



Coding of New Precede Operator

Nur Farah Alia Zahrullail, Tahir Ahmad

Department of Mathematical Sciences, Faculty of Science
Universiti Teknologi Malaysia, 81310 Johor Bahru, Malaysia
Corresponding author: tahir@utm.my

Abstract

The goal of this research is to use computer programming to generate the coding of a new precede operator for matrices. The code is designed in the MATLAB and provides for counting using a matrix-based language. The algorithms for precede operator and propositions are composed. They contain loop, 'if-else' statements. Precede operator's algorithm for square matrices is then verified. The algorithm is transformed into its coding. The coding is determined based on the matrix's entries. Furthermore, the precede operator's algorithm ensures that it can be applied to any square matrices.

Keywords: Precede Operator; MATLAB

1. Introduction

Epilepsy is a disorder caused by the central nervous system (neurological) in which the brain becomes abnormal, causing seizures or periods of behaviour that are out of control and sometimes loss of consciousness. Epilepsy sufferers include of male or female of all races, ethnic backgrounds and ages. Although the cause of the disease is unknown, the symptoms can be confirmed through medical method. Epilepsy can cause uncontrolled movements and loss of consciousness that can cause severe injury or even death. The Electroencephalogram (EEG) is widely uses for diagnosing and assessing brain activities and disorder (Lasefr et al., 2017).

Mathematicians have identified that Electroencephalogram (EEG) signal can be transform into matrices. The researchers found remarkable results, that is, epileptic patient's EEG signal are not chaotic but instead produced a pattern in the form of a simple algebraic structure in form of matrices. However, the obtained matrices need to be ordered. (Ahmad Fuad & Ahmad, 2021).

Programming is a process of designing and building computer programs to achieve certain computing results or to perform certain tasks. Programming can speed up the input and output processes of certain computation (Aisyah et al., 2021). Mostly, it is very useful and efficient in calculating big data and have ability to collect, manage and analyse the information.

In this project, MATLAB will be used. MATLAB is a programming language created by Math Works that allows engineers and scientists to study and design computer code in order to create a programme. MATLAB's core is a matrix-based language that provides for the most intuitive mathematical representation of operations. Because it is incredibly accurate, fast and simple, MATLAB has been used programmers.

Input and output, end and other MATLAB variables, functions, data types and commands are used in this project. It contains looping and if-else statements, as well as matrix. As a result, there are numerous conditions to be considered.

Precede Operator is a new mathematical operation introduced by Ahmad Fuad and Ahmad (2021). It can be used for recorded electroencephalography (EEG) signals, in which the signal can be presented as matrices. However, the procedure has never been automatized yet; i.e. to code it.

There are two objectives in this study. Firstly, to code the precede operation using MATLAB. Secondly, to show its possibilities for other possible of applications. This study will focus on developing algorithms for ordering matrices in C++ language using MATLAB. Some samples will be presented.

Previous results on the concept are reviewed. This research is aimed to write the coding for precede ordering technique. The resultant coding can transform and shorten the computing time for the EEG and ordering the obtained square matrices.

2. Literature Review

2.1. Definition of Matrices

A matrix defines as an orderly arrangement of some number or symbols in certain rows and columns enclosed by some brackets, subscribe by the magnitude of its order and denominated by some capital letters (Wanjohi, 2021). Generally, matrices are rectangular arrays that arranged in rows and columns, that are used to represent mathematical objects. For examples,

$$2x + 5y = 10, 9x + 4y = 27$$

$$\begin{bmatrix} 2 & 5 \\ 9 & 4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 10 \\ 27 \end{bmatrix}$$

Matrix consists of $m \times n$ array which is the size of a matrix. The m defines number of rows while n defines number of columns. From the example, we can say that size for matrix A is 2×2 and size for matrix B is 2×3 . Each number inside the matrix is called an entry and for each or specific entry of the matrix consists i, j th notation for some positive integer. The entries start with the row from top to bottom and for column from left to right. Notation of an entry starts from a_{11} until a_{mn} .

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$$

There are three common operations for matrices, namely addition, subtraction and multiplication. The use of matrix has been applied in various fields such as in medicine, communications, technology and many more.

