MATHS PROJECT

Convert Matrices into REF

MARCH 19, 2020

DHRUV MAHESHWARI
2019A7PS0020U

Acknowledgements

I would like to express my gratitude to Dr TK Dutta for providing me this opportunity wherein I could apply my theoretical knowledge into computer programming language.

I would also like to thank my parents for their help and support.

Index

Introduction

A *matrix* is a rectangular array of numbers or other mathematical objects for which operations such as addition and multiplication are defined.

The size of a matrix is defined by the number of rows and columns that it contains. A matrix with m rows and n columns is called an $m \times n$ matrix or m-by-n matrix, while m and n are called its *dimensions*.

For ex: $\begin{matrix} -3 & 2 & 9 \\ 2 & 5 & 4 \\ 0 & 2 & -7 \end{matrix}$ This matrix has a dimension of 3x3.

Few of the basic operations are listed below:

Operation	Definition	Example
Addition	The *sum* **A**+**B** of two *m*-by-*n* matrices **A** and **B** is calculated entrywise: $(\mathbf{A} + \mathbf{B})_{i,j} = \mathbf{A}_{i,j} + \mathbf{B}_{i,j}$, where $1 \le i \le m$ and $1 \le j \le n$.	$\begin{bmatrix} 1 & 3 & 1 \\ 1 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 5 \\ 7 & 5 & 0 \end{bmatrix} = \begin{bmatrix} 1+0 & 3+0 & 1+5 \\ 1+7 & 0+5 & 0+0 \end{bmatrix} = \begin{bmatrix} 1 & 3 & 6 \\ 8 & 5 & 0 \end{bmatrix}$
Scalar multiplication	The product *c***A** of a number *c* (also called a scalar in the parlance of abstract algebra) and a matrix **A** is computed by multiplying every entry of **A** by *c*: $(c\mathbf{A})_{i,j} = c \cdot \mathbf{A}_{i,j}$. This operation is called *scalar multiplication*, but its result is not named "scalar product" to avoid confusion, since "scalar product" is sometimes used as a synonym for "inner product".	$2 \cdot \begin{bmatrix} 1 & 8 & -3 \\ 4 & -2 & 5 \end{bmatrix} = \begin{bmatrix} 2\cdot 1 & 2\cdot 8 & 2\cdot -3 \\ 2\cdot 4 & 2\cdot -2 & 2\cdot 5 \end{bmatrix} = \begin{bmatrix} 2 & 16 & -6 \\ 8 & -4 & 10 \end{bmatrix}$
Transposition	The *transpose* of an *m*-by-*n* matrix **A** is the *n*-by-*m* matrix $\mathbf{A}^T$ (also denoted $\mathbf{A}^{tr}$ or ${}^t\mathbf{A}$) formed by turning rows into columns and vice versa: $(\mathbf{A}^T)_{i,j} = \mathbf{A}_{j,i}$.	$\begin{bmatrix} 1 & 2 & 3 \\ 0 & -6 & 7 \end{bmatrix}^T = \begin{bmatrix} 1 & 0 \\ 2 & -6 \\ 3 & 7 \end{bmatrix}$

Uses of matrices

Applications of matrices are found in most scientific fields.

- In every branch of physics, including classical mechanics, optics, electromagnetism, quantum mechanics, and quantum electrodynamics, they are used to study physical phenomena, such as the motion of rigid bodies.
- In computer graphics, they are used to manipulate 3D models and project them onto a 2-dimensional screen.
- In probability theory and statistics, stochastic matrices are used to describe sets of probabilities; for instance, they are used within the PageRank algorithm that ranks the pages in a Google search.
- Matrix calculus generalizes classical analytical notions such as derivatives and exponentials to higher dimensions.
- Matrices are used in economics to describe systems of economic relationships.

Row Echelon Form

A matrix is said to be in Row Echelon Form (REF) iff it satisfies following properties:

1. First non-zero element should be 1.
2. Each successive row has its 1^{st} non-zero element in a latter column.
3. All elements below the 1^{st} non-zero entry should be 0.
4. Zero rows appear at the bottom.

These conditions can be satisfied by using row operation such as

1. row addition, that is adding a row to another.
2. row multiplication, that is multiplying all entries of a row by a non-zero constant;
3. row switching, that is interchanging two rows of a matrix;

These operations are used in a number of ways, including solving linear equations and finding matrix inverses.

Code

The following programme has been written in C.

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();

float a[3][3],temp,row;
int i,j;

printf("Convert matrices into REF\nEnter elements (row wise): ");

for(i=0;i<3;i++)
        {for(j=0;j<3;j++)
                {scanf("%f", &a[i][j]);}
        }

        temp=a[0][0];
        a[0][0]/=a[0][0];
        a[0][1]=a[0][1]/temp;
        a[0][2]=a[0][2]/temp;

                row=a[1][0];
                a[1][0]=a[1][0]-(a[1][0]*a[0][0]);
                a[1][1]=a[1][1]-(row*a[0][1]);
                a[1][2]=a[1][2]-(row*a[0][2]);

                row=a[2][0];
                a[2][0]=a[2][0]-(a[2][0]*a[0][0]);
                a[2][1]=a[2][1]-(row*a[0][1]);
                a[2][2]=a[2][2]-(row*a[0][2]);
```

```
        temp=a[1][1];
        a[1][1]=a[1][1]/temp;
        a[1][2]=a[1][2]/temp;

                row=a[2][1];
                a[2][1]=a[2][1]-(row*a[1][1]);
                a[2][2]=a[2][2]-(row*a[1][2]);

        if (a[2][2]!=0)
        {a[2][2]/=a[2][2];}
else {a[2][2]=0;}

printf("\nREF of matrix is:\n\n");
for(i=0;i<3;i++)
        {for(j=0;j<3;j++)
                {printf("%f  ", a[i][j]);}
        printf("\n");
        }
getch();
}
```

Output

```
Compile Result

Convert matrices into REF
Enter elements (row wise): 9 3 1 4 2 1 4 -2 1

REF of matrix is:

1.000000   0.333333   0.111111
0.000000   1.000000   0.833333
0.000000   0.000000   1.000000

[Process completed (code 10) - press Enter]
```

Fig 1: REF for matrix circled in red

```
Compile Result

Convert matrices into REF
Enter elements (row wise): 4 5 6 7 8 9 1 2 3

REF of matrix is:

1.000000   1.250000   1.500000
0.000000   1.000000   2.000000
0.000000   0.000000   0.000000

[Process completed (code 10) - press Enter]
```

Fig 2: REF for matrix circled in red

Bibliography

- Class notes
- https://en.wikipedia.org/wiki/Matrix_(mathematics)