Science & Technology Facilities Council

Technical Report

RAL-TR-2017-003

The Hellerman-Rarick algorithm

IS Duff, AM Erisman, JK Reid

April 2017

Enquiries concerning this report should be addressed to:

RAL Library
STFC Rutherford Appleton Laboratory
Harwell Oxford
Didcot
OX11 0QX

Tel: +44(0)1235 445384
Fax: +44(0)1235 446403
email: libraryral@stfc.ac.uk

ISSN 1358-6254

The Hellerman-Rarick algorithm

I. S. Duff[1], A. M. Erisman[2], and J. K. Reid[1]

ABSTRACT

We describe an algorithm for achieving a bordered block triangular form, including the case where the diagonal blocks are themselves in this form. The original algorithm was created by Hellerman and Rarick (1971, 1972) for solving linear programming problems. We include an extension of this algorithm for overcoming the problem of structural singularity in intermediate steps introduced by Erisman, Grimes, Lewis and Poole (1985). Two areas requiring further work include: dealing with safeguards against numerical instability when factorizing the diagonal blocks of the form, and dealing with parallelization, cache management, and memory management for large problems. Only after doing these things would it be appropriate to explore what niche, if any, these algorithms might address when implemented in a modern computer architecture for the solution of very large problems.

Keywords: Bordered block triangular form, structural singularity, sparse matrices, solution of very large problems, modern computer architectures.

AMS(MOS) subject classifications: 65F05 Direct methods for linear systems and matrix inversion, 65F50 Sparse matrices, 90C05 Linear programming.

---

[1] Scientific Computing Department, STFC Rutherford Appleton Laboratory, Harwell Campus, Oxfordshire, OX11 0QX, UK.

[2] 11008 SE 24th Pl., Bellevue, Wa 98004, USA

Correspondence to: Iain.Duff@stfc.ac.uk

April 10, 2017

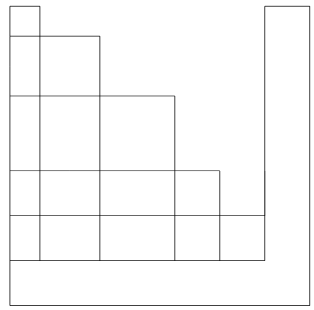

Figure 1: Bordered block triangular form.

1 Introduction

Between 1970 and the early 1980s, a large number of algorithms were developed for the preservation of sparsity in the factorization of large matrices. Most fall into three major categories: local orderings, dissection schemes, and banded (and variable banded) methods. There has been significant work on the algorithms to adapt them to modern computer architectures, enabling the solution of much larger problems. Unlike the early period when a good algorithm focused on reducing multiplication counts, modern architectures require careful consideration of caching, parallelization, data locations, and movement of data in general. This updated picture is captured in our book Direct Methods for Sparse Matrices (Duff, Erisman and Reid 2017).

However, there are some interesting algorithms from the early period that have fallen out of favour under the old rules. One is the Hellerman-Rarick algorithm for permuting to the bordered block triangular form (Figure 1). We included a careful write-up of it in the first edition of our book (Duff, Erisman and Reid 1986) because we could not find a clear presentation elsewhere. For many reasons, including competitiveness with local orderings, little attention to this algorithm has been paid since that early period, so we removed our write-up of it from the second edition. This report reproduces this write-up because on modern computer architectures, simply reducing multiplication counts is not a good enough measure of the success of a method. Careful memory and cache management and algorithmic parallelization are huge factors in the success of an algorithm in this environment. We do not know whether this algorithm could be competitive in this environment for some class of problems, but we invite this exploration.

To lay the groundwork for such studies, we describe the method here. Its intriguing structure, not neatly fitting in the three categories identified, suggests this might be a worthwhile pursuit.

2 The Boeing version of the Hellerman-Rarick algorithm

Hellerman and Rarick (1971) introduced an algorithm which they called the preassigned pivot procedure (P^3) and later (Hellerman and Rarick 1972) suggested an initial step of permuting to block lower triangular form to be followed by the application of the earlier (P^3) algorithm to each diagonal block. This later algorithm is called the partitioned preassigned pivot procedure (P^4).

A two-stage algorithm is normally used for permuting to block triangular form:

1. permute entries onto the diagonal, and

2. use symmetric permutations to find the block form itself.

