Thai Journal of Mathematics
Volume 19 Number 4 (2021)
Pages 1461–1476

http://thaijmath.in.cmu.ac.th

ISSN 1686-0209

Quadrant Interlocking Factorization Algorithm of Hourglass Matrix from Nonsingular Matrix

Olayiwola Babarinsa[1,3,*], **Hailiza Kamarulhaili**[2] **and Azfi Zaidi Mohammad Sofi**[1]

[1] *Faculty of Bioengineering & Technology, Universiti Malaysia Kelantan, Jeli, Malaysia*
e-mail : olayiwola@umk.edu.my (O. Babarinsa); azfi.ms@umk.edu.my (A. Z. M. Sofi)

[2] *School of Mathematical Sciences, Universiti Sains Malaysia, Penang, Malaysia*
e-mail : hailiza@usm.my (H. Kamarulhaili)

[3] *Department of Mathematical Sciences, Federal University Lokoja, Kogi State, Nigeria*

Abstract This paper presents the quadrant interlocking factorization (QIF) of nonsingular matrix, alternatively called WH factorization, to yield hourglass matrix (H-matrix). The WH factorization algorithm of H-matrix is synonymous to WZ factorization algorithm of Z-matrix, unlike LU factorization. We examine the conditions to generate the zero and nonzero entries of H-matrix from the factorization algorithm, and compare the H-matrix and Z-matrix. Then we conclude that the existence of WH factorization implies WZ factorization.

MSC: 15A23; 15B99; 65F35

Keywords: quadrant interlocking factorization; WZ-factorization; H-matrix; Z-matrix; LU decomposition

Submission date: 13.09.2017 / Acceptance date: 10.05.2020

1. Introduction

Demeure [1] coined the word hourglass matrix in describing the method of factorizing matrices, especially from Toeplitz matrix and Hankel matrix, from bowtie-hourglass factorization or quadrant interlocking factorization. He further explained that hourglass matrix is analogous to partitioned Z-matrix into Z-system (2×2 triangular block systems) [2]. Over time, hourglass matrix is used interchangeably with Z-matrix due to the structural form of the matrix in resemblance with hourglass device. Unfortunately, there are changes in structure of Z-matrix which depend on the type of matrix (Toeplitz, Hankel, Hermitian, centrosymmetric, diagonally dominant or tridiagonal matrix) being factorized when using quadrant interlocking factorization (QIF) [3]. However, Evans and Hatzopoulos [4] first posited QIF or WZ factorization of nonsingular matrix and gave details of the factorization as well as the avoidance of breakdown of the factorization algorithm. The stability of QIF comes from the centro-nonsingular matrix which is far

*Corresponding author.

Published by The Mathematical Association of Thailand.

reliable than any other type of factorization, such as LU factorization [5, 6]. LU factorization ($B = LU$) is the representation of a nonsingular matrix in the form of a lower triangular and an upper triangular matrix. Matrix inversion combined with the low computational complexity and partial pivoting techniques makes LU-factorization extremely efficient [7, 8]. LU factorization may fail to occur, but a proper permutation in rows or columns is sufficient for the LU factorization which makes it numerically stable with $\left(\frac{2}{3}n^3 + \frac{1}{2}n^2 - \frac{7}{6}n\right)$ arithmetic operations [9, 10].

WZ factorization offers parallelization to solve linear system in enhancing performance using OpenMP, OmpSs, CUDA, BLAS or EDK HW/SW codesign architecture on $SIMD$ or $MIMD$ shared memory parallel computers or mesh multiprocessors, see [11–16] and the references therein. The factorization is known for its adaptability to use direct method in solving $n \times n$ linear system defined as [17]

$$Bx = c, \tag{1.1}$$

where,

$$B = (b_{i,j})\ 1 \leq i, j \leq n\,, \quad x = (x_1, ..., x_n)^T, \quad c = (c_1, ..., c_n)^T; \quad x, c \in \mathbb{R}^n, \quad B \in \mathbb{R}^{n \times n}.$$

According to Evans and Hatzopoulos [18], Z-matrix exists together with a W-matrix during the factorization of nonsingular matrix B such that

$$B = WZ. \tag{1.2}$$

A matrix which is either a Z-matrix or a W-matrix is called butterfly matrix. Z-matrix and W-matrix are names suggested by the shapes of the set of all possible positions for nonzero entries given below

$$W = \begin{bmatrix} \bullet & & & & & & & \bullet \\ \bullet & \circ & & & & & \circ & \bullet \\ \bullet & \circ & \circ & & & \circ & \circ & \bullet \\ \bullet & \circ & \circ & \circ & \circ & \circ & \circ & \bullet \\ \bullet & \circ & \circ & \bullet & \bullet & & \circ & \bullet \\ \bullet & \circ & \bullet & & & \bullet & \circ & \bullet \\ \bullet & \bullet & & & & & \bullet & \bullet \\ \bullet & & & & & & & \bullet \end{bmatrix} \qquad Z = \begin{bmatrix} \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ & \circ & \circ & \circ & \circ & \circ & \bullet & \\ & & \circ & \circ & \circ & \bullet & & \\ & & & \circ & \bullet & & & \\ & & & \bullet & \circ & & & \\ & & \bullet & \circ & \circ & \circ & & \\ & \bullet & \circ & \circ & \circ & \circ & \circ & \\ \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \end{bmatrix}$$

The WZ factorization breaks up the nonsingular matrix to structural forms which are then regrouped and solved as sub-blocks [2, 19]. For the factorization, we compute $w_{i,k}^{(k)}$ and $w_{i,n-k+1}^{(k)}$ for W-matrix from Equation (1.3) by solving its 2×2 linear systems for every update of matrix B,

$$\begin{cases} z_{k,k}^{(k-1)} w_{i,k}^{(k)} + z_{n-k+1,k}^{(k-1)} w_{i,n-k+1}^{(k)} = z_{i,k}^{(k-1)}; \\ z_{k,n-k+1}^{(k-1)} w_{i,k}^{(k)} + z_{n-k+1,n-k+1}^{(k-1)} w_{i,n-k+1}^{(k)} = z_{i,n-k+1}^{(k-1)}, \end{cases} \tag{1.3}$$

where $k = 1, 2, ..., \lfloor \frac{n}{2} \rfloor$; $i = k+1, ..., n-k$. For the Z-matrix, its entries are obtained from Equation (1.4)

$$z_{i,j}^{(k)} = z_{i,j}^{(k-1)} - w_{i,k}^{(k)} z_{k,j}^{(k-1)} - w_{i,n-k+1}^{(k)} z_{n-k+1,j}^{(k-1)}, \tag{1.4}$$

where $j = k+1, ..., n-k$. The direct method to solve the linear systems of Equation (1.3) under the nonsingularity constraint presumed for their determinants solely depends on a conventional method called Cramer's rule [20, 21]. The unique solution provided by Cramer's rule to the system in Equation (1.1) is given by [22]

$$x = \frac{det(B_{i|c})}{det(B)}, \tag{1.5}$$

where $B_{i|c}$ is the matrix obtained from B by substituting the vector column of c to the ith column of B, for $i = 1, 2, ..., n$. The advantage of using Cramer's rule to solve all $\sum_{k=1}^{\lfloor \frac{n}{2}-1 \rfloor} (n-2k)$ of 2×2 linear systems in the factorization process is to check if the matrix is centro-singular and to adopt the least matrix norm [23, 24]. To properly portray the notion and context of hourglass matrix in Section 2, we restrict the computed entries $(h_{i,j}^{(k)})$ of hourglass matrix in the factorization process to be nonzero.

