# PRACTICA 0. INTRODUCCIÓN A MATLAB

**Javier José Guzmán Rubio y Lucas Álvarez**

## PARTE 1

### Ejercicio 1. Matrices y vectores.

1. **Cree la siguiente matriz A y el vector v:**

Los valores de la matriz los introducimos entre corchetes, y a través del punto y coma separamos las filas.

Para el vector, separamos cada fila con punto y coma.

Captura de pantalla de un celular

Descripción generada automáticamente

Resultado:

Imagen que contiene Gráfico

Descripción generada automáticamente

1. **Obtenga y visualice una matriz B concatenando la matriz A y el vector v.**

Para concatenar la matriz A y el vector v se hace a través de B=[A v]

Imagen que contiene Texto

Descripción generada automáticamente

Resultado:

Tabla

Descripción generada automáticamente con confianza media

1. **Obtenga y visualice un vector fila resultado de concatenar las filas de la matriz B.**

A través de un bucle for i=1:nfilas que va desde 1 hasta el numero de filas que tenga la matriz, concatenamos el vector fila con la fila correspondiente a i de la matriz.

Texto

Descripción generada automáticamente

Resultado:



1. **Obtenga y visualice un vector columna resultado de concatenar las columnas de la matriz B.**

Al igual que en el ejercicio anterior, a través de un bucle for i=1:ncolumnas, se recorre cada una de las columnas de nuestra matriz. La columna recorrida queda almacenada en un vector columna=B(:,i); Para después concatenarlo con el vector vColumnas que será el que contendrá las columnas de nuestra matriz.

Texto

Descripción generada automáticamente

Resultado:

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

### Ejercicio 2. Matrices y vectores.

1. **El script ha de generar una matriz, cuadrada y aleatoria de tamaño indicado por el usuario. En la línea de comandos se ha de visualizar el mensaje: “Indique el tamaño de la matriz”.**

En primer lugar, se debe pedir por pantalla el tamaño de la matriz, y una vez indicado a través de rand, generamos una matriz con el tamaño especificado.



Resultado:



1. **A partir de la matriz construida, el script deberá calcular y presentar por pantalla los siguientes datos:**
2. **Matriz generada.**

Al igual que hemos hecho anteriormente mostraremos la matriz mediante disp(A);



Resultado para n=4:

Texto

Descripción generada automáticamente

1. **Una segunda matriz formada por las columnas impares de la matriz inicial.**

Creamos una nueva matriz A que mediante B=A(:,1:2:n); hacemos que leyendo todas las columnas desde la 1 de 2 en 2 (para que sean las impares) hasta n que es el número de columnas, sea igual a B.

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

Resultado:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

1. **El valor de los elementos de la diagonal de la matriz generada.**

Para el valor de los elementos de la diagonal lo hemos hecho simplemente con diag(A).

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

Resultado:

Texto

Descripción generada automáticamente con confianza baja

1. **Valor máximo, mínimo, medio y varianza de cada fila. Estos valores se han de representar gráficamente, indicando en el eje de abscisas el número de fila.**

Hemos creado un vector para el máximo, mínimo, media y varianza. A través de un bucle for i=1:n, se recorre una a una las filas de la matriz para almacenar en el vector fila, la fila correspondiente. Mediante las funciones max, min, mean y var obtenemos el máximo, mínimo, media y varianza de la fila y a continuación lo concatenamos con el vector correspondiente.

Texto, Carta

Descripción generada automáticamente

Para la representación gráfica hemos usado la función subplot.

Texto

Descripción generada automáticamente

Resultado:

Gráfico

Descripción generada automáticamente

### Ejercicio 3. Matrices y vectores.

1. **Solicite al usuario las dimensiones de las matrices en formato [filas cols], (si se introduce un único número, la matriz será cuadrada).**

Lo solicitamos como hicimos en el ejercicio anterior.



Resultado:





Resultado:



1. **Genere dos matrices (A y B) de las dimensiones elegidas. Para rellenar las matrices, escriba una función en Matlab (en un fichero diferente) que reciba como parámetro las dimensiones deseadas [filas cols], y devuelva la matriz rellena.**

Para esto, en un fichero diferente hemos creado la función IntroducirMatriz, que recibe como parámetro el tamaño de la matriz y nos devuelve la matriz.





Resultado, para los tamaños indicados en el apartado anterior:

Tabla

Descripción generada automáticamente con confianza media

1. **La función debe pedir datos al usuario para cada posición de la matriz. En caso de que el usuario escriba ‘r’, la matriz se rellenará de valores aleatorios.**

Para indicar si el usuario desea que los valores sean aleatorios o no, nos mostrara el siguiente mensaje: 

Con un switch n



Desarrollamos los posibles casos:

Texto

Descripción generada automáticamente con confianza media

Resultado para cuando el usuario pulsa r:

Tabla

Descripción generada automáticamente con confianza media

Si el usuario escribe una r, se rellenará la matriz de valores aleatorios, mediante la función rand.

Texto

Descripción generada automáticamente con confianza media

En caso de que el usuario pulse n, deberá indicar los valores de la matriz.

