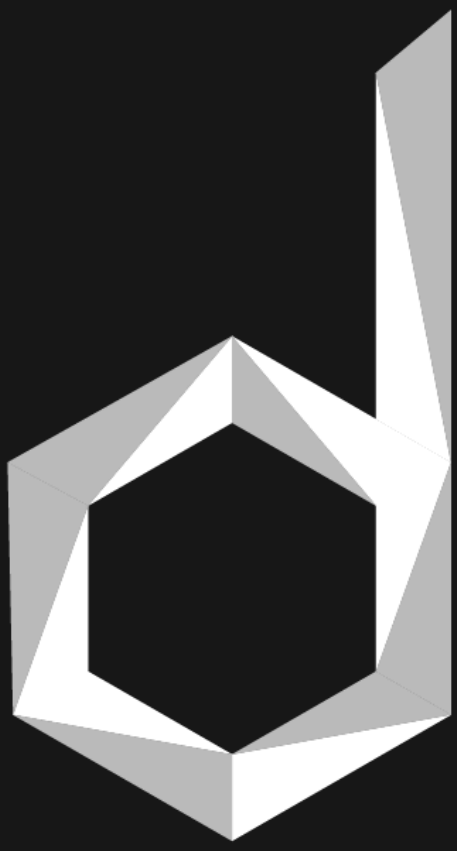
INGENIERÍA MECATRÓNICA



DI_CERO

DIEGO CERVANTES RODRÍGUEZ

INSTRUMENTACIÓN VIRTUAL

PYTHON 3.9.7, C# & LABVIEW

Multiplicación de Matrices

Contenido

Instrucciones – Multiplicación de Matrices:

El handbook of mathematics and computational science, Harris J. W. and Stocker H. Springer Verlag, contiene un algoritmo de la multiplicación de dos matrices. La siguiente información es extraída casi en su totalidad de dicho texto.

Use como ejemplo las siguientes operaciones de matrices:

$$\begin{pmatrix} \mathbf{1} & \mathbf{3} \\ \mathbf{2} & \mathbf{4} \end{pmatrix}\begin{pmatrix} \mathbf{2} & \mathbf{3} \\ \mathbf{0} & \mathbf{4} \end{pmatrix} = \begin{pmatrix} \mathbf{2} & \mathbf{15} \\ \mathbf{4} & \mathbf{22} \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{1} & \mathbf{2} & \mathbf{3} \\ \mathbf{4} & \mathbf{5} & \mathbf{6} \\ \mathbf{7} & \mathbf{8} & \mathbf{9} \end{pmatrix}\begin{pmatrix} \mathbf{1} \\ \mathbf{2} \\ \mathbf{3} \end{pmatrix} = \begin{pmatrix} \mathbf{14} \\ \mathbf{32} \\ \mathbf{50} \end{pmatrix}$$

1. El producto AB de la matriz A por la matriz B está definido sólo cuando el número de columnas de A son iguales al número de renglones de B.
2. Recuerde que, usualmente, AB ≠ BA
3. El producto matricial, o producto escalar de la matriz A y la matriz B produce la matriz C = AB, cuyos elementos son los productos escalares de los vectores renglón de A con los vectores columna de B. Si a_i denota el i−ésimo vector renglón de A y b_j el j−ésimo vector columna de B; entonces los elementos c_{ij} de C = AB están dados por la siguiente sumatoria:

$$c_{ij} = a_i * b_j = \sum_{k=1}^{l} a_{ik} * b_{kj}$$

   a. Donde los índices de los vectores renglón a_i son i = 1, …, m con m siendo el último renglón. En el caso de los vectores columna de b_j son j = 1, …, n, donde n corresponde a la última columna. Y los índices de los vectores columna de la matriz A corresponden a los índices de los vectores renglón de B y van de k = 1, …, l, donde l corresponde al número de columnas de la matriz A.
   b. Regla nemotécnica. Si l es el número de columnas que contiene la matriz A y l es el número de renglones de la matriz B, entonces la dimensión de la matriz C es (m n,), donde m es el número de renglones de A y n es el número de columnas de B, resumiendo se tiene:

$$c_{(m.n)} = A_{(m.l)} * B_{(l.n)}$$

Pseudocódigo.

