SISTEMAS ELECTRÓNICOS DE CONTROL
curso 2002-2003

PRÁCTICA 8: Análisis de Sistemas en el Espacio de Estados

1. Objetivos.

- Utilizar el *Toolbox* de control de Matlab (*Control System Toolbox*) para trabajar en el espacio de estados.
- Analizar la estabilidad de sistemas modelados en el espacio de estados.
- Estudiar la controlabilidad y la observabilidad de sistemas en el espacio de estados.
- Separar los sistemas modelados en el espacio de estados en sus subsistemas controlable y observable.

2. Introducción al toolbox de control.

Matlab posee un toolbox muy potente de control (*Control System Toolbox*). Este toolbox consta de una colección de funciones que implementan técnicas de diseño, análisis y modelado de sistemas de control lineales invariantes en el tiempo (*LTI: Linear Time-Invariant*). Sus características más importantes son:

- Los sistemas de control pueden ser modelados como funciones de transferencia o en el espacio de estados, permitiendo utilizar tanto técnicas clásicas como modernas.
- Permite trabajar con sistemas continuos y con sistemas discretos.
- Se pueden realizar conversiones entre las diferentes representaciones de los sistemas.
- Es posible calcular y representar repuestas en el dominio del tiempo, en el frecuencial, así como el lugar de las raíces.
- Existen funciones para asignación de polos, control óptimo, ...
- Posee una interface gráfica (*LTI Viewer*) para el análisis de la respuesta de los sistemas.

3. Modelado de sistemas LTI (*Linear Time-Invariant*).

Un sistema de control lineal invariante en el tiempo (LTI) es modelado en Matlab como un objeto. Este objeto es una estructura de datos que permite manipular el modelo del sistema como si fuera una única variable de Matlab.

3.1. Creación de modelos.

Vamos a ver como se define en Matlab un sistema modelado en representación interna, tanto en el caso continuo como en el discreto.

El sistema continuo lineal invariante en el tiempo modelado en representación interna como

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t)$$

se define en Matlab de la siguiente manera:

```
» sys = ss(A,B,C,D)
```

Ejemplo

```
» A=[0 1;-2 -3];
» B=[0 1]';
» C=[6 0];
» D=0;
» sys=ss(A,B,C,D)

a =
                    x1          x2
           x1            0     1.00000
           x2     -2.00000    -3.00000


b =
                    u1
           x1            0
           x2      1.00000


c =
                    x1          x2
           y1      6.00000           0


d =
                    u1
           y1            0

Continuous-time system.
»
```

Para definir el sistema discreto:

$$x[(k+1)T] = Ax(kT) + Bu(kT)$$
$$y(kT) = Cx(kT) + Du(kT)$$

se utilizaría la función **ss** incluyendo el periodo de muestreo (*Ts*):

```
» sysd=ss(A,B,C,D,Ts)
```

La variable *sys* creada es una estructura de datos específica del modelo que permite manipular el sistema como una única entidad.

Para devolver los datos asociados al sistema creado existen dos opciones:

- Utilizar la función **ssdata**:

```
» [A,B,C,D,Ts]=ssdata(sys)
```

- Acceder directamente al miembro de la estructura, por ejemplo:

```
» sys.A
» sys.B
» sys.Ts
```

3.2. Conversión de modelos.

Se ha mostrado en el punto anterior como se crea un modelo en representación interna. Además de utilizar la representación interna, es posible definir un modelo mediante representación externa.

Para crear un modelo para representar la función de transferencia

$$H(s) = \frac{num(s)}{den(s)}$$

se utiliza la siguiente función:

```
» h = tf(num,den)
```

donde *num* y *den* son vectores que especifican los coeficientes del numerador y el denominador.

Si se quiere definir el modelo en representación externa discreta, únicamente hay que utilizar la función **tf** especificando el periodo de muestreo:

```
» hd=tf(num,den,Ts)
```

Para realizar la conversión de un sistema entre diferentes representaciones se utilizan las mismas funciones que se usan para crear los modelos.

- Para obtener el modelo del sistema en representación externa a partir de su modelo en representación interna se ejecutaría en Matlab:

```
» systf=tf(sys)
```

  donde *sys* es el sistema modelado en representación interna y *systf* es el sistema modelado en representación externa.

