The 'weighted modification' incomplete factorisation method, Lapack Working Note 145, UT-CS-99-436

Victor Eijkhout*

1 December 1999

Abstract

Incomplete factorisation methods suffer from possible breakdown in the sense that pivots can become zero or negative. We propose a new factorisation that does not suffer from this defect, and that preserves several other useful properties. In numerical tests this method is seen to be more reliable than existing methods.

1 Introduction

For the efficient solution of sparse linear systems $Au = b$ by an iterative method, the choice of a proper preconditioner is crucial. A preconditioner is a matrix M that approximates A, but for which solving the system $Mu = b$ is computationally cheap. In addition, M itself should be easily constructable.

Since the original coefficient matrix A is sparse, people have sought to construct sparse factorisations $M = LU \approx A$. The exact LU factorisation of A is not sparse, so M is constructed by a so-called incomplete factorisation, where the update

$$a_{ij} \leftarrow a_{ij} - a_{ik}{a_{kk}}^{-1}a_{kj} \tag{1}$$

is executed subject to some decision process.

Ideally, the only question concerning incomplete factorisations would be their accuracy. For instance, typically the condition number $\kappa(A) \sim h^{-2}$ where h is the mesh width, and one would hope that $\kappa(M^{-1}A)$ is smaller, preferably of a lower order than h^{-2}. In practice, however, it is already hard to guarantee the existence of the factorisation. To begin with, if $i = j$ in equation (1), the

*This work was supported in part or in whole by Center for Parallel Computation, subcontract #292-3-54397, and Lawrence Livermore National Laboratory, subcontract #B503913, and Los Alamos National Laboratory, subcontract #D0252-0019-2G, and Los Alamos National Laboratory Copmputer Science Institute (LASCI) through LANL contract number 03891-99-23, as part of the prime contract (W-7405-ENG-36) between the Department of Energy and the Regents of the University of California.

updated value of a_{ii} can be zero leading to breakdown in the i-th elimination step. A negative value of a_{ii} is a problem too, since, if A is positive definite, we want M to be so in order to guarantee the applicability of iterative methods such as Conjugate Gradients, and consequently we need all pivots of the incomplete factorisation to be positive.

Incomplete factorisation methods are well-defined for M-matrices, but for any other type of matrix, even symmetric positive definite ones, they can suffer from breakdown of some form or other. A number of remedies have been proposed, but all suffer from certain disadvantages. In a companion paper [5] we have presented the an overview of existing methods, with the emphasis on how they tackle the existence problem.

In this paper we propose a new method: the 'weighted modification' algorithm. The name derives from the fact that a fill element in position (i, j) is multiplied by a factor based on weighing a_{ii} and a_{jj} before it is moved to the diagonal. In another departure from common algorithms, the weighted fill is added to both a_{ii} and a_{jj}. This new algorithm is first of all guaranteed not to break down for matrices with positive diagonal elements, a class that contains the positive definite matrices. Secondly, it satisfies several MILU-like conservation properties in the case of symmetric matrices, definite matrices, or M-matrices.

We conclude by reporing comparative tests on M-matrices, symmetric positive definite non-M-matrices, and nonsymmetric positive definite matrices, showing the relative efficiency of the various incomplete factorisation methods and the severity of their breakdown problems.

2 The 'weighted modification' factorisation algorithm

There are several matrix properties we want to conserve in devising a, positional or numerical, dropping strategy.

1. Any symmetry of the original matrix should be preserved.
2. The factorisation should be well-defined in the sense that, given a positive definite matrix, all pivots should be positive. In fact, the method we are proposing satisfies a stronger condition, namely that it will yield positive pivots if the original matrix has a positive diagonal.
3. The spectrum is not to be disturbed too much; in particular, we want the dropping strategy to reduce to MILU for M-matrices.
4. Applying the factorisation to an M-matrix should yield an M-matrix.

In this report we will give a novel incomplete factorisation algorithm that satisfies these criteria. We do this in two steps: first we describe a careful method for eliminating fill elements, then we incorporate that in a full-fledged factorisation algorithm.

