PRÁCTICA 0. INTRODUCCIÓN A MATLAB

SISTEMAS DE CONTROL INTELIGENTE

Álvaro Zamorano Ortega, Gabriel López Cuenca

PARTE 1

EJERCICIO 1. Matrices y vectores.

1.Cree la matriz A y el vector v

Los valores de la matriz los insertamos entre corchetes, y por medio del punto y coma separamos cada fila de la matriz.

Para el vector columna, el valor de cada fila lo separamos por punto y coma.

```
A = [1 2; 3 4; 5 6; 7 8];
v = [14;16;18;20];
```

2. Obtenga la matriz B concatenando la matriz A y el vector v.

Para concatenar la matriz con el vector, y para que este se añada como una nueva columna se hace mediante B = [A v];

```
1     2    14
3     4    16
5     6    18
7     8    20
```

3. Obtenga y visualice un vector fila concatenando las filas de la matriz B.

De esta forma concatenamos las filas de B en una sola fila, f = [B(1,:) B(2,:) B(3,:) B(4,:)];

```
1     2    14     3     4    16     5     6    18     7     8    20
```

4. Obtenga y visualice un vector columna concatenando las columnas de la matriz B.

En esta ocasión para tener todas las columnas como una única se hace mediante c = [B(:,1); B(:,2); B(:,3)];

```
 1
 3
 5
 7
 2
 4
 6
 8
14
16
18
20
```

EJERCICIO 2. Matrices y vectores.

1. Generar matriz cuadrada y aleatoria indicada por el usuario.

Con la función rand rellenamos la matriz cuadrada cuyas dimensiones son las introducidas por el usuario gracias a la función input.

La matriz obtenida para una dimensión de 3 es:

0.9649	0.9572	0.1419
0.1576	0.4854	0.4218
0.9706	0.8003	0.9157

2. a) Matriz generada.

Con la función disp mostramos los valores de la matriz A.

b) Matriz formada por columnas impares de la matriz inicial.

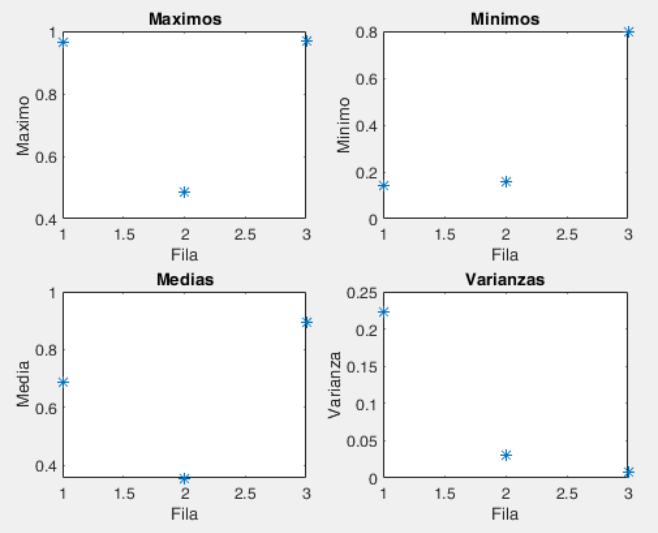
Por medio de un bucle for, en donde se suma la variable i de 2 en 2 hasta la longitud n de la matriz, vamos rellenando la matriz B con los valores de las columnas impares de la matriz A.

c) Valor de los elementos de la diagonal de la matriz generada.

Se obtiene con la función diag la cual nos devuelve un vector columna con los elementos de la diagonal pertenecientes a la matriz que se le pase como parámetro.

d) Valor máximo, mínimo, medio y varianza de cada fila. Estos valores se han de representar gráficamente.

Para obtener estos valores se han usado las funciones correspondientes aplicadas a cada una de las filas de la matriz y se han almacenado en vectores para su posterior representación. Para llevar a cabo dicha representación se ha usado la función subplot para representar varias gráficas dentro de una misma ventana.



EJERCICIO 3. Matrices y vectores.

