UNIVERSIDAD NACIONAL DE CÓRDOBA

FACULTAD DE CIENCIAS EXACTAS, FÍSICAS Y NATURALES

Sistema de Control 2

TAREA Nº 4

Alumno: Mugni, Juan Mauricio

Profesor: Laboret

2024

Datos asignados según el archivo Alumnos_Tarea2.pdf

Alfonso	MOUTON LAUDIN	2	0,3	1	10	135	-2
Juan Mauricio	MUGNI	2	0,4	1	10	180	-2
Matias Gustavo	OBREGÓN	1	0,3	1	10	135	-3

Estos valores se los utiliza para representar un péndulo simple con la ecuación:

$$ml^2\ddot{\theta} + b\dot{\theta} + mglsen\theta = T$$

Y se pretende que el péndulo se estabilice en el ángulo δ dado.
Tomando como estados, entradas y salida respectivamente:

$$x_1 = \theta - \delta = e$$
$$x_2 = \dot{\theta}$$
$$u = T$$
$$y = e$$

Notar que se ha desplazado el punto de equilibrio al origen tomando el error como salida.
El objetivo es:

- hallar el sistema dinámico en variable de estado:
$$\dot{x} = f(x,u)$$
$$y = h(x)$$
- Hallar el torque estático necesario u_f para que el sistema tenga como punto de equilibrio el origen, es decir, $f(0,u_f) = 0$
- Linealizar el sistema mediante la jacobiana:
$$\dot{x} = Ax + Bu$$
$$y = h(x)$$

En el punto:

$$\begin{bmatrix} x_1 \\ x_2 \\ u \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ u_f \end{bmatrix}$$

- Hallar los autovalores de A y determinar la estabilidad por el método indirecto de Lyapunov.
- Comparar los resultador obtenidos con los de la linealización por Matlab y Simulink.

Se comienza asignando los siguientes valores:

```
m = 2   ;          % masa
b = 0.4 ;          % coeficiente de rozamiento
l  = 1  ;          % longitud
G = 10  ;          % constante gravitatoria
delta = 180 ;      % ángulo de referencia
```

Simulamos el modelo lineal, para ello utilizamos la función *linmod*, que linealiza el modelo alrededor del ángulo δ.

```
[A,B,C,D] = linmod('pendulo_mod_tarea',delta*pi/180);
```

Notar que este ángulo es pasado a radianes, y el resultado de esta función es una representación en espacio de estado del sistema.

Donde:

A: matriz de estado.
B: matriz de entrada.
C: matriz de salida.
D: matriz de transmisión directa.

$$A = \begin{bmatrix} 0 & 1 \\ 10 & -0.2 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$$

$$C = [1 \quad 0]$$

$$D = 0$$

Luego calculamos los valores propios de la matriz A, para determinar la estabilidad y comportamiento del sistema. (Si alguno de los valores propios tiene una parte real positiva, el sistema es inestable).

```
eig(A)            % Calculo los valores propios de la matriz A
```

Esto da como resultado 3.0639 y -3.2639, podemos ver que el sistema es inestable. Ahora calculamos el rango de la matriz de controlabilidad, teniendo en cuenta que dicha matriz se define como:

$$ctrb(A, B) = [B, AB, A^2B, \dots, A^{n-1}B]$$

Donde n es el número de estados del sistema. Si el rango es igual al número de estados (dimensión de A), el sistema es completamente controlable, de lo contrario hay algunos estados que no pueden ser controlados.
Breve código que nos determina lo anterior:

```
if rank(ctrb(A,B)) == rank(A)
    disp('El sistema es completamente controlable');
else
    disp('El sistema no es completamente controlable');
end
```