2. Hourglass Matrix

Before we proceed, other notions of hourglass (stiffness) matrix that do not portray what we discuss in this paper are based on stabilization of hourglass control to reduce the hourglass effect, see for examples [25–27]. Now, hourglass matrix of order n ($n \geq 3$) is a nonsingular matrix given in Definition 2.1.

Definition 2.1. An hourglass matrix (H-matrix) is a nonsingular matrix of order n ($n \geq 3$) with nonzero entries from the ith to the $(n-i+1)$ element of the ith and $(n-i+1)$ row of the matrix, otherwise 0's, for $i = 1, 2, ..., \lfloor \frac{n+1}{2} \rfloor$.

Unlike Z-matrix with unrestricted nonzero entries, hourglass matrix with nonzero elements denoted with black dots has structural comparison with an hourglass device, see Figure 1.

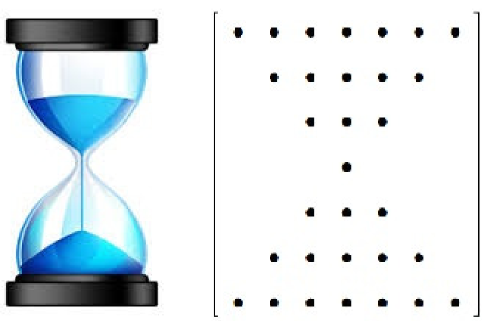

Figure 1. Structure of hourglass device and hourglass matrix.

In this article, quadrant interlocking factorization algorithm of nonsingular matrix to yield hourglass matrix will be referred as WH factorization. Like the factorization of Z-matrix, the factorization of H-matrix requires W-matrix to be computed during the WH factorization of nonsingular matrix B. Thus, H-matrix exists together with W-matrix such that

$$B = WH. \tag{2.1}$$

H-matrix and W-matrix of order n ($n \geq 3$) are generally defined as

$$H = \begin{cases} h_{ij}, & 1 \leq i \leq \lfloor\frac{(n+1)}{2}\rfloor \quad i \leq j \leq n+1-i; \\ h_{ij}, & \lceil\frac{(n+2)}{2}\rceil \leq i \leq n \quad n+1-i \leq j \leq i; \\ 0, & otherwise. \end{cases}$$

$$W = \begin{cases} (1, \underbrace{0, ..., 0}_{n-1}); \\ (w_{i,1}, ..., w_{i,i-1}, 1, \underbrace{0, ..., 0}_{n-2i+1}, w_{i,n-i+2}, ..., w_{i,n}), \quad i = 2, ..., \lfloor\frac{(n+1)}{2}\rfloor; \\ (w_{i,1}, ..., w_{i,n-i}, \underbrace{0, ..., 0}_{2i-n-1}, 1, w_{i,i+1}, ..., w_{i,n}), \quad i = \lfloor\frac{(n+1)}{2}\rfloor + 1, ..., n-1; \\ (\underbrace{0, ..., 0}_{n-1}, 1). \end{cases}$$

2.1. WH Factorization Algorithm

The QIF factorization of H-matrix and Z-matrix are quite similar, yet the factorization for H-matrix restricts the computed entries to be nonzero at every stage during the factorization. WH factorization specifies the number of times row-interchange can be done at each stage of the factorization if the computed entries yield zero, else the factorization breakdown. Based on the algorithm made by [28], we modify the sequential steps for the factorization are as follows:

Step 1: Let $B = H^{(0)}$ for initial update and check if the first row $\left(h^{(0)}_{1,j}\right)$ and last row $\left(h^{(0)}_{n,j}\right)$ of $H^{(0)}$ contains zero. If $h^{(0)}_{1,j} = 0$ or $h^{(0)}_{n,j} = 0$, then use suitable row-interchange in $H^{(0)}$, where $j = 1, 2, ..., n$. Then, we compute $w^{(1)}_{i,1}$ and $w^{(1)}_{i,n}$ in Equation (2.2) from matrix $H^{(0)}$ by solving 2×2 system of linear equations via Equation (1.5)

$$\begin{cases} h^{(0)}_{1,1}w^{(1)}_{i,1} + h^{(0)}_{n,1}w^{(1)}_{i,n} = h^{(0)}_{i,1}; \\ h^{(0)}_{1,n}w^{(1)}_{i,1} + h^{(0)}_{n,n}w^{(1)}_{i,n} = h^{(0)}_{i,n}, \end{cases} \tag{2.2}$$

to have

$$w^{(1)}_{i,1} = \frac{h^{(0)}_{n,n}h^{(0)}_{i,1} - h^{(0)}_{n,1}h^{(0)}_{i,n}}{h^{(0)}_{n,n}h^{(0)}_{1,1} - h^{(0)}_{1,n}h^{(0)}_{n,1}} \quad \text{and} \quad w^{(1)}_{i,n} = \frac{h^{(0)}_{1,1}h^{(0)}_{i,n} - h^{(0)}_{1,n}h^{(0)}_{i,1}}{h^{(0)}_{n,n}h^{(0)}_{1,1} - h^{(0)}_{1,n}h^{(0)}_{n,1}},$$

Whenever $h^{(0)}_{n,n}h^{(0)}_{1,1} - h^{(0)}_{1,n}h^{(0)}_{n,1} = 0$ use suitable row-interchange to avoid factorization breakdown. Then the values of $w^{(1)}_{i,1}$ and $w^{(1)}_{i,n}$ can be written in W-matrix as:

$$W^{(1)} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ w^{(1)}_{2,1} & 1 & \cdots & \iddots & w^{(1)}_{2,n} \\ \vdots & 0 & \ddots & 0 & \vdots \\ w^{(1)}_{n-1,1} & \iddots & \cdots & 1 & w^{(1)}_{n-1,n} \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}$$