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

La solicitud de dimensiones la realizamos por medio de la función input.

2 y 3. Rellenar dos matrices (A y B) de las dimensiones elegidas. Para ello, realice una función que reciba las dimensiones y devuelva la matriz rellena. La función debe pedir al usuario los

datos para cada posición de la matriz. En caso de que el usuario escriba 'r', la matriz se rellenará de valores aleatorios.

En primer lugar, pedimos al usuario si quiere que se rellene aleatoria o manualmente la matriz. Si lo quiere aleatoriamente, la matriz se rellenará aleatoriamente a través de la función rand. En caso contrario, pedimos al usuario el valor de cada posición recorriendo la matriz por medio de un bucle anidado, en el que recorremos filas y columnas. Finalmente, la función devuelve la matriz rellena con las dimensiones solicitadas anteriormente.

```
Indique el tamaño de la matriz 1 --> [2 3]
r para aleatorios, n para rellenar: r
Indique el tamaño de la matriz 2 --> [2 2]
r para aleatorios, n para rellenar: n
Comience a introducir valores
Posicion 1, 1: 2
Posicion 1, 2: 3
Posicion 2, 1: 4
Posicion 2, 2: 5
```

4. Calcular: ejemplo realizados con matrices aleatorias con dimensiones: A[3x3], B[3x2]

- **Matrices generadas.**
  Las mostramos por medio de la función disp.
- **Transpuesta e inversa de cada matriz.**
  Para el cálculo de la transpuesta, cambiamos filas por columnas. Para la inversa (en caso de ser la matriz cuadrada) utilizamos la función inv.

```
Inversas
   -1.9958    3.0630   -1.1690
    2.8839   -2.6919    0.6987
   -0.0291   -0.1320    1.1282

La matriz B no tiene inversa, no es cuadrada
```

- **Determinante y rango de cada matriz.**
  Para el cálculo del determinante (en caso de ser la matriz cuadrada) y del rango hacemos uso de las funciones det y rank, respectivamente.

```
Determinantes
   -0.2767

La matriz B no tiene determinante, no es cuadrada

Rangos
     3

     2
```

- **El producto de A y B (matricial y elemento a elemento).**
  Matricial: para su cálculo necesitamos que el número de columnas de A y filas de B sea el mismo. Tras considerar esto, realizamos el producto directamente (A*B).
  Elemento a elemento: en este caso es necesario que las dimensiones de ambas matrices sean iguales, el producto se hace mediante A.*B.

```
Producto matricial
    1.2004    1.4460
    1.5045    1.6116
    1.0673    0.9352

Producto elemento a elemento
No se puede realizar el producto
```

- **Un vector fila y columna obtenido concatenando la primera fila/columna de cada una de las matrices.**
  El vector fila se obtiene a través de v = [A(1,:) B(1,:)];, al igual el vector columna se hace con w = [A(:,1);B(:,1)];

```
Vector fila
    0.8147    0.9134    0.2785    0.9649    0.9572

Vector columna
    0.8147
    0.9058
    0.1270
    0.9649
    0.1576
    0.9706
```

EJERCICIO 4. Tiempo de cómputo y representación gráfica

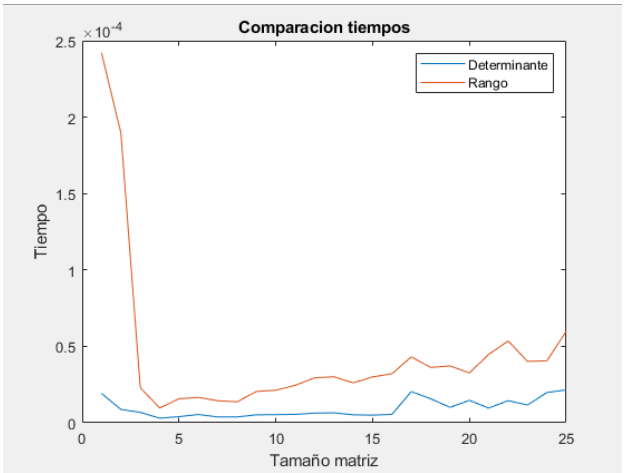
Realice un script en Matlab que permita obtener y representar el tiempo consumido para el cálculo del rango y el determinante de una matriz en función de su tamaño (entre 1x1 y 25x25). Tenga en cuenta que:

- **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.**

En primer lugar, realizamos un bucle for en el que creamos matrices cuadradas de dimensiones 1 hasta 25.

Para el cálculo de los tiempos de cada operación, utilizamos las funciones tic (empieza el cronómetro) y toc (finaliza el cronómetro) dentro de cada bucle. Los tiempos calculados los insertamos en dos vectores distintos para después representarlo de forma gráfica a través de plot.

Para etiquetar los ejes utilizamos las funciones xlabel e ylabel, y para visualizar la leyenda la función legend.



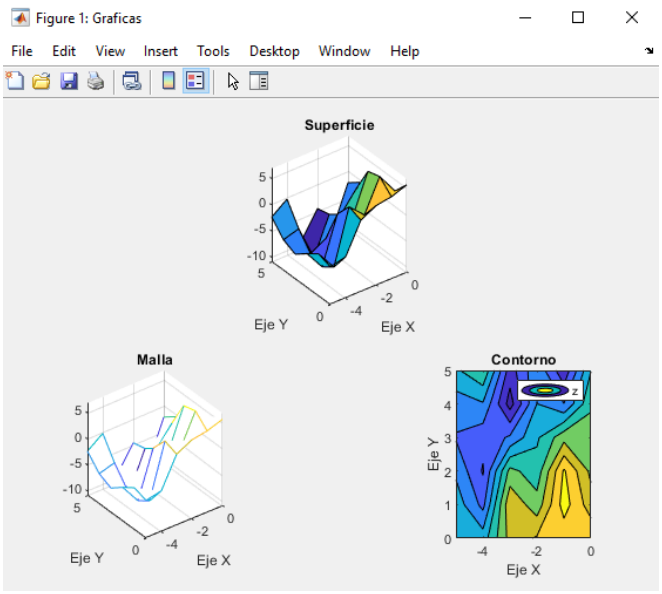
EJERCICIO 5. Representación gráfica en 3D

Realice un script en Matlab que dibuje sobre el área $-5 \leq x, y \leq 5$ la superficie, la superficie en forma de malla y el contorno de la función z.

- **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 dos vectores en los que representamos a **x** (de -5 a 0) y a **y** (de 0 a 5) y la función **Z**. Después creamos las matrices X (matriz de 6 filas donde cada fila es una copia de x) e Y (matriz de 6 columnas donde cada columna es una copia de y) por medio de la función meshgrid.

Finalmente, mostramos la gráfica de la superficie a través de surf, la gráfica de la superficie en forma de malla por medio de mesh y la gráfica del contorno a través de contourf. Las posiciones de cada gráfica las calculamos por medio de los argumentos de entrada de la función subplot (filas, columnas, posición).



EJERCICIO 6. Sistemas lineales

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

Para su posterior resolución, para cada sistema matricial hemos creado dos matrices, donde la primera representa los coeficientes y la segunda la solución, de manera AX = A1.

2. Escriba un script en que permita:

a) Obtener el número de condición de la matriz A respecto a la inversión

Para su cálculo hacemos uso de la función cond.

b) Resolver el sistema de ecuaciones para obtener la matriz x = [x1, x2, x3, x4]'.

Para la resolución del sistema utilizamos la función linsolve.

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

Para generar un vector de dichas características se ha hecho mediante v=a.*randn(1,10) + b, siendo a la desviación y b la media. Una vez generado, mediante un for, sumamos sus valores a cada uno de los coeficientes sin x del sistema.

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

```
Solucion S1 -> X1:-2.257095e+00, X2:4.933635e+00, X3:5.298567e+00, X4:3.564928e+00
Solucion S2 (sin ruido) -> X1:6.359299e+00, X2:6.369427e+00, X3:3.162992e+02, X4:2.162357e+02
Solucion S2 (con ruido) -> X1:6.668259e+00, X2:6.539580e+00, X3:3.166958e+02, X4:2.162066e+02
```

Compare los resultados obtenidos en cada caso. Al introducir ruido, los resultados del sistema apenas varían, esto quiere decir que nos encontramos ante un sistema cuyas ecuaciones son estables.

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.

a) Recoge los arrays con los que se crean los polinomios.
Realizado con la función input.

```
Polinomio 1 --> [1 2 2]
Polinomio 2 --> [1 3]
```

b) Solicita si la solución se aplica a uno de los polinomios o al producto: poli_1, poli_2, prod_poli.

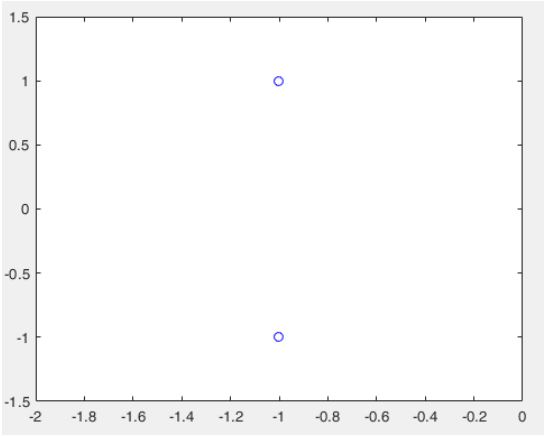
```
Escribir para que polinomio desea obtener la solución
pol_1, pol_2, prod_poli --> prod_poli
```

c) Devuelve las raíces del polinomio indicado y su clasificación (nº raíces reales y nº raíces complejas.

Para su resolución, hemos creado la función **raíces**, la cual nos devuelve las raíces del polinomio y dos vectores con las soluciones reales y complejas respectivamente. Para ello, hacemos uso de las funciones roots, la cual nos devuelve las raíces del polinomio. Almacenando éstas en un vector y recorriéndolas por medio de un bucle,

almacenamos en dos vectores las soluciones reales y complejas a través de la función isreal.

d) Representa en el plano complejo la ubicación de las raíces obtenidas.
Para poder representar las raíces mediante la función plot, se han tenido que separar en reales e imaginarias. En caso de que haya más de una, mediante la opción hold on se representarán todas ellas en la misma gráfica.



PARTE 2

EJERCICIO 1. Transformadas de señales.

1. **Obtenga la transformada z de la siguiente función: $f(k) = 2 + 5k + k$). Represente gráficamente las señales original y transformada.**

En primer lugar, mediante syms creamos la variable simbólica k. Tras esto, calculamos la transformada-z mediante la función ztrans de la función definida.

A la hora de representar las dos señales, hacemos uso de la función subs, en la cual damos valores de 0 a 10 a f(k) y a la transformada-z.

2. **Obtenga la transformada z de la siguiente función: $f(k) = \text{sen}(k) \cdot e^{-ak}$. Represente gráficamente, de nuevo, la señales original y transformada.**

Al igual que en el apartado anterior, creamos la variable simbólica k mediante syms, calculamos la transformada z de f(k) mediante ztrans y finalmente, representamos las dos señales dando valores a *k* y a *z* mediante subs.

A la variable constante *a* le hemos dado el valor 5.

3. **Dada la siguiente función de transferencia discreta**

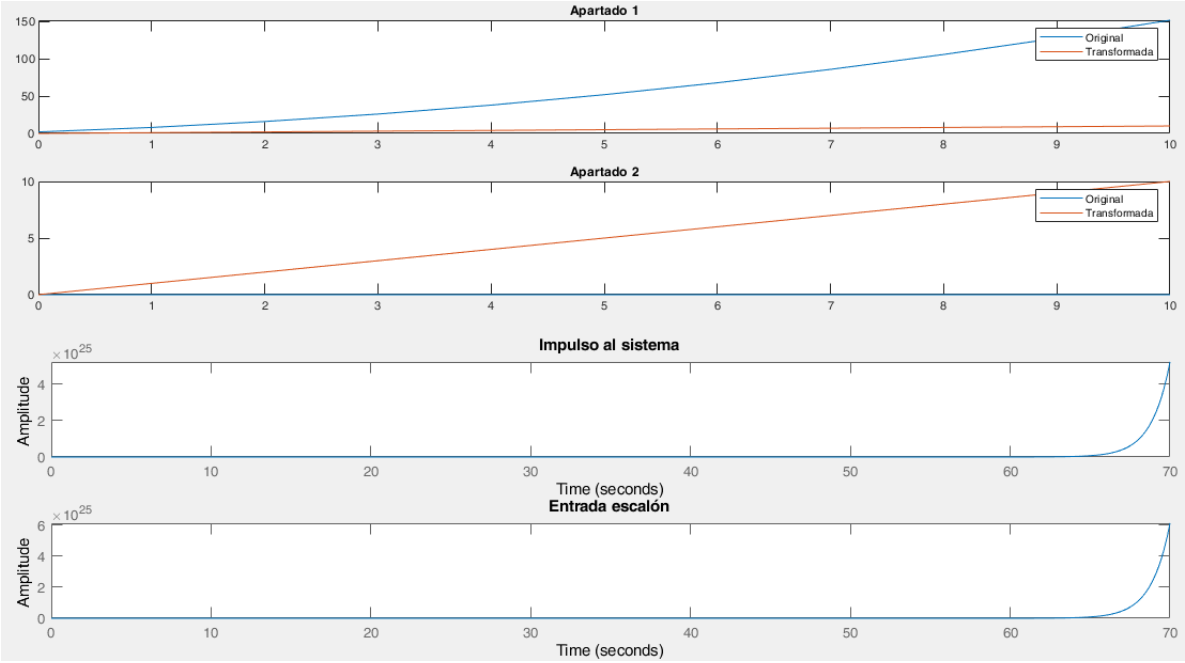
a) Obtenga y represente la respuesta al impulso del sistema.

En primer lugar, hemos especificado los coeficientes del numerador y denominador ordenados de forma descendiente de *z*, para así crear el modelo de función de transferencia mediante la función tf.

Finalmente, representamos la respuesta al impulso del sistema mediante *impulse*.

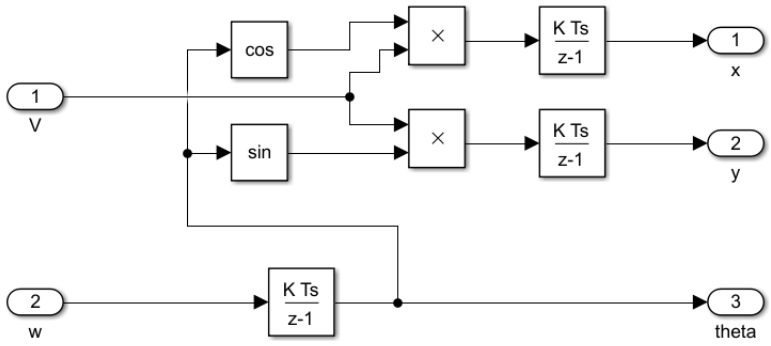
b) Obtenga y represente la respuesta del sistema ante una entrada escalón.

La obtención y representación del sistema ante una entrada escalón lo realizamos por medio de la función step.

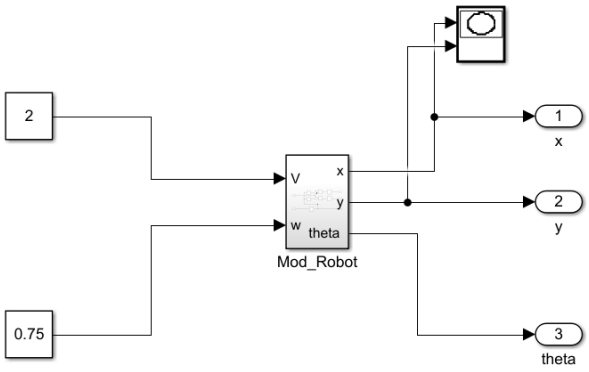


EJERCICIO 2. Modelado del comportamiento de un robot móvil en Simulink.

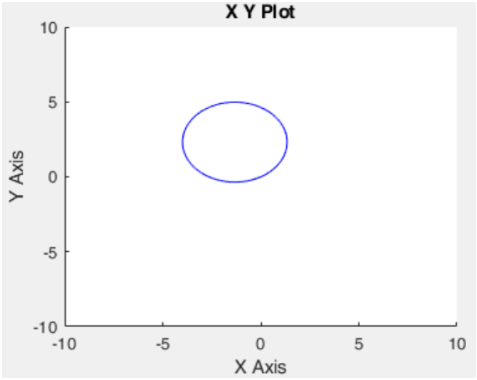
1. **Implemente el modelo de la Figura 2 (todos los bloques utilizados pueden encontrarse en la librería estándar de Simulink).**



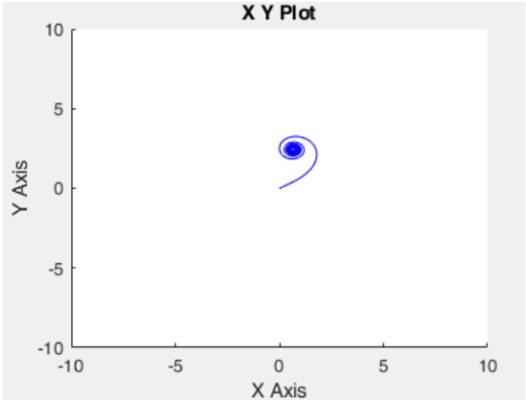
2. **Una vez completado el apartado anterior, cree el subsistema mostrado en la Figura 1 y simule su funcionamiento con velocidad lineal y angular constante creando el sistema mostrado en la Figura. Configure los parámetros de la simulación (menú "Simulation/Model Simulation Parameters" y menú Simulation/Pacing options) de acuerdo con la Figura 4.**



3. **Visualizando la gráfica generada por el módulo XY Graph, compruebe que el funcionamiento del robot se ajusta a lo esperado.**
   Al ser la velocidad angular y lineal constantes, lo esperado es que su movimiento sea en círculos.

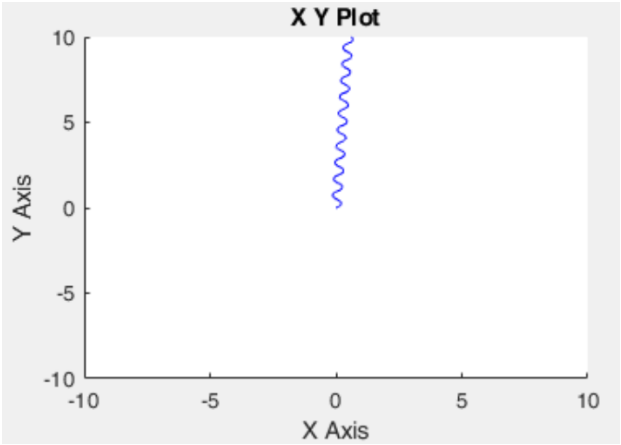


4. **Realice de nuevo la simulación con velocidades angulares no constantes (estas fuentes están disponibles en la librería de Simulink/sources):**
   a) **Rampa**
      Al ser la velocidad angular cada vez mayor, el movimiento se asemeja a una espiral.



b) Variación sinusoidal

La velocidad angular, al ser de tipo sinusoide, el movimiento se asemeja a la función seno.



5. Modifique la posición inicial del robot para que comience a moverse desde el punto (-4,-4) y realice estas simulaciones de nuevo.

Para modificar la posición inicial, dentro del modelo del robot, en ambos bloques de "Discrete-Time Integrator", se indica en Inicial Condition el -4.

Las trayectorias son las mismas que en las simulaciones anteriores, la única diferencia que se aprecia es el punto de comienzo del movimiento. Un ejemplo de ello: