Annales UMCS Informatica AI 6 (2007) 85-94

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI

http://www.annales.umcs.lublin.pl/

Markowitz scheme for the sparse WZ factorization

Beata Bylina*, Jarosław Bylina

Department of Computer Science, Institute of Mathematics, Marie Curie-Skłodowska University, pl. M. Curie-Skłodowskiej 1, 20-031 Lublin, Poland

Abstract

In this paper the authors present problems which can appear when a sparse square matrix (without any special structure) is factorized to a product of matrices **W** and **Z**. The fill-in problem is considered, and the manners of its solving – by permuting both rows and columns with a modified Markowitz scheme among others. The results of numerical experiments for sparse matrices of various sizes are presented and they show the Markowitz scheme applicability.

1. Introduction

In numerical methods of linear algebra, one of the most important issue is solving systems of linear equations of various forms – both when the coefficient matrix is dense (contains not too many zeros) and when it is sparse (contains a lot of zeros). For solving such systems the wide spectrum of methods is used – direct and iterative methods, among others. In direct methods the matrix is factorized to the products of two (LU, WZ, QR factorizations) or three (LDL^T factorization) factors. The factorization of sparse matrices (also called "sparse factorization" for short) usually consists of two stages. First (analysis), a symbolic factorization is made, by which it can be known what zero elements become non-zeros (because non-zeros have to be stored in memory, unlike zeros). Next, a numerical factorization is carried out.

Usually, during the factorization the number of non-zero elements grows rapidly and both the factors have much more non-zeros than the original matrix. To reduce growth of the number of non-zeros, the rows and columns of the original matrix may be permuted. There are $n!$ possible permutations and finding the best permutation (that is finding the one which gives the least growth of a number of non-zeros) is an NP complete problem.

*Corresponding author: *e-mail address*: beatas@hektor.umcs.lublin.pl

In this paper we consider WZ factorization (special kind of Gaussian elimination [1-3]) of a sparse matrix of dimension $n \times n$ with an arbitrary structure with the dominant diagonal.

Definition of a sparse matrix (a theoretical one). The matrix of dimension $n \times n$ is sparse when it contains "sufficiently few" non-zeros – usually $O(n)$ non-zeros.

Definition of a sparse matrix (a pragmatic one). The matrix is sparse when for a given computer system and a given method of computation, application of an algorithm taking into consideration the presence of zero elements allows to save computer memory or/and computation time.

For factorization of such a matrix, Gaussian elimination is used, well-known LU factorization the most frequent. Here, **L** is a lower triangular matrix with those on the diagonal and **U** is an upper triangular matrix.

The WZ factorization considered is a specific kind of Gaussian elimination. P. Yalamov and D.J. Evans [3] showed, for matrices with the dominant diagonal, that the WZ factorization can be faster than the LU factorization for some parallel computers. Here, we consider such a transformation of the original matrix, that reduces the number of non-zeros in factors – and simultaneously improves numerical properties of the factorization. The algorithm presented here is a greedy and heuristic algorithm – so it should be tested for many various sparse matrices.

The paper has the following layout. First (Section 2), we describe the WZ factorization, motivations for sparse factorization and the problem of the fill-in when we factorize an arbitrary structured matrix. Section 3 gives a simple example how the permutation of columns influences the sparsity of output matrices and that section describes the classical Markowitz scheme to reduce the number of output non-zeros in the LU factorization. Section 3 also presents a new modification of Markowitz scheme suitable for the WZ factorization, making it more stable and making its output more sparse. Section 4 describes an environment used for numerical experiments done with a number of matrices and shows its results. In that section we also deal with the time of the computations and the sparsity of the output matrices. The last section constitutes the conclusion.

2. The sparse WZ factorization

2.1. The WZ factorization

The WZ factorization was proposed by Evans and Hatzopoulos [1] as a proposition for SIMD computers. In [1-3] some modifications of the WZ factorization and its parallel implementations are considered.