El primer bucle for representa las filas mientras que el segundo las columnas.

El resultado para cuando el usuario pulsa n:



Nos va pidiendo el valor para cada posición de la matriz.



Una vez solicitados todos los valores, el resultado es:

Texto

Descripción generada automáticamente con confianza media

1. **Calcule y muestre por pantalla:**
2. **Las matrices generadas A y B.**

Como ya hemos hecho anteriormente, los mostraremos mediante la función disp().

Texto

Descripción generada automáticamente

Resultado:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

1. **La transpuesta e inversa de cada una de las matrices.**

Para la transpuesta, la hemos calculado con A´ y B´.

Imagen de la pantalla de un celular

Descripción generada automáticamente con confianza media

Resultado:

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

Para la inversa, debemos comprobar si es una matriz cuadrada ya que si no lo es, no tiene inversa. Para hacer la inversa, lo hacemos mediante la función inv().

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Para la matriz anterior, no hay inversa ya que no es cuadrada, por ello el resultado para este caso es el siguiente:

Texto

Descripción generada automáticamente

En caso de si ser una matriz cuadrada, el resultado para las dos siguientes matrices seria:

Tabla

Descripción generada automáticamente

Texto, Tabla

Descripción generada automáticamente

1. **El valor del determinante y el rango de cada una de las matrices.**

Para calcular el rango, lo hemos hecho mediante la función rank().

Para el determinante, comprobamos si la matriz es cuadrada, y en caso de que lo sea lo calculamos con la función det().

Texto

Descripción generada automáticamente

Resultado para la anterior matriz cuadrada:

Texto

Descripción generada automáticamente

1. **El producto de A y B (matricial y elemento a elemento).**

Para el producto matricial y elemento a elemento, debemos comprobar si el número de columnas de A es igual al número de filas de B.

En caso de que así sea, el productor matricial lo calculamos haciendo A\*B y el producto elemento a elemento, A.\*B

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

El resultado para las matrices anteriores es:

Texto

Descripción generada automáticamente

1. **Un vector fila obtenido concatenando la primera fila de cada una de las matrices.**

Para ello concatenamos A(1,:) y B(1,:).

Imagen que contiene Texto

Descripción generada automáticamente

Resultado:



1. **Un vector columna obtenido concatenando la primera columna de cada una de las matrices.**

Lo hacemos igual que el ejercicio anterior pero con las columnas. Concatenando A(:,1) y B(:,1).



Resultado:

Texto

Descripción generada automáticamente con confianza media

### Ejercicio 4. Tiempo de cómputo y representación gráfica.

**Realice un script en Matlab que permita obtener y representar el tiempo consumido para el cálculo del rango y el determinante de una matriz en función de su tamaño (entre 1x1 y 25x25). Tenga en cuenta que:**

* **La matriz se rellenará con valores aleatorios.**
* **El tiempo necesario para cada operación debe obtenerse por separado.**
* **Los tiempos de procesamiento para el cálculo del rango y del determinante se representarán en la misma gráfica, utilizando para ello diferentes colores.**
* **Deben añadirse etiquetas a los ejes, y una leyenda indicando que representa cada línea.**

Lo vamos a hacer para 250 matrices, para así poder apreciar mejor en la grafica la diferencia de tiempo en el cálculo de cada una de las matrices.

En un bucle for vamos a ir generando cada una de las matrices desde la matriz de 1x1 hasta la de 250x250.

Para cada matriz, en primer lugar calcularemos el determinante mediante la función det().

Para capturar el tiempo que tarda en calcular el determinante, ponemos tic para comenzar a capturar el tiempo que tarda. Una vez calculado el determinante, guardaremos dentro de un vector, tocDeterminante, el tiempo transcurrido para después concatenarlo con el vector en el que estarán los diferentes tiempos de cada una de las matrices.

Para el tiempo que tarda en calcular el rango de cada una de las matrices, haremos lo mismo que con el determinante. Almacenaremos en tiempoRango, el tiempo de cada una de las matrices.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

A continuación, debemos mostrar en una grafica el tiempo que tarda en calcular tanto el determinante como el rango de cada una de las 250matrices generadas.

Para ello, haremos un plot de los vectores, tiempoDeterminante y tiempoRango.

Texto

Descripción generada automáticamente

Resultado:

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

Como podemos apreciar, cuanto mas grande es la matriz, mas tiempo tarda en calcular tanto el rango como el determinante.

También podernos fijarnos en que tarda mas tiempo en calcular el rango que la matriz. El crecimiento de la pendiente del rango es bastante mayor al del determinante.

### Ejercicio 5. Representación gráfica en 3D.

**Realice un script en Matlab que dibuje sobre el área −5 ≤ x, y ≤ 5 la superficie, la superficie en forma de malla y el contorno de la función:**

****

* **En la misma figura dibuje en la parte superior y centrada la gráfica de la superficie, en la parte inferior izquierda la gráfica de la superficie en forma de malla y en la parte inferior derecha la gráfica del contorno. Además, añada la barra de color al contorno.**
* **Deben añadirse etiquetas a los ejes, y un título a cada gráfica.**

En primer lugar, creamos un vector x dividido en 20 valores desde -5 a 5, y hacemos los mismo para un vector y.

