ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 6, No. 1, 2011, *pp*. 049-054

Comparisons for Determinants of Special Matrices by Algorithm Proposed

Lugen M. Zake[1], Haslinda Ibrahim[2], Zurni Omar[3]

College of Arts and Sciences, Universiti Utara Malaysia, 06010 UUM Sintok, Kedah, Malaysia

(*Received October 19, 2010, accepted December 20, 2010*)

Abstract. This paper presents the results of an early which compared the new algorithm to special matrices, purpose to calculate the value of determinant. Matrices which are numerically reliable techniques are used for this purpose. This technique permits for accuracy in the sense that it breaks down the new determinant for comparison to improve the existing numerical algorithms and to calculate the determinant of matrix.

Keywords: comparisons for determinants, determinants of special matrices, special matrices for determinants

1. Introduction

There are more algorithms than determinants of matrices. Several algorithms for the determinant of matrices have been recognized [2, 4, 5]. Elimination algorithm is not successful that it produces fractions. This is called naive the technique, because it does not avoid the problem of dividing by zero. This point has to be taken into account when implementing this technique on computers. A method for improving the above elimination algorithm was already known to elimination algorithm with Partial Pivot and complete pivot. This algorithm increase the operation of flop point. The algorithm of permutation obtains the determinant of matrices from its definition by generating all the permutations and copying the entries according to each permutation. The algorithm of permutation produces the determinant of matrices without additional fractions. But it is not easy to get all the permutations from the set of integers. The generating of permutations is actually one of the complete problems. To improve the permutation of algorithm for determinant, we must bring new algorithm for generating the permutations. Khanit (2007) found an algorithm for generating permutation and he used it in algorithm for determinant of any matrix, but he didn't presented comparisons to the test the new algorithm. Dong (2002) has presented a new algorithm for the generation of permutation and used this algorithm to find algorithm for determinant. However the empirical tests proved that algorithm gave the same operation flop point of permutation algorithm for determinant is ($n.n!$). He didn't write program for this algorithm for doing comparisons with other existing algorithms. We present a new algorithm for generating of permutation and use this algorithm to find a new algorithm for determinant. We present many comparisons with other existing algorithms by using special matrices in computer.

2. Construction of New Algorithm

As described in the previous section, there are different algorithms for the calculation of the determinant of matrices, but here we present a new algorithm for calculating the determinant of matrices by using permutation. This algorithm is based on a new algorithm for generating permutations, which gives the determinant of an n-by-n matrix.

To calculate the determinant of matrices, let A be a matrix of $n\times n$ where n is the size of the matrix. Then the steps that lead to the computation of the new algorithms are as follows:

1. Input the matrix A which you want for computation algorithm for determinant of matrices and find the number of all permutations by using the factorial rule as the follow: $n!=n\times(n-1)\times(n-2)\times(n-3)\times.....\times 1!$

[1] Corresponding author. *E-mail address*: lujaenalsufar@yahoo.com.
[2] *E-mail address*: linda@uum.edu.my.
[3] *E-mail address*: zurni@uum.edu.my.

2. Construction algorithm generating of permutation : Generating all the permutations by using the new algorithm for generating of permutation by three steps first, fixing one element then find the number permutations by $(n-1)!$, second, delete the equivalents' permutations by $(n-1)!/2$, three find all permutations by cycle the reminder permutations, for example $n = 4$, the permutations generating in the following:

1 2 3 4				
1 2 4 3	$\rightarrow$	1 2 3 4	$\rightarrow$	1 2 3 4, 2 3 4 1, 3 4 1 2, 4 1 2 3
1 3 2 4		1 2 4 3		1 2 4 3, 2 4 3 1, 4 3 1 2, 3 1 2 4
1 3 4 2	$\rightarrow$	1 4 2 3	$\rightarrow$	1 4 2 3, 4 2 3 1, 2 3 1 4, 3 1 4 2
1 4 2 3				
1 4 3 2				
Step1		**Step2**		**Step 3**

