ISSN (Print): 0974-6846
Indian Journal of Science and Technology, *Vol 11(45), DOI: 10.17485/ijst/2018/v11i45/138543, December 2018* ISSN (Online): 0974-5645

Application of Modified Cramer's Rule in Quadrant Interlocking Factorization

Olayiwola Babarinsa[1*] and Hailiza Kamarulhaili[2]

[1]Department of Mathematical Sciences, Federal University Lokoja, 1154 Lokoja, Kogi State, Nigeria; babs3in1@gmail.com
[2]School of Mathematical Sciences, Universiti Sains Malaysia, 11800 Pulau Pinang, Penang, Malaysia; hailiza@usm.my

Abstract

Objectives: To show that modified Cramer's rule is better than classical Cramer's rule for solving linear systems in quadrant interlocking factorization or WZ factorization. **Methods:** The relative residual measurement of modified Cramer's rule was compared with classical Cramer's rule. Furthermore, we apply the rules in WZ factorization and evaluate their matrix norm on AMD and Intel processor. **Findings:** This study shows that the residual measurements of modified Cramer's rule are 20% better than Cramer's rule. It also shows that the matrix norm of Cramer's rule in WZ factorization is higher than using modified Cramer's rule in the factorization. **Application/improvements:** Modified Cramer's rule can be used to solve simple linear system. Applying the modified Cramer's rule in *WZ* factorization using parallel computer or shared memory *multiprocessor* networks such as Intel Xeon Phi, Sunway Taihulight or OLCF-4 should be strongly considered.

Keywords: Cramer's Rule, Matrix Norm, Quadrant Interlocking Factorization, *WZ* Factorization, *Z*- matrix

1. Introduction

Quadrant interlocking factorization (QIF) or *WZ* factorization produces a *W*-matrix together with a *Z-matrix* (or hourglass matrix) from nonsingular matrix A[1,2] such that

$$A = WZ \qquad 1$$

The necessary condition for nonsingular matrix $A=[a_{i,j}]_{i,j=1}^{n}$ to be a WZ factorization is that the central submatrices $A_{n-m+1}^{c}=[a_{i,j}]_{i,j=1}^{n-m+1}$ are centro-nonsingular[3]. Where n is even order of matrix and *c* the centered submatrix of *A*, for m = 1,2,…, $\left[\frac{n}{2}\right]$. A matrix which is either a *Z*-matrix or a *W* -matrix is called butterfly matrix. These names are suggested by the shapes of the set of all possible positions for nonzero entries, which are as follows

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

To obtain the entries of *Z*-matrix by updating matrix *A*, the column of *W*-matrix must be computed from 2×2 systems of linear equations in equation 2 using *Cramer's rule*

$$\begin{cases} z_{(m,m)}^{(m-1)} w_{(i,m)}^{m} + z_{(n-m+1,m)}^{(m-1)} w_{(i,n-m+1)}^{m} = -z_{(i,m)}^{(m-1)} \\ z_{(m,n-m+1)}^{(m-1)} w_{(i,m)}^{m} + z_{(n-m+1,n-m+1)}^{(m-1)} w_{(i,n-m+1)}^{m} = -z_{(i,n-m+1)}^{(m-1)} \end{cases} \qquad (2)$$

to obtain equation (3) as

$$z_{i,j}^{m} = z_{i,j}^{m-1} + w_{i,m}^{m} z_{m,j}^{m-1} + w_{i,n-m+1}^{m} z_{n-m+1,j}^{m-1} \qquad (3)$$

*Author for correspondence

For $i, j = m + 1, \ldots, n - m$

For vivid understanding of *WZ* factorization, readers may see[4–11] and the references therein. One of the advantages of Cramer's rule is to determine if a system of linear equations is inconsistent or indeterminate because Cramer's rule gives a clear representation of an individual component unconnected to all other components[12,13]. This specific advantage of Cramer's rule is useful in *WZ* factorization to solve for the 2×2 linear systems which is the major factor to know if the matrix is centro-nonsingular. The *matrix norm* of *WZ* factorization is the Frobenius norm of the matrix given as[14]

$$||A||_F = ||A\text{-}WZ||$$

The Frobenius norm or the Hilbert-Schmidt norm is equivalent to the Euclidean norm on $K^{n \times n}$ [15].

In Section 2, we apply the two modified methods of Cramer's rule proposed by[16] in *WZ* factorization. We further evaluate their matrix norm using MATLAB on two distinct physical processors (AMD and Intel processor). The results obtained from the matrix norm can be improved if parallel computer or mesh multiprocessors are used.

2. Applying Modified Cramer's Rule in QIF or WZ factorization

2.1 Modified Cramer's Rule

Irrespective of its high computational time, Cramer's rule is hypothetically significance for solving systems of linear equations because accurate method to evaluate determinants can make Cramer's rule numerically stable[17–19]. Thus, much deductions have been made on Cramer's rule to solve simple and large-scale systems of linear equations, see for example[20–22].

Theorem 2.1.1 [Cramer's rule]. Let $Ax = c$ be an $n \times n$ system of linear equation and A an $n \times n$ matrix such that $|A| \neq 0$, then the unique solution $x = (x_1, x_2, \ldots, x_n)^T$ to the system of linear equations is given by

$$x_i = \frac{|A_{i|c}|}{|A|} \tag{4}$$

where, $A_{(i|c)}$ is the matrix obtained from A by substituting the column vector c to the ith column of A, for $i = 1, 2, \ldots, n$

In[16], the following corollaries were obtained from Theorem 2.1.1 which are based on one of the properties of determinant.

Corollary 2.1.1 Let $Ax = c$ be an $n \times n$ system of linear equation and A an $n \times n$ matrix such that $|A| \neq 0$, then the unique solution $x = (x_1, x_2, \ldots, x_n)^T$ to the system of linear equations is given by

$$x_i = \frac{|A_{i+c}|}{|A|} - 1 \tag{5}$$

where, $A_{(i+c)}$ is the matrix obtained from A by adding the column vector c to the ith column of A, for $i = 1, 2, \ldots, n$

Corollary 2.1.2. Let $Ax = c$ be an $n \times n$ system of linear equation and A an $n \times n$ matrix such that $|A| \neq 0$, then the unique solution $x = (x_1, x_2, \ldots, x_n)^T$ to the system of linear equations is given by

$$x_i = 1 - \frac{|A_{i-c}|}{|A|} \tag{6}$$

where, $A_{(i-c)}$ is the matrix obtained from A by subtracting the column vector c from the ith column of A, for $i = 1, 2, \ldots, n$.

The authors proved that Corollary 2.1.1 and Corollary 2.1.2 are equal to classical Cramer's rule. Now, we further the findings by providing the algorithm for the above corollaries in Figure 1.

Even though the modified methods of Cramer's rule use extra n arithmetic operations for every solution of the linear systems, their computational time is like Cramer's rule. Therefore, the modified Cramer's rule may nowhere better than other direct methods (such as GE or LU decomposition) in terms of computational time complexity or numerical stability for higher linear systems. Though, our main objective is to apply the modified Cramer's rule in QIF but the methods can be compared with Cramer's rule in terms of their relative residual error, $\frac{||A\bar{x} - c||}{||A||||\bar{x}||}$, which is given in Table 1.

Figure 2 shows that the relative residual of Method I and Method II are almost similar and are better than residual of Cramer's rule. Among the three algorithms, Method II shows to be better than Cramer's rule and Method I. Our algorithms, for Method I and Method II, are about 20% better than Cramer's rule.

2.2 Application of Modified Cramer's Rule

Due to the lack of parallel computer, the MATLAB codes application of this paper are limited to AMD and Intel processor with standard hardware in Table 2.

Algorithm 1 Modified Cramer's rule algorithm

```
procedure MODIFIED CRAMER'S RULE
    A ← n × n coefficient matrix
    c ← column vector
    x_i ← solutions of linear system
    for i do1n
        K ← the ith column of A ± c
        det(A) ≠ 0 ← determinant of A
        det(K) ← determinant of K
        if K = S + c then
            x_i ← 1 − (det(K)/det(A))
        else
            x_i ← (det(K)/det(A)) − 1
        end if
    end for
    return x_i
end procedure
```

Figure 1. Algorithm of modified Cramer's rule.

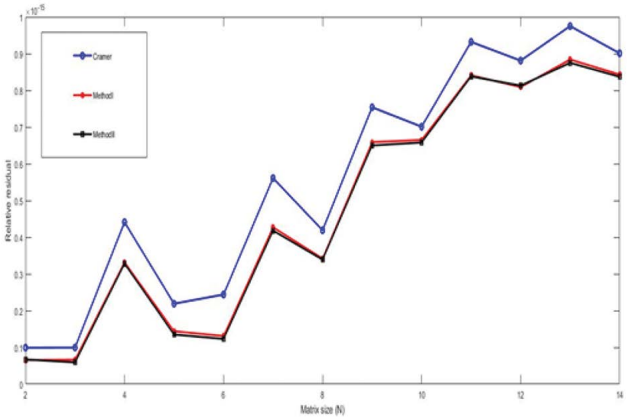

Figure 2. Residual measurement of Cramer's rule, Method I and Method II.

To apply the above corollaries in *WZ* factorization algorithm, we reconstruct equation (2) as

$$\underbrace{\begin{bmatrix} z_{m,m}^{m-1} & z_{n-m+1,m}^{m-1} \\ z_{m,n-m+1}^{m-1} & z_{n-m+1,n-m+1}^{m-1} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} w_{i,m}^{m} \\ w_{i,n-m+1}^{m} \end{bmatrix}}_{x} = \underbrace{\begin{matrix} -z_{i,m}^{m-1} \\ -z_{i,n-m+1}^{m-1} \end{matrix}}_{c} \tag{7}$$

Now, we apply Corollary 2.1.1 to compute $w_{i,m}^{m}$ and $w_{i,n-m+1}^{m}$ from equation (7) to obtain

$$w_{i,m}^{m} = \frac{\left|\left[w_{i,m}^{m}\right]_{i+c}\right|}{|z|} - 1 \text{ and } w_{i,n-m+1}^{m} = \frac{\left|\left[w_{i,n-m+1}^{m}\right]_{i+c}\right|}{|z|} - 1 \tag{8}$$

where,

$$|z| = -z_{n-m+1,m}^{m-1} z_{m,n-m+1}^{m-1} + z_{n-m+1,n-m+1}^{m-1} z_{m,m}^{m-1} \neq 0$$

$$\left|\left[w_{i,m}^{m}\right]_{i+c}\right| = z_{n-m+1,m}^{m-1} z_{i,n-m+1}^{m-1} - z_{n-m+1,n-m+1}^{m-1} z_{i,m}^{m-1}$$

Table 1. Relative residual measurements of Cramer's rule, Method I and Method II

Matrix size (N)	*Cramer*	*Method*	*Method*
2×2	9.93E-17	6.53E-17	6.71E-17
3×3	9.97E-17	6.65E-17	5.93E-17
4×4	4.41E-16	3.32E-16	3.28E-16
5×5	2.19E-16	1.44E-16	1.35E-16
6×6	2.44E-16	1.31E-16	1.23E-16
7×7	5.61E-16	4.27E-16	4.18E-16
8×8	4.19E-16	3.42E-16	3.39E-16
9×9	7.54E-16	6.59E-16	6.50E-16
10×10	7.01E-16	6.65E-16	6.58E-16
11×11	9.32E-16	8.41E-16	8.38E-16
12×12	8.81E-16	8.09E-16	8.13E-16
13×13	9.75E-16	8.84E-16	8.75E-16
14×14	9.01E-16	8.43E-16	8.37E-16