1. Lectura de las matrices A y B. Donde la matriz A se expresa como $A_{(i.k)}$, la matriz B como $B_{(k.j)}$.
2. Obtención de m, l y n, donde m es el número de renglones de A, l es el número de columnas de A y n es el número de columnas de B.
3. Cálculo de los elementos $C_{(i.j)}$ de la matriz C, por medio de tres ciclos for anidados.

```
for (int i = 1; i < m; i++) {

    for (int j = 1; j < n; j++) {
```

```
                sum = 0;
                    for (int k = 1; k < n; k++) {
                        sum = sum + a[i, k]*b[k, j];
                    }
                c[i, j] = sum;
            }
        }
```

4. Mostrar la salida de C en pantalla.

Código Python – Visual Studio Code (Logo Azul):

```
# -*- coding: utf-8 -*-

#Comentario de una sola linea con el simbolo #, en Python para nada se deben poner acentos sino el programa
#puede fallar o imprimir raro en consola, la siguiente línea de código es para que no tenga error, pero aún
#así al poner un acento, la palabra saldrá rara en consola.
#La línea debe ponerse tal cual como aparece y justo al inicio.

#Con cls se borra el contenido de la consola y con el botón superior izquierdo de Play se corre el programa.

#CÓDIGO PARA MULTIPLICAR DOS MATRICES:

#El producto AB de la matriz A por la matriz B está definido sólo cuando el número de columnas de A son iguales al
#número de renglones de B.
#• En multiplicación de matrices el orden del producto si altera el resultado, AB ≠ BA.
#• El producto de la matriz A y la matriz B produce la matriz C = AB, cuyos elementos son el resultado de la
#multiplicación de los vectores fila de A por los vectores columna de B. Si ai denota el i-ésimovector
#fila de A y bj el j-ésimovector columna de B; entonces los elementos cij de C = AB están dados por:
#cij = ai*bj = Σai*bj

#Declaración e iniciación de las matrices en un tipo de dato de Python llamado lista, siguiendo la nomenclatura
#descrita a continuación, tomando en cuenta que el número de elementos de las filas debe ser el mismo:
#Matriz_lista = [[fila1],[fila2],...,[fila_n]], FILA X COLUMNA
#vector_renglon = [1,2,3]
#vector_columna = [[1],[2],[3]]
A = [[1,3],[2,4]] #Matriz A (tipo de dato lista), 2X2
B = [[2,3],[0,4]] #Matriz B (tipo de dato lista), 2X2
#Se debe declarar la matriz vacía que almacenará el resultado de la multiplicación:
C = [[0,0],[0,0]] #Matriz C = A*B, 2X2
```

```
#Obtención de las dimensiones de la matriz para accesar a las posiciones de sus elementos y realizar la multiplicación,
#ya que debe coincidir el tamaño de las columnas de la matriz A con el tamaño de las filas de la matriz B para que se
#pueda hacer su producto.
#len(): Este método sirve para ver el tamaño de una matriz, que en Python a ese tipo de dato se le llama lista.
m = len(A) #Número de filas o renglones de la matriz A
#Si una lista tiene posiciones con listas internas, como lo es en el caso de las matrices, lo que se hace para saber
#el tamaño de las listas internas es que se indica que se quiere saber el tamaño de la posición de una matriz, como
#todas las filas deben tener el mismo número de elementos, no importa que posición se elija.
n = len(B[0]) #Número de las columnas de la matriz B
#El número de columnas de la matriz A debe ser igual al número de filas de la mariz B
l = len(A[0]) #l = len(A[0]) = len(B)

#BUCLE FOR: En Python este no utiliza llaves de apertura o cierre tampoco, solamente se utilizan dos puntos para indicar