- Para obtener el modelo del sistema en representación interna a partir de su modelo en representación externa se ejecutaría:

```
» sys=ss(systf)
```

  donde *systf* es el sistema modelado en representación externa y *sys* es el sistema modelado en representación interna.

3.3. Conversión sistemas continuos / sistemas discretos.

Las funciones **c2d**, **d2c** y **d2d** realizan conversiones de continuo a discreto, discreto a continuo y discreto a discreto (asignación de nuevo periodo de muestreo), respectivamente:

```
» sysd=c2d(sysc,Ts)   % Conversión continuo - discreto
» sysc=d2c(sysd)      % Conversión discreto - continuo
» sysd2=d2d(sysd,Ts)  % Modificación periodo muestreo
```

Al discretizar se considera, por defecto, que se añade un retenedor de orden cero en la entrada del sistema. Es posible seleccionar un método de discretización diferente. Consúltese la ayuda para obtener más información.

4. Herramientas de análisis.

Existen multitud de funciones que permiten analizar los sistemas creados, tanto en el caso de sistemas modelados en representación interna, como en el de los modelados en representación externa. A continuación se muestran las funciones más útiles para analizar sistema modelados en el espacio de estados.

4.1. Características generales.

Las siguientes funciones sirven para obtener información relativa a las características generales asociadas a un sistema:

```
» size(sys) % Indica el núm. de entradas, salidas y estados
» isct(sys) % Devuelve 1 si el sistema es continuo
» isdt(sys) % Devuelve 1 si el sistema es discreto
```

4.2. Dinámica del modelo.

Las funciones **pole** y **eig** permiten determinar la estabilidad absoluta del sistema.

```
» pole(sys)      % Calcula los polos del sistema
» eig(sys)       % Calcula los polos del sistema
```

Nota: La función **eig** puede ser invocada pasándole como parámetro únicamente la matriz *A* del sistema, es decir, ejecutando **eig(A)**.

4.3. Funciones específicas para los sistemas modelados en el espacio de estados.

- Función **ss2ss**. Permite realizar transformaciones lineales de estado.

  ```
  » sys = ss2ss(sys,T)
  ```

  La instrucción anterior realiza la transformación $z(t) = Tx(t)$ sobre el vector de estados *x* del sistema *sys*. El sistema resultante es:

$$\dot{z}(t) = TAT^{-1}z(t) + TBu(t)$$
$$y(t) = CT^{-1}z(t) + Du(t)$$

  Nota: Esta transformación es diferente de la definición clásica de transformación lineal:

$$x(t) = Tw(t)$$

$$\dot{w}(t) = T^{-1}ATw(t) + T^{-1}Bu(t)$$
$$y(t) = CTw(t) + Du(t)$$

- Función **ctrb**. Calcula la matriz de controlabilidad del sistema.

  ```
  » Q = ctrb(sys)
  ```

  ```
  » Q = ctrb(A,B)
  ```

  Cualquiera de las instrucciones anteriores calcula la matriz de controlabilidad del sistema:

$$Q = \begin{bmatrix} B & AB & A^2B & \cdots & A^{n-1}B \end{bmatrix}$$

  Para saber si el sistema es controlable, únicamente hay que comprobar si el rango de la matriz *Q* coincide con la dimensión del sistema (*n*). Para obtener el rango de la matriz *Q* utilizaremos la función de Matlab **rank**:

  ```
  » rank(Q)
  ```

- Función **obsv**. Calcula la matriz de observabilidad del sistema.

  ```
  » P = obsv(sys)
  ```

  ```
  » P = obsv(A,C)
  ```

  Cualquiera de las instrucciones anteriores calcula la matriz de observabilidad del sistema:

$$P = \begin{bmatrix} C \\ CA \\ CA^2 \\ \cdots \\ CA^{n-1} \end{bmatrix}$$

  Para saber si el sistema es observable, simplemente hay que comprobar si el rango de la matriz *P* coincide con la dimensión del sistema (*n*). Para obtener el rango de la matriz *P* utilizaremos la función de Matlab **rank**.

- Función **ctrbf**. Descompone un sistema en los subespacios controlable y no controlable. Sin embargo la descomposición que realiza difiere de la definida teóricamente, por lo que no utilizaremos esta función. Consúltese la ayuda de Matlab para obtener más información relativa a esta función.

- Función **obsvf**. Descompone un sistema en los subespacios observable y no observable. Sin embargo, al igual que ocurría con la función anterior, la descomposición que realiza difiere de la definida teóricamente, por lo que tampoco utilizaremos esta función. Consúltese la ayuda de Matlab para obtener más información relativa a esta función.

5. Respuesta temporal del sistema.

Existen diversas funciones que permiten obtener la repuesta del sistema considerando distintos tipos de entradas. A continuación se detallan las más importantes:

- Función **step**. Respuesta ante escalón unitario, asumiendo condiciones iniciales nulas. Este comando puede ejecutarse de dos modos:

  ```
  » step(sys)
  ```
  En este caso muestra una gráfica con la respuesta del sistema *sys*, pero no permite observar la trayectoria de los estados.

  ```
  » [y,t,x]=step(sys);
  ```
  En lugar de representarse la respuesta del sistema se obtiene un vector *y* con la respuesta del sistema, un vector *t* con el tiempo usado para simulación, y una matriz *x* con la trayectoria de los estados.

- Función **initial**. Respuesta del sistema a las condiciones iniciales considerando que la entrada es nula. Al igual que en el caso anterior, este comando puede ejecutarse de dos maneras:

  » initial(sys,x0)
  Muestra una gráfica con la respuesta del sistema *sys* a las condiciones iniciales expresadas en el vector *x0*.

  » [y,t,x]= initial(sys,x0);
  Obtiene un vector *y* con la respuesta del sistema a las condiciones iniciales expresadas en el vector *x0*, un vector *t* con el tiempo usado para simulación, y una matriz *x* con la trayectoria de los estados.

- Función **lsim**. Respuesta ante una entrada arbitraria. Este comando, al igual que en los casos anteriores, puede ejecutarse de dos modos:

  » lsim(sys,u,t,x0)
  En este caso muestra una gráfica con la respuesta del sistema *sys*, pero no permite observar la trayectoria de los estados. El vector *t* indica el tiempo de las señal de entrada, que se expresa en el vector *u*. Las condiciones iniciales se expresan en el vector *x0*. Si se considera condiciones iniciales nulas, no es necesario incluir este parámetro.

  » [y,ts,x]= lsim(sys,u,t,x0);
  En este caso en lugar de representarse la respuesta del sistema se obtiene un vector *y* con la respuesta del sistema, un vector *ts* con el tiempo usado para simulación, y una matriz *x* con la trayectoria de los estados.

El toolbox de control incorpora una interface interactiva que facilita el análisis de la respuesta de los modelos LTI, denominada *LTI Viewer*. Para acceder a esta herramienta simplemente hay que ejecutar en Matlab:

» ltiview

La interface permite obtener la respuesta de cualquier sistema definido ante diferentes señales de entradas.

6. Ejercicios.

Ejercicio 1. Considérese el siguiente sistema continuo definido en representación interna:

$$\begin{bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\\ \dot{x}_3(t)\end{bmatrix}=\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ -50 & -20 & -10\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\\ x_3(t)\end{bmatrix}+\begin{bmatrix}0\\ 0\\ 1\end{bmatrix}u(t)$$

$$y(t)=\begin{bmatrix}30 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\\ x_3(t)\end{bmatrix}$$

Utilizando las funciones oportunas del toolbox de control de Matlab, deben realizarse los siguientes apartados:

1- Representar el sistema como un objeto LTI (*Linear Time-Invariant*).
2- Obtener en Matlab la respuesta del sistema continuo, así como la trayectoria de los estados, considerando una entrada en escalón unitario y condiciones iniciales nulas.
3- Simular el mismo sistema en Simulink y comprobar que se obtienen los mismos resultados que utilizando Matlab.
4- Discretizar el modelo continuo considerando un periodo de 0.1 segundos y un retenedor de orden cero a la entrada del sistema.
5- Obtener en Matlab la respuesta del sistema discreto, así como la trayectoria de los estados, considerando una entrada en escalón unitario y condiciones iniciales nulas.
6- Simular el sistema discreto en Simulink y comprobar que se obtienen los mismos resultados que utilizando Matlab.
7- Comparar la respuesta del modelo continuo y discreto, así como la trayectoria de estados.

Ejercicio 2. Estudiar la estabilidad absoluta de los siguientes sistemas:

(i)

$$\begin{bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\\ \dot{x}_3(t)\end{bmatrix}=\begin{bmatrix}3 & 1 & 2\\ 4 & 0 & 1\\ 5 & 6 & 0\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\\ x_3(t)\end{bmatrix}+\begin{bmatrix}0\\ 0\\ 1\end{bmatrix}u(t)$$

$$y=\begin{bmatrix}2 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\\ x_3(t)\end{bmatrix}$$

(ii)

$$\begin{bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\end{bmatrix}=\begin{bmatrix}0 & 1\\ -3 & -5\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\end{bmatrix}+\begin{bmatrix}0\\ 1\end{bmatrix}u(t)$$

$$y=\begin{bmatrix}10 & 0\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\end{bmatrix}$$

Ejercicio 3. Determinar si los siguientes sistemas son controlables.

(i)

$$\begin{bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\\ \dot{x}_3(t)\end{bmatrix}=\begin{bmatrix}-1 & 1 & 0\\ 0 & -1 & 0\\ 0 & 0 & -2\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\\ x_3(t)\end{bmatrix}+\begin{bmatrix}0\\ 4\\ 3\end{bmatrix}u(t)$$

(ii)

$$\begin{bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\\ \dot{x}_3(t)\\ \dot{x}_4(t)\\ \dot{x}_5(t)\end{bmatrix}=\begin{bmatrix}-2 & 1 & 0 & 0 & 0\\ 0 & -2 & 1 & 0 & 0\\ 0 & 0 & -2 & 0 & 0\\ 0 & 0 & 0 & -5 & 1\\ 0 & 0 & 0 & 0 & -5\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\\ x_3(t)\\ x_4(t)\\ x_5(t)\end{bmatrix}+\begin{bmatrix}4\\ 2\\ 1\\ 3\\ 0\end{bmatrix}u(t)$$

(iii)

$$\begin{bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\end{bmatrix}=\begin{bmatrix}-1 & 0\\ 0 & -2\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\end{bmatrix}+\begin{bmatrix}2\\ 0\end{bmatrix}u(t)$$

Ejercicio 4. Separar los sistemas del ejercicio 3 que no sean completamente controlables en los subsistemas controlable y no controlable.

Ejercicio 5. Determinar la observabilidad de los siguientes sistemas:

(i)

$$\begin{bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\\ \dot{x}_3(t)\end{bmatrix}=\begin{bmatrix}2 & 1 & 0\\ 0 & 2 & 1\\ 0 & 0 & 2\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\\ x_3(t)\end{bmatrix}+\begin{bmatrix}1\\ 0\\ 1\end{bmatrix}u(t)$$

$$\begin{bmatrix}y_1(t)\\ y_2(t)\end{bmatrix}=\begin{bmatrix}3 & 0 & 0\\ 4 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\\ x_3(t)\end{bmatrix}$$

(ii)

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \\ \dot{x}_5(t) \end{bmatrix} = \begin{bmatrix} 2 & 1 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & -3 & 1 \\ 0 & 0 & 0 & 0 & -3 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \\ 1 \\ 0 \\ 0 \end{bmatrix} u(t)$$

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \end{bmatrix}$$

(iii)

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -2 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u(t)$$

$$y = \begin{bmatrix} 1 & 3 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

Ejercicio 6. Separar los sistemas del ejercicio 5 que no sean completamente observables en los subsistemas observable y no observable.

Ejercicio 7. Determinar que sistemas son controlables y observables al mismo tiempo en el ejercicio 5. Realizar una separación simultánea de los subsistemas controlable y observable en aquellos sistemas que no lo sean.

Importante

- Debe entregarse un informe detallando cada uno de los ejercicios realizados en la práctica.
- El plazo de entrega del informe de la práctica finaliza el día 9 de mayo de 2003.