Far East Journal of Mathematical Sciences (FJMS)
Volume 49, Number 1, 2011, Pages 43-55
Published Online: February 25, 2011
This paper is available online at http://pphmj.com/journals/fjms.htm

NEW DIVISION FREE ALGORITHM FOR FINDING THE DETERMINANT

SHARMILA KARIM*, HASLINDA IBRAHIM and ZURNI OMAR

Quantitative Sciences Building
College of Arts and Sciences
Universiti Utara Malaysia
06010 Sintok, Kedah, Malaysia
e-mail: mila@uum.edu.my

Abstract

In this study, we investigate division free method for computing determinant. Then, we improve one of division free methods, that is a cross multiplication method to any matrix $n \times n$. The algorithm was designed based on recursive circular permutation method where the division procedure is not require. To do so, this circular permutation method is derived from the diagonal column indices in cross multiplication method. The analysis indicates that this new division free algorithm is better in time computation compared to other division free algorithms.

1. Introduction

Numerous studies have been done for determinant computation where determinant plays an important role to solve systems of linear equations. These computing methods can be classified into two main categories: the division free method (DFM) and the non-division free method (nDFM). DFM are cross multiplication, cofactor expansion, clow sequence method, and permutation method,

2010 Mathematics Subject Classification: 15A15.

Keywords and phrases: determinant, division free algorithm, permutation.

*Corresponding author

Received November 22, 2010

while nDFM are Gauss elimination, Cholesky decomposition method, and Condensation method.

Between these two computing methods, nDFM is better in term of time computation and uses only $O(n^3)$ operations, but the floating point error still arise in nDFM [3]. For example, Shin [7] has made comparisons of Gauss elimination method with cofactor expansion method, and with permutation approach for determining the symbolic determinant over time complexity. As a result, the permutation approach is the best method after the Gauss elimination method since the latter method encounters the fraction problem. Nevertheless, the fraction problem can be overcome by using DFM.

In this paper, we attempted to construct a new division free algorithm for computing the determinant by applying our new permutation method [6]. Then we compared our algorithm to the cofactor expansion, and permutation method, namely, Langdon [2] algorithm, and Thongchiew [9] algorithm in term of time computation. In our analysis, we assumed that the entries of the matrix are all non-zero and real number.

2. Related Study for Division Free Methods

In this section, each of division free method for finding the determinant will be discussed briefly. Let $A = [a_{ij}]$ represent an arbitrary $n \times n$ matrix. Then the determinant of A is denoted by $|A|$ or $\det(A)$. The arbitrary determinant,

$$\det(A) = | a_{ij} |_n = | C_1 \, C_2 \, C_3 \cdots C_n |$$

represented in column indices.

2.1. Cross multiplication

Pierre Frédéric Sarrus (1853) introduced the cross multiplication method. This method also called as *Sarrus's rule*. The cross multiplication method is easy to use especially when the size of square matrix is small $(n \leq 3)$.

Example 1. Sarrus's rule for 2×2 matrix.

By line drawn with arrows on the two diagonals,

$$\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}$$

The multiplication of the elements on each line gives the products. Then the subtraction of the products from the up-going lines from the products of the down-going lines gives the determinant.

For 3×3 matrix, we need to append the first two columns to the right of the matrix. Then put the lines on all diagonals, the same hold as for 2×2 matrix. See the following example.

Example 2.

$$\det(A) = \left| \begin{array}{ccc|cc} a_{11} & a_{12} & a_{13} & a_{11} & a_{12} \\ a_{21} & a_{22} & a_{23} & a_{21} & a_{22} \\ a_{31} & a_{32} & a_{33} & a_{31} & a_{32} \end{array} \right|$$

$$= [a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32}] - [a_{13}a_{22}a_{31} + a_{11}a_{23}a_{32} + a_{12}a_{21}a_{31}].$$

However, this method can be used when the order of the matrix less than four $(n \leq 3)$.

In spite of Sarrus's work, Hajrijaz [1] introduced three methods to determine the determinant of the third order matrix. For each method, six diagonals will be formed.

Let us consider matrix A:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}.$$

One of his methods, where element a_{13} and element a_{33} will be placed before first row and third row, respectively. Then element a_{11} and element a_{31} will be placed after first row and third row, respectively. See Example 3:

Example 3.

$$\begin{array}{c|ccc|c} a_{13} & a_{11} & a_{12} & a_{13} & a_{11} \\ & a_{21} & a_{22} & a_{23} & \\ a_{33} & a_{31} & a_{32} & a_{33} & a_{31}. \end{array}$$

The process of elements product and its sign for six diagonals are similar to Sarrus's rule for 3×3 matrix.