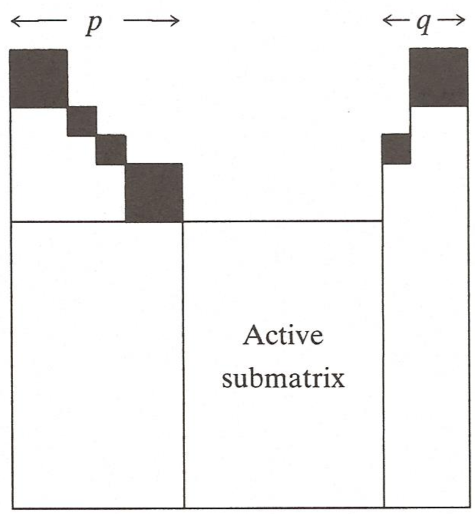

Figure 2: The permuted matrix at a typical intermediate stage of the P^5 algorithm.

Algorithms based on the work of Kuhn (1955), which use a depth-first search, are efficient for the first stage and are discussed in Section 6.4 of Duff et al. (2017). The algorithm of Tarjan (1972) is efficient for the second stage and is discussed in Section 6.5.3 of Duff et al. (2017). We believe that this two-stage process is the most efficient algorithm available for practical problems. We therefore describe only the treatment of each block on the diagonal of the block triangular form. Each such block is irreducible (cannot be permuted to block triangular form), so it suffices to limit our description to the case where the original matrix is irreducible.

In this section, we consider a variant of the P^3 algorithm due to Erisman et al. (1985) because it is a simplification and is therefore easier to describe. They call it P^5 (precautionary partitioned preassigned pivot procedure). It produces a bordered block triangular form (Figure 1). All the blocks on the diagonal of the block triangular submatrix are dense and the algorithm tries to make the border thin. The original P^3 algorithm also produces a bordered block triangular form, but tries harder to make the border thin and the diagonal blocks may themselves be bordered block triangular matrices. We describe this in the next section.

We assume that the sparse matrix that we are seeking to process is square of order n. At a typical intermediate stage the permuted matrix has the form illustrated in Figure 2. The leading $p \times p$ submatrix is block lower triangular with dense diagonal blocks. The diagonal entries are **assigned pivots**. The q columns in the border are called **spike columns** by Hellerman and Rarick for reasons that will be apparent in the next section. Each has a leading dense block in rows corresponding to a block of the block triangular submatrix. Each of these spike columns extends at least as far up the matrix as its predecessors. The submatrix of rows 1 to p and columns $p+1$ to $n-q$ is zero. We call the submatrix of rows $p+1$ to n and columns $p+1$ to $n-q$ the **active submatrix** since it is within this submatrix that further permutations take place. We commence with $p = q = 0$ and the whole matrix active and end with $p+q = n$ and no columns left in the active submatrix.

In the first major step of the algorithm, m columns are chosen, where m is the minimum number of entries in a row. Some or all of them make up the leading block column and the rest

	1	2	3	4	5	6
1	×	×	×		×	
2	×			×		×
3	×	×	×	×		
4	×			×		×
5		×	×	×	×	×
6	×	×	×		×	

Figure 3: A 6×6 matrix pattern.

	6	4	2	5	3	1
4	×	×				×
2	×	×				×
3		×	×		×	×
1			×	×	×	×
5	×	×	×	×	×	
6			×	×	×	×

Figure 4: The matrix of Figure 3 after the first major stage.

make up the end of the border. A column with most entries in rows with count m (we defer the resolution of ties for the present) is chosen first. After removing this column from the active submatrix, we will have the greatest possible number of rows with the minimum row count of $m-1$. Next a column with most entries in rows with count $m-1$ is chosen. The process continues similarly until m columns have been chosen. If the last column chosen has s singletons (rows with one entry), the rows containing these singletons are permuted to the front and assigned as pivotal rows. The last s columns selected ($s < m$, since the matrix is irreducible) are permuted to the front and assigned as pivotal columns. The remaining $m - s$ columns are permuted to the back and become the end of the border.

In the simple example shown in Figure 3, the minimum row count m is 3 and column 1, being a column with most entries in rows having 3 entries, is chosen first. We revise the row counts to exclude this column and find that column 4 has most entries (2) in rows with the new minimum row count of 2, so this is chosen next. Now column 6 has singletons in rows 2 and 4. Therefore we permute rows 2 and 4 and columns 6 and 4 forward to the pivotal block, and permute column 1 to the back to become a border spike. The resulting matrix is shown in Figure 4. The active submatrix is now rows 3 to 6 and columns 3 to 5 of the permuted matrix.

After the first major step, the matrix will always have the form illustrated in Figure 5. There is a leading block of order s, there are $m - s$ spikes at the end of the matrix and the first s rows are otherwise zero. Since every row in the original matrix had at least m entries, the $s \times s$ pivotal block and the first s rows of the border are dense.