Table 2. Hardware specifications

CPU	Memory
AMD APU Core A10-9600P 2.4GHz	8 GB
Intel Core i7-4600U 2.1GHz	8 GB

$$-z_{n-m+1,m}^{m-1} z_{m,n-m+1}^{m-1} + z_{n-m+1,n-m+1}^{m-1} z_{m,m}^{m-1}$$

$$\left|\left[w_{i,n-m+1}^{m}\right]_{i+c}\right| = z_{m,n-m+1}^{m-1} z_{i,m}^{m-1} - z_{i,n-m+1}^{m-1} z_{m,m}^{m-1}$$

$$-z_{n-m+1,m}^{m-1} z_{m,n-m+1}^{m-1} + z_{n-m+1,n-m+1}^{m-1} z_{m,m}^{m-1}$$

The factorization obtained from using Corollary 2.1.1 is called W^1Z^1 factorization and its MATLAB code is given in Figure 3.

More so, if we apply Corollary 2.1.2 to compute $w_{i,m}^{m}$ and $w_{i,n-m+1}^{m}$ from equation (7), we will have

$$w_{i,m}^{m} = 1 - \frac{\left|\left[w_{i,m}^{m}\right]_{i-c}\right|}{|z|} \text{ and } w_{i,n-m+1}^{m} = 1 - \frac{\left|\left[w_{i,n-m+1}^{m}\right]_{i+c}\right|}{|z|} \tag{9}$$

where,

$$|z| = -z_{n-m+1,m}^{m-1} z_{m,n-m+1}^{m-1} + z_{n-m+1,n-m+1}^{m-1} z_{m,m}^{m-1} \neq 0$$

```
function Z = W1Z1factorization
% step of elimination - from A to Z
A=input('matrix A =');
n = size(A, 1);
W = zeros(n);
for k = 1:ceil((n-1)/2)
    k2 = n - k + 1 ;
    det = A(k,k) * A(k2,k2) - A(k2,k) * A(k,k2);
    if det == 0
        exitflag = 0;
        for i1 = k:k2
            for i2 = i1:k2
                det = A(i1,k) * A(i2,k2) - A(i2,k) * A(i1,k2);
                if det ~= 0
                    tmp = A(i1,k:k2);
                    A(i1,k:k2) = A(k,k:k2);
                    A(k,k:k2) = tmp;
                    tmp = A(i2,k:k2);
                    A(i2,k:k2) = A(k2,k:k2);
                    A(k2,k:k2) = tmp;
                    exitflag = 1;
                    break
                end
            end
        end
        if exitflag == 0
            Z = A;
            return
        end
    end
    % finding elements of W
    W(k+1:k2-1,k)=((A(k2,k)*A(k+1:k2-1,k2)-A(k2,k2)*A(k+1:k2-1,k)-A(k2,k)*A(
        k,k2)+A(k2,k2)*A(k,k))/det)-1;
    W(k+1:k2-1,k2)=((A(k,k2)*A(k+1:k2-1,k)-A(k+1:k2-1,k2)*A(k,k)-A(k2,k)*A(k
        ,k2)+A(k2,k2)*A(k,k))/det)-1;
    % updating A
    A(k+1:k2-1,k+1:k2-1) = A(k+1:k2-1,k+1:k2-1) + W(k+1:k2-1,k) * A(k,k+1:k2
        -1) + W(k+1:k2-1,k2) * A(k2,k+1:k2-1);
    Z = A;
end
```

Figure 3. MATLAB code of W^1Z^1 factorization.

```
%finding elements of W
W(k+1:k2-1,k)=1-((A(k2,k2)*A(k+1:k2-1,k)-A(k2,k)*A(k+1:k2-1,k2)-A(k2,
    k)*A(k,k2)+A(k2,k2)*A(k,k))/det);
W(k+1:k2-1,k2)=1-((A(k,k)*A(k+1:k2-1,k2)-A(k,k2)*A(k+1:k2-1,k)-A(k2,k
    )*A(k,k2)+A(k2,k2)*A(k,k))/det);
```

Figure 4. MATLAB code of W^2Z^2 factorization.

```
%finding elements of W
W(k+1:k2-1,k)=(A(k2,k)*A(k+1:k2-1,k2)-A(k2,k2)*A(k+1:k2-1,k))/det;
W(k+1:k2-1,k2)=(A(k,k2)*A(k+1:k2-1,k)-A(k,k)*A(k+1:k2-1,k2))/det;
```

Figure 5. MATLAB code of WZ factorization.

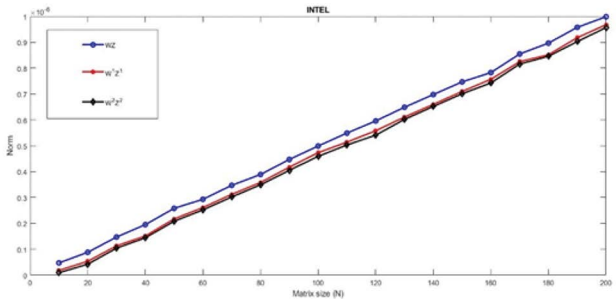

Figure 6. Norm of WZ, W^1Z^1 *and* W^2Z^2 and factorization on Intel processor.

Table 3. Norms of WZ, W^1Z^1 and W^2Z^2 factorization on Intel processor

Matrix size (N)	$\|A- WZ\|$	$\|A-W^2Z^2\|$	$\|A-W^2Z^2\|$
10×10	0.47E-7	0.18E-7	0.09E-7
20×20	0.88E-7	0.53E-7	0.42E-7
30×30	1.47E-7	1.13E-7	1.04E-7
40×40	1.95E-7	1.51E-7	1.44E-7
50×50	2.58E-7	2.17E-7	2.08E-7
60×60	2.93E-7	2.61E-7	2.52E-7
70×70	3.47E-7	3.12E-7	3.02E-7
80×80	3.89E-7	3.58E-7	3.49E-7
90×90	4.47E-7	4.17E-7	4.05E-7
100×100	4.99E-7	4.73E-7	4.59E-7
110×110	5.49E-7	5.14E-7	5.03E-7
120×120	5.96E-7	5.58E-7	5.41E-7
130×130	6.49E-7	6.11E-7	6.02E-7
140×140	6.98E-7	6.60E-7	6.52E-7
150×150	7.47E-7	7.10E-7	7.01E-7
160×160	7.83E-7	7.57E-7	7.43E-7
170×170	8.55E-7	8.25E-7	8.16E-7
180×180	8.97E-7	8.52E-7	8.46E-7
190×190	9.58E-7	9.18E-7	9.03E-7
200×200	9.99E-7	9.68E-7	9.57E-7

$$\left|\left[w^m_{i,m}\right]_{i-c}\right| = z^{m-1}_{n-m+1,n-m+1}z^{m-1}_{i,m} + z^{m-1}_{n-m+1,n-m+1}z^{m-1}_{m,m}$$

$$-z^{m-1}_{n-m+1,m}z^{m-1}_{i,n-m+1} - z^{m-1}_{n-m+1,m}z^{m-1}_{m,n-m+1}$$

$$\left|\left[w^m_{i,n-m+1}\right]_{i-c}\right| = z^{m-1}_{i,n-m+1}z^{m-1}_{m,m} + z^{m-1}_{n-m+1,n-m+1}z^{m-1}_{m,m}$$

$$-z^{m-1}_{m,n-m+1}z^{m-1}_{i,} - z^{m-1}_{n-m+1,m}z^{m-1}_{m,n-m+1}$$

We refer the factorization obtained from using Corollary 2.1.2 as W^2Z^2 factorization and its MATLAB code is given in Figure 4, where we replaced line 32 and line 33 in Figure 3 with line 2 and line 3 of Figure 4.

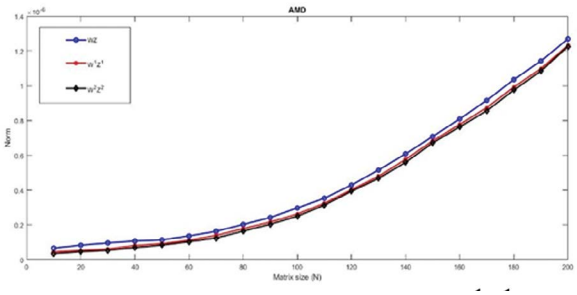

Figure 7. Norm of WZ, W^1Z^1 and W^2Z^2 and factorization on AMD processor.