Let **A** be a nonsingular matrix. In the WZ factorization we present **A** as a product of matrices **W** and **Z** of the following forms (for an odd $n = 5$):

$$\mathbf{W} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ w_{21} & 1 & 0 & 0 & w_{25} \\ w_{31} & w_{32} & 1 & w_{34} & w_{35} \\ w_{41} & 0 & 0 & 1 & w_{45} \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{W} = \begin{bmatrix} z_{11} & z_{12} & z_{13} & z_{14} & z_{15} \\ 0 & z_{22} & z_{23} & z_{24} & 0 \\ 0 & 0 & z_{33} & 0 & 0 \\ 0 & z_{42} & z_{43} & z_{44} & 0 \\ z_{51} & z_{52} & z_{53} & z_{54} & z_{55} \end{bmatrix}.$$

Fig. 1 presents the matrices **A** and **Z** (**A** becomes **Z**) during the factorization (actually, during its *k*th step) in a graphical form. This factorization in a sequential form is presented in Figure 2 (as a MATLAB-like program).

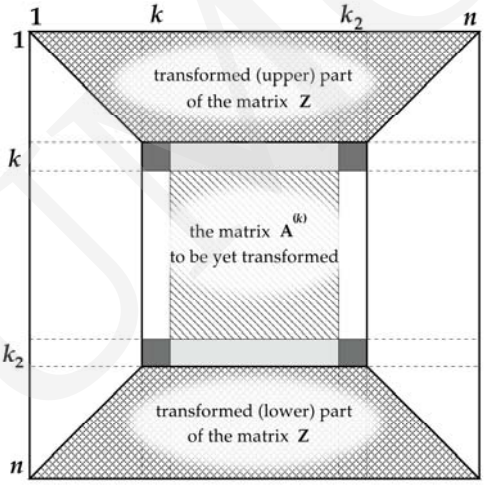

Fig. 1. The WZ factorization in a graphical form ($k_2 = n - k + 1$)

```
% elimination loop – reduction steps from A to Z
for k = 0 : m
   k2 = n-k-1;
   det=A(k,k)*A(k2,k2)-A(k2,k)*A(k,k2);
% finding elements of W
   for i = k+1 : k2-1
      wk1 = (A(k2,k)*A(i,k2)-A(k2,k2)*A(i,k))/det;
      wk2 = (A(k,k2)*A(i,k)-A(k,k)*A(i,k2))/det;
% updating A
      for j = k+1 : k2-1
         A(i,j) = A(i,j)+wk1*A(k,j)+wk2*A(k2,j);
```

Fig. 2. The sequential WZ factorization

2.2. The motivation for the sparse WZ factorization

The WZ factorization can be used, for example, to solve linear systems of the form $\mathbf{Ax} = \mathbf{b}$; after the factorization $\mathbf{A} = \mathbf{WZ}$ we have to solve a pair of systems:

$$\mathbf{Wc} = \mathbf{b},$$
$$\mathbf{Zx} = \mathbf{c}$$

($\mathbf{c}$ being an auxiliary vector). The other example is solving systems of the form $\mathbf{AX} = \mathbf{B}$ where $\mathbf{X}$ and $\mathbf{B}$ are the matrices of the same dimension – here the factorization is done only once.

All the problems mentioned above have often a sparse matrix $\mathbf{A}$ as its main matrix. Among others, such system appears during Markovian modeling of computer systems and networks [4], where elements a_{ij} of the matrix $\mathbf{A}$ are transition rates between given states. Usually there are only a few transitions from given states so the matrix $\mathbf{A}$ is sparse.

2.3. The fill-in problem

When a sparse matrix $\mathbf{A}$ is presented as a product of matrices, fill-in appears. That means that the output matrices have much more non-zeros than the input matrix $\mathbf{A}$. Table 1 shows number of non-zeros in the input matrix and in the output matrices for exemplary sparse matrices in the WZ factorization.

Table 1. Growth of the number of non-zeros for various sparse matrices

	number of non-zeros...	
size	...before factorization	...after factorization
200	2 340	30 940
1 000	11 917	742 892
2 000	23 928	2 923 322
3 000	35 933	6 624 009
5 000	59 922	18 417 468

To factorize a sparse matrix we need to use such algorithms and data structures, which can reduce fill-in and will be sufficiently fast.