Step 2: We, therefore, update matrix $H^{(0)}$ to $H^{(1)}$ for the first update by evaluating its entries as

$$h_{i,j}^{(1)} = h_{i,j}^{(0)} - w_{i,1}^{(1)} h_{1,j}^{(0)} - w_{i,n}^{(1)} h_{n,j}^{(0)}, \tag{2.3}$$

where $i, j = 2, ..., n-1$. If one of the computed entry $h_{2,j}^{(1)} = 0$ or $h_{n-1,j}^{(1)} = 0$ in Equation (2.3), then apply row-interchange in $H^{(1)}$ at $h_{i,j}^{(1)}$ for $i, j = 2, ..., n-1$ in no more than $(n-2)$ times, else the factorization breakdown. Thus, updating $h_{i,j}^{(1)}$ we have a matrix of the form

$$H^{(1)} = \begin{bmatrix} h_{1,1}^{(0)} & h_{1,2}^{(0)} & \cdots & \cdots & h_{1,n-1}^{(0)} & h_{1,n}^{(0)} \\ 0 & h_{2,2}^{(1)} & \cdots & \cdots & h_{2,n-1}^{(1)} & 0 \\ \vdots & \vdots & & & \vdots & \vdots \\ \vdots & \vdots & & & \vdots & \vdots \\ 0 & h_{n-1,2}^{(1)} & \cdots & \cdots & h_{n-1,n-1}^{(1)} & 0 \\ h_{n,1}^{(0)} & h_{n,2}^{(0)} & \cdots & \cdots & h_{n,n-1}^{(0)} & h_{n,n}^{(0)} \end{bmatrix}$$

Step 3: Next, ensure $h_{2,j}^{(1)} \neq 0$ and $h_{n-1,j}^{(1)} \neq 0$ and compute $w_{i,2}^{(2)}$ and $w_{i,n-1}^{(2)}$ from $H^{(1)}$ by solving 2×2 systems in Equation (2.4) to have

$$\begin{cases} h_{2,2}^{(1)} w_{i,2}^{(2)} + h_{n-1,2}^{(1)} w_{i,n-1}^{(2)} = h_{i,2}^{(1)}; \\ h_{2,n-1}^{(1)} w_{i,2}^{(2)} + h_{n-1,n-1}^{(1)} w_{i,n-1}^{(2)} = h_{i,n-1}^{(1)}. \end{cases} \tag{2.4}$$

Then,

$$w_{i,2}^{(2)} = \frac{h_{n-1,n-1}^{(1)} h_{i,2}^{(1)} - h_{n-1,2}^{(1)} h_{i,n-1}^{(1)}}{h_{n-1,n-1}^{(1)} h_{2,2}^{(1)} - h_{2,n-1}^{(1)} h_{n-1,2}^{(1)}} \text{ and } w_{i,n-1}^{(2)} = \frac{h_{2,2}^{(1)} h_{i,n-1}^{(1)} - h_{2,n-1}^{(1)} h_{i,2}^{(1)}}{h_{n-1,n-1}^{(1)} h_{2,2}^{(1)} - h_{2,n-1}^{(1)} h_{n-1,2}^{(1)}}.$$

Thus, we write the values of $w_{i,2}^{(2)}$ and $w_{i,n-1}^{(2)}$ in W-matrix as:

$$W^{(2)} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ w_{2,1}^{(1)} & 1 & 0 & \cdots & 0 & 0 & w_{2,n}^{(1)} \\ \vdots & w_{3,2}^{(2)} & 1 & \cdots & \iddots & w_{3,n-1}^{(2)} & \vdots \\ \vdots & \vdots & 0 & \ddots & 0 & \vdots & \vdots \\ \vdots & w_{n-2,2}^{(2)} & \iddots & \cdots & 1 & w_{n-2,n-1}^{(2)} & \vdots \\ w_{n-1,1}^{(1)} & 0 & 0 & \cdots & 0 & 1 & w_{n-1,n}^{(1)} \\ 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix}$$

Step 4: We update matrix $H^{(1)}$ to $H^{(2)}$ for the second update by evaluating its entries as

$$h_{i,j}^{(2)} = h_{i,j}^{(1)} - w_{i,2}^{(2)} h_{2,j}^{(1)} - w_{i,n-1}^{(2)} h_{n-1,j}^{(1)} \neq 0, \tag{2.5}$$

where $i, j = 3, ..., n-2$. If one of the computed entry $h_{3,j}^{(2)} = 0$ or $h_{n-2,j}^{(1)} = 0$ in Equation (2.5), then apply row-interchange in $H^{(2)}$ at $h_{i,j}^{(2)}$ for $i, j = 3, ..., n-2$ in no more than $(n-4)$ times, else the factorization breakdown. Thus, updating $h_{i,j}^{(1)}$ to $h_{i,j}^{(2)}$ we have

$$H^{(2)} = \begin{bmatrix} h_{1,1}^{(0)} & h_{1,2}^{(0)} & h_{1,3}^{(0)} & \cdots & \cdots & h_{1,n-2}^{(0)} & h_{1,n-1}^{(0)} & h_{1,n}^{(0)} \\ 0 & h_{2,2}^{(1)} & h_{2,3}^{(1)} & \cdots & \cdots & h_{2,n-2}^{(1)} & h_{2,n-1}^{(1)} & 0 \\ 0 & 0 & h_{3,3}^{(2)} & \cdots & \cdots & h_{3,n-2}^{(2)} & 0 & 0 \\ \vdots & \vdots & \vdots & & & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & & & \vdots & \vdots & \vdots \\ 0 & 0 & h_{n-2,3}^{(2)} & \cdots & \cdots & h_{n-2,n-2}^{(2)} & 0 & 0 \\ 0 & h_{n-1,2}^{(1)} & h_{n-1,3}^{(1)} & \cdots & \cdots & h_{n-2,n-2}^{(1)} & h_{n-1,n-1}^{(1)} & 0 \\ h_{n,1}^{(0)} & h_{n,2}^{(0)} & h_{n,3}^{(0)} & \cdots & \cdots & h_{n,n-2}^{(0)} & h_{n,n-1}^{(0)} & h_{n,n}^{(0)} \end{bmatrix}$$

Step 5: Now, we compute $w_{i,k}^{(k)}$ and $w_{i,n-k+1}^{(k)}$ from matrix $H^{(k-1)}$ by solving 2×2 linear systems in Equation (2.6) to generalize for every update of $H^{(k)}$ and proceed similarly for the inner square matrices of size $(n-2k)$ and so on. That is,

$$\begin{cases} h_{k,k}^{(k-1)} w_{i,k}^{(k)} + h_{n-k+1,k}^{(k-1)} w_{i,n-k+1}^{(k)} = h_{i,k}^{(k-1)}; \\ h_{k,n-k+1}^{(k-1)} w_{i,k}^{(k)} + h_{n-k+1,n-k+1}^{(k-1)} w_{i,n-k+1}^{(k)} = h_{i,n-k+1}^{(k-1)}, \end{cases} \tag{2.6}$$

where $k = 1, 2, ..., \lfloor \frac{n-1}{2} \rfloor$; $i = k+1, ..., n-k$. Then,

$$w_{i,k}^{(k)} = \frac{h_{n-k+1,n-k+1}^{(k-1)} h_{i,k}^{(k-1)} - h_{n-k+1,k}^{(k-1)} h_{i,n-k+1}^{(k-1)}}{h_{n-k+1,n-k+1}^{(k-1)} h_{k,k}^{(k-1)} - h_{n-k+1,k}^{(k-1)} h_{k,n-k+1}^{(k-1)}}$$

and

$$w_{i,n-k+1}^{(k)} = \frac{h_{k,k}^{(k-1)} h_{i,n-k+1}^{(k-1)} - h_{k,n-k+1}^{(k-1)} h_{i,k}^{(k-1)}}{h_{n-k+1,n-k+1}^{(k-1)} h_{k,k}^{(k-1)} - h_{n-k+1,k}^{(k-1)} h_{k,n-k+1}^{(k-1)}}.$$

