Programas Matlab para el modelado de un motor DC. Anexo al documento "aplicacion-MotorDC-Modelado.pdf"

Félix Monasterio-Huelin y Álvaro Gutiérrez

19 de abril de 2016

Índice

1. Introducción

En este documento se explican someramente los programas Matlab del algoritmo de modelado de un motor DC descrito en [1]. A este documento se adjunta el código de cada programa en el fichero comprimido **simulaboModeladoMotorDC.tar.gz**. Este documento debe leerse en compañía de [1] y de cada uno de los programas mencionados aquí, ya que se hace constante referencia a ellos. Cada programa tiene unos comentarios iniciales que explican el significado de las variables principales de entrada y salida, y sin estos comentarios, lo que se dice en este documento estaría incompleto.

En lo que sigue se supone que se han obtenido datos experimentales de la salida de un motor DC real cuando la entrada es una señal cuadrada de amplitud $V_j, j \in \{1, 2, \cdots, Q\}$, y que estos datos están almacenados en Q ficheros de texto de dos columnas: la primera representa el número de muestra k, y la segunda el número de pulsos $N_j(k)$ obtenido con un encoder situado en el eje del motor. La primera fila de cada fichero representa la condición inicial, con el número de muestra $k = 0$, y puesto que se supone que la condición inicial es nula, el número de pulsos $N_j(0) = 0$.

Para interpretar correctamente estos ficheros es necesario saber cuál es el periodo de muestreo uniforme T en segundos utilizado en los experimentos, la resolución del encoder, CPR, en pulsos/vuelta y los parámetros de la señal cuadrada: la duración del escalón de subida t_1 en segundos, la duración del escalón de bajada t_2 en segundos y la amplitud del escalón de subida V_j en voltios.

Por ejemplo, si $t_1 = t_2 = 0{,}6$ segundos y $T = 1$ milisegundo, cada fichero tendrá 1201 datos.

El valor de CPR es necesario para calcular la posición angular $\theta_j(k)$ en radianes del motor y la velocidad angular del motor, $\dot{\theta}_j(k)$ en rad/s, a partir del número de pulsos. Puesto que se supone que las condiciones iniciales son nulas, $\theta_j(0) = \dot{\theta}_j(0) = 0$.

El cálculo en radianes de $\theta_j(k)$ se realiza con la siguiente expresión,

$$\theta_j(k) = \frac{2\pi N_j(k)}{q}, \quad k \geq 0 \tag{1.1}$$

donde $q = CPR$.

En los programas desarrollados se calcula la velocidad angular $\dot{\theta}_j(k)$ en rad/s utilizando la primera aproximación de Euler de la derivada,

$$\dot{\theta}_j(k) = \frac{2\pi \left(N_j(k) - N_j(k-1)\right)}{q}, \quad k \geq 1 \tag{1.2}$$

2. ModeladoMotorDCVelAngPoloBucleB.m

El programa **ModeladoMotorDCVelAngPoloBucleB.m** es el programa principal, cuya función es ejecutar todos los programas necesarios para realizar el modelado experimental del motor DC. Estos programas se encuentran en el fichero comprimido **simulaboModeladoMotorDC.tar.gz**.

En primer lugar deben definirse las constantes necesarias, así como los nombres de los ficheros que deberán escribirse y leerse a lo largo de todo el proceso. Esto debe hacerse manualmente como se explica a continuación. Una vez hecho, puede ejecutarse el programa **ModeladoMotorDCVelAngPoloBucleB.m** en el entorno de Matlab. Su cabecera es la siguiente:

```
[pM, KM, Gzero,J,VeqV, a,b]=ModeladoMotorDCVelAngPoloBucleB
```

donde $VeqV$ es el vector de tensiones equivalentes (sección 2.3 del documento [1]), J es el valor de la función de error que permite obtener la ganancia a bajas frecuencias óptima (sección 2.3.1 del documento [1]), a, b son los vectores de parámetros del polinomio de interpolación de la función no lineal f y f^{-1} respectivamente (secciones 2.4 y 4 del documento [1]), y donde

$$G(s) = \frac{KM}{s + pM} \tag{2.1a}$$

$$Gzero = G(0) = \frac{KM}{pM} \tag{2.1b}$$

es decir, pM y KM son los parámetros estimados de la función de transferencia de la parte lineal del modelo del motor DC.

Los ejemplos que se citen en este documento, se corresponden con el ejemplo de modelado de la sección 3 del documento [1].

Los programas requieren que se definan algunas constantes cuyos nombres de las variables deben respetarse. Por ejemplo:

```
%%%%%%%%%%%%%%%%%%%%%%%
TensionVector=[1 2 3 4 5 6 7 8 9];
TensionVectorL=length(TensionVector);
DuracionSubida=0.6;% segundos
Periodo=0.001;% en segundos
CPR=12;% pulsos/vuelta
z1=0.5;% ponderación de los polos de subida y bajada
opCurvas=[1 1 1];% se explica en ModeladoMotorDCCurvasB.m
```

A su vez, deben definirse los nombres y ubicaciones de algunos ficheros. Debe haber cuatro tipos de ficheros. En el ejemplo que se pone a continuación tienen extensiones distintas: ".mean", ".resumen", ".polo" y ".poli". A continuación se explica cada uno de ellos.

Lo que debe respetarse son los nombres de las variables de cada uno de ellos: "NomFicheroLecturaMean", "NomFicheroEscrituraResumen", "NomFicheroEscrituraPolo" y "NomFicheroEscrituraPoli". Las variables "NomFicheroLecturaMean" y "NomFicheroEscrituraPolo" tienen la estructura de un vector de celdas ("cell array") ya que hay Q ficheros de nombres distintos.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% ejemplo NOMBRE DE FICHEROS:
%%  - con extensiones .mean, .resumen, .polo y .poli,
          que se leen o se escriben en el subdirectorio ./PololuModeladoDat
(que se encuentra en el fichero: simulaboModeladoMotorDC.tar.gz)
%%  - ficheros de datos:
% ./PololuModeladoDat/trapXV_0ms600ms600ms_T1ms_ST.mean,
                donde X es la tensión en {1,2,...,9}
%%  - restantes ficheros:
% ./PololuModeladoDat/MotorDCPololu.V_0ms600ms600ms_T1ms_ST.XV.polo,
                donde X es la tensión en {1,2,...,9}
% ./PololuModeladoDat/MotorDCPololu.V_0ms600ms600ms_T1ms_ST.resumen
% ./PololuModeladoDat/MotorDCPololu.V_0ms600ms600ms_T1ms_ST.poli
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
PrefNomMeanA='./PololuModeladoDat/trap';
PrefNomMeanB='0ms600ms600ms_T1ms_ST.mean';
PrefNomFicheroEscritura='./PololuModeladoDat/MotorDCPololu';
PrefNomFichero2=strcat(PrefNomFicheroEscritura,'.',PrefNomMeanB);
NomFicheroEscrituraResumen=strcat(PrefNomFichero2,'.resumen');
NomFicheroEscrituraPoli=strcat(PrefNomFichero2,'.poli');
NomFicheroLecturaResumen=NomFicheroEscrituraResumen;
for i=1:TensionVectorL
Tension=TensionVector(i);
NTension=num2str(Tension);
NomFicheroEscrituraPolo{i}=strcat(PrefNomFichero2,'.',NTension,'V.polo');
NomFicheroLecturaMean{i}=strcat(PrefNomMeanA,NTension,'V_',PrefNomMeanB);
end
NomFicheroLecturaPolo=NomFicheroEscrituraPolo;
```

El Programa principal **ModeladoMotorDCVelAngPoloBucleB.m** ejecuta los siguientes programas, y en el orden indicado. Todos los programas tienen un comentario inicial que explica su funcionamiento. Solo el programa **ModeladoMotorDCVelAngPoloB.m** requiere que se introduzcan manualmente valores numéricos para el cálculo de los polos de subida y de bajada.

1. Función **LecturaExperimentoMultiV.m** Lee, en un bucle de longitud **TensionVectorL=Q**, cada uno de los ficheros de datos de los experimentos asociados a la variable de tipo celda ("cell array") "NomFicheroLecturaMean", creando el vector "Tiempo" (número de pulso, índice temporal k) y el vector de celdas ("cell array") "Pulsos" (valor del pulso, $N(k)$). Se supone que el vector "Tiempo" es el mismo para los Q ficheros de datos, por lo que no es necesario que sea un vector de celdas.

```
[Tiempo,Pulsos{i}]=LecturaExperimentoMultiV(TensionVector(i),
        NomFicheroLecturaMean{i},Periodo);
```

2. Programa **ModeladoMotorDCVelAngPoloB.m**. Este programa se ejecuta en un bucle de longitud **TensionVectorL=Q**.

```
ModeladoMotorDCVelAngPoloB(i,TensionVector(i),CPR,Periodo,Tiempo,Pulsos{i},
        NomFicheroEscrituraResumen,NomFicheroEscrituraPolo{i});
```

   El programa **ModeladoMotorDCVelAngPoloB.m** requiere que se introduzcan manualmente los valores extremos de los intervalos temporales para el cálculo del polo de varianza mínima de subida, y para el cálculo del polo de bajada, tal y como se describe en las secciones 2.2 y 2.2.1 del documento [1]. Un ejemplo es el siguiente:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
IniS=1;% inicial subida
InfS=601;% final subida
InfB=1201;% subida + bajada
kTs=401;% frontera régimen permanente subida
DeltatS=60;
DnS=2;%>=2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

   Al ejecutar en un bucle el programa **ModeladoMotorDCVelAngPoloB.m** se crea y actualiza el fichero asociado a la variable "NomFicheroEscrituraResumen" y Q ficheros asociados a la variable de tipo celda "NomFicheroEscrituraPolo". El formato de cada uno de estos ficheros se explica en los comentarios iniciales de este programa.

   El programa principal **ModeladoMotorDCVelAngPoloBucleB.m** lee el fichero asociado a la variable "NomFicheroLecturaResumen" (que normalmente coincidirá con el fichero asociado a la variable "NomFicheroEscrituraResumen") ya que utiliza la información creada en el bucle del programa **ModeladoMotorDCVelAngPoloB.m** para calcular los valores de "pM","KM", "Gzero", "J" y "VeqV".

3. Función **InterpolacionPolinomioSimetrico.m**. Calcula el vector de parámetros, a, del polinomio de interpolación f de la tensión equivalente y de la tensión experimental, tal y como se explica en la sección 2.4 del documento [1]. Permite también obtener el vector de los parámetros, b, de la función inversa f^{-1}, necesarios para la implementación del controlador, según se explica en la Sección 4 del documento [1].

   Para obtener los vectores de parámetros "a" y "b" se ejecutarían las siguientes instrucciones:

```
[a,aR]=InterpolacionPolinomioSimetrico(VeqVP,TensionVectorP);%f
[b,bR]=InterpolacionPolinomioSimetrico(TensionVectorP,VeqVP);%f^{-1}
```

   Los vectores "VeqVP" y "TensionVectorP" deben coincidir con los los vectores "VeqV" y "TensionVector" salvo que se pretenda obtener un polinomio de interpolación extendido, como

se ha hecho en el ejemplo de modelado de la sección 3 y de diseño del controlador de la sección 4 del documento [1]. Por ejemplo, deben definirse los vectores "VeqVP" y "TensionVectorP" antes de ejecutar la función **InterpolacionPolinomioSimetrico.m**:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
TensionVectorP=TensionVector;
VeqVP=VeqV;
%%%%%%%%%%%%%%
%% Si se hace con un polinomio extendido: por ejemplo,
%V9=[9.4 9.6 9.8]
%TensionVectorP=[TensionVectorP V9];
%VeqVP=[VeqVP V9];
```

Los resultados de la función **InterpolacionPolinomioSimetrico.m** se guardan en el fichero "NomFicheroEscrituraPoli", operación que se realiza en el mismo programa principal **ModeladoMotorDCVelAngPoloBucleB.m**. El formato de este fichero se explica en los comentarios iniciales de este programa.

4. Programa **ModeladoMotorDCCurvasB.m**. Crea, como máximo, tres tipos de figuras, según se indique en la variable "opCurvas". Por ejemplo, si "opCurvas=[1 0 1]" solo se crearán las figuras de tipo "a" y de tipo "c". En los comentarios iniciales del programa **ModeladoMotorDCCurvasB.m** se explica cada uno de estos tipos.

   El programa **ModeladoMotorDCCurvasB.m** tiene la siguiente cabecera:

```
ModeladoMotorDCCurvasB(opCurvas,DuracionSubida,TensionVector,
        NomFicheroLecturaPolo,Tiempo,PulsosRad,PS,PB,KS,KB,
        PSB,KSB,VeqV,pM,KM,a,b,TensionVectorP,VeqVP)
```

   Este programa tiene una primera instrucción, que puede comentarse si se considera conveniente, que cierra todas las figuras abiertas,

```
delete(findall(0,'Type','figure')) % cierra todas las figuras
```

   El programa **ModeladoMotorDCCurvasB.m** ejecuta, como máximo, tres programas, dependiendo del valor de "opCurvas": **ModeladoMotorDCCurvasPolo.m**, **ModeladoMotorDCCurvasVelAng.m** y **ModeladoMotorDCCurvasPoli.m**. El primero de ellos lee datos del conjunto de ficheros asociado a la variable de tipo celda "NomFicheroLecturaPolo", que normalmente coincidirá con el conjunto de ficheros asociados a la variable de tipo celda "NomFicheroEscrituraPolo".

   La cabecera del programa **ModeladoMotorDCCurvasPolo.m** es la siguiente:

```
ModeladoMotorDCCurvasPolo(TensionVector,NomFicheroLecturaPolo)
```

   La cabecera del programa **ModeladoMotorDCCurvasVelAng.m** es la siguiente:

```
ModeladoMotorDCCurvasVelAng(DuracionSubida,TensionVector,Tiempo,
        PulsosRad,PS,PB,KS,KB,PSB,KSB,VeqV,pM,KM)
```

   La cabecera del programa **ModeladoMotorDCCurvasPoli.m** es la siguiente:

```
ModeladoMotorDCCurvasPoli(a,b,TensionVectorP,VeqVP)
```

Las figuras generadas con este programa, y con el ejemplo que se está utilizando en este escrito, son las que aparecen en el documento [1]. La leyenda de la figura generada por el programa **ModeladoMotorDCCurvasVelAng.m** se ha separado para realizar una captura de pantalla, de tal manera que puedan ser visibles tantos las curvas como el contenido de la leyenda.

Bibliografía

[1] F. Monasterio-Huelin and A. Gutiérrez, *Modelado experimental de un Motor DC real*, 2016. [Online]. Available: http://robolabo.etsit.upm.es