The algorithm now treats the active submatrix in exactly the same way, continuing until it produces the form illustrated in Figure 2. The active submatrix is rectangular, so it is possible that all the columns selected are assigned as pivotal. The simplest case is when $m = 1$, in which case a row singleton is moved to the leading position and assigned immediately as a pivot.

In describing the algorithm, we purposely omitted to say which column is chosen if several

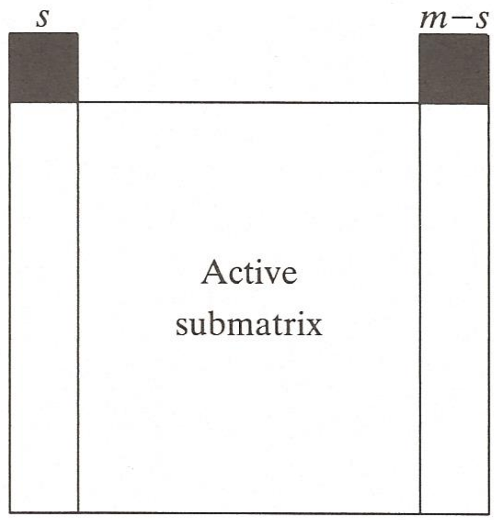

Figure 5: The general form after selection of the first set of spikes.

have the maximum number of entries in rows with minimum row count. In this case, Hellerman and Rarick aim to reduce the number of spikes in later stages by choosing the candidate that has greatest column count unless they have a single entry in a row of minimum count. In the latter case, account is first taken of the number of entries in rows of second least row count, then a remaining column with greatest count is taken.

This completes our description of the algorithm. For ease of reference, we summarize it in pseudo-algol in Figure 6. It is convenient to describe each column as being permuted to the end of the active submatrix as it is chosen. If it has s singletons then it and the appropriate number of its successors are moved forward into the pivotal block.

Erisman et al. (1985) experimented with the use of the implicit factorization

$$\begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \\ \mathbf{A}_{21}\mathbf{A}_{11}^{-1} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ & \mathbf{A}_{22} \end{pmatrix}, \tag{1}$$

where $\mathbf{A}_{11}$ is the block lower triangular submatrix of the P^5 ordering and the rest is the border, thereby confining the fill-in to the block $\mathbf{A}_{22}$. Their test comparisons with the algorithm of Markowitz showed the two to be broadly comparable, with neither consistently better than the other. Unfortunately they did not have any proposals for safeguards against numerical instability, so the algorithm cannot be regarded as a serious challenger to that of Markowitz. Because their goal was structural stabilization rather than numerical stabilization, they did not introduce strategies for dealing with potentially unstable pivots. Before going further this would need to be done. Duff et al. (2017) outline some strategies for dealing with small numerical pivots in a sparse numerical factorization.

3 The Hellerman-Rarick ordering

The aim of Hellerman and Rarick was not so much to produce a bordered block triangular matrix as to produce a spiked matrix of the form illustrated in Figure 7. By a **spike**, we mean the part

Count the number of entries in the rows and columns;
Initialize the active submatrix to be the whole $n \times n$ matrix;
$j := 1$; **while** $j \leq n$ **do**
begin

> find the minimum row count m;
> **for** $m1 := m$ **step** -1 **until** 1 **do**
> **begin**
>
>> find the set T of columns with maximum number s of entries in rows with count of $m1$;
>> **if** there is more than one column in T and $s = 1$ **then** find the second least row count m' of rows with entries in columns of T and reduce T to those columns with maximum number of entries in rows with count m';
>> choose a column $\widehat{j}$ of T with greatest column count;
>> exchange column $\widehat{j}$ with the last column of the active submatrix;
>> remove column $\widehat{j}$ from the active submatrix and revise the row counts to correspond;
>
> **end**;
> $j := j + m$;
> assign pivots

end;

procedure assign pivots
begin

> permute the rows so that the s rows that have just had count 1 are the leading rows of the active submatrix;
> **for** $i := 1$ **step** 1 **until** $\min(s, m)$ **do** move the first column in the border ahead of the active submatrix; $i = \min(s, m)$;
> remove the leading i rows from the active submatrix;

end;

Figure 6: A summary of the P^5 version of the Hellerman-Rarick algorithm in pseudo-algol.

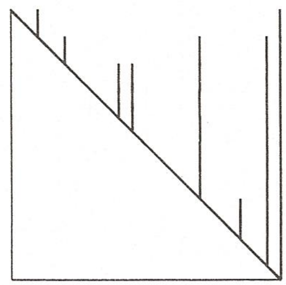

Figure 7: A spiked matrix.

