

LABORATORIO DE AUTOMÁTICA - MINAS

PRÁCTICA V

1. DATOS DEL GRUPO

Fecha:
Grupo:
Nombre: dni

2. GENERACIÓN DE DATOS

```
rand('state',dni);  
r = rand;  
Ts = 0.02*r + 0.05*(1-r); % periodo de muestreo
```

3. INTRODUCCIÓN

En esta práctica se utilizarán funciones del `toolbox` de identificación de `MATLAB` y el `real-time toolbox` de `simulink` para identificar un motor de corriente continua (Sistema Feedback) con entrada en tensión y salida en rotación. Acto seguido se elaborará un compensador para obtener prestaciones exigidas en respuesta a escalón en lazo cerrado.

El sistema Feedback como planta operativa está constituido por:

1. Ordenador con Matlab-Simulink instalado
2. Tarjeta de entrada-salida PCLAB
3. Motor CC con Preamplificador, Fuente, Tacómetro, Carga (Freno) y Potenciómetro.
4. Cables, resistencias y zócalos para conexión.

La tarjeta PCLAB requiere algunas explicaciones ya que ella es responsable de la transformación entre variables con contenido energético (tensiones y corrientes) y representaciones numéricas discretas en un determinado rango de valores (0 4095). Para este primer curso, los fenómenos de conversión y sus consecuencias no serán llevados en cuenta. Estas prácticas serán realizadas dentro de una gama dinámica en que estos aspectos no sean sensibles. Conforme se puede observar en el diagrama hay dos pistas de conexión. Cada pista tiene 20 puertos. La pista con puertos GND sirve para conexión a masa. (Es una regla de oro que siempre se conecten a masa común todos los equipos participantes de la práctica.) La otra pista posee tres tipos de puertos cuyos propósitos básicos son:

1. Tensiones de referencia. (RV1, VR2)
2. Salidas analógicas. (DA1, DA2). Estas salidas realizan la conversión entre grandezas numéricas en el ordenador para señales que transportan energía (tensiones y corrientes). El rango de tensiones para la salida DA1 vale $(-5 < v < 0)$ y para la salida DA2 vale $(0 < v < 5)$. Así una señal numérica

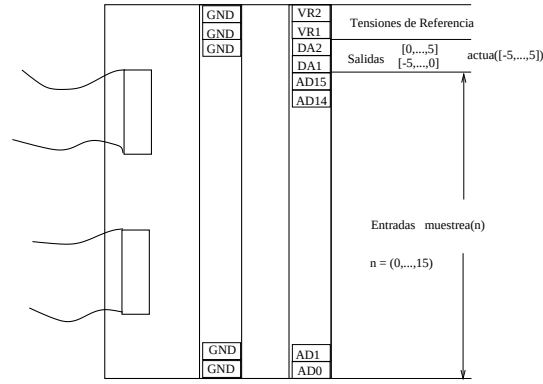


FIGURA 1. Tarjeta PCLAB

en el rango $(-5 < v < 5)$ se realiza simultáneamente a través de DA1 y DA2 siendo que cuando positivo el número se convierte por DA2 y cuando negativo por DA1.

3. Entradas digitales. (AD0, ... , AD15). El propósito de estos puertos es transformar las medidas realizadas sobre señales del mundo exterior, en valores numéricos en un determinado rango y precisión que puedan ser procesador dentro de un programa. El rango máximo permitido (sin saturación) es de $v \in [-5, 5]$. siendo que las medidas se efectúan en los puertos $n \in [0, ..., 15]$

Claro está que en el mundo real las variables de energía, tanto las generadas cuanto las medidas, pueden aparecer en los rangos más diversos. Esto en nada invalida lo ya expuesto pues si necesario se pueden condicionar las señales a través de transformaciones de escala lo que básicamente se realiza con ganancias y traslaciones.

3.1. Conexiones para la Actuación. La salida analógica a través de la tarjeta PCLAB debe escalarse para obtener señales de mando al nivel de potencia requerido. El diagrama que sigue da una idea de una posible conexión. Un divisor de tensión reduce las señal originada por la tarjeta a ≈ 0.1 de su valor. La señal en seguida es inyectada en un pre amplificador que la eleva a ≈ 20 siendo que al final resultan valores del orden de ≈ 2 la salida de la tarjeta PCLAB pero con una característica muy importante: ahora las señales transportan la polaridad y energía suficientes para comandar el motor de CC. Las entradas deben también escalarse