2.1 Elimination of fill-in

Let the framework for a factorisation algorithm be given as in figure 1. Note

for $k = 1, \ldots$
    for $j > k$
        for $k < i < j$
            process fill elements (i,j) and (j,i)
        process diagonal fill element (j,j)

Figure 1: The main ijk loop of an incomplete factorisation algorithm.

that the inner loops describe only half of the index space $\{i,j > k\}$: the innermost loop body tackles simultaneously the fill in (i,j) and (j,i) positions.

In this section we describe an algorithm for eliminating the off-diagonal fill elements in (i,j) and (j,i) elements by added weighted combinations of them to the (i,i) and (j,j) diagonal elements.

Consider then the 2×2 block

$$A \equiv A_{[i,j]} = \begin{pmatrix} a_{ii} & a_{ij} \\ a_{ji} & a_{jj} \end{pmatrix}.$$

1. Let
$$\rho_x = |a_{ij}|/\sqrt{a_{ii}a_{jj}}, \quad \rho_y = |a_{ji}|/\sqrt{a_{ii}a_{jj}}.$$
(This comparison against a geometric mean of diagonal elements was also used in [1].)

2. If $\rho_x \geq 1$, choose $0 < \sigma_x < 1$ and define $x = \mathrm{sign}(a_{ij})\sigma_x\sqrt{a_{ii}a_{jj}}$; otherwise, $x = a_{ij}$. Likewise, if $\rho_y \geq 1$, choose $0 < \sigma_y < 1$ and define $y = \mathrm{sign}(a_{ji})\sigma_y\sqrt{a_{ii}a_{jj}}$; otherwise, $y = a_{ji}$.

   The quantities x and y will be the fraction of the fill that is moved to the diagonal. We call the cases $\rho_x \geq 1$, $\rho_y \geq 1$ 'degenerate', since they correspond to the case where the fill elements are too large to be moved without making the modified factorisation indefinite. In practical applications we expect $\rho_x, \rho_y < 1$ (and as shown below, this is guaranteed to be the case for spd matrices); the introduction of x, y serves to keep the algorithm well-defined when we positionally drop large off-diagonal elements.

3. For positional dropping, decide to drop a_{ij} and a_{ji} based on some sparsity set; for numerical dropping,

   - drop a_{ij} if $|a_{ij}| \leq \gamma\sqrt{a_{ii}a_{jj}}$,
   - drop a_{ji} if $|a_{ji}| \leq \gamma\sqrt{a_{ii}a_{jj}}$.

   where $\gamma < 1$ is some drop tolerance fraction.

4. Let

$$F_x = \begin{pmatrix} -\frac{x}{2}\sqrt{\frac{a_{ii}}{a_{jj}}} & a_{ij} \\ 0 & -\frac{x}{2}\sqrt{\frac{a_{jj}}{a_{ii}}} \end{pmatrix}, \quad F_y = \begin{pmatrix} -\frac{y}{2}\sqrt{\frac{a_{ii}}{a_{jj}}} & 0 \\ a_{ji} & -\frac{y}{2}\sqrt{\frac{a_{jj}}{a_{ii}}} \end{pmatrix}, \qquad (2)$$

and $F = F_x + F_y$; we modify $A \leftarrow A - F_x$ if a_{ij} is to be dropped, $A \leftarrow A - F_y$ if a_{ji} is to be dropped, and $A \leftarrow A - F$ if both are to be dropped.

We give the algorithm in figure 2.

---

let $\sigma < 1$
let $m_{ij} = a_{ik}{a_{kk}}^{-1}a_{kj}$ and $m_{ji} = a_{jk}{a_{kk}}^{-1}a_{ki}$
let $\rho_x = |m_{ij}|/\sqrt{a_{ii}a_{jj}}$ and $\rho_y = |m_{ji}|/\sqrt{a_{ii}a_{jj}}$
if $\rho_x < 1$ then $x = m_{ij}$
  else $x = sign(m_{ij})\sigma\sqrt{a_{ii}a_{jj}}$
if $\rho_y < 1$ then $y = m_{ji}$
  else $y = sign(m_{ji})\sigma\sqrt{a_{ii}a_{jj}}$
let $f_i^x = -x\sqrt{a_{ii}/a_{jj}}/2$, $f_j^x = -x\sqrt{a_{jj}/a_{ii}}/2$, and $f_i^y = -y\sqrt{a_{ii}/a_{jj}}/2$, $f_j^y = -y\sqrt{a_{jj}/a_{ii}}/2$
$a_{ii} \leftarrow a_{ii} - f_i^x - f_i^y$
$a_{jj} \leftarrow a_{jj} - f_j^x - f_j^y$

Figure 2: The algorithm for weighted motion of fill elements to the diagonal.