3. Generate a matrix depending on the new permutation.

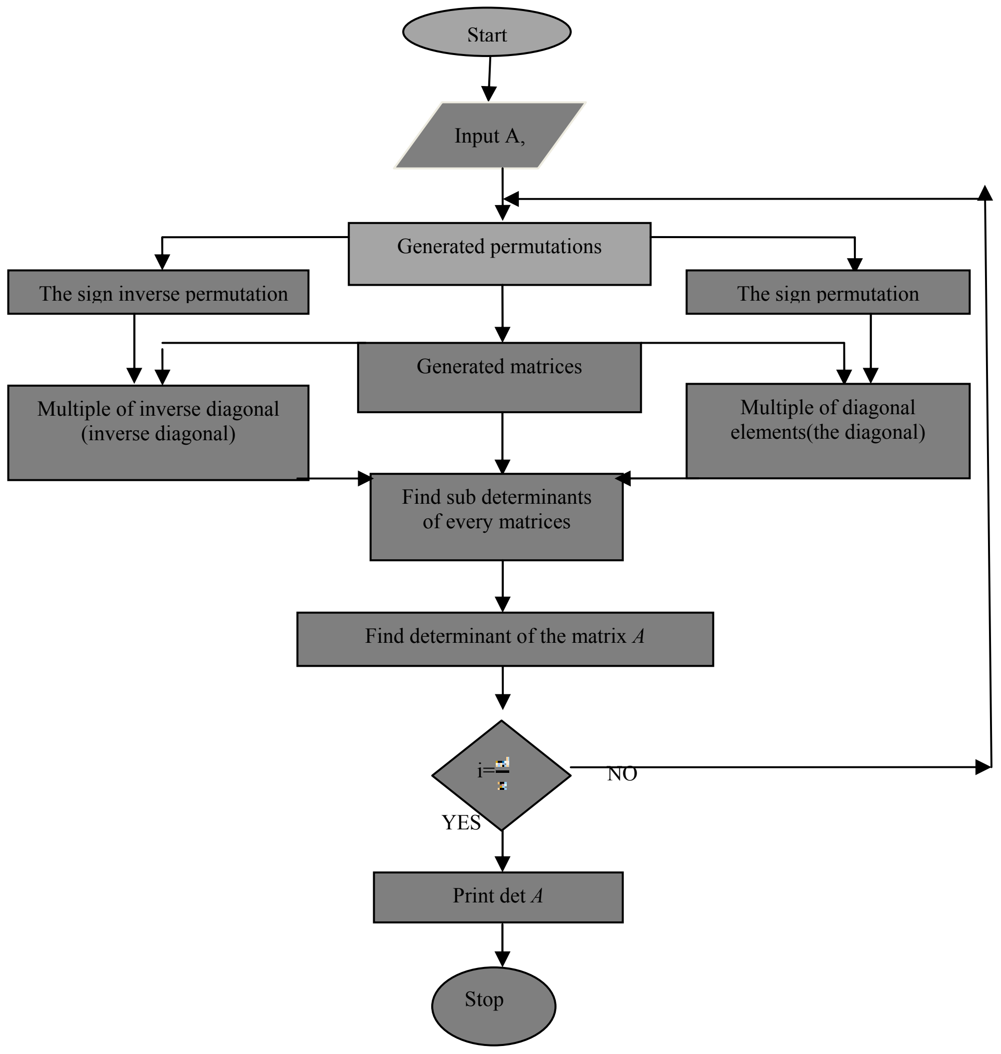
4. Find the sign permutation and sign inverse permutation.

The sign for permutation

$$sign\ (1\ 2\ 3\ 4\ \ldots.\ n) =$$
$$(-1)^{*\text{number of inversion permutation}} = (-1)^{*0} = 1 = +1$$

and sign for inverse permutation

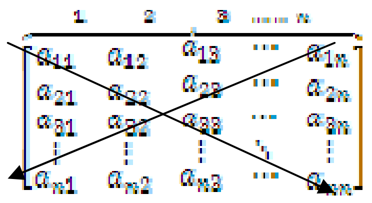
$$sign\ (n\ \ldots\ldots 4\ 3\ 2\ 1) =$$
$$(-1)^{*\text{number of inversion inverse permutation}} = (-1)^{*(n-(n-1)+n(n-2)+\ldots\ +2-1)} = \begin{cases} 1 & \text{if } n \text{ is even} \\ -1 & \text{if } n \text{ odd} \end{cases}.$$



Start
Input A,
Generated permutations
The sign inverse permutation
The sign permutation
Generated matrices
Multiple of inverse diagonal (inverse diagonal)
Multiple of diagonal elements(the diagonal)
Find sub determinants of every matrices
Find determinant of the matrix A
i=
NO
YES
Print det A
Stop

Fig.1: Flow Chart for New Algorithm

5. Find the sub determinant for a matrix by using multiplication of diagonal elements of matrix and inverse diagonal elements of matrix, then combination to gather .