3. The fill-in and the Markowitz scheme

To reduce the amount of fill-in we need to reorganize row and/or columns by permuting them. For example, for a given matrix of the dimension $n = 7$ with 19 non-zero elements (* denotes an arbitrary non-zero element):

$$\mathbf{A} = \begin{bmatrix} * & * & * & * & * & * & * \\ * & & & & & & \\ * & & & & & & \\ * & & & & & & \\ * & & & & & & \\ * & & & & & & \\ * & * & * & * & * & * & * \end{bmatrix}.$$

After the factorization we get matrices **W** and **Z** totally filled:

$$\mathbf{W} = \begin{bmatrix} 1 & & & & & & \\ * & 1 & & & & & * \\ * & * & 1 & & & * & * \\ * & * & * & 1 & * & * & * \\ * & * & & & 1 & * & * \\ * & & & & & 1 & * \\ & & & & & & 1 \end{bmatrix},$$

$$\mathbf{A} = \begin{bmatrix} * & * & * & * & * & * & * \\ & * & * & * & * & * & \\ & & * & * & * & & \\ & & & * & & & \\ & & * & * & * & & \\ & * & * & * & * & * & \\ * & * & * & * & * & * & * \end{bmatrix}.$$

However, after the permutation of columns 1 and 4 (and corresponding permutation of variables in vector **x** in the linear system solving, of course) we have the matrix

$$\mathbf{W} = \begin{bmatrix} * & * & * & * & * & * & * \\ & & & * & & & \\ & & & * & & & \\ & & & * & & & \\ & & & * & & & \\ & & & * & & & \\ * & * & * & * & * & * & * \end{bmatrix}.$$

Now we have no fill-in, because **Z** = **A** and **W** = **I**.

For the structured matrices (like symmetric ones) we can use the minimum degree algorithm [5] or the nested dissection [6].

Of course, we do not always know the structure of the matrix so there are heuristic algorithms which reorganize the matrix. Some of them include the Markowitz scheme [7] and the Markowitz scheme with threshold pivoting (for stability) [8]. In papers [9] and [10] some modifications of the Markowitz scheme are considered – namely pivoting constrained to the diagonal elements.

3.1. The Markowitz scheme for the LU factorization

The original Markowitz scheme was first presented in [7]. Let the matrix $\mathbf{A}^{(k)}$ of the dimension $(n-k+1)\times(n-k+1)$ be a matrix after kth step of the Gaussian elimination. Let $r_i^{(k)}$ be the number of non-zeros in the ith row of the matrix $\mathbf{A}^{(k)}$. Let $c_j^{(k)}$ be the number of non-zeros in the jth column of the matrix $\mathbf{A}^{(k)}$. We choose $w_{ij}^{(k)} = \min\limits_{k\le i,j\le n} r_i^{(k)} c_j^{(k)}$ and swap the kth row with the ith row and the kth column with the jth column. Unfortunately, such an algorithm can lead to a zero pivot and hence make the factorization fail. There are modifications of the Markowitz scheme which ensure success of factorization (as in [8]).

3.2. The Markowitz scheme for the WZ factorization

Now we present a new modification of the Markowitz scheme, suitable for the WZ factorization. Let the matrix $\mathbf{A}^{(k)}$ of the dimension $(n-2k+2)\times(n-2k+2)$ be a matrix after kth step of the WZ factorization. Let $r_{i_1}^{(k)}$ be the number of non-zeros in the i_1st row of the matrix $\mathbf{A}^{(k)}$. Let $r_{i_2}^{(k)}$ be the number of non-zeros in the i_2nd row of the matrix $\mathbf{A}^{(k)}$.

We choose $w_i^{(k)} = \min\limits_{k\le i\le k_2} r_i^{(k)}$ and $w_j^{(k)} = \min\limits_{k\le j\le k_2, j\ne i} r_j^{(k)}$ and swap the kth row with the ith row and the k_2nd row with the jth row.

Of course, the determinant $a_{kk}^{(k)} a_{k_2k_2}^{(k)} - a_{kk_2}^{(k)} a_{k_2k}^{(k)}$ (which is the pivot by which we divide in the WZ factorization), after the swapping of the rows, can be equal to zero, so choosing i and j we have to pay attention not to choose rows which give us zero determinant.

In our scheme there is a need to make a lot of comparisons, which can influence the time of computations as discused in Section 4.