Theorem 1 *The above algorithm satisfies the four criteria given at the start of section 2:*

1. *Symmetry is preserved in the sense that, with numerical dropping, $a_{ij} = a_{ji}$ implies that both elements are dropped or accepted together. With positional dropping, symmetry depends on that of the sparsity set.*
2. *If the original matrix has positive diagonal elements, so have $A - F_x$, $A - F_y$, and $A - F$.*
3. *If $\rho_x, \rho_y < 1$, the modification $F = F_x + F_y$ is semi-definite, with a zero eigenvalue in the case of symmetry. The modification is negative semi-definite if the off-diagonal elements of A are non-positive, positive semi-definite if they are non-negative.*
4. *In the case of symmetry, the modification preserves the product with the vector $v = (\sqrt{a_{jj}}, \sqrt{a_{ii}})^t$. This vector is pointwise positive, and, in the case of non-degenerate modification, Av is pointwise positive too.*

Proof. The dropping criterium obviously preserves symmetry, since a_{ij} and a_{ji} are compared to the same value $\gamma\sqrt{a_{ii}a_{jj}}$, so their being equal implies that we eliminate both a_{ij} and a_{ji}.

The next problem is preserving positive diagonal entries. The diagonal elements of $A - F$ are positive, since, for instance,

$$a_{ii} + \frac{x+y}{2}\sqrt{\frac{a_{ii}}{a_{jj}}} \geq a_{ii} - \frac{|x|+|y|}{2}\sqrt{\frac{a_{ii}}{a_{jj}}} > a_{ii} - \tau\sqrt{a_{ii}a_{jj}}\sqrt{\frac{a_{ii}}{a_{jj}}} = (1-\tau)a_{ii} > 0.$$

where $\tau = \max\{\sigma_x, \sigma_y, \gamma\} < 1$.

Then, the determinant of $F = F_x + F_y$ is

$$\begin{aligned} |F| &= \frac{1}{2}\sqrt{\frac{a_{ii}}{a_{jj}}}(x+y)\frac{1}{2}\sqrt{\frac{a_{jj}}{a_{ii}}}(x+y) - a_{ij}a_{ji} = \frac{1}{4}(x+y)^2 - a_{ij}a_{ji} \\ &= \frac{1}{4}(x-y)^2 - (a_{ij}a_{ji} - xy). \end{aligned}$$

Thus, if $\rho_x, \rho_y < 1$, we have $a_{ij}a_{ji} = xy$ and consequently F is semi-definite, with both eigenvalues non-negative if the off-diagonal elements are non-positive, and vice versa. In the case of symmetry, and with $\sigma_x = \sigma_y$ for degenerate modification, F has a zero eigenvalue.

For the positive vector $v = (\sqrt{a_{jj}}, \sqrt{a_{ii}})^t$ we have

$$Av = \begin{pmatrix} a_{ii}\sqrt{a_{jj}} + a_{ij}\sqrt{a_{ii}} \\ a_{jj}\sqrt{a_{ii}} + a_{ji}\sqrt{a_{jj}} \end{pmatrix}$$

so $Av > 0$ if

$$-a_{ij} < \sqrt{a_{ii}a_{jj}}, \qquad -a_{ji} < \sqrt{a_{ii}a_{jj}},$$

which is true in the non-degenerate case or for positive off-diagonal elements. Now,

$$Fv = \begin{pmatrix} \frac{x-y}{2}\sqrt{a_{ii}} \\ \frac{y-x}{2}\sqrt{a_{jj}} \end{pmatrix}$$

so, in the case of symmetry, $(A - F)v = Av$.
Qed.

Lemma 2 *If the matrix is symmetric positive definite, there will be no degenerate modification.*

Proof. For a positive definite matrix A, the determinant $a_{ii}a_{jj} - a_{ij}a_{ji}$ of A is positive. For a symmetric matrix this implies

$$a_{ij} = a_{ji} < \sqrt{a_{ii}a_{jj}},$$

so that the conditions for degeneracy are not met. Qed.