A partir de estos dos vectores, queremos obtener una matriz X y una matriz Y.

Es por ello que utilizamos la función, meshgrid(), que necesita como parámetro el vector\_x y el vector\_y para poder generar la matriz X e Y (20x20).

A partir de estas dos matrices, se generaría una matriz Z con los valores de las matrices X e Y.

Texto, Carta

Descripción generada automáticamente

Una vez tenemos las tres matrices, queremos representar gráficamente la superficie, la malla y el contorno.

Para la superficie utilizamos la función surf(X,Y,Z), para la malla mesh(X,Y,Z) y para el contorno contourf(X,Y,Z).

Todo estos lo queremos hacer dentro de una misma ventana y es por ello que utilizamos subplot().

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Resultado:

Gráfico, Gráfico de superficie

Descripción generada automáticamente

### Ejercicio 6. Sistemas lineales.

**Dados los siguientes sistemas lineales de 10 ecuaciones con 4 incógnitas (x1, x2, x3, x4)**

**Texto, Tabla

Descripción generada automáticamente con confianza media**

1. **Exprese el sistema de forma matricial en Matlab. Para ello, cree las matrices A y b.**

Lo expresaremos de la siguiente forma:

Texto

Descripción generada automáticamente

1. **Escriba un script en que permita:**
2. **Obtener el número de condición de la matriz A respecto a la inversión.**

Para esto vamos a utilizar la función cond(A).

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

Resultado:



1. **Resolver el sistema de ecuaciones para obtener la matriz x = [x1, x2, x3, x4].**

Para resolver el sistema utilizamos la función linsolve().



Resultado:



1. **Añadir ruido a la matriz b, sumándole un vector aleatorio de media 0 y desviación 1, y resuelva el sistema de ecuaciones resultante.**

Para aplicar ruido, hemos creado un vector aleatorio de desviación 0 y media 1, como nos indica el enunciado.



Para crear una matriz con ruidos, le sumaremos a cada uno de los valores de la matriz de términos independientes el vector ruido. Estas sumas las almacenamos en un vector BTiR.

Para obtener la solución de la matriz B con ruidos, utilizamos la función linsolve(B, BTiR´).

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Resultado:



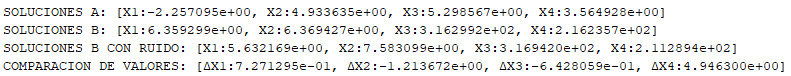
1. **Mostrar el resultado (con y sin ruido añadido) por pantalla.**

Mostramos los resultados.

Texto

Descripción generada automáticamente

Resultado:



### Ejercicio 7. Polinomios.

**Realice una función de Matlab que permita obtener las raíces de un producto de polinomios y las clasifique en reales y complejas. Para ello ha de realizar los siguientes pasos:**

**Las entradas y salidas de la función son las que se especifican, según la siguiente sintaxis:**

Texto

Descripción generada automáticamente con confianza media

1. **Recoge los arrays con los que se crean los polinomios.**

Para recoger los arrays, pediremos al usuario por pantalla que introduzca el polinomio 1 que se guardará en poli1 y el polinomio 2 que será almacenado en poli2.



1. **Solicita si la solución se aplica a uno de los polinomios o al producto: poli\_1, poli\_2, prod\_poli.**

Para esto, pediremos al usuario por pantalla para que polinomio desea la solución.

La respuesta del usuario quedará almacenada en opción.



1. **Devuelve las raíces del polinomio indicado y su clasificación (nº raíces reales y nº raíces complejas.**

A partir de un switch con la variable opción que he comentado en el apartado anterior, accedemos al caso que haya sido seleccionado por el usuario.

Texto

Descripción generada automáticamente

Para conocer tanto la solución como el número de raíces reales y complejas, llamamos a la función raíces.

Texto

Descripción generada automáticamente

En esta función, en primer lugar, inicializamos un vector reales y otro complejas.

Con un bucle for recorremos el vector soluciones en el que están todas las raíces del polinomio y vamos comprobando si la raíz es real o compleja. En caso de que la raíz sea real, lo concatenamos con el vector reales, en el caso de que sea una raíz compleja, concatenamos la raíz con el vector de complejas.

Por último, en el script del ejercicio 7, mostramos las raíces del polinomio, el número de raíces reales y el número de raíces complejas.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Resultado para el producto del polinomio 1 y 2 del enunciado:

Texto

Descripción generada automáticamente

1. **Representa en el plano complejo la ubicación de las raíces obtenidas.**

Con la función plot(), representamos gráficamente las raíces reales y complejas.

En el eje de las x estará el número de aparición de la raíz real y compleja, mientras que en el eje de las y, estará el valor de las raices.

Texto

Descripción generada automáticamente

Resultado para el producto del polinomio 1 y del polinomio 2 del enunciado:

Gráfico

Descripción generada automáticamente