2.2. Type of Matrices

The different types of matrices are:

Table 2.2.1: Type of matrices

Type of Matrix	Detail	Example
Row Matrix	$A = [a_{ij}]_{1 \times n}$	$A = [-5 \quad 3 \quad -1]$
Column Matrix	$A = [a_{ij}]_{1 \times m}$	$A = \begin{bmatrix} -5 \\ 3 \\ -1 \end{bmatrix}$
Zero / Null Matrix	$A = [a_{ij}]_{m \times n}$ where $a_{ij} = 0$	$A = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$
Singleton Matrix	$A = [a_{ij}]_{m \times n}$ where $m = n = 1$	$[-5], [3], [-1]$
Horizontal Matrix	$A = [a_{ij}]_{m \times n}$ where $n > m$	$\begin{bmatrix} 1 & 3 & 5 \\ 2 & 6 & 4 \end{bmatrix}$
Vertical Matrix	$A = [a_{ij}]_{m \times n}$ where $m > n$	$\begin{bmatrix} 1 & 3 \\ 5 & 2 \\ 6 & 4 \end{bmatrix}$
Square matrix	$A = [a_{ij}]_{m \times n}$ where $m = n$	$\begin{bmatrix} 5 & 1 & 8 \\ 3 & 7 & 9 \\ 2 & 4 & 6 \end{bmatrix}$
Diagonal Matrix	$A = [a_{ij}]_{m \times n}$ when $i \neq j$	$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 6 \end{bmatrix}$

Scalar Matrix	$A = [a_{ij}]_{m \times n}$ where $a_{ij} = \begin{cases} 0, & i \neq j \\ k, & i = j \end{cases}$ where k is constant	$\begin{bmatrix} \sqrt{5} & 0 \\ 0 & \sqrt{5} \end{bmatrix}$
Identity Matrix	$A = [a_{ij}]_{m \times n}$ where $a_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
Equal Matrix	$A = [a_{ij}]_{m \times n}$ and $B = [b_{ij}]_{r \times s}$ where $a_{ij} = b_{ij}$, $m = r$ and $n = s$	$A = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 6 & 4 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{bmatrix}$
Triangular Matrix	Can either be upper triangular ($a_{ij} = 0$ when $i > j$) or lower triangular ($a_{ij} = 0$ when $i < j$)	Upper: $\begin{bmatrix} 1 & 3 & 7 \\ 0 & 5 & 9 \\ 0 & 0 & 4 \end{bmatrix}$ Lower: $\begin{bmatrix} 1 & 0 & 0 \\ 3 & 5 & 0 \\ 7 & 4 & 4 \end{bmatrix}$
Singular Matrix	$ A = 0$	$A = \begin{bmatrix} 3 & 6 \\ 2 & 4 \end{bmatrix}$ $ A = (3 \times 4) - (6 \times 2)$ $= 12 - 12$ $= 0$
Non-Singular Matrix	$ A \neq 0$	$A = \begin{bmatrix} 1 & -4 \\ 3 & 5 \end{bmatrix}$ $ A = (1 \times 5) - (3 \times (-4))$ $= 5 - (-12)$ $= 17$
Transpose Matrix	$A = [a_{ij}]_{m \times n}$ $A^T = [a_{ij}]_{n \times m}$	$A = \begin{bmatrix} 6 & 1 \\ 4 & -9 \\ 24 & 8 \end{bmatrix}$ $A^T = \begin{bmatrix} 6 & 4 & 24 \\ 1 & -9 & 8 \end{bmatrix}^T$
Symmetric Matrices	$A = [a_{ij}]$ where $a_{ij} = a_{ji}$	$A^T = A$ $\begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 8 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 8 \end{bmatrix}^T$
Skew-Symmetric Matrices	$A = [a_{ij}]$ where $a_{ij} = -a_{ji}$	$A^T = -A$ $A = \begin{bmatrix} 0 & 3 & 4 \\ -3 & 0 & 7 \\ -4 & -7 & 0 \end{bmatrix}$ $A^T = \begin{bmatrix} 0 & -3 & -4 \\ 3 & 0 & -7 \\ 4 & 7 & 0 \end{bmatrix}$ $-A = \begin{bmatrix} 0 & -3 & -4 \\ 3 & 0 & -7 \\ 4 & 7 & 0 \end{bmatrix}$
Conjugate Matrix	\bar{A}	$A = \begin{bmatrix} 1+i & 2+3i \\ 4-2i & 6 \end{bmatrix}$ $\bar{A} = \begin{bmatrix} 1-i & 2-3i \\ 4+2i & 6 \end{bmatrix}$
Hermitian Matrix	$A = A^\theta$ or $(\bar{A})^T$ (conjugate transpose of matrix A)	$A = \begin{bmatrix} 3 & 1-i \\ 1+i & -2 \end{bmatrix}$ $\bar{A} = \begin{bmatrix} 3 & 1+i \\ 1-i & -2 \end{bmatrix}$ $A^\theta = \begin{bmatrix} 3 & 1-i \\ 1+i & -2 \end{bmatrix} = A$
Skew-Hermitian Matrix	$A^\theta = -A$	$A = \begin{bmatrix} 3i & 1+i \\ -1+i & -i \end{bmatrix}$ $A^T = \begin{bmatrix} 3i & -1+i \\ 1+i & -i \end{bmatrix}$