2.2 Preserving the M-matrix property

The problem in using the weighted fill-elimination strategy given above lies in preserving the M-matrix property during the fatorisation. In traditional modified methods the positive vector v to be used in the modification (usually taken as $v = (1, 1, \ldots)^t$) is determined a priori, whereas the weighted modification strategy constructs it a posteriori: eliminating fill elements in positions (i, j) and (j, i) fixes the ratio of v_i/v_j.

It is then possible to find contradictory demands on the elements of v, e.g., when eliminating fill elements (i_1, j_1), $(i_2 = j_1, j_2)$, $(i_3 = j_2, j_3 = i_1)$. The first two steps determine the ratios v_{i_1}/v_{j_1} and v_{i_2}/v_{j_2}, leaving no degree of freedom for v_{i_3}/v_{j_3}. We repair this by keeping track of the components of v that have been 'fixed into place' in this manner, and we skip certain elimination steps if necessary. This conditional execution of the fill statements, in effect, makes the algorithm a cross between modified LU and SSOR.

Additionally, we must make the diagonal fill-in step contingent upon that row having had its off-diagonal fill moved. Performing the diagonal fill step regardless would make the method subject to the breakdown seen in the unmodified ILU methods. Figure 3 gives an algorithm; it is theoretically fully

```
fix ← 0, dia ← 1
for k
    for j > k
        for k < i < j
            if fix(i) = 0 or fix(j) = 0
              eliminate fill in (i, j) and (j, i) locations as described in figure 2,
              set fix(i) ← 1, fix(j) ← 1.
            else set dia(i) = dia(j) = 0
        if dia(j) = 1
            a_jj ← a_jj − a_jk a_kk^{-1} a_kj.
```

Figure 3: Altered inner factorisation loop to preserve M-matrix property.

guaranteed to preserve the M-matrix property.

2.3 Fine tuning

The main problem with incomplete factorisations, giving rise to non-positive pivots, is that fill subtracted from the diagonal usually is positive, thereby decreasing the diagonal element. While in exact factorisation there are theoretical guarantees that the diagonal will never be decreased below zero, in incomplete methods it raises the possibility of nonpositive pivots.

In some cases such as convection-dominated problems, however, fill elements are often negative, and they increase the size of the diagonal element. (This observation can be taken to be basis of the methods of Jennings and Malik [9], and of Robert [13].) Thus, for positive fill elements there is no need for the elaborate weighing strategy, and we do not have to make the diagonal fill conditional as described in the previous subsection.

The over-specification argument above implies that we have to ignore certain fill elements. Intuitively, it is more important to handle large elements correctly than small elements. Therefore, we sort the elements in a row and column currently being eliminated by magnitude. In practical tests we have seen this to make a considerable difference in the number of iterations, up to almost a factor of two on the Harwell-Boeing matrix `bcsstk03`.

While the above theory is based on an exact move of the – weighted – fill to the diagonal, in a practical application we multiply the fill by a factor slightly less than one. Such a relaxation has been advocated for various reasons in [2, 3, 4, 6, 14].

We arrive at the inner loop described in figure 4;

$fix \leftarrow 0$, $dia \leftarrow 1$
for $j > k$
  for $k < i < j$
    let $m_{ij} = a_{ik}{a_{kk}}^{-1}a_{kj}$ and $m_{ji} = a_{jk}{a_{kk}}^{-1}a_{ki}$
    if $m_{ij} < 0$
      let $f_i^x = m_{ij}$ and $f_j^x = 0$
    otherwise compute f_i^x, f_j^x as in figure 2 updating fix and dia.
    if $m_{ji} < 0$
      let $f_i^y = 0$ and $f_j^y = m_{ji}$
    otherwise compute f_i^y, f_j^y as in figure 2, updating fix and dia.
    $a_{ii} \leftarrow a_{ii} - f_i^x - f_j^x$
    $a_{jj} \leftarrow a_{jj} - f_j^x - f_j^y$
  if $dia(j) = 1$ or $a_{jk}{a_{kk}}^{-1}a_{kj} < 0$
    $a_{jj} \leftarrow a_{jj} - a_{jk}{a_{kk}}^{-1}a_{kj} < 0$

Figure 4: The weighted inner loop, with conditionals disabling distribution in 'unsafe' cases.

---

3 Tests

We ran a number of tests to judge the performance of the various factorisation methods. We tested both simple and more complicated problems, so that the efficacy of the pivot repair stategies could be judged, as well as their influence in cases where they would not be needed.