1 2 3 ...... n
a_{11} a_{12} a_{13} ... a_{1n}
a_{21} a_{22} a_{23} ... a_{2n}
a_{31} a_{32} a_{33} ... a_{3n}
a_{n1} a_{n2} a_{n3} ... a_{nn}

sign (π^{-1}) a_{1n}..... .....a_{n1} *sign* (π) $a_{11}.a_{22}.a_{33}.a_{44}$ ......$.a_{nn}$

for all matrices products generated from step 3.

6. If the number permutation equal $n!/2$ Combination of all the products of the sub determinants for all sub matrices which are produced from step 5. To find the final result for the determinant to the original matrix, result the determinant of matrix A. If no do repetition of all steps from step 2 ─ 6.

3. Analysis the New Algorithm

The new formula for algorithm of the determinant of matrices involves half the sum of the all permutations of columns. In other words, this algorithm involves the sum over half of the permutations of columns. The algorithm for determinant of matrices has generated permutations to be prevented from an increase of time. The new algorithm that you need to find is the half of the number of permutations *((n!/2))*. Using an efficient algorithm to find that permutations is described above. This study will help us to provide half of the time that is spent for algorithm of the determinant of matrices which is compared with general algorithm. The option for memory saving is copying $(n-1)/2$ entries after each process of permuting indices and moving quickly the current indices to obtain the next permutations. With the strategy for new algorithm, the prototype of execution time function is

$$O(n^2) = (n-1)\,A(algorithm) + n!/2\ \ B(algorithm) + (n - 1)/2 \times n!/2 \qquad (1)$$

where A(algorithm) and B(algorithm) are the two the time functions. The former has the dependencies on the algorithm used to the generating of permutations, and the latter $((n\ \text{-}1)/2{\times}n!/2)$ is the time for generating multiplication signs between entries. A(algorithm) implies the time required to obtain all the permutations from column indices and to copy the entries. Similarly, B(algorithm), implies the time for decision plus/minus sign for a product. So, selecting/devising fast and simple algorithms for those two time functions implies the successful implementation of this algorithm. Then the operations of the new possible algorithm is calculated as follows:

$$= n.\ \frac{(n-1)!}{2} + 1 + \sum_{p=2}^{n}\left(\frac{n!}{2p}\right).p(p-1)/2p. + \ldots + \frac{(n-1)n!}{2} + \frac{n!}{2}. \qquad (2)$$

where $n\ .\ \frac{(n-1)!}{2}$ signifies the time for copying all the entries, $\left(\frac{n!}{2p}\right)$. The number of permutations requiring same modulus operations in step2, p the number of copying process when reversing a portion of array in step3, $(p-1)/2p$ the number of divisions that determines reversing point in step5, $\frac{(n-1)n!}{2}$ the total number of multiplication signs generation between entries in step4. The last term $\frac{n!}{2}$ is for the time to determine plus/minus signs of products and the reason of its simplicity is that the summing process of step6 can be replaced by another reference table of integers. This time function is valid for $n > 4$. Computing this time function results in the estimation below.

$$= n.\ \frac{(n-1)!}{2} + 1 + \left(\frac{n!}{2.2}\right).2(2-1)/2 + \left(\frac{n!}{2.3}\right).3(3-1) + \left(\frac{n!}{2.4}\right).4(4-1) + \cdots + (n-1).\frac{n!}{2}$$
$$+ \frac{n!}{2},$$

$$= n.\ \frac{(n-1)!}{2} + 1 + \left(\frac{n!}{2}\right) + \left(\frac{n!}{2}\right).(2) + \left(\frac{n!}{2}\right).(3) + \cdots + (n-1).\frac{n!}{2} + \frac{n!}{2},$$

$$= n.\ \frac{(n-1)!}{2} + 1 + constant.\left(\frac{n!}{2}\right) + (n-1).\frac{n!}{2} + \frac{n!}{2}\ \cong\ n.\frac{n!}{2}. \qquad (3)$$

This algorithm has an *O (n. (n! /2))* procedure. Then the time function has more influential term (*n. n! /2)* of the time complexity of the entire algorithm for determinant of matrices.

4. Empirical Tests and Considerations

We implemented our algorithms on computer and investigated the efficiency for several test matrices. The tables show the timing data on laptop Acer 5635Z (Pentium 800 MHz CPU, 3 Mb of main memory).

The permutation algorithm for calculating determinants is believed to be superior to elimination algorithms in most cases, especially for special matrices. Therefore we compared two algorithms: (1) The recursive general permutation algorithm; (2) Our efficient permutation algorithm, for the following dense matrices for which is suited:

i. A Toeplitz Matrix: $A_{i,j} = A_{i-1,j-1}$, If the i,j element of A is denoted $A_{i,j}$,

ii. A Hilbert Matrix: $A_{ij} = \frac{1}{i+j-1}$,

iii. A Hessenberg Matrix: $A_{ij} = a_{ij}$, matrix with $a_{i,j} = 0$ for $i > j + 1$.

Table 1 shows the results. We see that the new permutation algorithm is quite efficient for the Toeplize matrix. This result is reasonable because, for Toeplize matrices, the fraction-free is not serious and the general permutation algorithm is efficiently performed, as it is the case for numeric matrices.

Hilbert Matrix has fraction; therefore, it is inefficient for all algorithm. But the new algorithm is efficient more than the general permutation algorithm. In Hessenberg cases, however, the fraction-free algorithm is quite inefficient because of the intermediate expression expansion. Table 1 shows these results for tests in the following:

Table 1. Timings by Two Algorithms for Three Kinds of Matrices

Matrix	***n***	**New permutation algorithm**	**General permutation algorithm**
(i)	4	0.0780	0.0470
	5	0.2650	1.1410
	6	0.9690	1.6570
	7	8.4850	13.6410
	8	96.531	129.344
(ii)	4	0.0310	0.0310
	5	0.1250	0.5460
	6	1.0160	1.9690
	7	9.1090	15.859
	8	105.922	156.547
(iii)	4	0.0450	0.0930
	5	0.1090	0.2660
	6	1,0160	1.8440
	7	10.234	15.7190
	8	118.265	150.844