		$(\bar{A})^T = \begin{bmatrix} -3i & -1-i \\ 1-i & i \end{bmatrix}$ $= -A$
Orthogonal Matrix	$AA^T = A^T A = I_n$	$A = \begin{bmatrix} \cos x & \sin x \\ -\sin x & \cos x \end{bmatrix}$ $A^T = \begin{bmatrix} \cos x & -\sin x \\ \sin x & \cos x \end{bmatrix}$ $AA^T = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I$
Idempotent Matrix	$A^2 = A$	$A^2 = \begin{bmatrix} 5 & 10 \\ -2 & -4 \end{bmatrix} \times \begin{bmatrix} 5 & 10 \\ -2 & -4 \end{bmatrix}$ $= \begin{bmatrix} 5 & 10 \\ -2 & -4 \end{bmatrix} = A$
Involuntary Matrix	$A^2 = I$ or $A^{-1} = A$	$A^2 = \begin{bmatrix} 2 & -1 \\ 3 & -2 \end{bmatrix}^2$ $= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Nilpotent Matrix	$\exists p \in N$ such that $A^p = 0$	<p>Let $p = 2,$</p> $A^2 = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ -1 & -2 & -3 \end{bmatrix}^2$ $= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

2.3. Precede Operator

Ahmad Fuad & Ahmad (2021) generated an Ordering of Transformed Recorded Electroencephalography (EEG) Signals by a Novel Precede Operator. The main purpose of it is to introduce a new technique of ordering transformed EEG signals in term of square matrices. There are few theorems generated using the precede operator.

Definition 1 (Fuad, et al., 2021). *Let C and C' be n × n matrices and C ≠ C'. Matrix C is said to precede C', written as C > C', whenever the first c_{ij} > c'_{ij} exists for some i, j. The comparison must be made in the sequence of rows, i.e., R₁, R₂, ..., R_n, until the first c_{ij} > c'_{ij} is discovered and denoted as > (C, C') = c_{ij}. Otherwise, if c_{ij} < c'_{ij}, then C' > C. When C = C', i.e., all the corresponding entries for each matrix are the same, then > (C, C') = c₁₁. Consider C, C' ∈ M_n such that*

$$C = \begin{pmatrix} c_{11} & \dots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{n1} & \dots & c_{nn} \end{pmatrix}, \quad C' = \begin{pmatrix} c'_{11} & \dots & c'_{1n} \\ \vdots & \ddots & \vdots \\ c'_{n1} & \dots & c'_{nn} \end{pmatrix}$$

2.4. Concept of Ordering Matrices

Concept of ordering matrices has sparked interest over a past few decades by mathematicians. According to Ahmad Fuad & Ahmad (2021), several order relations for matrix algebra have been introduced in connection to a series of applications relevant to different branches of mathematics and its applications. A matrix represents transformation on a space. It deals with numbers in a given space.

3. Methodology

3.1. Research Design and procedure

The research begins with the introduction of the precede several properties on its symmetric matrices are generated. Next, the algorithm is developed using MATLAB for simulation purpose.

3.2. Proposed Work

The algorithm for using the precede operator in step-by-step mode is described in this section. Input-output, matrix declaration, 'if-else' statements and end functions are all included.

3.3.1 Algorithms

a) Precede Operator

Step 1: Start.

Step 2: Insert the input (matrices A and B).

Step 3: Accept input.

Step 4: Declare the entries of each matrix.

Step 5: Accept declarations and display on command window.

Step 6: Create a condition for first entry of matrix A if bigger than first entry of matrix B. If yes, ($A_1 > B_1$), stop run. Else go to Step 8.

Step 7: Display 'matrix A is bigger than matrix B'.