In all of the next tables we gave iteration numbers; 'inf' denotes that the method showed no perceptible convergence at a certain cutoff point, typically 1000 iterations; '-1 denotes that the iterative method broke down, typically after only a few iterations.

n	SSOR	ILU	jILU	wILU	μILU
20	19	17	20	18	16
40	35	30	36	31	23
80	34	50	68	51	34

Table 1: Number of iterations of CG with a 'D'-variant incomplete factorisation on the 5-point Laplacian stencil on an $n \times n$ grid.

3.1 Numerical results

We use a number of representative factorization methods to solve systems with various symmetric and nonsymmetric coefficient matrices. This methods are described in detail in [5]. In the tables, the following names are used:

gILU A modified method preceded by Gustafsson's modification; see [8]. This method eliminates all positive off-diagonal elements prior to the (modified) incomplete factorisation.

jILU The Jennings and Malik method; see [9]. This method adds the absolute value of fill to the diagonal.

kILU The Kershaw method; see [10]. This method sets any arising negative pivots to an appropriate positive value. We omitted updating the Schur complement whenever pivot repair was needed.

mILU Manteuffel's method; see [11]. This method adds a sufficient amount to the diagonal to prevent negative pivots; it was implemented as follows. Let v be a vector such that $v_i = \max\{0, -A_{ii} + \sum_{j\neq i} |A_{ij}|\}$. Since $A + diag(v)$ is (non-strictly) diagonally dominant, there is an $0 \leq \alpha \leq 1$ such that $A + diag(v)$ has a well-defined factorisation. The algorithm then successively tries $\alpha = 0, .1, .2., 3, \ldots$. While this is not optimal, at least it illustrates the principle.

μILU Modified ILU; see [7].

wILU The weighted modification ILU introduced above.

3.2 Symmetric positive definite M-matrices

On SPD M-matrices, ILU and μILU are well-defined. Thus there is no need for repair strategies. Some repair strategies, such as in the Kershaw and Manteuffel methods, are indeed not invoked. Others, such as in the Jennings and Weighted method, are always invoked; with these problems we thus only test any possible performance degradation due to these strategies. Furthermore, the Gustafsson method reduced to modified incomplete factorisation. We tested two model problems: the 5-point Laplacian, reported in table 1, and the 9-point fourth order Laplacian, reported in table 2. We used a standard Conjugate Gradient

n	SSOR	ILU	jILU	wILU	μILU
20	19	18	21	19	17
40	31	30	38	34	24
80	58	53	71	58	35

Table 2: Number of iterations of CG with a 'D'-variant incomplete factorisation on the fourth-order 9-point finite difference Laplacian stencil on an $n \times n$ grid.

n	SSOR	ILU	gILU	jILU	kILU	mILU	wILU	μILU
20	70	-1	39	113	inf	79	98	-1
40	199	-1	75	372	inf	252	322	-1
80	683	-1	147	1063	inf	789	905	-1

Table 3: Number of iterations of CG with a 'D'-variant incomplete factorisation on the biharmonic flake problem on an $n \times n$ grid.

method with a right hand side of all ones; the stopping test was on a relative reduction of 10^{-6}, and the cutoff point was at 1000 iterations[1].

We see that the Jennings and Malik method increases the number of iterations with respect to simple ILU; Eijkhout's weighted method gives a slight increase in that sense, and it does not grow like modified ILU, in spite of its theoretical resemblances.

3.3 Symmetric positive definite non-M-matrices

For matrices that are SPD but are not M-matrices, a full factorisation is defined, but an incomplete factorisation need not be. However, ILU and μILU are not immediately guaranteed to breakdown. In order to get an indication of the likelihood of breakdown, and the efficacy of methods where existence is guaranteed, we tested two stencils for the biharmonic equation. The iterative method was set up as above.

In tables 3, 4, 5, and reftab:star1, we give the results for the following two stencils:

- the 'biharmonic flake' stencil found by multiplying the Laplace stencil by itself, and

[1] In table 3 it was clear that one method had almost converged so we reran the test with a slightly higher maximum number of iterations.

n	SSOR	ILU	gILU	jILU	kILU	mILU	wILU	μILU
20	70	-1	45	84	inf	48	70	15
40	199	-1	85	282	inf	156	217	25
80	683	-1	171	792	inf	491	626	48

Table 4: Number of iterations of CG with a level-1 incomplete factorisation on the biharmonic flake problem on an $n \times n$ grid.

n	SSOR	ILU	gILU	jILU	kILU	mILU	wILU	μILU
20	52	40	44	79	40	40	88	-1
40	127	113	88	257	113	113	264	-1
80	441	375	177	703	375	375	870	-1

Table 5: Number of iterations of CG with a 'D'-variant incomplete factorisation on the biharmonic star problem on an $n \times n$ grid.

n	SSOR	ILU	gILU	jILU	kILU	mILU	wILU	μILU
20	52	-1	40	74	inf	43	69	18
40	127	-1	74	234	inf	119	205	34
80	441	-1	148	676	inf	416	699	65

Table 6: Number of iterations of CG with a level-1 incomplete factorisation on the biharmonic star problem on an $n \times n$ grid.

- the 'biharmonic – 9-point – star' stencil that uses only connections along the coordinate axes.

The results allow us to draw the following conclusions:

- ILU can break down, but need not, as in the case of the star stencil. In the case it does not, the Kershaw and Manteuffel methods coincide with ILU.
- Modified ILU is just as risky as ILU on such non-M-matrices.
- If ILU breaks down, the Kershaw method offers no solace; the Jennings and Malik, Manteuffel and Eijkhout methods do converge, though not necessarily faster than SSOR.
- Gustafsson's method is superior on these problem, due to the fact that after preprocessing the reduced stencil is an M-matrix, for which the subsequent modified ILU factorisation performs very well. We can not expect this behaviour to persist beyond this special case.

3.4 Nonsymmetric positive definite matrices

We generate a model convection-diffusion problem by the five-point central difference discretisation of

$$-\Delta u + s\bar{v} \cdot u = f \qquad (3)$$

where $\bar{v} = (\sin\alpha, \cos\alpha)^t$ and $s > 0$. The matrix from this stencil may have positive off-diagonal coefficients, and may not be diagonally dominant.

In a practical situation the convection part is smaller by a factor of h, so these adverse properties only hold for matrices up to a certain size. In our tests

n	SSOR	ILU	gILU	jILU	kILU	mILU	wILU	μILU
20	18	16	12	20	16	16	17	12
40	24	23	14	31	23	23	24	14

Table 7: Convergence results on a 5-point convection diffusion stencil with factor 3 upwind, preconditioned with a D-variant method on an $n \times n$ grid.

n	SSOR	ILU	gILU	jILU	kILU	mILU	wILU	μILU
20	26	15	inf	inf	182	inf	25	12
40	56	22	inf	inf	910	inf	59	13

Table 8: Convergence results on a 5-point convection diffusion stencil with factor 20 upwind, preconditioned with a D-variant method on an $n \times n$ grid.

we have explicitly let the size of the convection part be fixed with respect to the diffusion part. Specifically, we used the stencil

$$\begin{array}{ccc} -1 & -4 & -1 \\ -4 & 20 & -4 \\ -1 & -4 & -1 \end{array} \quad + \quad c \cdot \begin{array}{ccc} & 1 & \\ & -2 & 1 \\ & & \end{array}$$

where $c = 3$ in table 7, and $c = 20$ in table 8. In the latter case, the initial matrix will already have positive off-diagonal elements.

We used a BiConjugate Gradient method with a right hand side of all ones; the stopping test was on a relative reduction of 10^{-6}, and the cutoff point was at 1000 iterations.

We draw the following conclusions.

- On the problem with weak convection, all methods converge, and in a very similar number of iterations.
- Surprisingly, ILU and μILU converge on the problem with strong convection.
- Contrary to the case in table 3, in the strong-convection problem the Kershaw method converged whereas the Jennings and Malik method did not. However, the convergence of the Kershaw method was much slower than of the other converging methods.
- We note that among the methods that are guaranteed not to break down, only the 'weighted modification' method actually converged.

3.5 Non-model matrices

We tested a number of matrices from the Harwell-Boeing collection, both symmetric and nonsymmetric. From the results in table 9 we draw the following conclusions:

matrix	ssor	ILU	gILU	jILU	kILU	mILU	wILU	μILU
bcsstk14	188	33	450	270	33	33	184	-1
bcsstk26	885	103	inf	1196	103	103	620	-1
gre115	inf	inf	56	107	inf	28	105	104
gre185	inf	150	199	150	371	172	221	206
gre512	157	156	295	129	156	156	127	129
orsirr1	184	19	-1	14	inf	inf	14	14

Table 9: Convergence results on various test problems, using a level-1 factorisation.

- In these test problems, we see instances of matrices for which SSOR does not converge, or converges slowly. In previous tests, SSOR looked like a fairly attractive method.
- Surprisingly, ILU(1) will converge in some cases where there is no theoretical guarantee. In `gre115` we have an instance of ILU(1) not breaking down, but also not converging. In `orsirr1` we have a matrix where ILU(1) breaks down, but the BiConjugate Gradient method converges nevertheless. Repairing the breakdown with the Kershaw trick leads to non-convergence.
- The gILU method can not normally break down; however, on the `orsirr1` matrix division by zero occurs because of zero pivots. This phenomenon was explained in [6, 12].

4 Conclusion

The existence problem of incomplete factorisations, that is, the matter of guaranteeing positive pivots in an incomplete factorisation where a full factorisation carries such a guarantee, is a hard one. Several methods exist that will give positive pivots, but several of them can be characterised as little more than stop-gap measures. The tests in this paper illustrate that such methods can have severe convergence problems.

We have introduced a new method, the ‘weighted modification’ factorisation, which guarantees positive pivots for any matrix with positive diagonal elements, a strict superset of the positive definite matrices. This method is not uniformly faster to converge when other methods converge, but it is more robust, converging on every problem where any other method converges. Note that we are not saying that it will give a converging iterative method on every matrix. Still, we hope to have added one more trick to the literature of incomplete factorisation methods.

References

[1] O. Axelsson and N. Munksgaard. Analysis of incomplete factorizations with fixed storage allocation. In D. Evans, editor, *Preconditioning Methods – Theory and Applications*, pages 265–293. Gordon and Breach, New York, 1983.

[2] Owe Axelsson and Gunhild Lindskog. On the rate of convergence of the preconditioned conjugate gradient method. *Numer. Math.*, 48:499–523, 1986.

[3] R. Beauwens. On Axelsson's perturbations. *Lin. Alg. Appl.*, 68:221–242, 1985.

[4] T. Dupont, R. Kendall, and H. Rachford. An approximate factorization procedure for solving self-adjoint elliptic difference equations. *SIAM J. Numer. Anal.*, 5:559–573, 1968.

[5] Victor Eijkhout. On the existence problem of incomplete factorisation methods. Technical Report Lapack working note 144, UT-CS-99-435, Computer Science Department, University of Tennessee.

[6] Victor Eijkhout. Beware of unperturbed modified incomplete point factorizations. In Robert Beauwens and Pieter de Groen, editors, *Proceedings of the IMACS International Symposium on Iterative Methods in Linear Algebra, Brussels Belgium*, pages 583–591, 1992.

[7] I. Gustafsson. A class of first-order factorization methods. *BIT*, 18:142–156, 1978.

[8] Ivar Gustafsson. An incomplete factorization preconditioning method based on modification of element matrices. *BIT*, 36:86–100, 1996.

[9] A. Jennings and G.M. Malik. Partial elimination. *J. Inst. Maths Applics*, 20:307–316, 1977.

[10] D.S. Kershaw. The incomplete cholesky-conjugate gradient method for the iterative solution of systems of linear equations. *J. Comp. Phys.*, 26:43–65, 1978.

[11] T.A. Manteuffel. An incomplete factorization technique for positive definite linear systems. *Math. Comp.*, 34:473–497, 1980.

[12] Y. Notay. Solving positive (semi)definite linear systems by preconditioned iterative methods. In O. Axelsson and L.Yu. Kolotilina, editors, *Preconditioned Conjugate Gradient Methods*, pages 105–125, Nijmegen, 1989. Lecture Notes in Mathematics, vol. 1457.

[13] Yves Robert. Regular incomplete factorizations of real positive definite matrices. *Lin. Alg. Appl.*, 48:105–117, 1982.

[14] Henk van der Vorst. High performance preconditioning. *SIAM J. Sci. Stat. Comput.*, 10:1174–1185, 1989.