Figure 3 presents the sequential WZ factorization with the use of the modified Markowitz scheme.

```
% elimination loop – reduction steps from A to Z
for k = 0 : m
   k2 = n-k-1;
% here we choose and swap rows – as described in the article
% . . .
% now, after swapping
   det=A(k,k)*A(k2,k2)-A(k2,k)*A(k,k2);
% finding elements of W
   for i = k+1 : k2-1
      wk1 = (A(k2,k)*A(i,k2)-A(k2,k2)*A(i,k))/det;
      wk2 = (A(k,k2)*A(i,k)-A(k,k)*A(i,k2))/det;
% updating A
      for j = k+1 : k2-1
         A(i,j) = A(i,j)+wk1*A(k,j)+wk2*A(k2,j);
```

Fig. 3. The sequential WZ factorization with Markowitz scheme

4. The numerical experiment

For the implementation of the WZ factorization with our modification of the Markowitz scheme we used the language C. For storing all the matrices (namely, **A**, **W**, **Z**) we use two-dimensional arrays totally fitting into RAM. The tests were done using a Intel Pentium 2.8GHz computer with 1GB RAM powered by the GNU/Linux operating system. The programs were compiled with the *gcc* compiler with `-O3` optimization option. Tests were done for matrices generated by the authors (randomly but with a dominant diagonal) which all have approximately the same density coefficient equal to 12 elements in a row. Figures 4-8 present the results of the experiments.

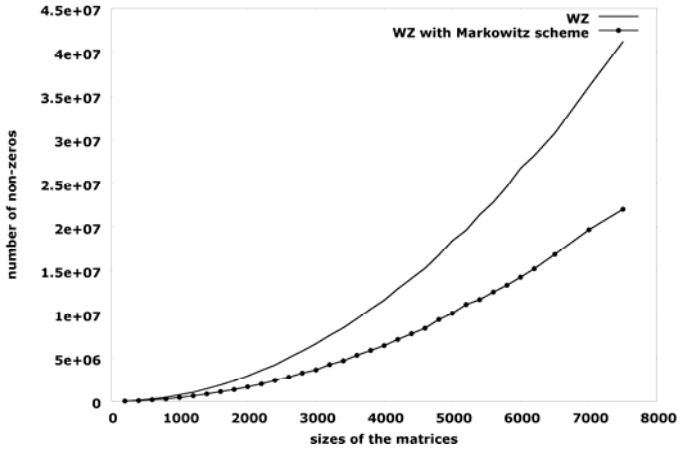

Fig. 4. Number of non-zeros in the output matrices for both kinds of WZ

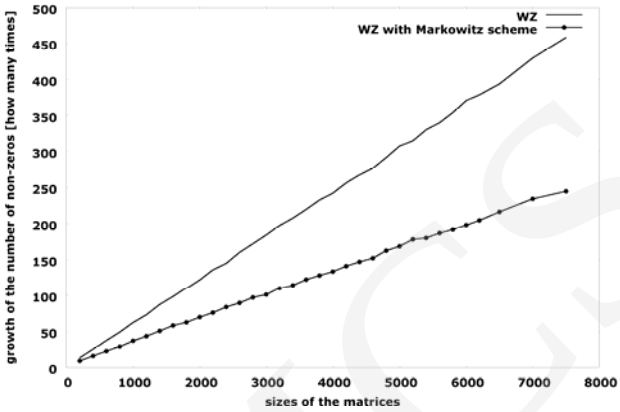

Fig. 5. Growth of the number of non-zeros in the output matrices for both kinds of WZ

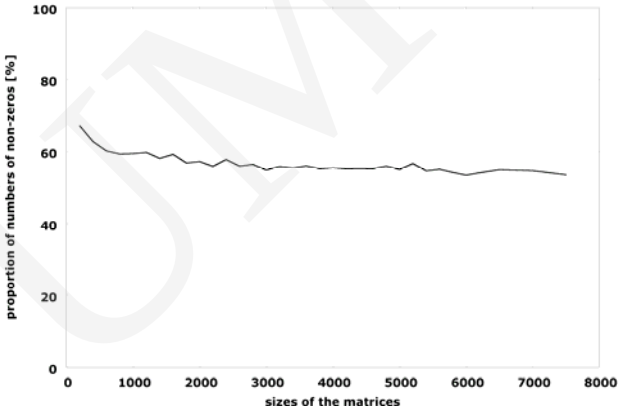

Fig. 6. Proportion of the number of non-zeros in WZ factorization with the Markowitz scheme to that number in the traditional WZ factorization

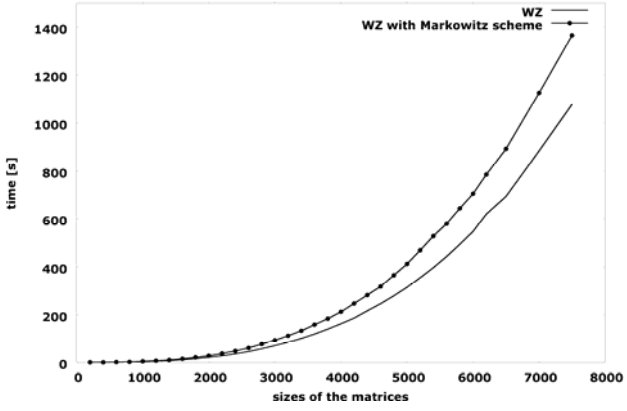

Fig. 7. Times of computations

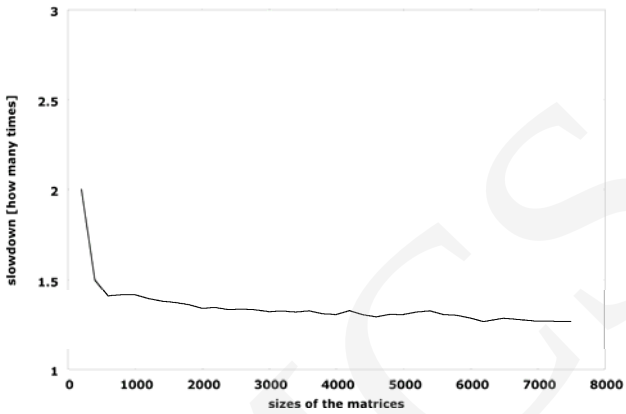

Fig. 8. Slowdown (proportion of times) of computations

The use of the Markowitz scheme improved considerably (that is reduced by about 40%) the number of non-zero elements in the output matrices but lengthened the time of the computation by about 50% (decreasing for larger matrices).

5. Conclusion

In the paper we show purposefulness of dealing with sparse matrices. Next we present the fill-in problem and the need to permute rows and/or columns of factorized matrices. We also describe a modification of the Markowitz scheme suitable for the WZ factorization and its effects in real computations.

It is worth mentioning that for compressed matrices slowdown of the factorization caused by the Markowitz scheme can be much lower because the structures are smaller and there are fewer elements to check in the Markowitz phase of the factorization.

References

[1] Evans D. J., Hatzopoulos M., *The parallel solution of linear system*. Int. J. Comp. Math., 7 (1979) 227.

[2] Chandra Sekhara Rao S., *Existence and uniqueness of WZ factorization*. Parallel Computing, 23 (1997) 1129.

[3] Yalamov P., Evans D. J., *The WZ matrix factorization method*. Parallel Computing, 21 (1995) 1111.

[4] Bylina B., Bylina J., *The Vectorized and Parallelized Solving of Markovian Models for Optical Networks*. Lecture Notes in Computer Science, 3037 (2004) 578.

[5] Tinney W .F., Walker J. W., *Direct solution of sparse network equations by optimally ordered triangular factorization*. Proc. IEEE, 55 (1967) 1801.

[6] Reid J., Duff I. S., Erisman A. M., *On George's nested dissection method*. SIAM J. Numer. Anal., 13 (1976) 686.

[7] Markowitz H. M., *The elimination form of the inverse and its application to linear programming*. Management Science, 3 (1957), 255.
[8] Duff I. S., Erisman A. M., Reid J., *Direct Methods for Sparse Matrices*. Oxford University Press, New York, (1986).
[9] Amestoy P., Li X. S., Ng E. G., *Diagonal Markowitz Scheme with Local Symmetrization* (Report LBNL-53854, Dec. 2003). SIAM J. Matrix Anal. and Appl., 29 (2007) 228.
[10] Amestoy P., Pralet S., *Unsymmetric Ordering Using a Constrained Markowitz Scheme*, submitted to SIAM J. Matrix Anal. and Appl. (Report LBNL-56861, Jan. 2005).