Step 8: Check if first entry of matrix B if bigger than first entry of A, ($B_1 > A_1$). If yes, stop run. Else go to Step 10.

Step 9: Display 'matrix B is bigger than matrix A'.

Step 10: Check if first entry of matrix A is equal to the first entry of matrix B. If yes, ($A_1 == B_1$), create a condition for second entry of matrix A if bigger than second entry of matrix B. If yes, ($A_2 > B_2$), stop run. Else go to Step 12.

Step 11: Display 'matrix A is bigger than matrix B'.

Step 12: Check if second entry of matrix B if bigger than second entry of matrix A, ($B_2 > A_2$). If yes, stop run. Else go to Step 14.

Step 13: Display 'matrix B is bigger than matrix A'.

Step 14: Check if second entry of matrix A is equal to the second entry of matrix B. If yes, ($A_2 == B_2$), create a condition for third entry of matrix A if bigger than third entry of matrix B. If yes, ($A_3 > B_3$), stop run. Else go to Step 16.

Step 15: Display 'matrix A is bigger than matrix B'.

Step 16: Check if third entry of matrix B if bigger than third entry of matrix A, ($B_3 > A_3$). If yes, stop run. Else go to Step 18.

Step 17: Display 'matrix B is bigger than matrix A'.

Step 18: Check if third entry of matrix A is equal to the third entry of matrix B. If yes, ($A_3 == B_3$), create a condition for fourth entry of matrix A if bigger than fourth entry of matrix B. If yes, ($A_4 > B_4$), stop run. Else go to Step 20.

Step 19: Display 'matrix A is bigger than matrix B'.

Step 20: Check if fourth entry of matrix B if bigger than fourth entry of matrix A, ($B_4 > A_4$).

Step 21: Display 'matrix B is bigger than matrix A'.

Step 22: End.

b) Proposition 1

Step 1: Start.

Step 2: Insert the input (matrices A and B).

Step 3: Accept input.

Step 4: Declare the entries of each matrix.

Step 5: Accept declarations and display on command window.

Step 6: Display 'This is Transpose Matrix A and Matrix B:'.

Step 7: Declare matrix transpose as A' and B' .

Step 8: Accept declaration and display on command window.

Step 9: End.

c) Proposition 2

Step 1: Start.

Step 2: Insert the input (matrices A and B).

Step 3: Accept input.

Step 4: Declare the entries of each matrix.

Step 5: Accept declarations and display on command window.

- Step 6: Display 'This is Negative Matrix A and Matrix B:'.
- Step 7: Declare negative matrix as $-A$ and $-B$.
- Step 8: Accept declaration and display on command window.
- Step 9: End.

d) Proposition 3

- Step 1: Start.
- Step 2: Insert the input (matrices A and B).
- Step 3: Accept input.
- Step 4: Declare the entries of each matrix.
- Step 5: Accept declarations and display on command window.
- Step 6: Declare k value as 3.
- Step 7: Accept k declaration.
- Step 8: Display 'This is matrix-positive-scalar multiplication preserves:'.
- Step 7: Declare the matrices as $k*A$ and $k*B$.
- Step 8: Accept declaration and display on command window.
- Step 9: End.

e) Proposition 4

- Step 1: Start.
- Step 2: Insert the input (matrices A and B).
- Step 3: Accept input.
- Step 4: Declare the entries of each.
- Step 5: Accept declarations and display on command window.
- Step 6: Display 'This are skew symmetric matrix A and B:'.
- Step 7: Declare negative matrix as $-A'$ and $-B'$.
- Step 8: Accept declaration and display on command window.
- Step 9: End.

4. Results and discussion

4.1. Codes

a) Precede Operator

```
% insert matrices A and B

A1 = 3;
A2 = 4;
A3 = 15;
A4 = 3;
B1 = 3;
B2 = 4;
B3 = 15;
B4 = 14;

% declare the entries of the matrices
A = [A1 A2; A3 A4]
B = [B1 B2; B3 B4]
```

```

% apply condition to compare the entries
if (A1 > B1)
    disp ('matrix A is bigger than matrix B')
elseif (B1 > A1)
    disp ('matrix B is bigger than matrix A')
elseif (A1 == B1)
    if (A2 > B2)
        disp ('matrix A is bigger than matrix B')
    elseif (B2 > A2)
        disp ('matrix B is bigger than matrix A')

        elseif (A2 == B2)
    if (A3 > B3)
        disp ('matrix A is bigger than matrix B')
    elseif (B3 > A3)
        disp ('matrix B is bigger than matrix A')

        elseif (A3 == B3)
    if (A4 > B4)
        disp ('matrix A is bigger than matrix B')
    elseif (B4 > A4)
        disp ('matrix B is bigger than matrix A')
    end
    end
    end
end

```

In this coding, the entries of the matrices are inserted to produce the output of the matrices in command window. By entering elements in each row as comma or space separated numbers and using semicolons to mark the end of each row. Then, declare the entries of the matrices as $A = [A1 \ A2; \ A3 \ A4]$ and $B = [B1 \ B2; \ B3 \ B4]$. Several conditions are obtained to compare the entries in order to implement precede ordering. Since both matrices have the same size (2x2), then each pair corresponding entry is compared as follow,

Step 1: given matrices $C, C' \in M_n$ and $C \neq C'$, determine row i for C and C' , i.e., $R_i(C) = (c_{i1}, c_{i2}, \dots, c_{in})$ and $R_i(C') = (c'_{i1}, c'_{i2}, \dots, c'_{in})$.

Step 2: determine matrix $[\omega_{ij}]$ whereby $\omega_{ij}: R_i(C) \times R_i(C') \rightarrow M_n$, such that $\omega_{ij} = \begin{cases} c_{ij}, & \text{when } c_{ij} > c'_{ij} \\ c'_{ij}, & \text{when } c'_{ij} > c_{ij} \end{cases}$ for $i, j = 1, 2, \dots, n$.

Step 3: determine matrix $[\omega'_{ij}]$ whereby $\omega'_{ij}: M_n \rightarrow M_{1 \times n^2}$, such that $\omega'_{ij}([\omega_{ij}]) = (\text{Row}_1([\omega_{ij}]) \text{ Row}_2([\omega_{ij}]) \text{ Row}_3([\omega_{ij}]) \dots \text{Row}_n([\omega_{ij}])) = (\omega_{11} \ \omega_{12} \ \omega_{13} \ \dots \ \omega_{1n} \ \omega_{21} \ \omega_{22} \ \omega_{23} \ \dots \ \omega_{2n} \ \dots \ \omega_{n1} \ \omega_{n2} \ \omega_{n3} \ \dots \ \omega_{nn})$
 $= (\omega'_{11} \ \omega'_{12} \ \omega'_{13} \ \dots \ \omega'_{1n} \ \omega'_{(n+1)} \ \omega'_{(n+2)} \ \dots \ \omega'_{1(2n)} \ \dots \ \omega'_{1(n^2-n)} \ \dots \ \omega'_{1(n^2-2)} \ \omega'_{1(n^2-1)} \ \omega'_{1(n^2)})$.

Step 4: determine $\omega^*: M_{1 \times n} \rightarrow \mathbb{R}$ such that $\omega^* = \begin{cases} \omega'_{ij} & \text{when } c_{ij} \neq c'_{ij} \\ \omega'_{1(j+1)}, & \text{when } c'_{ij} = c_{ij} \end{cases}$ for $i, j = 1, 2, \dots, n$ and $jt = 1, 2, \dots, n^2$.

Figure 4.1 Precede operator's algorithm

After apply the conditions, the program then prints an output in two conditions:

- a) If the entry in matrix A is bigger than entry in matrix B, then it will display "matrix A is bigger than matrix B".
- b) Otherwise, it will proceed to next condition.

This shows the results on MATLAB after apply the condition based on precede operator's algorithm:

Command Window	Command Window
<pre>>> PrecedeOperator A = 4 4 15 3 B = 3 4 15 14 matrix A is bigger than matrix B fx >></pre>	<pre>>> PrecedeOperator A = 4 4 15 3 B = 4 10 15 14 matrix B is bigger than matrix A fx >></pre>

The output on command window if ($A_{11} > B_{11}$)

The output on command window if ($B_{12} > A_{12}$)

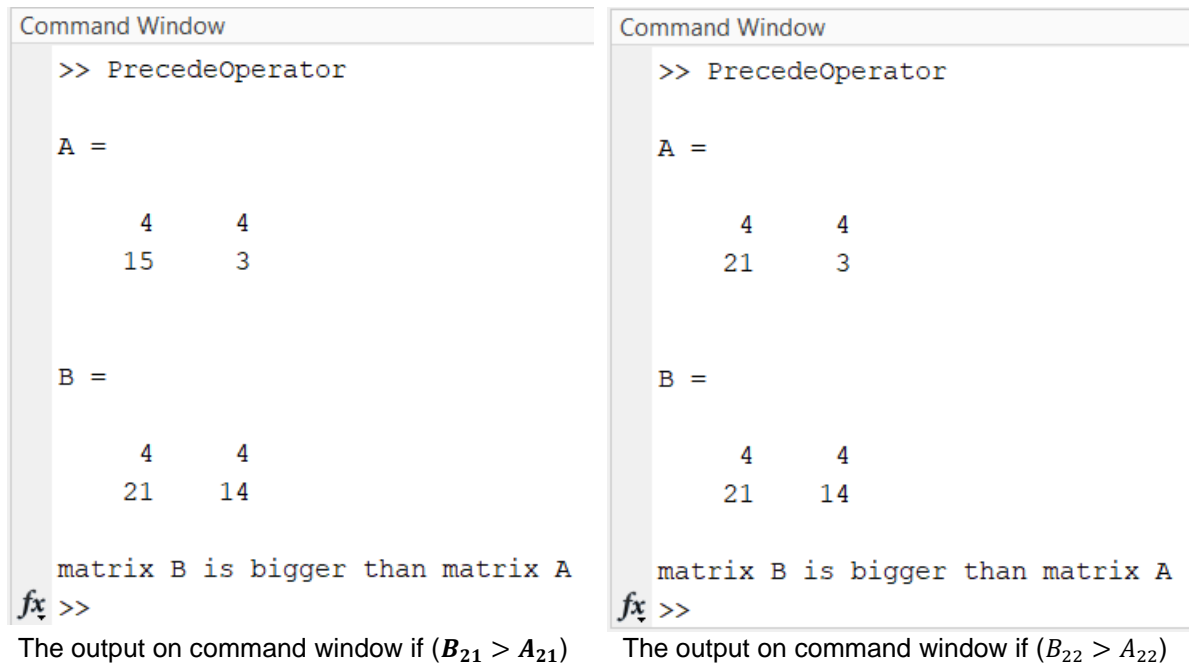


Figure 4.2 Results coding on the MATLAB

b) Proposition 1

`% Proposition 1`

`% Matrix A and Matrix B`

`A1 = 4;`

`A2 = 4;`

`A3 = 21;`

`A4 = 20;`

`B1 = 4;`

`B2 = 4;`

`B3 = 21;`

`B4 = 14;`

`A = [A1 A2; A3 A4]`

`B = [B1 B2; B3 B4]`

`% Transpose Matrix A and Matrix B`

`disp('This is Transpose Matrix A and Matrix B:')`

`A'`

`B'`

The output for proposition 1 on the MATLAB:

```

Command Window
>> Proposition1

A =

     4     4
    21    20

B =

     4     4
    21    14

This is Transpose Matrix A and Matrix B:

ans =

     4    21
     4    20

ans =

     4    21
     4    14

fx >>
    
```

Figure 4.3 the output for proposition 1 on MATLAB

c) Proposition 2

`% Proposition 2`

`% Matrix A and Matrix B`

`A1 = 4;`

`A2 = 4;`

`A3 = 21;`

`A4 = 20;`

`B1 = 4;`

`B2 = 4;`

`B3 = 21;`

`B4 = 14;`

`A = [A1 A2; A3 A4]`

`B = [B1 B2; B3 B4]`

`% Negative Matrix A and Matrix B`

`disp('This is Negative Matrix A and Matrix B:')`

`-A`

-B

The output for proposition 2 on the MATLAB:

```
Command Window

A =

     4     4
    21    20

B =

     4     4
    21    14

This is Negative Matrix A and Matrix B:

ans =

    -4    -4
   -21   -20

ans =

    -4    -4
   -21   -14

fx >>
```

Figure 4.4 the output for proposition 2 on MATLAB

d) Proposition 3

```
% Proposition 3
```

```
% Matrix A and Matrix B
```

```
A1 = 4;
```

```
A2 = 4;
```

```
A3 = 21;
```

```
A4 = 20;
```

```
B1 = 4;
```

```
B2 = 4;
```

```
B3 = 21;
```

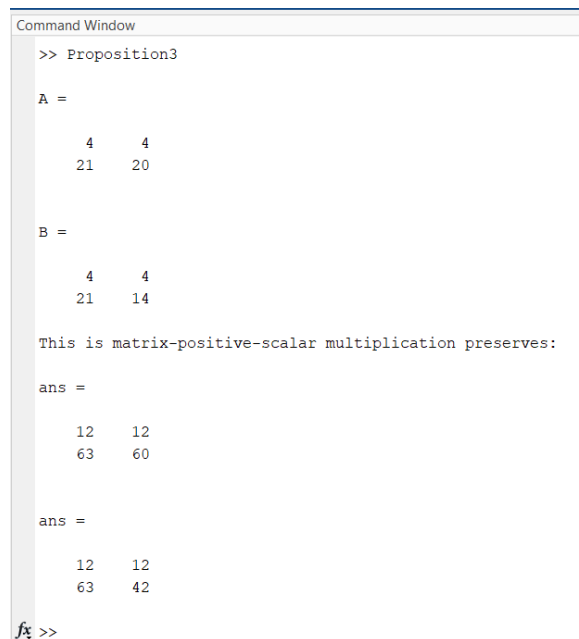
```
B4 = 14;
```

```
A = [A1 A2; A3 A4]
```

```
B = [B1 B2; B3 B4]
```

```
% declare k value  
k = 3;  
  
% Matrix A and Matrix B with k  
disp('This is matrix-positive-scalar multiplication preserves:')  
k*A  
k*B
```

The output for proposition 3 on the MATLAB:



```
Command Window  
>> Proposition3  
  
A =  
    4    4  
   21   20  
  
B =  
    4    4  
   21   14  
  
This is matrix-positive-scalar multiplication preserves:  
  
ans =  
    12    12  
    63    60  
  
ans =  
    12    12  
    63    42  
fx >>
```

Figure 4.5 the output for proposition 3 on MATLAB

e) Proposition 4

```
% Proposition 4  
% Matrix A and Matrix B
```

```
A1 = 4;  
A2 = 4;  
A3 = 21;  
A4 = 20;  
B1 = 4;  
B2 = 4;  
B3 = 21;  
B4 = 14;
```

```
A = [A1 A2; A3 A4]
B = [B1 B2; B3 B4]

% declaration of skew symmetric matrix A and B
disp('This are skew symmetric matrix A and B:')
-A'
-B'
```

The output for proposition 4 on the MATLAB:

```
Command Window
>> Proposition4

A =

     4     4
    21    20

B =

     4     4
    21    14

This are skew symmetric matrix A and B:

ans =

    -4   -21
    -4   -20

ans =

    -4   -21
    -4   -14

fx >>
```

Figure 4.6 the output for proposition 4 on MATLAB

5. Conclusion

Before constructing a programme, an algorithm is necessary. The 'if-else' statements, matrix arrays, input, output, display and other features are available in MATLAB. It can also run on any type of data, including integers, doubles and strings. The programme can validate the algorithm of the Precede Operator. When the code is implemented, the comparison is revealed.

Acknowledgement

Alhamdulillah, Thanks to God's grace, I finally managed to spend two semester of final year project at Universiti Teknologi Malaysia. First of all, I would like to thank all the parties whether directly or indirectly involved during my period of study. I would also like to thank my supervisor, Prof Dr Tahir bin Ahmad, for his guidance and encouragement during these two semester. Without his guidance, how could I possibly have completed this final year project so well. Since I underwent final year project at home, my family members and housemate always helped and supported me during that period. I want to thank my family and friends for supporting and staying with me to deal with all the stress and obstacles. I am

very grateful for the favours from everyone who contributed to the effort and assist in this final year project.

References

- [1] Ahmad Fuad, A. A., & Ahmad, T. (2021). Ordering of Transformed Recorded Electroencephalography (EEG) Signals by a Novel Precede Operator. *Journal of Mathematics*, 2021.
- [2] Aisyah, N., Nor, M., & Ahmad, T. (2021). An Investigation On The Possible Relation Of Krohn-Rhodes Theorem-Goldbach Conjecture-Jordan-Chevalley Decomposition Theorem. In *Undergraduate Project Proceedings*. UTM.
- [3] Lasefr, Z., Ayyalasomayajula, S. S. V. N. R., & Elleithy, K. (2017). Epilepsy seizure detection using EEG signals. *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference, UEMCON 2017, 2018-January*, 162–167.
- [4] Wanjohi, E. (2021). *Applications of Matrices to Cryptography*.
- [5] Applications of MATLAB in Mathematics. (n.d.). Retrieved May 12, 2022,