#el inicio del bucle. Además después de la palabra resevrada "for" se declara una variable local que solo se usará en el
#bucle y será la que cuente hasta que este llegue al extremo indicado dentro del paréntesis de la palabra reservada
#range() para terminar el bucle, dentro del paréntesis de la palabra reservada "in range()" se coloca el inicio del
#conteo y el final para indicar cuantas veces se ejecutará el bucle.
for i in range(0, m): #Lectura de filas de la matriz A
    for j in range(0, n): #Lectura de columnas de la matriz B
        sum = 0 #Inicialización de la variable que almacenará el valor de cada elemento de la matriz resultante C
        for k in range(0, l): #Lectura de las columnas de la matriz A y filas de la matriz B
            sum = sum +  A[i][k]*B[k][j] #Llenado de los elementos de la matriz C: cij = ai*bj = Σai*bj
        #for k
        C[i][j] = sum #Asignación del valor de cada elemento de la matriz resultante C
    #for j
#for i

#Impresión de la matriz C
print("El resultado de la multiplicación de las matrices A: ", A, " y B: ", B, " es igual a: ", C)

#BIBLIOTECA NUMPY: Esta librería de Python importa mucho material para poder hacer operaciones entre matrices sin
#necesidad de hacerlo con bucles, para importar librerías primero esta se debe descargar con la consola de Windows
#CMD, Powershell o GitBash, luego se usa la palabra reservada import, el nombre de la librería, la palabra reservada
#as y el nuevo nombre con el que se quiere usar los métodos de la librería dentro del código.
import numpy as np

#Ejemplo de multiplicación de matrices con 3 filas y 3 columnas, osea FXC = 3X3
#Las matrices declaradas con el método array de la librería numpy no son listas, son totalmente otro tipo de valor
#perteneciente a la librería numpy, llamado numpy array y para declarar arrays de este tipo de dato se usa la siguiente
#nomenclatura:
#Matriz_numpy_array = numpy.array([[fila1],[fila2],...,[fila_n]]), FILA X COLUMNA
#La diferencia con la matriz declarada en forma de lista es la siguiente:
#Matriz_lista = [[fila1],[fila2],...,[fila_n]], FILA X COLUMNA
D = np.array([[1,2,3],[4,5,6],[7,8,9]]) #Matriz D (tipo de dato numpy array), 3X3
```

```python
E = np.array([[1],[2],[3]]) #Matriz E (tipo de dato numpy array), 3X1
#Recordemos que para que se pueda realizar la multiplicación entre matrices, el número de columnas de la primera matriz
#y el número de filas de la segunda matriz debe ser el mismo y el resultado será del tamaño que sobre de las matrices
#originales, cuando se está usando la librería numpy no es necesario declarar la matriz vacía que almacenará el resultado
#de la multiplicación.

#numpy.dot: Este método de la librería numpy sirve para realizar el producto de dos matrices sin necesidad de usar un
#bucle, en su primer parámetro se pone la primera matriz que se quiere multiplicar y en el segundo la segunda matriz,
#recordemos que en matrices el orden de los elementos si afecta al resultado del producto
F = np.dot(D,E)

#Impresión de la matriz F, cuando se imprima el resultado de matrices hechas con la librería numpy se debe dar saltos
#de línea entre el texto y donde se muestre el valor de las matrices porque por default estas dan un salto de línea
print("El resultado de la multiplicación de las matrices D: \n", D, "\nY E: \n", E, "\nEs igual a: \n", F)
```

Resultado del Código Python

```
PS C:\Users\diego\OneDrive\Documents\Aprendiendo\Python\1.-Instrumentación Virtual> & D:/Users/diego/AppData/Local/Programs/Python/Python39/python.exe
"c:/Users/diego/OneDrive/Documents/Aprendiendo/Python/1.-Instrumentación Virtual/5.-Multiplicación de Matrices.py"
El resultado de la multiplicación de las matrices A:  [[1, 3], [2, 4]]  y B:  [[2, 3], [0, 4]]  es igual a:  [[2, 15], [4, 22]]
El resultado de la multiplicación de las matrices D:
 [[1 2 3]
 [4 5 6]
 [7 8 9]]
Y E:
 [[1]
 [2]
 [3]]
Es igual a:
 [[14]
 [32]
 [50]]
PS C:\Users\diego\OneDrive\Documents\Aprendiendo\Python\1.-Instrumentación Virtual>
```

Código C# (.Net Framework) – Visual Studio (Logo Morado):

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace _4.__Multiplicación_de_matrices
{
    class Program
    {
        //El código en C# se corre presionando CTRL + F5
        //CÓDIGO PARA MULTIPLICAR DOS MATRICES:
        static void Main(string[] args)
        {
            /*Arrays: Cuando se declara un array en C# se debe indicar su tipo de dato, el número de elementos
            que vaya a contener y su nombre, como el array declarado es un objeto de la clase Array, se debe
            crear una instancia por medio de la palabra reservada new, pero cuando no se sabe el tamaño que
            tendrá el array, simplemente se pone una coma dentro de los corchetes que van después del tipo de
            dato que almacenará el array.*/
            /*tipo_de_dato_array[,] Nombre_Matriz = new tipo_de_dato_array[,] {{fila1},{fila2},...,{fila_n}},
            FILA X COLUMNA.
            tipo_de_dato_array[,] vector_renglon = new tipo_de_dato_array[,] {{1,2,3}}
            tipo_de_dato_array[,] vector_columna = new tipo_de_dato_array[,] {{1},{2},{3}}*/
            //Declaración e iniciación de las matrices A y B
            int[,] A = new int[,] { { 1, 2, 3 }, { 4, 5, 6 }, { 7, 8, 9 } };//Matriz A, 3X3
            int[,] B = new int[,] { { 1 }, { 2 }, { 3 } };//Matriz B, 3X1
            /*En C# para poder imprimir el contenido de un vector o matriz se debe hacer uso de Bucles for:*/
            Console.WriteLine("El resultado de la multiplicación de las matrices A:");
```

```
    for (int i = 0; i < A.GetLength(0); i++)
    {
        for (int j = 0; j < A.GetLength(1); j++)
        {
            Console.Write(A[i, j] + " ");
        }
        Console.WriteLine();
    }
    /*En C# para poder imprimir el contenido de un vector o matriz se debe hacer uso de Bucles for:*/
    Console.WriteLine("\n Y B: ");
    for (int i = 0; i < B.GetLength(0); i++)
    {
        for (int j = 0; j < B.GetLength(1); j++)
        {
            Console.Write(B[i, j] + " ");
        }
        Console.WriteLine();
    }
    Console.WriteLine("\nEs igual a: ");
    /*El producto AB de la matriz A por la matriz B está definido sólo cuando el número de columnas de A
    son iguales al número de renglones de B y el resultado tendrá el número de filas de A y el número de
    columnas de B.*/
    /*Obtención de las dimensiones de la matriz para accesar a las posiciones de sus elementos y realizar
    la multiplicación, ya que debe coincidir el tamaño de las columnas de la matriz A con el tamaño de las
    filas de la matriz B para que se pueda hacer su producto.*/
    /*GetLength(): Este método sirve para ver el tamaño de un array, en su parámetro se le debe pasar la
    dimensión del array que se quiere ver su tamaño*/
    //Número de filas o renglones de la matriz A, la dimensión es cero porque es la de hasta afuera.
    int m = A.GetLength(0);
    /*Si un array tiene posiciones con arrays internos, como lo es en el caso de las matrices, lo que se
    hace para saber el tamaño de las listas internas es que se indica al método GetLength() que se quiere
    saber el tamaño de la dimensión 1 del array, así se meterá a los arrays internos y verá su tamaño.*/
    //Número de columnas de la matriz B, la dimensión es uno porque es la de un array interno.
    int n = B.GetLength(1);
    /*El número de columnas de la matriz A debe ser igual al número de filas de la mariz B para que se pueda
    hacer la multiplicación.*/
    //l = A.GetLength(1) = B.GetLength(0)
    int l = A.GetLength(1);

    //Se debe declarar la matriz vacía que almacenará el resultado de la multiplicación:
    int[,] C = new int[m, n];//Matriz C = A*B, 3X1

    /*Declaración de la variable que almacenará el valor de cada elemento de la matriz resultante C, esta se
    debe declarar fuera del siguiente bucle for sino el resultado de la operación será erróneo.*/
    int sum;

    /*BUCLE FOR: El bucle for en C# se declara como en la mayoría de los lenguajes de programación, indicando
    una variable local de tipo primitivo número entero, hasta donde va a llegar el conteo y el paso con el
    que irá contando.*/
    for (int i = 0; i < m; i++)//Bucle que lee el contenido de las filas de la matriz A
    {
        for (int j = 0; j < n; j++)//Bucle que lee el contenido de las columnas de la matriz B
        {
            //Inicialización de la variable que almacenará el valor de cada elemento de la matriz resultante C
            sum = 0;
            for (int k = 0; k < l; k++)//Lee el contenido de las columnas de la matriz A y filas de la matriz B
            {
                //Cálculo de cada uno de los elementos de la matriz C = A*B
                sum = sum + A[i, k] * B[k, j];
            }
            //Asignación de cada uno de los elementos a su respectiva posición en la matriz C
            C[i, j] = sum;
        }
    }