The determinant of this matrix is equal to determinant in Example 2.

Unfortunately, this new method only works for 3×3 matrix. The other two methods also generated six diagonals and also work for 3×3 matrix.

2.2. Cofactor expansion

The mathematical formulation for the cofactor expansion method is

$$\det(A) = \sum^{n} (-1)^{i+j} a_{ij} M_{ij}, \tag{1}$$

where M_{ij} is the determinant of the $(n-1) \times (n-1)$ matrix obtained from A by omitting the i-th row and j-th column of A, M_{ij} is called the *minor of entry* a_{ij} and $(-1)^{i+j} M_{ij}$ is the *cofactor* of a_{ij}. However, we need to reduce the size of a matrix to 2×2 or 3×3. Furthermore, this method requires computing of $n!$ products and highly dependent to previous cofactor where the cofactors themselves are determinants.

2.3. Clow sequence method

This combinatorial method was introduced by Mahajan and Vinay [3] in extending the definition of permutation to a clow sequence. The clow sequences can be shown in the graph-theoretical model. The mathematical formulation of this method is

$$\det(A) = \sum_{\mathcal{W}:\text{ a clow sequence}} \operatorname{sgn}(\mathcal{W})\, w(\mathcal{W}). \tag{2}$$

A clow sequence is a sequence of clows $\mathcal{W} = C_1, \ldots, C_k$ of clows with a strictly increasing sequence of the heads: $\text{head}(C_1) < \text{head}(C_2) < \cdots < \text{head}(C_k)$.

$$w(\mathcal{W}) = \prod_i w(C_i). \tag{3}$$

Equation (3) means that the weight of the clow sequence $\mathcal{W}$ is the product of the weights of edges in the walk, where the edges are $\langle w_i, w_{i+1} \rangle$. Soltys [8] observed that some clow sequences did not correspond to the cycle covers (permutation in graph theoretical from), and he proposed a new clow sequence.

2.4. Permutation method

The permutation method is a classical method to determine the determinant of square matrix where the summation is over all permutation on n elements which was introduced by Leibniz in 1678 [4],

$$\det(A) = \sum_{\sigma \in S_n} \operatorname{sign}(\sigma) \prod_{i}^{n} a_{i\sigma(i)}. \tag{4}$$

The sign of a permutation is defined in terms of the total number of inversions:

$$\operatorname{sign}(\sigma) = (-1)^{\text{total number of inversions in } \sigma}. \tag{5}$$

This method did not state any specification of permutation patterns or types that should be employed. Thus, Thongchiew [9] proposed a new permutation generation method and applied it for finding the determinant. However, he had not attempt to compare his work to other division free algorithms.

In our study, we would like to generalize the cross multiplication method a.k.a the Sarrus's rule to any matrix size. In doing so, we need to construct a new DFM by employing circular permutation method which followed to Sarrus's rule. In other words, we attempted to extend the cross multiplication method to any size of square matrix using circular permutation method. This, new circular permutation method is generated based on starter sets to begin with [6]. The advantage of this circular permutation method is that $\frac{n!}{2}$ permutations are generated, while the other $\frac{n!}{2}$ permutations can be obtained from the reversing order of the previous permutations. So in next section, we will discuss the derivation of the algorithm based on our circular permutation method for finding the determinant.

3. Algorithm Development for Circular Permutation Method

In this section, we started by providing some definitions that will be needed in the algorithm derivation.

Definition 3.1. Main diagonal of square matrix which generated based on any column array is a diagonal from the left-hand top corner to the right-hand bottom corner of square matrix.

Definition 3.2. Secondary diagonal of square matrix which generated based on any column array is the diagonal from the right-hand top corner to the left-hand bottom corner of square matrix.

Now we explore the relationship between circular permutation and Sarrus's rule for case $n = 3$.

Example 4.

$$\left|\begin{array}{ccc|cc} a_{11} & a_{12} & a_{13} & a_{11} & a_{12} \\ a_{21} & a_{22} & a_{23} & a_{21} & a_{22} \\ a_{31} & a_{32} & a_{33} & a_{31} & a_{32} \end{array}\right|.$$

Looking at the column indices of elements in main diagonal, $[a_{11}, a_{22}, a_{33}]$, we extracted the column indices and we had 123. Next to main diagonal, $[a_{12}, a_{23}, a_{31}]$, the column indices are 231 and next diagonal, $[a_{13}, a_{21}, a_{32}]$, the column indices are 312. Then we do the same process for secondary diagonal and it parallel diagonal. We obtained the following permutations for main diagonal, secondary diagonal and their parallel diagonals column indices as shown below:

$123 \rightarrow 231 \rightarrow 312$ (main diagonal and its parallel diagonal),

$321 \rightarrow 132 \rightarrow 213$ (secondary diagonal and its parallel diagonal).

The circular permutation pattern appeared in main diagonal and its parallel diagonal column indices where all elements are cycled. The similar pattern also appeared in the secondary diagonal and its parallel diagonal column indices. Then we rearranged that result as shown in Table 1:

Table 1. A pairs of main diagonal and secondary diagonal column indices

Main diagonal and its parallel diagonal	Secondary diagonal and its parallel diagonal
123	321
231	132
312	213

As shown in Table 1, the secondary diagonal column indices are a reverse of the main diagonal column indices. It is also apply to other parallel main diagonal to parallel secondary diagonal. From Sarrus's rule, the determinant is

$$[a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32}] - [a_{13}a_{22}a_{31} + a_{11}a_{23}a_{32} + a_{12}a_{21}a_{33}]. \quad (6)$$

Then we rearranged the result in equation (6), with respect to the circular permutation of element column indices in pair of diagonals as shown in Table 1, we have

$$[a_{11}a_{22}a_{33} - a_{13}a_{22}a_{31}] + [a_{12}a_{23}a_{31} - a_{11}a_{23}a_{32}] + [a_{13}a_{21}a_{32} - a_{12}a_{21}a_{33}]. \quad (7)$$

Thus, we extended this concept discussed in Example 4, for any $n \times n$ matrix.

Definition 3.3. The Main Diagonal Product (MDP) is a product of all n entries in the main diagonal of the square matrix:

$$MDP = \operatorname{sign}(\sigma) \prod_{i=1}^{n} a_{i\sigma(i)}. \quad (8)$$

Definition 3.4. The Secondary Diagonal Product (SDP) is a product of all n entries in the secondary diagonal of the square matrix:

$$SDP = \operatorname{sign}(\sigma) \prod_{i=1}^{n} a_{i\sigma(n-i+1)}. \quad (9)$$

The signs of MDP and SDP follow rule in equation (5). Now this notation is extended for circular process. For each k cycle, $k \in \{0, 1, 2, 3, \ldots, n-1\}$, and $j = \left(1, 2, \ldots, \frac{(n-1)!}{2}\right)$, the summation of MDP and its SDP equal to Product Diagonal (PD),

$$\begin{aligned} PD(A_{j,k}) &= (MDP + SDP)(A)_{j,k} \\ &= \operatorname{sign}(\sigma)\left[\prod_{i=1}^{n} a_{(i)(\sigma(i)_k)}\right] + \operatorname{sign}(\sigma)\left[\prod_{i}^{n} a_{(i)(\sigma(n+1-i)_k)}\right]. \end{aligned}$$

When $k = 0$; $PD(A_{j,0})$ is the PD of starter matrix of A_j.

In term of for any n cases, where n is the order of square matrix:

$$\det(A) = \sum_{j=1}^{\frac{(n-1)!}{2}} \left[\sum_{k=0}^{n-1} (PD(A_{j,k}))\right]. \quad (10)$$

The essential characteristic of main diagonal and its secondary diagonal is it sign $(+1)$ and (-1). Thus we set up the property such as follows:

$$\text{sign of } SDP = \begin{cases} \text{sign of } MDP, & \text{if } n \equiv 0 \text{ or } 1 \pmod 4, \\ (-1) \cdot (\text{sign of } MDP), & \text{otherwise.} \end{cases}$$

Now we demonstrated our algorithm for $n = 4$. Since $n = 4 \equiv 0 \pmod 4$, the sign of SDP equal to sign of MDP:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}.$$

Step 1. Determine the starter sets.

Without loss of generality, suppose the starter sets based on column indices are $[1, 2, 3, 4]$, $[1, 3, 4, 2]$, $[1, 4, 3, 2]$.

Step 2. Generate matrix based on the each starter:

Starter: $[1, 2, 3, 4]$

$$A_1 = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}.$$

Starter: $[1, 3, 4, 2]$

$$A_2 = \begin{bmatrix} a_{11} & a_{13} & a_{14} & a_{12} \\ a_{21} & a_{23} & a_{24} & a_{22} \\ a_{31} & a_{33} & a_{34} & a_{32} \\ a_{41} & a_{43} & a_{44} & a_{42} \end{bmatrix}.$$

Starter: $[1, 4, 2, 3]$

$$A_3 = \begin{bmatrix} a_{11} & a_{14} & a_{12} & a_{13} \\ a_{21} & a_{24} & a_{22} & a_{23} \\ a_{31} & a_{34} & a_{32} & a_{33} \\ a_{41} & a_{44} & a_{42} & a_{43} \end{bmatrix}.$$

Step 3. Calculate the sign of diagonal product and sum up the PD of each A_j.

Let $j = 1$. Then

$$A_1 = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}.$$

If the order of matrix, $n \equiv 0$ or $1 \pmod 4$, then the sign of its secondary diagonal equal to its main diagonal sign.

$$PD(A_{1,0}) = (-1)^0 a_{11}a_{22}a_{33}a_{44} + a_{14}a_{23}a_{32}a_{41},$$

$$\text{Total } PD = PD(A_{1,0}).$$

Step 4. Use circular operation on A_1 and total up the $PD(A_{1,1})$. Simultaneously sum the PD:

$$[2, 3, 4, 1]$$

$$A_{1,1} = \begin{bmatrix} a_{12} & a_{13} & a_{14} & a_{11} \\ a_{22} & a_{23} & a_{24} & a_{21} \\ a_{32} & a_{33} & a_{34} & a_{31} \\ a_{42} & a_{43} & a_{44} & a_{41} \end{bmatrix},$$

$$PD(A_{1,1}) = (-1)^3 a_{12}a_{23}a_{34}a_{41} - a_{11}a_{24}a_{33}a_{42},$$

$$\text{Total } PD = PD(A_{1,0}) + PD(A_{1,1}) + PD(A_{1,2})$$

$$\text{count circular} = 1.$$

Step 5. Repeat step 4 until number of circular equal to three:

$$[3, 4, 1, 2]$$

$$A_{1,2} = \begin{bmatrix} a_{13} & a_{14} & a_{11} & a_{12} \\ a_{23} & a_{24} & a_{21} & a_{22} \\ a_{33} & a_{34} & a_{31} & a_{32} \\ a_{43} & a_{44} & a_{41} & a_{42} \end{bmatrix},$$

$$PD(A_{1,2}) = (-1)^4 a_{13}a_{24}a_{31}a_{42} + a_{12}a_{21}a_{34}a_{43},$$

Total $PD = PD(A_{1,0}) + PD(A_{1,1}) + PD(A_{1,2})$

count circular $= 2$

[4, 1, 2, 3]

$$A_{1,3} = \begin{bmatrix} a_{14} & a_{11} & a_{12} & a_{13} \\ a_{24} & a_{21} & a_{22} & a_{23} \\ a_{34} & a_{31} & a_{32} & a_{33} \\ a_{44} & a_{41} & a_{42} & a_{43} \end{bmatrix},$$

$PD(A_{1,3}) = (-1)^3 a_{14}a_{21}a_{32}a_{43} - a_{13}a_{22}a_{31}a_{44},$

Total $PD = PD(A_{1,0}) + PD(A_{1,1}) + PD(A_{1,2}) + PD(A_{1,3})$

count circular $= 3$.

Step 6. Go to step 2 for next $j = 2$ and repeat steps 3, 4 and 5. Stop when $j = \frac{(n-1)!}{2}$.

Step 7. Calculate the $\det(A)$:

$$\det(A) = \left[\sum_{k=0}^{3} (PD(A_{1,k}) + PD(A_{2,k}) + PD(A_{3,k}))\right].$$

It can be simplified as follows:

$$\det(A) = \sum_{j=1}^{3} \left[\sum_{k=0}^{3} (PD(A_{j,k}))\right].$$

This sequential steps from 2 until 7 are shown for starter [1-4]. We use steps 2-7 for other two starters.

The mathematical formula of our algorithm for any $n \times n$ matrix A, is

$$\det(A) = \sum_{j=1}^{\frac{(n-1)!}{2}} \left[\sum_{k=0}^{n-1} (PD(A_{j,k}))\right].$$

4. Results and Discussion

We used the three division free algorithms, namely, the cofactor expansion algorithm, Langdon [2] algorithm, and Thongchiew [9] algorithm to compare with our new algorithm. Specifically Langdon method is fall under permutation method. Langdon method was developed by Langdon [2] for permutation generation and Sedgeweck [5] highlighted that the Langdon method is the best method for circular permutation generation. Since also our method is based on circular permutation, we also applied Langdon method for finding the determinant. All sequential algorithms are implemented in C++ language and run in laptop which has Pentium Dual-Core 1.7GHz processor and 2.99 GB RAM. We test the algorithms for real number matrix. The result is given in Table 2 and represents computation time, in seconds.

Table 2. The computation time (in seconds)

n	Our new algorithm	Langdon	Thongchiew	Cofactor expansion
7	0.000	0.000	0.00	0.016
8	0.031	0.032	0.031	0.125
9	0.172	0.297	0.328	1.203
10	2.000	3.313	3.719	12.360
11	23.610	41.203	46.422	133.453
12	337.140	570.844	637.250	1616.547

As can be seen in Table 2, the execution time grows dramatically when size of the matrix increases, and our new algorithm is less time than comparable permutation program Langdon [2] and Thongchiew [9]. Meanwhile the cofactor expansion algorithm is the slowest among them. The result from comparison of the program is indicated that the new algorithm is better over time computation than these three division free algorithms for finding determinant. For $n \geq 13,$ the process becomes longer and the laptop memory capability is limited. However, we expected that our new algorithm still performed better.

Based on the result from Table 2, and from our algorithm development in Section 3, we can summarize two advantages of this new algorithm compared to

Langdon [2] and Thongchiew [9] algorithms. Firstly, their algorithms generated all $n!$ permutations whereas our algorithm generated only $\frac{n!}{2}$ permutations. The next $\frac{n!}{2}$ permutations will be generated by reversing the order of permutation of the first $\frac{n!}{2}$ permutations as explained in Example 4. Secondly, we only calculated the sign of main diagonal column indices and the sign of secondary diagonal column indices are depended to the sign of main diagonal column indices, whereas Langdon [2] and Thongchiew [9] algorithms, signs of each $n!$ product terms are computed. We can conclude that these two essential factors contributed to less computation time for our new algorithm.

5. Conclusion

In this paper, the new division free algorithm for finding determinant via circular permutation has been discussed and compared in terms of time computation in order to indicate its beneficial. This circular permutation algorithm also can be used to calculate the determinant for any size of square matrix using cross multiplication method. Future work should encompass on implementation of our sequential algorithm for parallel computational environment. This exploration will be contributed for time execution reduction of the algorithm.

References

[1] D. Hajrizaj, New method to compute the determinant of a 3×3 matrix, International J. Algebra 3 (2009), 211-219.

[2] G. G. Langdon, An algorithm for generating permutations, Communication of ACM 10 (1967), 298-299.

[3] M. Mahajan and V. Vinay, Determinants: Combinatorics, algorithms, and complexity, Chicago J. Theoretical Computer Science 5 (1997), 730-738.

[4] H. Schneider and G. P. Barker, Matrices and Linear Algebra, Dover Publications Inc., New York, 1989.

[5] R. Sedgewick, Permutation generation methods, Comput. Surveys 9 (1977), 137-164.

[6] K. Sharmila, I. Haslinda, O. Zurni, O. Kharil Iskandar and S. Mohamed, Generating permutation based on starter sets, Electronic Proceeding of 5th Asian Mathematical

Conference, Vol. 1, 2009, pp. 90-96, Available
url:http://www.mat.usm.my/AMC 20200920 Proceedings/Pure/P134.PDF

[7] D. W. Shin, The permutation algorithm for non-sparse matrix determinant in symbolic computation, Proc. on the 15th CISL winter workshop, Japan, 2002, Available
url: http://icat.snu.ac.kr:3333/ww/pdf/ww200210/pdf.

[8] M. Soltys, Berkowitz's algorithm and Clow sequences, Electronic J. Linear Algebra 9 (2002), 42-54.

[9] K. Thongchiew, A computerize algorithm for generating permutation and its application in determining a determinant, Proc. World Academy of Science, Engineering and Technology 21 (2007), 178-183.