In all, if we apply Theorem 2.1.1 to compute $w^m_{i,m}$ and $w^m_{i,n-m+1}$ from equation (7) we obtain

$$w^m_{i,m} = \frac{\left|\left[w^m_{i,m}\right]_{i|c}\right|}{|z|} \text{ and } w^m_{i,n-m+1} = \frac{\left|\left[w^m_{i,n-m+1}\right]_{i|c}\right|}{|z|} \qquad (10)$$

where,

$$|z| = -z^{m-1}_{n-m+1,m}z^{m-1}_{m,n-m+1} + z^{m-1}_{n-m+1,n-m+1}z^{m-1}_{m,m} \neq 0$$

$$\left|\left[w^m_{i,m}\right]_{i|c}\right| = z^{m-1}_{n-m+1,m}z^{m-1}_{i,n-m+1} - z^{m-1}_{n-m+1,n-m+1}z^{m-1}_{i,m}$$

$$\left|\left[w^m_{i,n-m+1}\right]_{i+c}\right| = z^{m-1}_{m,n-m+1}z^{m-1}_{i,m} - z^{m-1}_{i,n-m+1}z^{m-1}_{m,m}$$

We then refer the factorization obtained from using Theorem 2.1.1 as WZ factorization and its MATLAB code is given in Figure 5, where we replaced line 32 and line 33 in Figure 3 with line 2 and line 3 of Figure 5.

Next, we compute the Frobenius norm of WZ, W^1Z^1 and W^2Z^2 factorization on AMD and Intel processor and the results are recorded in Table 3 and Table 4 respectively.

In Figures 6 and in 7, the norm of W^1Z^1 and W^2Z^2 factorization are relatively close, but they are better than the norm of WZ factorization on both Intel and AMD processor. Due to round off error in Cramer's rule, the norm of WZ factorization is higher. We can deduce that on Intel processor WZ factorization execution has better sequential algorithms than AMD processor. Applying the modified Cramer's rule in WZ factorization using parallel computer or mesh multiprocessors with better FLOPS on shared memory such as Intel Xeon Phi or OLCF-4 is passionately advocated.

Table 4. Norms of WZ, W^1Z^1 and W^2Z^2 factorization on AMD processor

Matrix size (N)	$\|\|A-WZ\|\|$	$\|\|A-W^1Z^1\|\|$	$\|\|A-W^2Z^2\|\|$
10×10	0.66E-7	0.44E-7	0.34E-7
20×20	0.83E-7	0.53E-7	0.45E-7
30×30	0.96E-7	0.61E-7	0.53E-7
40×40	1.07E-7	0.81E-7	0.69E-7
50×50	1.13E-7	0.92E-7	0.83E-7
60×60	1.36E-7	1.11E-7	1.03E-7
70×70	1.63E-7	1.39E-7	1.24E-7
80×80	2.02E-7	1.78E-7	1.64E-7
90×90	2.42E-7	2.16E-7	2.03E-7
100×100	2.97E-7	2.62E-7	2.49E-7
110×110	3.53E-7	3.24E-7	3.13E-7
120×120	4.29E-7	4.02E-7	3.93E-7
130×130	5.15E-7	4.79E-7	4.67E-7
140×140	6.08E-7	5.76E-7	5.59E-7
150×150	7.08E-7	6.83E-7	6.71E-7
160×160	8.09E-7	7.77E-7	7.63E-7
170×170	9.16E-7	8.74E-7	8.56E-7
180×180	10.35E-7	9.92E-7	9.75E-7
190×190	11.42E-7	10.98E-7	10.84E-7
200×200	12.69E-7	12.32E-7	12.25E-7

3. References

1. Evans D, Hatzopoulos M. A parallel linear system solver. International Journal of Computer Mathematics. 1979;7(3):227–38. https://doi.org/10.1080/00207167908803174
2. Babarinsa O, Kamarulhaili H. Quadrant interlocking factorization of hourglass matrix. AIP Conference Proceedings, 2018, Kuantan: AIP Publishing; 2018. https://doi.org/10.1063/1.5041653
3. Heinig G, Rost K. Fast algorithms for Toeplitz and Hankel matrices. Linear Algebra and its Applications. 2011;435(1):1–59. https://doi.org/10.1016/j.laa.2010.12.001
4. Evans DJ, Hadjidimos A. A modification of the quadrant interlocking factorisation parallel method. International Journal of Computer Mathematics. 1980;8(2):149–66. https://doi.org/10.1080/00207168008803201
5. Chawla M, Passi K. A new quadrant interlocking factorization for parallel solution of tridiagonal linear systems. International Journal of Computer Mathematics. 1991;39(1):99–107. https://doi.org/10.1080/00207169108803982
6. Asenjo R, Ujaldon M, Zapata EL. Parallel WZ factorization on mesh multiprocessors. Microprocessing and Microprogramming. 1993;38(1–5):319–26. https://doi.org/10.1016/0165-6074(93)90161-D
7. Yalamov P, Evans DJ. The WZ matrix factorisation method. Parallel Computing. 1995;21(7):1111–20. https://doi.org/10.1016/0167-8191(94)00088-R
8. Rao SCS. Existence and uniqueness of WZ factorization. Parallel Computing. 1997;23(8):1129–39. https://doi.org/10.1016/S0167-8191(97)00042-2
9. Bylina B. Solving linear systems with vectorized WZ factorization. Annales UMCS, Informatica. 2003;1(1):1–9.
10. Golpar-Raboky E. A new approach for computing WZ factorization. Applications and Applied Mathematics. 2012;7(2):571–84.
11. Bylina B, Bylina J. The WZ factorization in MATLAB. IEEE 2014 Federated Conference on Computer Science and Information Systems (FedCSIS); 2014.
12. Poole D. Linear algebra: A modern introduction: Cengage Learning; 2014.
13. Shores TS. Applied linear algebra and matrix analysis: Springer Science & Business Media; 2007. https://doi.org/10.1007/978-0-387-48947-6. PMCid:PMC2117545
14. Bylina B, Bylina J. 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. 2015;5:629–39. https://doi.org/10.15439/2015F354
15. Golub GH, Van Loan CF. Matrix computations. Johns Hopkins studies in the mathematical sciences. Johns Hopkins University Press, Baltimore, MD; 1996.
16. Babarinsa O, Kamarulhaili H. Modification of Cramer's rule. Journal of Fundamental and Applied sciences. 2017;9(5):556–67.
17. Brunetti M, Renato A. Old and New Proofs of Cramer' s Rule History, notations and tools. Applied Mathematical Sciences. 2014;8(133):6689–97. https://doi.org/10.12988/ams.2014.49683
18. Habgood K, Arel I. A condensation-based application of Cramer's rule for solving large-scale linear systems. Journal

of Discrete Algorithms. 2012; 10:98–109. https://doi.org/10.1016/j.jda.2011.06.007

19. Babarinsa O. Variation of Cramer's rule. Journal of the Nigerian Association of Mathematical Physics. 2014;28(2):57–60.
20. Li H, Huang T-Z, Gu T-x, Liu X-P. From Sylvester's determinant identity to Cramer's rule. arXiv preprint arXiv:14071412; 2014:1–15.
21. Kyrchei I. Analogs of Cramer's rule for the minimum norm least squares solutions of some matrix equations. Applied Mathematics and Computation. 2012;218(11):6375–84. https://doi.org/10.1016/j.amc.2011.12.004
22. Song G-J, Dong C-Z. New results on condensed Cramer's rule for the general solution to some restricted quaternion matrix equations. Journal of Applied Mathematics and Computing. 2015;53(1):1–21.