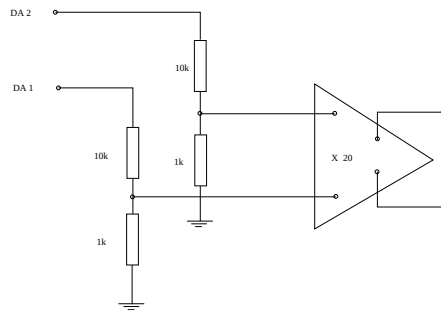


FIGURA 2. Diagrama de la actuación

en la mayoría de los casos. En el Sistema Feedback hay básicamente dos medidas de interés:

1. Posición obtenida a través de un potenciómetro
2. Velocidad obtenida a través de un generador tacométrico.

Una posible conexión para adquirir las medidas se puede observar en la figura que sigue:

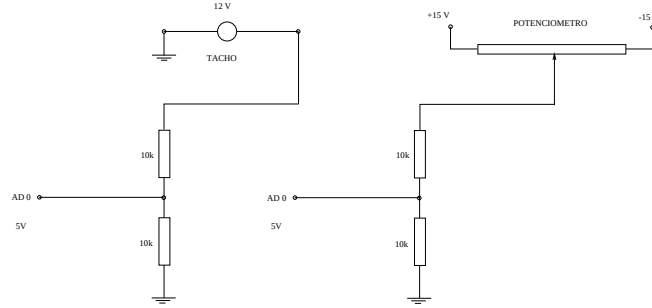


FIGURA 3. Adecuación de las medidas tacométricas

3.2. Modelo simplificado del Sistema Feedback. El sistema feedback utilizado para prácticas puede ser representado por el diagrama que sigue: se pueden destacar la siguientes funcionalidades

1. Controlador, aquí representado por la caja ORDENADOR. (Incorpora la UCP y la PCLAB).
2. Planta, representado por el conjunto AMPL. POT. + MOTOR + Carga. (Esta última está incorporada en la caja MOTOR).
3. Observadores representados por las cajas TACHO. y POT. de Generador Tacométrico y Potenciómetro

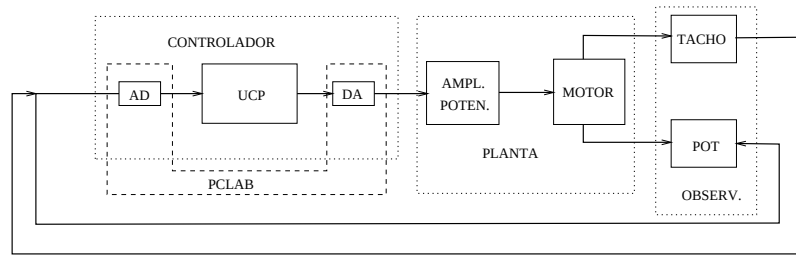


FIGURA 4. Arreglo del conjunto

El generador tacométrico al girar genera una tensión proporcional a la rotación, pudiendo ser utilizado para observar la rotación del motor principal. Ya el potenciómetro genera una tensión proporcional al ángulo módulo $2\pi\eta$ donde η es un factor de reducción. El motor normalmente debe conectarse a una carga externa

representada por un freno magnético que actúa de forma creciente con la rotación, y con el área expuesta al campo magnético.

Un aspecto muy importante en este arreglo experimental es el del mínimo tiempo en que se puede reaccionar a cambios en por ejemplo, variables de consigna. Este tiempo está acotado inferiormente por el tiempo de transporte de una señal a través del lazo. En los equipos de laboratorio el tiempo necesario para transformar AD procesar en la UCP y convertir DA, sumado al tiempo de gestión del sistema operativo y del gestor de MATLAB acotan inferiormente este tiempo en ≈ 0.01 [s]. De donde se concluye que las prácticas realizadas con este arreglo deberán atender especificaciones de desempeño con constantes de tiempo del orden de $\approx 20 \times 0.01$ [s] como mínimo.