Then, we put the values $w_{i,k}^{(k)}$ and $w_{i,n-k+1}^{(k)}$ in a W-matrix of the form as

$$W^{(k)} = \begin{bmatrix} 1 & & & & & & 0 \\ & \ddots & & & & \iddots & \\ & & 1 & & 0 & & \\ & & w_{k+1,k}^{(k)} & \ddots \quad\quad \iddots & w_{k+1,n-k+1}^{(k)} & & \\ & & \vdots & \ddots & \vdots & & \\ & & w_{n-1,k}^{(k)} & \iddots \quad\quad \ddots & w_{n-k,n-k+1}^{(k)} & & \\ & & 0 & & 1 & & \\ & \iddots & & & & \ddots & \\ 0 & & & & & & 1 \end{bmatrix}$$

Step 6: We finally compute for kth steps of $h_{i,j}^{(k)}$ as:

$$h_{i,j}^{(k)} = h_{i,j}^{(k-1)} - w_{i,k}^{(k)} h_{k,j}^{(k-1)} - w_{i,n-k+1}^{(k)} h_{n-k+1,j}^{(k-1)}, \tag{2.7}$$

where $j = k+1, ..., n-k$. From Equation (2.7), if one of the computed entries is zero, then apply possible row-interchange in no more than $(n-2k)$ times in $H^{(k-1)}$ and re-factorize, else the factorization breakdown to produce H^k (H-matrix). After the successful kth steps we get hourglass matrix of the form:

$$
H=\begin{bmatrix}
h^{(0)}_{1,1} & h^{(0)}_{1,2} & h^{(0)}_{1,3} & \cdots & \cdots & \cdots & \cdots & \cdots & h^{(0)}_{1,n-2} & h^{(0)}_{1,n-1} & h^{(0)}_{1,n} \\
0 & h^{(1)}_{2,2} & h^{(1)}_{2,3} & \cdots & \cdots & \cdots & \cdots & \cdots & h^{(1)}_{2,n-2} & h^{(1)}_{2,n-1} & 0 \\
0 & 0 & h^{(2)}_{3,3} & \cdots & \cdots & \cdots & \cdots & \cdots & h^{(2)}_{3,n-2} & 0 & 0 \\
\vdots & 0 & 0 & \ddots & \vdots & \vdots & \vdots & \iddots & 0 & 0 & \vdots \\
\vdots & \vdots & \vdots & & h^{(k-1)}_{k,k} & \cdots & h^{(k-1)}_{k,n-k+1} & & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \vdots & & \vdots & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & & h^{(k-1)}_{n-k+1,k} & \cdots & h^{(k-1)}_{n-k+1,n-k+1} & & \vdots & \vdots & \vdots \\
\vdots & 0 & 0 & \iddots & \vdots & \vdots & \vdots & \ddots & 0 & 0 & \vdots \\
0 & 0 & h^{(2)}_{n-2,3} & \cdots & \cdots & \cdots & \cdots & \cdots & h^{(2)}_{n-2,n-2} & 0 & 0 \\
0 & h^{(1)}_{n-1,2} & h^{(1)}_{n-1,3} & \cdots & \cdots & \cdots & \cdots & \cdots & h^{(1)}_{n-1,n-2} & h^{(1)}_{n-1,n-1} & 0 \\
h^{(0)}_{n,1} & h^{(0)}_{n,2} & h^{(0)}_{n,3} & \cdots & \cdots & \cdots & \cdots & \cdots & h^{(0)}_{n,n-2} & h^{(0)}_{n,n-1} & h^{(0)}_{n,n}
\end{bmatrix}
$$

From the above algorithmic steps, The MATLAB code to compute H-matrix from WH factorization of nonsingular matrix is given in Listing 1.

Listing 1. MATLAB code of WH factorization.

```
function H = WHfactorization(B)
 % step of elimination - from B to H
 B=input('matrix B =');
 n = size(B, 1);
 W = zeros(n);
 counter = 0;
 tic
 for k = 1:ceil((n-1)/2)
     k2 = n - k + 1 ;
     % insert code here to check for non-zero
     % check first row
     first_flag = any(B(k,k:k2) == 0);
     % check last row
     last_flag = any(B(k2, k:k2) == 0);
     % let zero_row_count be the number of row needed
     zero_row_count = first_flag + last_flag;
     % detect if we have enough such rows between them
     if zero_row_count > 0
         potential_non_zero = zeros(n,1);
         for between_index = (k+1) : (k2-1)
             if all(B(between_index, k:k2))
                 potential_non_zero(between_index)=1;
             end
         end
         potential_non_zero = find(potential_non_zero);
         if length(potential_non_zero) < zero_row_count
             % if there isn't enough such row exit
             H = B;
              disp('cannot perform switching')
             return
         else
             chosen_index = datasample(potential_non_zero, zero_row_count, 'replace', false);
             % if there is sufficient, swap with those.
             if zero_row_count > 1
                 % switch both
                 tmp = B(chosen_index(1),k:k2);
                 B(chosen_index(1), k:k2) = B(k, k:k2);
                 B(k, k:k2) = tmp;
                 tmp = B(chosen_index(2), k:k2);
                 B(chosen_index(2), k:k2) = B(k2, k: k2);
                 B(k2, k: k2) = tmp;
                 counter = counter + 2;
             else
```

```
                  % switch one of them
                  if first_flag
                      tmp = B(chosen_index(1),k:k2);
                      B(chosen_index(1), k:k2) = B(k, k:k2);
                      B(k, k:k2) = tmp;
                      counter = counter + 1;
                  else
                      tmp = B(chosen_index(1), k:k2);
                      B(chosen_index(1), k:k2) = B(k2, k: k2);
                      B(k2, k: k2) = tmp;
                      counter = counter + 1;
                  end
              end
          end
      end
     % end of inserting code
      determinant = B(k,k) * B(k2,k2) - B(k2,k) * B(k,k2);
%       disp('determinant =')
%       disp(determinant)
      if determinant == 0
          exitflag = 0;
          for i1 = k:k2
              for i2 = i1:k2
                  determinant = B(i1,k) * B(i2,k2) - B(i2,k) * B(i1,k2);
                  if determinant ~= 0
                      disp('input matrix cannot be factorized to hourglass matrix'
                          )
                      tmp = B(i1,k:k2);
                      B(i1,k:k2) = B(k,k:k2);
                      B(k,k:k2) = tmp;
                      tmp = B(i2,k:k2);
                      B(i2,k:k2) = B(k2,k:k2);
                      B(k2,k:k2) = tmp;
                      exitflag = 1;
                      break
                  end  % end if determinant ~= 0
              end  % end of i2
          end  % end of i1
          if exitflag == 0
              H = B;
              return
          end
      end  % end if determinant == 0
    % finding elements of W
       W(k+1:k2-1,k)=(B(k2,k2)*B(k+1:k2-1,k)-B(k2,k)*B(k+1:k2-1,k2))/determinant;
       W(k+1:k2-1,k2)=(B(k,k)*B(k+1:k2-1,k2)-B(k,k2)*B(k+1:k2-1,k))/determinant;
     for m=1:n
     W(m,m)=1;
     W(m,n+1-m);
   end
    % updating B
     B(k+1:k2-1,k) = 0;
     B(k+1:k2-1,k2) = 0;
     B(k+1:k2-1,k+1:k2-1) = B(k+1:k2-1,k+1:k2-1) - W(k+1:k2-1,k) * B(k,k+1:k2-1)
         - W(k+1:k2-1,k2) * B(k2,k+1:k2-1);
     if B(k+1:k2-1,k+1:k2-1)==0
         error ('computed entries cannot form Hourglass matrix')
     end
     H = B;
      permutation = counter
      W = W - diag(diag(W)) + eye(n);
      disp(W)
      disp(H)
toc
end
```