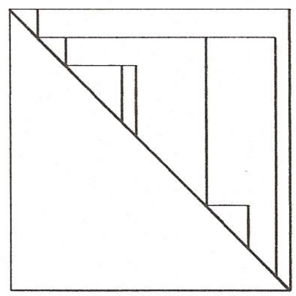

Figure 8: The matrix of Figure 7, regarded as a nested bordered block triangular form.

of a spike column that lies on and above the diagonal. Any pair of spikes have the property that the set of rows for the first is either contained in or disjoint from the set of rows for the second. This property corresponds to being able to regard each spike as the border of a block lower triangular form in a properly nested set, as illustrated in Figure 8.

If Gaussian elimination without interchanges is applied to such a sparse matrix, fill-ins can take place only within spike columns. Indeed the fill-in may be confined to the spikes themselves if the matrix is reduced to a diagonal matrix by a sequence of elementary row operations that create zeros successively in the lower triangular part of row k then in the upper triangular part of column k, for $k = 2, 3, \ldots, n$. Hellerman and Rarick aim to minimize both the number of spikes and the extent to which they project above the diagonal.

The total number of spikes produced by Hellerman and Rarick is the same as for the P^5 variant described in the last section, but they try harder to move spikes forward. If an active submatrix has minimum row count m and the last of the m columns chosen has more than m singletons, then some of the previously assigned spikes may be moved forward to become pivotal,

$$
\begin{array}{c|cc|cc}
\times & & & \times & \times \\
\hline
\times & \times & & \times & \times \\
\times & \times & & \times & \times \\
\times & \times & \times & & \times \\
\times & \times & \times & \times & \times
\end{array}
\qquad
\begin{array}{ccccc}
\times & & & \times & \times \\
\times & \times & & \times & \times \\
\times & \times & \times & & \times \\
\times & \times & & \times & \times \\
\times & \times & \times & \times & \times
\end{array}
\qquad
\begin{array}{ccccc}
\times & & \times & & \times \\
\times & \times & \times & & \times \\
\times & \times & \times & & \times \\
\times & \times & & \times & \times \\
\times & \times & \times & \times & \times
\end{array}
$$

Figure 9: A simple example showing extra spikes being moved forward.

$$
\begin{array}{cccccccc}
\times & & & & \times & & \times & \times \\
 & \times & \times & \times & & & \times & \times \\
 & \times & \times & \times & & & \times & \times \\
 & \times & \times & \times & & & \times & \times \\
 & \times & \times & \times & & & \times & \times \\
\times & \times & \times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times & \times & \times
\end{array}
$$

Figure 10: The 8×8 example of Erisman *et al.*

which shortens them. A trivial illustration is shown in Figure 9. On the left, the active submatrix is in rows 2 to 5 and columns 2 and 3, and column 2 contains two singletons. Erisman et al. (1985) would accept entry (2,2) as a pivot, then find the singleton in column 3 and assign this to give the form shown in the middle of Figure 9. Hellerman and Rarick, on the other hand, after finding 2 singletons in column 2 would bring the spike forward to give the last form in Figure 9.

If extra spikes are always brought forward in this way, we may place a zero in a pivotal position, as is the case for the matrix of Erisman et al. (1985) shown in Figure 10, where column 5 is a spike column that has been moved forward. In this particular case, not only is the fifth pivot zero, but it remains zero during Gaussian elimination. Hellerman and Rarick do not make clear how they treat such a situation. Erisman et al. (1985) make the interpretation that if a column has s singletons, $s-m$ previously assigned spikes are always moved forward, and found that this variant of the algorithm failed quite often. Since the algorithm is widely used in linear programming codes, it seems more likely that the authors intended spikes to be brought forward only if nonzero pivots can be assigned, perhaps with the help of permutations of the singleton rows. The P^5 approach of never assigning previous spikes seems over-cautious. The pseudo-algol procedure assign_pivots in Figure 11 summarizes this part of the algorithm and replaces the procedure of the same name in Figure 6. It uses our interpretation of the algorithm, rather than that of Erisman et al. (1985)

These extra movements can mean that the leading square submatrix is no longer a block lower triangular matrix with dense diagonal blocks. Instead it is a bordered block lower triangular matrix (Figure 1) whose diagonal blocks are themselves bordered block triangular matrices, nested to any depth. When an extra spike is moved forward we add a border to a block triangular matrix because spikes always start in the same row as a block.

For actual factorization of the matrix, further permutations may be needed, since the diagonal blocks can be singular or nearly so, even with the P^5 algorithm which avoids structural singularity. For instance the Figure 12 example (Westerberg, private communication 1974) is already permuted to the form the algorithm (both versions) leaves. Even though the matrix is structurally nonsingular, pivoting in the order given will yield a seventh pivot that is zero (that is the leading 7×7 submatrix is structurally singular). Thus the normal implementation of the Hellerman and Rarick algorithm would fail. Note, however, that if we consider the matrix of Figure 12 to have a border of order 3 and regard the leading block diagonal matrix of order 6 as $\mathbf{A}_{11}$ in the matrix (2), then the Schur complement $\mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12}$ is not structurally singular

procedure assign pivots
begin
  for $i := 1$ **step** 1 **until** s **do**
  begin
    move the first spike column in the border ahead of the active submatrix;
    if it is possible to permute the rows so that the i diagonal coefficients
    ahead of the active submatrix are entries
    then do so
    else move the column ahead of the active matrix back to the border and
    goto quit;
  end $i := s{+}1$;
  quit: remove the leading $i - 1$ rows from the active submatrix;
end

Figure 11: The alternative procedure assign pivots, that converts the pseudo-algol program of Figure 6 to the Hellerman-Rarick algorithm.

$$\begin{matrix}
\times & & & & & & \times & \times & \times \\
& \times & & & & & \times & \times & \times \\
& & \times & & & & \times & \times & \times \\
& & & \times & & & \times & \times & \times \\
& & & & \times & \times & & \times & \times \\
& & & & \times & \times & & \times & \times \\
& & & & \times & \times & & \times & \times \\
\times & \times & \times & \times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times & \times & \times & \times
\end{matrix}$$

Figure 12: Example with structurally singular leading 7×7 submatrix.

and any method which forms this and allows pivoting when solving for the border variables will succeed (provided $\mathbf{A}_{11}$ is not numerically singular).

Hellerman and Rarick (1971) suggested permuting the spikes whenever necessary, and indeed interchanging columns 7 and 9 cures the structural singularity of the Figure 12 example. In general, however, such interchanges alter the form of the matrix even to the extent that the spikes no longer give a properly nested bordered block triangular form unless some of them are artificially elongated. For numerical stability it is necessary to avoid all small pivots, even in non-spike columns. Saunders (1972) suggests column interchanges on these grounds too. This may further worsen the structure since additional spikes are introduced, but in practice the problem is not serious if a mild relative pivot tolerance is used. The structure is not altered by row or column interchanges within inner blocks (for instance we did not alter the structure of the Figure 12 matrix) so it may be worthwhile to try to use such interchanges first.

4 Concluding remarks

The bordered block triangular form offers potential value for a parallel factorization. The steps of the algorithm outlined in this paper also have potential for investigating an efficient parallel implementation. However, the investigation into the implementation of the Hellerman-Rarick algorithm had not been done in time to publish the second edition of Direct Methods for Sparse Matrices, so we chose to eliminate this algorithm from the second edition. Rather than abandon the algorithm, which we believe has potential promise, we have outlined it and identified what needs to be done for further consideration.

Acknowledgements

We would like to thank Mike Saunders (Stanford) for his encouragement to issue this as a technical report and for comments on a draft version.

References

Duff, I. S., Erisman, A. M. and Reid, J. K. (1986), *Direct Methods for Sparse Matrices*, Oxford University Press, Oxford, UK.

Duff, I. S., Erisman, A. M. and Reid, J. K. (2017), *Direct Methods for Sparse Matrices, Second Edition*, Oxford University Press, Oxford, UK.

Erisman, A. M., Grimes, R. G., Lewis, J. G. and Poole, W. G. J. (1985), 'A structurally stable modification of Hellerman-Rarick's P^4 algorithm for reordering unsymmetric sparse matrices', *SIAM J. Numerical Analysis* **22**, 369–385.

Hellerman, E. and Rarick, D. C. (1971), 'Reinversion with the preassigned pivot procedure', *Mathematical Programming* **1**, 195–216.

Hellerman, E. and Rarick, D. C. (1972), The partitioned preassigned pivot procedure (P^4), *in* D. J. Rose and R. A. Willoughby, eds, 'Sparse Matrices and their Applications', Plenum Press, New York, pp. 67–76.

Kuhn, H. W. (1955), 'The Hungarian method for solving the assignment problem', *Naval Research Logistics Quarterly* **2**, 83–97.

Saunders, M. A. (1972), Product form of the Cholesky factorization for large-scale linear programming, Technical Report STAN-CS-72-301, Department of Computer Science, Stanford University, Stanford, California.

Tarjan, R. E. (1972), 'Depth-first search and linear graph algorithms', *SIAM J. Computing* **1**, 146–160.