4. IDENTIFICACIÓN DEL MOTOR CC

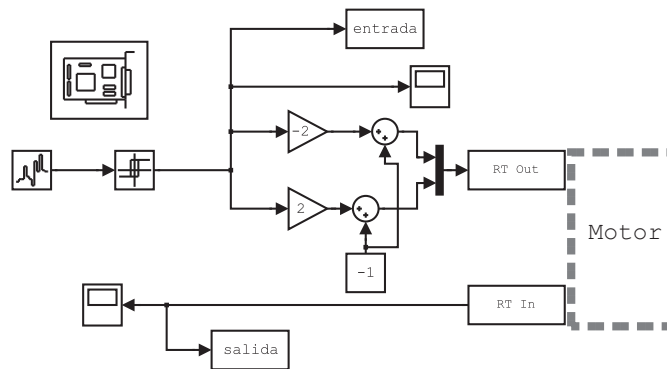


FIGURA 5. Arreglo de simulink utilizando funciones del real-time toolbox

5. DESARROLLO DE LA PRÁCTICA

La identificación de la planta se efectúa utilizando funciones del toolbox de identificación de MATLAB. Con auxilio del arreglo exhibido en la figura 5 construido utilizando simulink con la toolbox de tiempo real se obtienen dos vectores de puntos representativos de la *entrada* y *salida* del motor CC. Con estos vectores de medidas obtenidos a intervalos de $T = T_s$ [s] se efectúa la identificación paramétrica, (*pem*, *arx*) y no paramétrica (*impulse*, *step*, *spa*). También se observan los resultados comparativos (*step*, *bode*)

```
load entrada
load salida
% El vector salida contiene las medidas de rotacion.
% El vector entrada contiene las medidas de tension.
% La entrada esta generada por una secuencia aleatoria
% de valores binarios (0,1) de tension. El periodo de
% muestreo es de Ts s.
%
% Primero generemos un objeto del tipo IIDATA:
T = Ts;
motor = iddata(salida,entrada,T);
% Rotulemos la entrada y salida
motor.InputName = 'Tension';
motor.OutputName = 'Rotacion';
```

```

n = length(entrada);
N = n - 30;
N2 = ceil(N/2);
% Desechamos las 30 primeras medidas y las dividimos en dos grupos
% Seleccionemos el primer grupo para elaborar el modelo:
ze = motor(30:N2+30);
% La grafica para las muestras 30 to N2+30:
figure(1)
plot(ze)
% Remove las constantes y elaborar datos con media nula:
ze = dtrend(ze);
% Estimar la respuesta impulsiva del sistema por analisis
% de correlacion para tener una idea de las constantes de tiempo
figure(2)
impulse(ze,'sd',3)
% Las lineas a trazos marcan un intervalo de 99% de confianza.
% Podemos notar que hay un retardo de 1 intervalo.
% Incluyendo el parametro 'fill' tendremos representacion
% alternativa
figure(3)
impulse(ze,'sd',3,'fill')
% El modo mas simple de empezar es construir un modelo lineal
% cuyo orden es determinado automaticamente, utilizando para
% tanto el metodo PEM (Prediction Error Method)
m1 = pem(ze);
m1
% Los parametros del modelo se obtienen facilmente
% como sigue. Para saber los parametros que pueden
% recuperarse se efectua el comando get(m1)
A = m1.a
B = m1.b
C = m1.c
D = m1.d
K = m1.k
% Cuan bueno es el modelo? Una forma de responder esta pregunta
% es comparar la respuesta simulada con la respuesta medida.
% Seleccionemos un conjunto de medidas que no haya sido utilizado
% para construir el modelo: por ejemplo los datos del segundo grupo
zv = motor(N2+30+1:n);
zv = dtrend(zv);
figure(4)
compare(zv,m1);
% El ajuste es bastante bueno
% La grafica de bode del modelo se obtiene con
figure(5)
bode(m1, subplot(211), grid, subplot(212), grid
% Se puede comparar la respuesta a escalon del modelo
% con una respuesta computada a partir de los datos de
% forma no parametrica
figure(6)
step(m1,ze)
% Para generar un modelo de estructura determinada
% utilizamos el comando arx con parametros relativos
% a dos polos un cero y un retardo:
m2 = arx(ze,[2 2 1]);
% el modelo resulta:
m2
% Para comparar sus prestaciones con el modelo m1
% se efectua:
figure(7)
compare(zv,m1,m2);
% Ambos modelos se comportan de modo muy similar.
% Ahora podemos visualizar el comportamiento comparativo
% en frecuencia de ambos modelos
figure(8)
bode(m1,m2, subplot(211), grid, subplot(212), grid
% Podemos tambien comparar ambos modelos (m1, m2)
% con la funcion de transferencia obtenida por
% metodos no parametricos (transformada de Fourier)
gs = spa(ze);
figure(9)
bode(m1,m2,gs), subplot(211), grid, subplot(212), grid

```

```
% Las respuestas en frecuencia de ambos modelos casi
% coinciden lo que acredita la bondad de los
% modelos obtenidos.
```

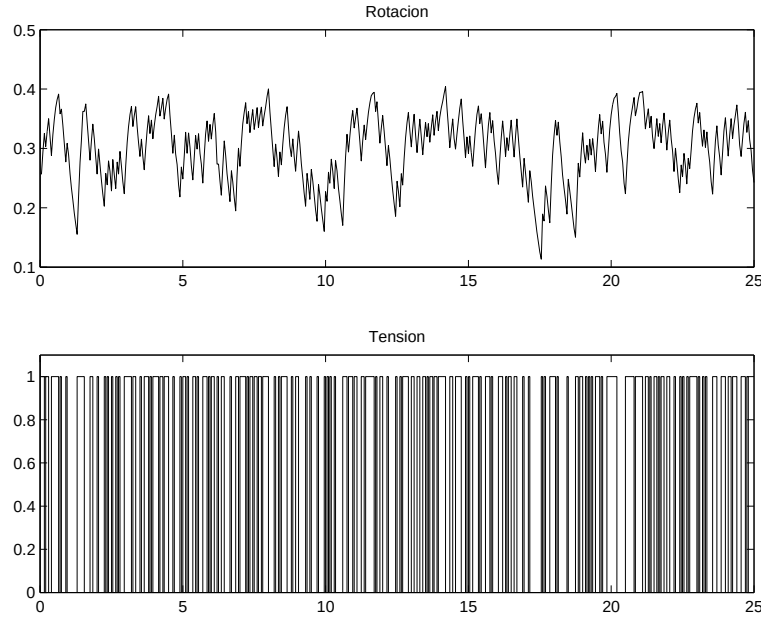


FIGURA 6. Datos de entrada-salida

5.1. Cálculo del compensador.

- Utilizando la gráfica obtenida 7 se pide elaborar un controlador PI con pre-filtro que conlleve a las prestaciones:
 - Error nulo en régimen permanente para entrada en escalón.
 - Sobre-oscilación $SO \leq 10\%$.
 - Tiempo de asentamiento $t_a \leq 1[s]$
- Comparar la respuesta a escalón con controlador proporcional y con el PI calculado.
- Verificar los márgenes de fase y ganancia.
- Efectuar correcciones en el controlador caso se haga necesario.

6. OBSERVACIONES

Una vez obtenido el controlador en la forma de una fracción polinomial de numerador $N_c(s)$ y denominador $D_c(s)$ el controlador debe ser discretizado para que pueda llevar en cuenta de modo eficiente la cadencia impuesta $T = Ts$. Esto se obtiene con el comando MATLAB

```
[dNc,dDc] = c2dm(Nc,Dc,T,'zoh');
```

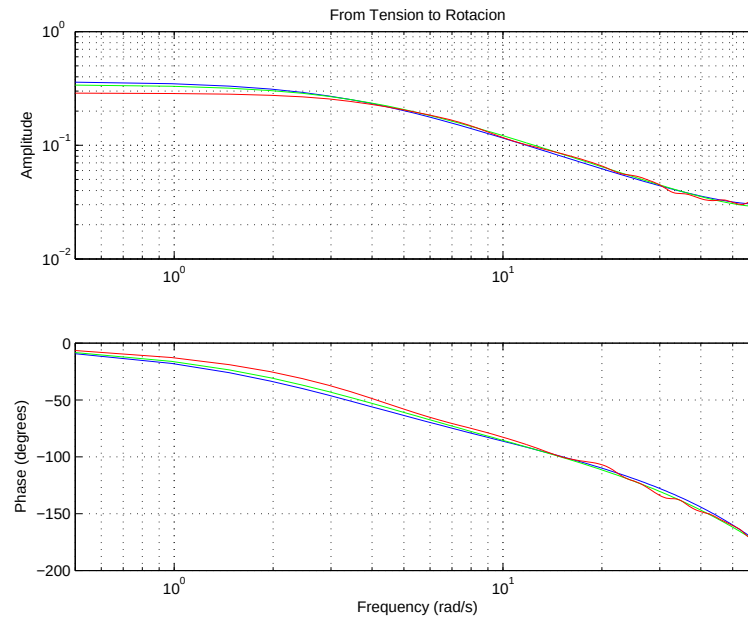
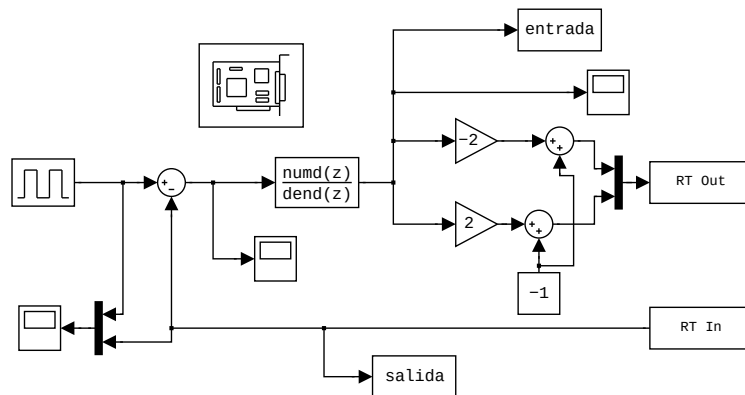


FIGURA 7. Gráfica de bode comparativa de la función de transferencia

FIGURA 8. Arreglo de `simulink` utilizando funciones del `real-time toolbox` para implementación del control a través de un PI