LISTING 2. MATLAB code for random hourglass matrix.

```
function B = random_H(N,k)
B = tril(ones(N));
non_zero_size = nchoosek(N+1,2);
v = zeros(non_zero_size ,1);
for i = 1:nchoosek(N+1,2)
    v(i) = gen(k);
    gen(k)=(2 * (unidrnd(2) -1) -1) * unidrnd(k);
end
B(~~B) = v;
for j = 1:floor(N/2)
    B(j, j+1: N-j) = B(j+1: N-j, j);
    B(j+1: N-j, j) = zeros(N- 2* j, 1);
    B(j, N-j+1) = gen(k);
end
```

2.2. Time Complexity and Stability of WH Factorization

Recall that $k = 1, 2, ..., \lfloor \frac{n-1}{2} \rfloor$ and that there are $\lfloor \frac{n-1}{2} \rfloor$ stages in the factorization. From every successful loops $i, j = k+1, k+2, ..., n-k$ for each stage, there are $(n-2k)$ of 2×2 linear systems to be solved in $(n-2k)$ times which account for the elements in W-matrix and Z-matrix. Next, there are two real addition (2 RA) and two real multiplication (2 RM) require to compute $h_{i,j}^{(k)}$ in $(n-2k)$ four times at every stage of the factorization. Thus, the complexity of the total number of arithmetic operations $T(n)$, used is

$$T(n) = \sum_{k=1}^{\lfloor \frac{n-1}{2} \rfloor} 3(n-2k) + \sum_{k=1}^{\lfloor \frac{n-1}{2} \rfloor} (n-2k) \sum_{i=k+1}^{n-k} 8 + \sum_{k=1}^{\lfloor \frac{n-1}{2} \rfloor} (n-2k) \sum_{i=k+1}^{n-k} \frac{1}{2} \sum_{j=k+1}^{n-k} 4. \quad (2.8)$$

By further simplifying Equation (2.8), we have

$$T(n) = \sum_{k=1}^{\lfloor \frac{n-1}{2} \rfloor} (n-2k) \left[3 + \sum_{i=k+1}^{n-k} 2 \left(4 + \frac{1}{2} \sum_{j=k+1}^{n-k} 4 \right) \right].$$

Thus,

$$\begin{aligned} T(n) &= \frac{8n^3 - 14n - 36}{12} \\ &= \frac{4n^3 - 7n - 18}{6} \\ &\approx \frac{2}{3}(n^3). \end{aligned}$$

The beauty of WH factorization is that it works for nonsingular matrix that is either well-conditioned (such as Teoplitz matrix) or ill-conditioned (such as Hilbert matrix). If the matrix is nonsingular, to avoid breakdown at its submatrices there must be row-interchange in the factorization process. This row-interchange is carried over in exactly the same way at every stage of the factorization to ensure that the 2×2 submatrix has the least condition number adopting any matrix norm. Although, swapping or row-interchange at every stage in WH factorization increases the overall time of the algorithm. This happens as the time required for the algorithm to moved and sort data in and out of the processor also increases. However, applying row-interchange is crucial, when necessary, for the factorization to work thereby making it stable. Since WH factorization must fulfil the requirements of WZ factorizations in order to be applied and that stability of WZ factorization based on Factorization Theorem depends on invertible submatrices, see Theorem 2.5.

Furthermore, the numerical accuracy $\left(-log_{10}\frac{\|B-WH\|}{n\cdot\|B\|}\right)$ of WH factorization depends on the matrix size but more on the matrix norms. The matrix norm of WH factorization is the Frobenius norm. The Frobenius norm of WH factorization is given as

$$\|B-WH\|_F = \sqrt{\left(\sum_{i=1}^{n}\sum_{j=1}^{n}|b_{i,j}-w_{i,j}h_{i,j}|\right)}. \tag{2.9}$$

2.3. Comparison between H-Matrix and Z-Matrix

Although, H-matrix and Z-matrix (especially when factorized from Hankel and Toeplitz matrix) share most things in common yet Z-matrix does not always imply H-matrix since Z-matrix is more general than H-matrix [29]. The WZ factorization is possible provided the submatrices of the nonsingular matrix are invertible, while WH factorization is possible provided the submatrices of the nonsingular matrix are invertible as well as all the elements in the first row and in the last row of its submatrix are nonzero. Assuming the entries $h_{i,j}$ is analogous to $z_{i,j}$, then Z-matrix will imply H-matrix provided that the computed $z_{i,j}^{(k-1)}$ and $z_{n,j}^{(k-1)}$ are strictly nonzero, for $k = 1, 2, ..., \lfloor\frac{n-1}{2}\rfloor$. However, the entries of Z-matrix are unbound to be nonzero. Therefore, quadrant interlocking factorization of symmetric positive definite or diagonally dominant does not guarantee that the factored matrix is H-matrix, however it often guarantees that it is Z-matrix. Then it is obvious that it will no longer be H-matrix if one of its strictly nonzero elements is replaced with zero. The WZ factorization exists for every nonsingular matrix often with pivoting whereas WH factorization may fail to exist even if the matrix is nonsingular. Unlike the factorization of Z-matrix, the factorization of an H-matrix may not necessarily be from a symmetric positive definite or diagonally dominant matrix but definitely not from a tridiagonal matrix. In general, every H-matrix is theoretically a Z-matrix but the converse may not always true, see Figure 2.

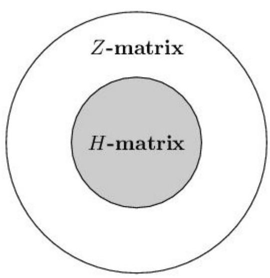

Figure 2. H-matrix as a subset of Z-matrix.