Da como resultado que el sistema es completamente controlable.
El trabajo aclara que se desea diseñar para el péndulo linealizado en el punto de operación dado, un controlador con acción integral:

$$u = k_1(\theta - \delta) + k_2\dot{\theta} + k_3\sigma$$

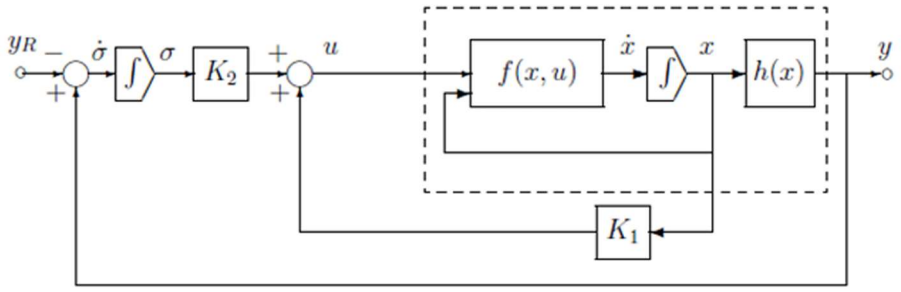
$$\dot{\sigma} = \theta - \delta$$

Que no es otra cosa más que un PID en la forma PI+D, ya que el torque vale:

$$T = k_1 e + k_3 \int_0^t e(\tau)d\tau + k_2\dot{\theta}$$

Donde el error se definió en el sentido tradicional como $e = \delta - \theta$.

El esquema de la situación es el siguiente:



Control por realimentación de estados y salidas con acción integral

Continuamos encontrando las matrices del sistema ampliado, donde:

$$A_{ampliado} = \begin{bmatrix} A & 0 \\ C & 0 \end{bmatrix} \; B_{ampliado} = \begin{bmatrix} B \\ 0 \end{bmatrix}$$

```
% Matrices ampliadas
Aa = [A, zeros(size(A,1), 1);
      C, 0];
Ba = [B;
      0];
```

Y esto resulta ser:

$$A_{ampliado} = \begin{bmatrix} 0 & 1 & 0 \\ 10 & -0.2 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

$$B_{ampliado} = \begin{bmatrix} 0 \\ 0.5 \\ 0 \end{bmatrix}$$

Se calculan los valores propios de la matriz $A_{ampliada}$:

```
% Cálculo de los valores propios de la matriz ampliada
eigenvalues = eig(Aa);
```

Esto da como resultado los siguientes polos 0, 3.0639 y -3.2639, podemos ver que el sistema ampliado también es inestable.
Verificamos la controlabilidad de la matriz ampliada calculando el rango de la matriz de controlabilidad ampliada y comparando con el número de filas de la matriz $A_{ampliado}$.

```
if rank(ctrb(Aa, Ba)) == size(Aa, 1)
    disp('El sistema ampliado es completamente controlable');
else
    disp('El sistema ampliado no es completamente
controlable');
end
```

Dando como resultado que el sistema ampliado es completamente controlable.

Notar que se utilizó `size(Aa, 1)` ya que devuelve el número de filas, que me indica el número de estados del sistema ampliado.
Se pide diseñar un controlador utilizando la fórmula de Ackerman, que será de la siguiente manera:

$$u = -[k_1 \quad k_2 \quad k_3]\begin{bmatrix} x_1 \\ x_2 \\ \sigma \end{bmatrix}$$

Lo primero que hacemos es asignar los polos, esto es dato:

```
p1 = -2 ; % posición del polo triple
p2 = p1 ;
p3 = p2 ;
```

y ahora se calcula el controlador con la función *acker*.

```
K = acker(Aa,Ba,[p1 p2 p3]);
```

K es el siguiente vector:

$$K = [44 \quad 11.6 \quad 16]$$

Podemos verificar la ubicación de los polos a lazo cerrado:

```
disp('Verificación de la ubicación de los polos asignados: ');
eig(Aa-Ba*K)
```

y esto da como resultado: $-2 + i0, -2 + i0, -2 + i0$

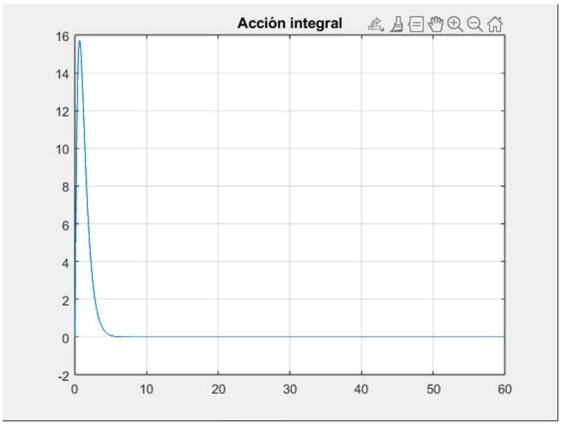
Se calcula ahora el tiempo de respuesta, $t_{ss} \cong \frac{7.5}{-p}$:

