We have also given descriptions of particular versions of the Bi-CGSTAB algorithm of Van der Vorst, furnished with ILLU as robust preconditioner. These versions have been generalized to the case of a system of PDEs. The left preconditioned version is especially suited for problems stemming from the field of semiconductor equations. This version might be of equal importance in other applications, e.g. fluid dynamics.

Acknowledgements

The author wishes to thank Prof. Dr. Henk A. Van der Vorst for providing his subroutine that discretizes the aquifer problem of section 4.1.

References

[1] R.E. Alcouffe, A. Brandt, J.E. Dendy Jr. and J.W. Painter, *The multi-grid method for the diffusion equation with strongly discontinuous coefficients*, SIAM J. Sci. Statist. Comput. **2**(4), pp. 430-454, 1981.

[2] O. Axelsson, *A General Incomplete Block-Matrix Factorization Method*, Linear Algebra and its Applications **74**, pp. 179-190, 1986.

[3] O. Axelsson, *A survey of preconditioned iterative methods for linear systems of algebraic equations*, BIT **25**, pp. 166-187, 1985.

[4] P. Concus, G.H. Golub and G. Meurant, *Block preconditioning for the conjugate gradient method*, SIAM J. Sci. Statist. Comput., **6** (1), pp. 220-252, 1985.

[5] W. Hackbusch, *Multi-Grid Methods and Applications*, Springer Ser. Comput. Math. **4**, Berlin, 1985.

[6] P.W. Hemker and P.M. de Zeeuw, *Some implementations of multigrid linear system solvers* in: D.J. Paddon and H. Holstein (Eds.) *Multigrid methods for integral and differential equations*, Inst. Math. Appl. Conf. Ser. New Ser. (Oxford Univ. Press, New York), pp. 85-116, 1985.

[7] R. Kettler, *Analysis and comparison of relaxation schemes in robust multigrid and preconditioned conjugate gradient methods*, in: W. Hackbusch and U. Trottenberg (Eds.), Lecture Notes in Math. **960** (Springer, Berlin), pp. 502-534, 1981.

[8] R. Kettler and J.A. Meijerink, *A multigrid method and a combined multigrid-conjugate gradient method for elliptic problems with strongly discontinuous coefficients in general domains*, Shell Publ. 604, KSEPL, Rijswijk, The Netherlands, 1981.

[9] P. Sonneveld, P. Wesseling and P.M. de Zeeuw, *Multigrid and conjugate gradient methods as convergence acceleration techniques*, in: D.J. Paddon and H. Holstein (Eds.) *Multigrid methods for integral and differential equations*, Inst. Math. Appl. Conf. Ser. New Ser. (Oxford Univ. Press, New York), pp. 117-167, 1985.

[10] R.R. Underwood, *An approximate factorization procedure based on the block Cholesky decomposition and its use with the conjugate gradient method*, Report NEDO-11386, General Electric Co., Nuclear Energy Div., San Jose, California, 1976.

[11] H.A. Van der Vorst, *Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput. **13**(2), 1992.

[12] P. Wesseling, *A robust and efficient multigrid method*, in: W. Hackbusch and U. Trottenberg (Eds.), Lecture Notes in Math. **960** (Springer, Berlin), pp. 614-630, 1981.

[13] P. Wesseling, *An Introduction to Multigrid Methods*, John Wiley & Sons Ltd, Chichester, England, 1991.

[14] P. Wesseling, P. Sonneveld, *Numerical experiments with a multiple grid and a preconditioned Lanczos type method*, in: R. Rautmann (ed.), *Approximation methods for Navier-Stokes problems*, Proceedings, Paderborn, Germany 1979, Lecture Notes in Mathematics, Springer, Berlin-etc., 1980.

[15] P.M. de Zeeuw, *Incomplete line LU for discretized coupled PDEs as preconditioner in Bi-CGSTAB*, Report, Centre for Mathematics and Computer Science, Amsterdam, to appear.

[16] P.M. de Zeeuw, *Matrix-dependent prolongations and restrictions in a blackbox multigrid solver*, Journal of Computational and Applied Mathematics, **33**, pp. 1-27, 1990.

[17] P.M. de Zeeuw and E.J. van Asselt, *The convergence rate of multi-level algorithms applied to the convection-diffusion equation*, SIAM J. Sci. Statist. Comput. **6**(2), pp. 492-503, 1985.