Proposition 2.2. [28] *Let H be an hourglass matrix of order n $(n \geq 3)$, $H_{T(nz)}$ the total number of nonzero entries and $H_{T(z)}$ be the total number of zero entries in hourglass* *matrix. Then,*

$$H_{T(nz)} = \frac{n^2 + 2n - \left|(n+1) \bmod 2 - 1\right|}{2}$$

and

$$H_{T(z)} = \frac{n^2 - 2n + \left|(n+1) \bmod 2 - 1\right|}{2}.$$

Though not always that properties of hourglass matrix and Z-matrix are similar, the entries in H-matrix are linearly independent. Like Z-matrix, the transpose of hourglass matrix does not retain the shape of the matrix but rather form a bowtie matrix or butterfly matrix. Inverse and nth root of hourglass matrix is again hourglass matrix. The minimum order of hourglass matrix is 3 and its rank is n. Regardless of order of hourglass matrix, the total number of zero entries is even. The minimum matrix density of H-matrix is $\lim_{n\to\infty} \frac{\frac{n^2+2n-\left|(n+1) \bmod 2-1\right|}{2}}{n^2} = 0.5.$

2.4. On Hourglass Matrix

Definition 2.3. [28] Filanz submatrix, denoted as $f_m^{1\le i\le\lceil\frac{n-1}{2}\rceil}$, is a 2×2 non-singular matrix obtained by taking the first and the last nonzero elements of the ith and $(n+1-i)$th row of H-matrix given as

$$f_m^{1\le i\le\lceil\frac{n-1}{2}\rceil} = \begin{bmatrix} h_{i,i}^{(i-1)} & h_{i,n+1-i}^{(i-1)} \\ h_{n+1-i,i}^{(i-1)} & h_{n+1-i,n+1-i}^{(i-1)} \end{bmatrix}_{1\le i\le\lceil\frac{n-1}{2}\rceil} \tag{2.10}$$

Every permutation matrix P is a product of elementary row-interchange matrices, it is important to know that at any stage k if suitable row-interchange is applied then

$$H = \left(W^{(k-1)}P^{(k-1)}W^{(k-2)}P^{(k-2)} \ldots W^{(2)}P^{(2)}W^{(1)}P^{(1)}\right) B.$$

The determinant of matrix B can be evaluated as

$$det(B) = det\left(W^{(k-1)} \cdot P^{(k-1)} \cdot \ldots \cdot W^{(2)} \cdot P^{(2)} \cdot W^{(1)} \cdot P^{(1)} \cdot\right)^{-1} H.$$

Due to 1's in the diagonal and 0's in the anti-diagonal of W-matrix, it is easy to deduce that

$$det\left(W^{(k-1)} \cdot \ldots \cdot W^{(2)} \cdot W^{(1)}\right)^{-1} = 1,$$

while

$$det\left(P^{(k-1)} \cdot \ldots \cdot P^{(2)} \cdot P^{(1)}\right)^{-1} = (-1)^{p_n}.$$

Thus,

$$(-1)^{p_n} = \begin{cases} 1 & \text{if even number of rows are interchanged,} \\ -1 & \text{if odd number of rows are interchanged.} \end{cases}$$

Therefore,

$$det(B) = (-1)^{p_n} det\,(H)$$

where p_n is the total number of permutation matrix (successful row interchange) occurs in the factorization.

Proposition 2.4. [28] *Let det(H) be the determinant of hourglass matrix of order* $(n \geq 3)$ *Then,*

$$det\,(H) = \begin{cases} \displaystyle\prod_{i=1}^{\lceil \frac{n-1}{2} \rceil} \begin{vmatrix} h_{i,i}^{(i-1)} & h_{i,n+1-i}^{(i-1)} \\ h_{n+1-i,i}^{(i-1)} & h_{n+1-i,n+1-i}^{(i-1)} \end{vmatrix} & \textit{if } n \textit{ is even;} \\ h_{\left(\frac{n+1}{2},\frac{n+1}{2}\right)} \displaystyle\prod_{i=1}^{\lceil \frac{n-1}{2} \rceil} \begin{vmatrix} h_{i,i}^{(i-1)} & h_{i,n+1-i}^{(i-1)} \\ h_{n+1-i,i}^{(i-1)} & h_{n+1-i,n+1-i}^{(i-1)} \end{vmatrix} & \textit{if } n \textit{ is odd.} \end{cases}$$

In evaluating the determinants of H-matrix, each filanz minor (determinant of filanz submatrix of Equation (2.10)) uses 2 multiplications and 1 subtraction. If n is even, then there are $\frac{3n-2}{2}$ multiplications and $\frac{n}{2}$ subtractions. However. if n is odd, then there are $\frac{3n-3}{2}$ multiplications and $\frac{n-1}{2}$ subtractions to have $T(n) \approx 2n$.

Theorem 2.5. (Factorization Theorem [30]). *Let $B \in R^{n\times n}$ be a nonsingular matrix that has a unique QIF factorization, then $B = WZ$ if and only if the submatrices of B are invertible.*

Theorem 2.6. *If there exists WH factorization for a nonsingular matrix B, then there exists WZ factorization.*

Proof. First, we assume matrix B has even order (the assumption is also true for odd order). If $B = WH$, then the central submatrices $\nabla_h = h_{i,j}^{(k-1)}$ of B are nonsingular according to its factorization algorithm otherwise the factorization fails. That is,

$$\nabla_h = \begin{bmatrix} h_{k,k}^{(k-1)} & \cdots & h_{k,n-k+1}^{(k-1)} \\ \vdots & & \vdots \\ h_{n-k+1,k}^{(k-1)} & \cdots & h_{n-k+1,n-k+1}^{(k-1)} \end{bmatrix}_{1\leq k\leq \frac{n}{2}}.$$

This assumption is also applicable to $B = WZ$ according to Theorem 2.5, if and only if its centro-nonsingular submatrix $\Delta z = z_{i,j}^{(k-1)}$ are invertible, such that

$$\Delta z = \begin{bmatrix} z_{k,k}^{(k-1)} & \cdots & z_{k,n-k+1}^{(k-1)} \\ \vdots & & \vdots \\ z_{n-k+1,k}^{(k-1)} & \cdots & z_{n-k+1,n-k+1}^{(k-1)} \end{bmatrix}_{1\leq k\leq \frac{n}{2}}.$$

If a nonsingular matrix B with centro-nonsingular submatrix assumes WH factorization such that $det\,(\nabla_h) = h_{n-k+1,n-k+1}^{(k-1)} h_{k,k}^{(k-1)} - h_{n-k+1,k}^{(k-1)} h_{k,n-k+1}^{(k-1)} \neq 0$, then the matrix also assumes WZ factorization such that $det\,(\Delta z) = z_{n-k+1,n-k+1}^{(k-1)} z_{k,k}^{(k-1)} - z_{n-k+1,k}^{(k-1)} z_{k,n-k+1}^{(k-1)} \neq 0$. However, the computed entry $z_{i,j}^{(k-1)}$ may or may not be nonzero for $i, j = k, k+1, ..., n-k+1$. This is because WZ factorization only requires invertibility of Δz, whereas WH factorization ensures that row interchange exists for ∇_h to contain only nonzero entries and still being invertible. In a case where $z_{i,j}^{(k-1)} \neq 0$ then $z_{i,j}^{(k-1)} = h_{i,j}^{(k-1)}$, but if an entry in $z_{i,j}^{(k-1)}$ is zero then $z_{i,j}^{(k-1)} \neq h_{i,j}^{(k-1)}$, since $h_{i,j}^{(k-1)}$ cannot be zero, even though $det\,(\Delta z) \neq 0$ and $det\,(\nabla_h) \neq 0$. ∎

Next, we investigate the performance time and matrix norms of LU factorization with row pivoting against WH factorization on nonsingular dense matrices via MATLAB R2017b and the results were recorded in Table 1. Due to the lack of parallel computer or mesh multiprocessors with high multicores, we limit our MATLAB codes on Intel processor (Core i7-4600U 2.1GHz, 8GB RAM) and AMD processor (Ryzen 5 1500X 2.1GHz, 8GB RAM) with standard hardware.

TABLE 1. Performance time and matrix norm of LU and WH factorization on Intel and AMD processor via MATLAB R2017b.

Matrix name	Intel				AMD			
	Performance time		Matrix norm		Performance time		Matrix norm	
	LU	WH	$\|B-LU\|$	$\|B-WH\|$	LU	WH	$\|B-LU\|$	$\|B-WH\|$
500 × 500	7.06	1.08	1.54E-14	0.18E-14	10.53	3.65	1.96E-14	0.24E-14
1000 × 1000	18.93	9.82	3.24E-14	2.49E-14	27.35	19.45	3.62E-14	2.53E-14
1500 × 1500	42.91	23.22	5.33E-14	4.91E-14	65.11	43.02	5.82E-14	4.92E-14
2000 × 2000	124.10	99.52	8.37E-14	7.30E-14	203.13	169.18	8.91E-14	7.35E-14
2500 × 2500	248.93	203.74	1.18E-13	0.90E-13	449.24	343.01	1.57E-13	0.98E-13
3000 × 3000	448.34	399.75	3.21E-13	2.42E-13	681.97	581.73	4.22E-13	2.57E-13
3500 × 3500	635.90	569.32	5.67E-13	4.76E-13	1035.81	936.16	6.31E-13	4.86E-13
4000 × 4000	961.67	781.63	7.89E-13	6.71E-13	1645.35	1284.07	8.49E-13	6.85E-13
4500 × 4500	1441.01	1229.63	1.28E-12	0.94E-12	2368.34	1788.69	1.63E-12	0.97E-12
5000 × 5000	2110.47	1823.76	1.39E-12	0.96E-12	3461.00	2753.27	1.89E-12	0.99E-12
5500 × 5500	2628.34	2264.21	2.86E-12	2.36E-12	4290.46	3457.11	3.26E-12	2.46E-12
6000 × 6000	3289.31	2923.41	3.43E-12	2.73E-12	5235.85	4637.36	3.98E-12	2.81E-12
6500 × 6500	4002.68	3575.92	5.51E-12	4.80E-12	6584.34	6054.05	6.12E-12	4.93E-12
7000 × 7000	4991.64	4514.47	6.52E-12	5.26E-12	8734.14	7239.35	7.08E-12	5.39E-12
7500 × 7500	6194.87	5703.16	8.31E-12	6.76E-12	10634.65	9616.25	9.24E-12	6.96E-12
8000 × 8000	8279.02	6901.07	9.29E-12	7.87E-12	12683.64	12007.31	9.80E-12	8.01E-12

In Figure 3, the performance time of WH factorization on Intel processor is about 23% better than the performance time of WH factorization on AMD processor but the performance time of LU factorization on AMD processor is 19% better than the performance time of WH factorization on Intel processor. However, the performance time of LU and WH factorization increase as the dimension of the matrix increases irrespective of the processor used. From our results in Figure 4, we deduced that norms of WH factorization is better than LU factorization on Intel and AMD processor. More so, the matrix norms of LU and WH factorization increase as the size of their matrices increase.

3. Conclusion

The notion and context of hourglass matrix and its WH factorization algorithm have been successively discussed. Like WZ factorization, the performance time and matrix norms on Intel and AMD processor of WH factorization are better than LU factorization. In all, Intel processor gives better result on WH factorization and LU factorization than on AMD processor. Besides, the comparison made between H-matrix and Z-matrix concludes that the existence of WH factorization implies WZ factorization.

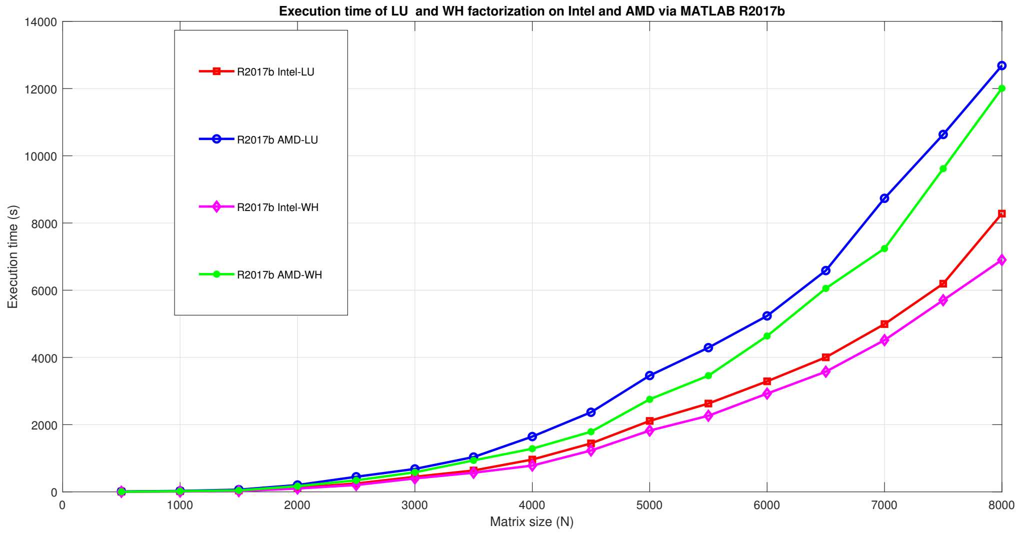

Figure 3. Performance time of LU and WH factorization on MATLAB R2017b.