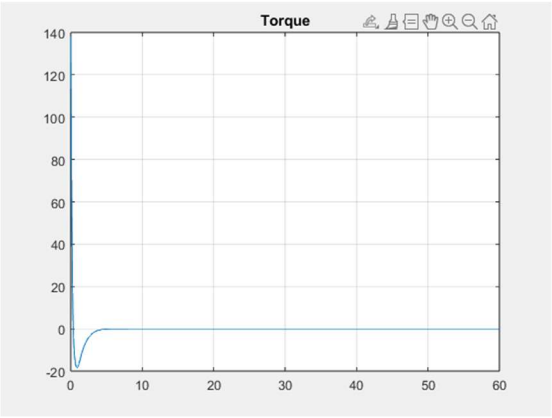
```
tscalc = 7.5/(-p1) % Tiempo de respuesta
```

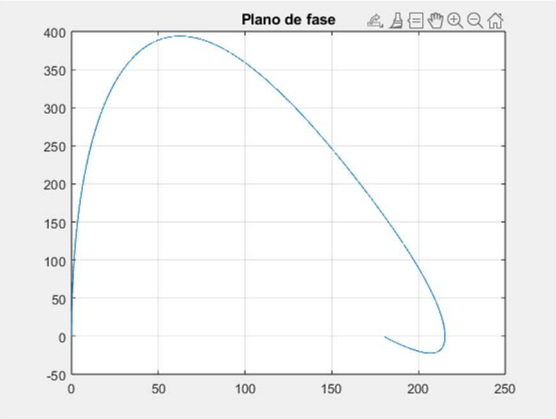
Se continúa simulando el péndulo con PID, partiendo del origen con velocidad nula y referencia δ.

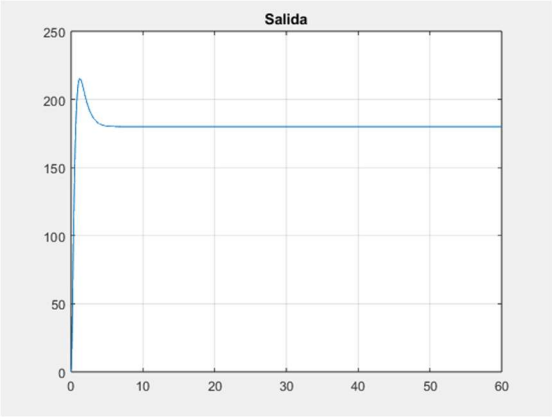
```
sim('pendulo_pid_tarea')
```

Y se realizan los siguientes gráficos:









El código para obtenerlos es el siguiente:

```
% Gráficos
figure(1);
plot(tout, yout);
title('Salida');
grid on;

figure(2);
plot(yout, velocidad); % Plano de fase
title('Plano de fase')
grid on;

figure(3);
plot(tout, torque);
title('Torque'); % Torque
grid on;

figure(4);
plot(tout, -accint); % Acción integral
title('Acción integral');
grid on;
```

Procedemos a obtener los siguientes valores:

- Máximo valor de salida: $y_{max} = 215.244$
- Sobrepaso en porcentaje: $S = 19.58$
- Error final: $efinal = 0$
- Tiempo de establecimiento: $t_{SS} = 3.5533$
- Salida al tiempo de establecimiento: $y_{te} = 183.6$
- Torque final: $u_f = -8.8541x10^{-13}$
- Acción integral final: $int_f = -2.6633x10^{-11}$

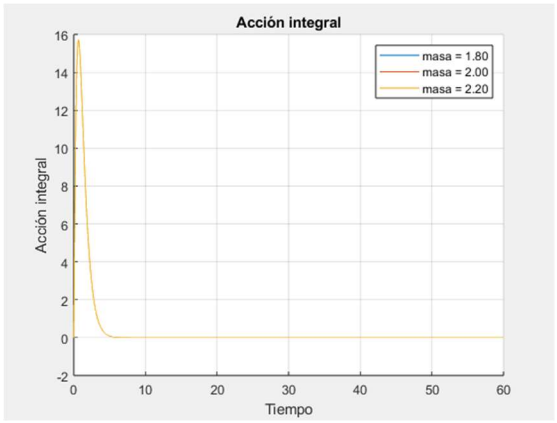
El código empleado para obtener los resultados anteriores es el siguiente:

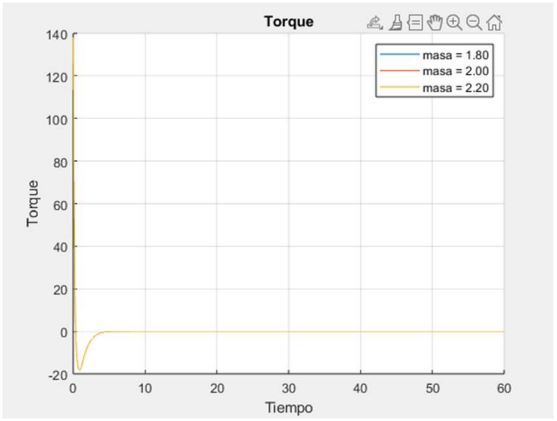
```
ymax = max(yout); % máximo valor de salida
S = (ymax - delta)/delta*100; % sobrepaso en %
erel = (delta - yout)/delta; % error relativo
efinal = erel(end); % error final, debe ser cero
ind = find(abs(erel)>0.02); % índice elementos con error relativo absoluto menor a 2%
tss = tout(ind(end)); % tiempo de establecimiento (último valor del vector)
yte = yout(ind(end)); % salida al tiempo ts
uf = torque(end); % torque final
intf = -accint(end); % acción integral final
```

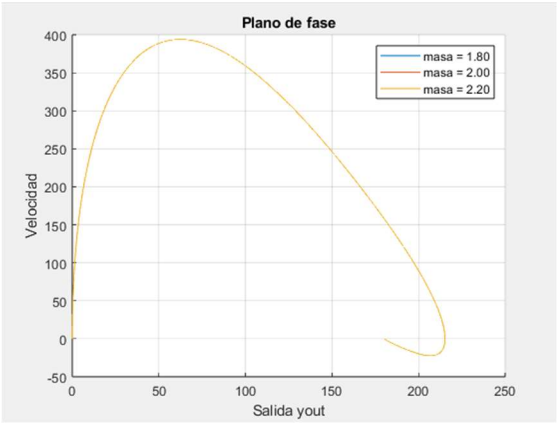
Por último, se analiza la robustez variando la masa del péndulo en más y menos 10%, y se obtiene la siguiente tabla:

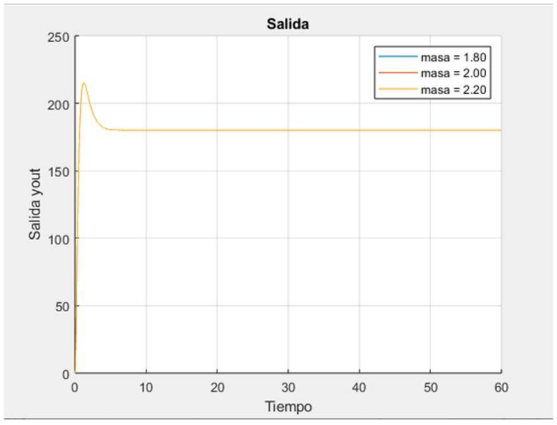
masa	y_{max}	s	e_{final}	t_{ss}	y_{te}	u_f	int_f
$m-10\%$	215.244	19.58	0	3.5533	183.6	$-8.8566x10^{-13}$	$-8.8566x10^{-13}$
m	215.244	19.58	0	3.5533	183.6	$-8.8541x10^{-13}$	$-2.6633x10^{-11}$
$m+10\%$	215.244	19.58	0	3.5533	183.6	$-8.8517x10^{-13}$	$-2.9297x10^{-11}$

Y en el gráfico podemos ver como son prácticamente iguales si la masa varía $\pm 10\%$.


Acción integral
masa = 1.80
masa = 2.00
masa = 2.20
Acción integral
Tiempo


Torque
masa = 1.80
masa = 2.00
masa = 2.20
Torque
Tiempo


Plano de fase
masa = 1.80
masa = 2.00
masa = 2.20
Velocidad
Salida yout


Salida
masa = 1.80
masa = 2.00
masa = 2.20
Salida yout
Tiempo