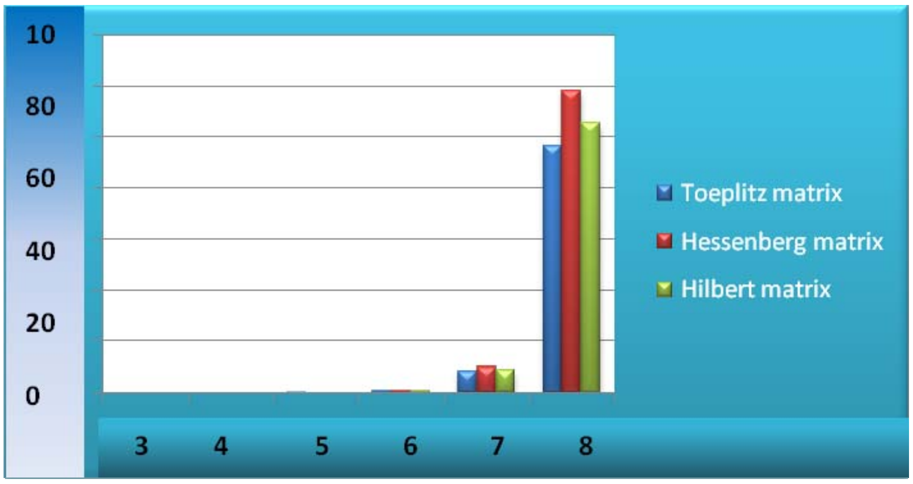


Fig. 2: Comparison for Three Kinds of Matrices by New Algorithm

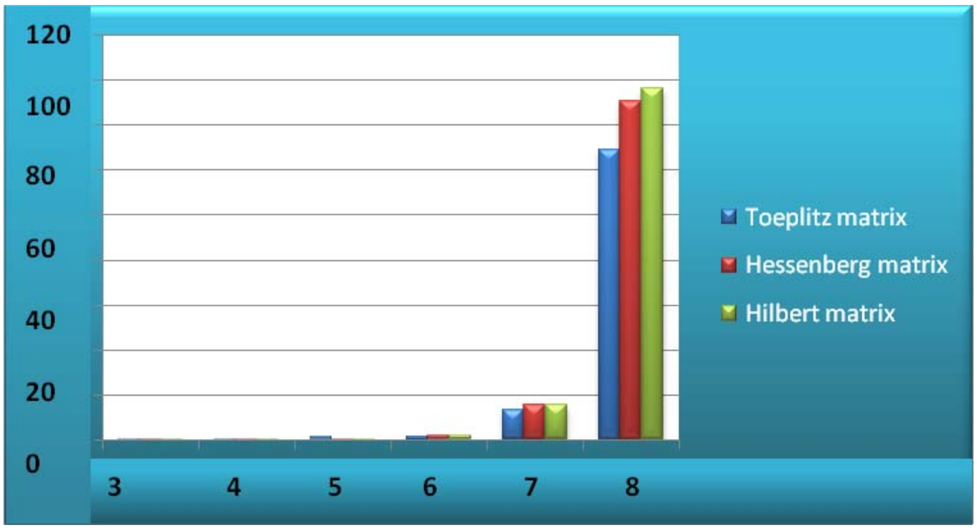


Fig. 3: Comparison for Three Kinds of Matrices by General Algorithm

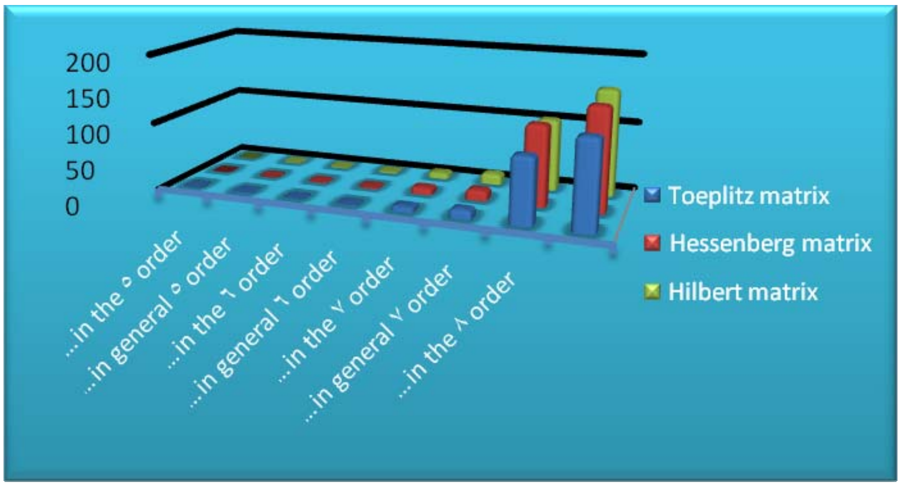


Fig.4: Comparison by Two Algorithms for Three Kinds Matrices

We expect that our technique will improve permutation algorithm in calculating determinants of matrices, in terms of time, efficiency and performance.

5. Conclusion and Discussion

The three types of matrices seem to be availed measurement to compare the efficiency of the new algorithm by computer, which is clearly explained in the table 1. We note in the figure 2 and 3, dearly test the speed for the three special matrices of the two algorithms. We see the success of the two methods in the matrix toeplitz. Figure 4 shows clearly the time spent for the two algorithm in the three matrices together. Also, it clearly shows the superiority and efficiency of the proposed algorithm for the general algorithm in the three types of matrices.

6. References

[1] B. Martin. *Differential Equations and Their Applications*(4th edition). Springer-Verlag, 1993.

[2] H. P. William, F. P. Brian, A. Saul, B. Teulsky, & T. V. William. *Numerical Recipes, the art of scientific computing.* Cambridge University Press, 1st edition1. 1986, **1**.

[3] K. Erwin. *Advanced Engineering Mathmatheics*. John Wiley and Sons, 1988.

[4] T. Khanit. A Computerize Algorithm for generating permutation and its Application in determining a determinant. *World academy of Science, Engineering and Technology.* 2007, 27.

[5] W. S. Dong. The Permutation Algorithm for Non-Sparse Matrix Determinant in Symbolic Computation. *Proceedings on the 15th CISL Winter Workshop*. Kushu, Japan. 2002.