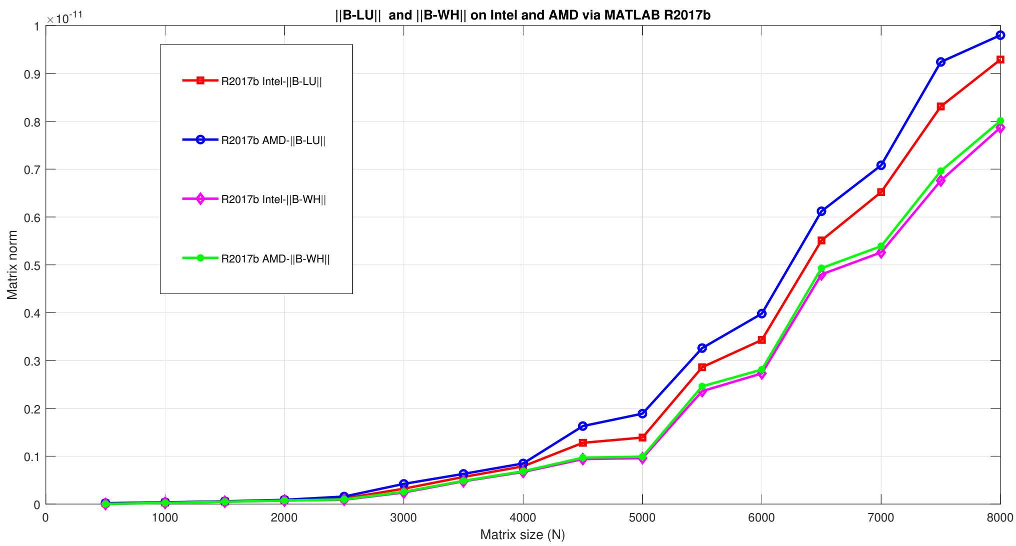

Figure 4. Matrix Norms of LU and WH factorization on MATLAB R2017b.

Acknowledgements

This research is funded by RU (Research University) Grant, Universiti Sains Malaysia, Grant number 1001/PMATHS/811337.

References

[1] C. Demeure, Bowtie factors of toeplitz matrices by means of split algorithms, IEEE Transactions on Acoustics, Speech, and Signal Processing 37 (1989) 1601–1603.

[2] G. Heinig, K. Rost, Schur-type algorithms for the solution of Hermitian Toeplitz systems via factorization, Operator Theory: Advances and Applications, Vol. 160, Birkhäuser Basel (2005), 233–252.

[3] B. Bylina, J. Bylina, Strategies of parallelizing nested loops on the multicore architectures on the example of the wz factorization for the dense matrices, Annals of Computer Science and Information Systems 5 (2015) 629–639.

[4] D. Evans, M. Hatzopoulos, The parallel calculation of the eigenvalues of a real matrix A, Computers and Mathematics with Applications 4 (1978) 211–218.

[5] A.R. Rhofi, M. Ameur, Double power method iteration for parallel eigenvalue problem, Int. J. Pure Appl. Math. 108 (2016) 945–955.

[6] J. Bylina, B. Bylina, Incomplete WZ factorization as an alternative method of preconditioning for solving Markov chains, In International Conference on Parallel Processing and Applied Mathematics (2007) 99–107.

[7] J. Bunch, R. Hopcroft, E. John, Triangular factorization and inversion by fast matrix multiplication, Mathematics of Computation 28 (1974) 231–236.

[8] S. Chinchole, A. Bhadane, Lu factorization method to solve linear programming problem, International Journal of Emerging Technology and advanced Engineering 4 (2014) 176–180.

[9] A. Townsendfar, L. Trefethen, Continuous analogues of matrix factorizations, Proceedings of the Royal Society A 471 (2015) 585–606.

[10] X. Wang, P. Jones, J. Zambreno, A configurable architecture for sparse LU decomposition on matrices with arbitrary patterns, ACM SIGARCH Computer Architecture News 43 (2016) 76–81.

[11] E. Golpar-Raboky, N. Mahdavi-Amiri, Wz factorization via abaffy-broyden-spedicato algorithms, Bulletin of the Iranian Mathematical Society 40 (2014) 399–411.

[12] E. Golpar-Raboky, Abs algorithms for integer wz factorization, Malaysian Journal of Mathematical Sciences 8 (2014) 69–85.

[13] J. Bylina, B. Bylina, Parallelizing nested loops on the intel xeon phi on the example of the dense WZ factorization, In 2016 Federated Conference on Computer Science and Information Systems (FedCSIS) (2016) 655–664.

[14] J. Shanehchi, D. Evans, Further analysis of the quadrant interlocking factorisation (qif) method, International Journal of Computer Mathematics 11 (1982) 49–72.

[15] J. Chudik, G. David, V. Kotov, N. Mirenkov, J. Ondas, I. Plander, V. Valkovskii, Algorithms, Software and Hardware of Parallel Computers, Springer Science and Business Media, Berlin, 2013.

[16] R. Asenjo, M. Ujaldon, E. Zapata, Parallel WZ factorization on mesh multiprocessors, Microprocessing and Microprogramming 38 (1993) 319–326.

[17] G. Heinig, K. Rost, Fast algorithms for toeplitz and hankel matrices, Linear Algebra and Its Applications 435 (2011) 1–59.

[18] D. Evans, M. Hatzopoulos, A parallel linear system solver, International Journal of Computer Mathematics 7 (1979) 227–238.

[19] P. Huang, A. MacKay, D. Teng, A hardware/software codesign of WZ factorization to improve time to solve matrices, In Canadian Conference on Electrical and Computer Engineering (2010) 1–5.

[20] M. Brunetti, A. Renato, Old and new proofs of Cramer's rule, Applied Mathematical Sciences 8 (2014) 6689–6697.

[21] O. Babarinsa, H. Kamarulhaili, Modification of Cramer's rule, Journal of Fundamental and Applied sciences 9 (2017) 556–567.

[22] N. Higham, Accuracy and Stability of Numerical Algorithms, SIAM, New York, 2002.

[23] D. Levin, D. Evans, The inversion of matrices by the double-bordering algorithm on MIMD computers, Parallel Computing 17 (1991) 591–602.

[24] J. Bylina, B. Bylina, The WZ factorization in MATLAB, In 2014 Federated Conference on Computer Science and Information Systems (FedCSIS) (2014) 561–568.

[25] C.R. McGann, P. Arduino, P. Mackenzie-Helnwein, Stabilized single-point 4-node quadrilateral element for dynamic analysis of fluid saturated porous media, Acta Geotechnica 7 (2012) 297–311.

[26] L. Li, Y. Peng, D. Li, A stabilized underintegrated enhanced assumed strain solid-shell element for geometrically nonlinear plate/shell analysis, Finite Elements in Analysis and Design 47 (2011) 511–518.

[27] M. Warburton, S. Maddock, Physically based forehead animation including wrinkles, Computer Animation and Virtual Worlds 26 (2015) 55–68.

[28] O. Babarinsa, H. Kamarulhaili, Quadrant interlocking factorization of hourglass matrixn, In AIP Conference Proceedings of the 25th National Symposium on Mathematical Sciences (2018) 1–9.

[29] K.H. Han, S.-H. Kye, Construction of multi-qubit optimal genuine entanglement witnesses, Journal of Physics A: Mathematical and Theoretical 49 (2016) 303–312.

[30] S.C. Rao, Existence and uniqueness of WZ factorization, Parallel Computing 23 (1997) 1129–1139.