    //Despliegue de la matriz C en pantalla
    for (int p = 0; p < m; p++)//Bucle que lee el contenido de las filas de la matriz A
    {
        for (int q = 0; q < n; q++)//Bucle que lee el contenido de las columnas de la matriz B
        {
            /*Método para imprimir en consola, usando la clase Console y su método Write para imprimir en
            consola y después no ejecutar un salto de línea.*/
            Console.Write(C[p, q]); //Impresión en pantalla de todos los valores de la matriz resultante C
            Console.Write(" ");
        }
        /*Impresión en consola con un salto de línea para diferenciar los valores de la matriz C.*/
        Console.WriteLine();
    }
}//Método main: Desde el método main se ejecutan todas las partes del proyecto
```

```
    }//Clase del programa
}//Espacio de nombres: En esta parte del código se pueden declarar más de una clase que conforme el proyecto
```

Resultado del Código C#

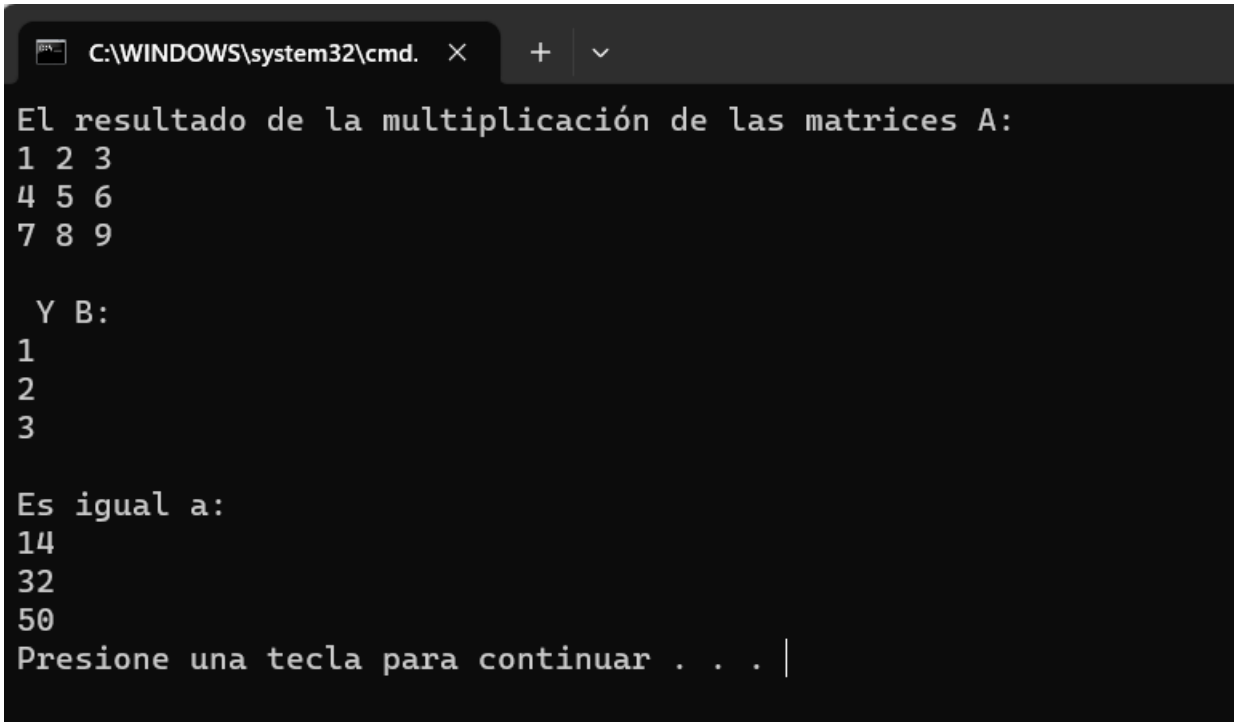


Diagrama LabVIEW: