Escuela Politécnica Superior de Elche

CONTROL AVANZADO DE SISTEMAS
CONTROL INTELIGENTE
4º Ingeniería Industrial

PRÁCTICA 1:
Diseño de reguladores borrosos

Departamento de Ingeniería de Sistemas Industriales
Área de Ingeniería de Sistemas y Automática

CONTROL AVANZADO DE SISTEMAS

PRÁCTICA 1: Diseño de reguladores borrosos

1. Objetivos

Los objetivos de esta práctica son:

1. Aprender a manejar una herramienta de diseño de reguladores borrosos.
2. Comparar el comportamiento de un sistema borroso ante distintos valores de sus parámetros.
3. Utilizar un regulador borroso para controlar un sistema continuo sencillo.

2. Introducción

Un sistema borroso puede simularse e implementarse directamente usando un lenguaje de programación como el que incorpora Matlab. En esta práctica no vamos a implementar el sistema borroso completo, sino que se va a utilizar una *toolbox* ya construida que permite definir e implementar distintos sistemas borrosos. Por este motivo es necesario comenzar introduciendo el manejo de esta herramienta que nos va a permitir analizar el comportamiento de un sistema borroso y su utilización como regulador en un bucle de control.

3. *toolbox* de lógica borrosa

En Matlab existe una *toolbox* dedicada a los sistemas borrosos. Esta toolbox permite definir un sistema borroso por medio de diálogos y ventanas que facilitan la introducción de los datos. Además, dispone de un conjunto de funciones para analizar el comportamiento de dichos sistemas. A lo largo de este apartado se estudiarán las etapas que se deben seguir en la edición o modificación de un sistema borroso.

3.1 Estructura de datos

Matlab almacena toda la información relevante de un sistema borroso (funciones de pertenencia, reglas, métodos de implicación y desborrosificación) en una matriz siguiendo un determinado formato. Dicha matriz puede almacenarse en la memoria de trabajo y referenciarse mediante una variable de Matlab, o bien almacenarse en un fichero que puede ser cargado directamente en memoria, cargado en el editor de sistemas borrosos o utilizado en una simulación de *Simulink*.

Por ejemplo, si queremos crear el sistema borroso estudiado en teoría, podemos ejecutar los siguientes comandos:

```
a = newfis('ejemplo');
a = addvar(a,'input','x',[0 8]);
a = addmf(a,'input',1,'pequeño','trapmf',[0 0 1 3]);
a = addmf(a,'input',1,'medio','trapmf',[1 3 3 7]);
a = addmf(a,'input',1,'grande','trapmf',[3 7 8 8]);
 showfis(a)
```

El comando ***newfis*** crea un sistema borroso nuevo (*fis* es la abreviatura de *fuzzy inference system*). El comando ***addvar*** añade una variable de entrada o salida al sistema. Tiene 4 parámetros:

- El nombre de la variable de Matlab en la que está almacenado el sistema borroso.
- Una cadena que indica si la variable que queremos introducir es de entrada (‘input’) o de salida (‘output’).
- El nombre de la variable que queremos añadir.
- Un vector que indica el rango de valores que puede tomar la variable.

El comando ***addmf*** añade una función de pertenencia a un sistema borroso (*mf* es la abreviatura de *membership function*). Esta función tiene 6 parámetros:

- El nombre de la variable de Matlab en la que está almacenado el sistema borroso.
- Una cadena que indica si la variable que queremos introducir es de entrada (‘input’) o de salida (‘output’).
- El índice de la variable a la que se quiere añadir la función de pertenencia.
- Una cadena que representa el nombre de la nueva función de pertenencia.
- Una cadena que representa el tipo de la nueva función de pertenencia.
- El vector de parámetros que define la función de pertenencia.

En nuestro ejemplo se han añadido las tres funciones de pertenencia correspondientes a la variable de entrada x. Estas tres funciones son trapezoidales (*trapmf*), y para definir una función de este tipo es necesario especificar un vector con 4 parámetros: los 4 puntos que definen sobre el eje X el trapecio.

La estructura del sistema borroso se guarda en una matriz, en nuestro ejemplo en la matriz del entorno ‘a’. El comando ***showfis*** nos muestra el contenido de la matriz ya interpretado.

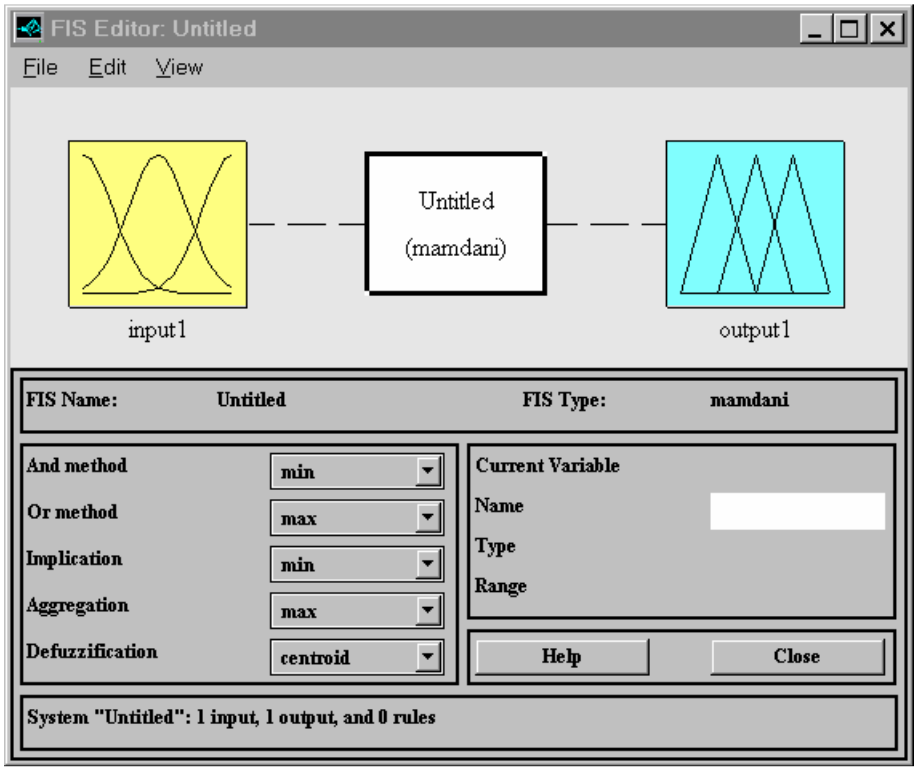
Esta toolbox dispone de un conjunto de funciones para la creación de conjuntos borrosos como los que se han estudiado en teoría. Los nombres de las funciones más significativas son:

Función	Expresión
Gaussiana	gaussmf(dominio, [σ, centro])
forma π	pimf(dominio, [a, b, c, d])
Sigmoide	sigmf(dominio, [amplitud, centro])
forma S	smf(dominio, [a, b])
Trapezoidal	trapmf(dominio,[a, b, c, d])
Triangular	trimf(dominio, [a, b, c])
forma Z	zmf(dominio, [a, b])

Como la edición de los parámetros de un sistema borroso es muy costosa si se utilizan directamente funciones de Matlab como las descritas, existe un interfaz de ventanas que permite la edición de las funciones de pertenencia, de la base de reglas y de las características de dicho sistema.

3.2 El editor del sistema borroso

Para abrir la ventana de edición del sistema borroso, puede ejecutarse el comando ***fuzzy***, si se quiere crear un sistema nuevo, o ***fuzzy(nombre_fis)*** si se quiere editar el sistema borroso definido en la matriz almacenada en la variable *nombre_fis*, o ***fuzzy('nombre_fichero')*** si se quiere editar el sistema almacenado en el fichero *nombre_fichero*. Aparece entonces la siguiente ventana:



La primera acción a realizar es definir el número de entradas y salidas que debe tener el sistema borroso utilizando la opción *Edit->Add input* o *Edit->Add output*.

En nuestro primer ejemplo se va a trabajar con el sistema más sencillo, con una sola entrada y salida (sistema SISO), por lo que el número de variables que aparecen por defecto es el correcto.

Se pueden cambiar los nombres de las variables de entrada y salida seleccionando el icono que representa la variable correspondiente y modificando el valor que aparece en el campo *Name*. **Modificarlos para que los nombres de las variables sean x e y.**

En esta ventana se declaran las características globales que se asumen para todo el sistema a través de las listas desplegables correspondientes :

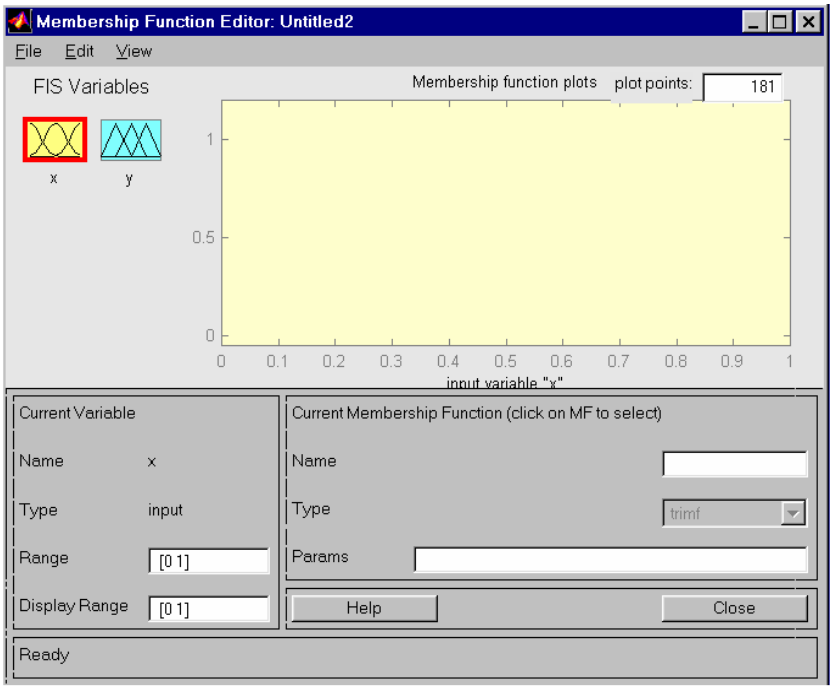
- t-norma utilizada en antecedentes: *And method*
- s-norma: *Or method*
- Método de implicación : *Implication*
- Método de composición o agregación : *Aggregation*
- Método de desborrosificación: *Defuzzification*

Debe tenerse en cuenta que siempre se supone que el motor de inferencia opera en el modo de reglas individuales, y que el borrosificador no puede ser seleccionado, siendo siempre de tipo unitario o *singleton*, aplicado a la entrada del sistema.

En nuestro ejemplo, vamos a dejar todos los parámetros con sus valores por defecto.

3.3 El editor de funciones de pertenencia

A continuación se deben definir las funciones de pertenencia de cada variable. Para ello se realiza una doble pulsación sobre uno de los iconos de las variables o se activa la opción *View->Edit membership functions*, con lo que se abre la siguiente ventana:



Lo primero que se debe hacer es declarar el dominio de cada variable, que en este caso es un intervalo que se introduce en el campo *Range*. El campo *Display Range* indica el rango que se utilizará en el subpanel que muestra las funciones de pertenencia**.** Siguiendo el ejemplo visto en clase, **introduciremos un rango comprendido entre 0 y 8**, tanto para la variable de entrada como para la de salida.

Para añadir funciones de pertenencia seleccionar la opción *Edit->Add MFs*. A través del cuadro de diálogo que aparece se define el tipo de función elegido y el número de funciones que se van a utilizar para esa variable. Al aceptar los datos se creará una distribución uniforme de funciones de pertenencia con un solapamiento del 100% sobre el dominio definido. **En nuestro ejemplo introduciremos 3 funciones trapezoidales para las variables x e y, y posteriormente ajustaremos sus parámetros.**

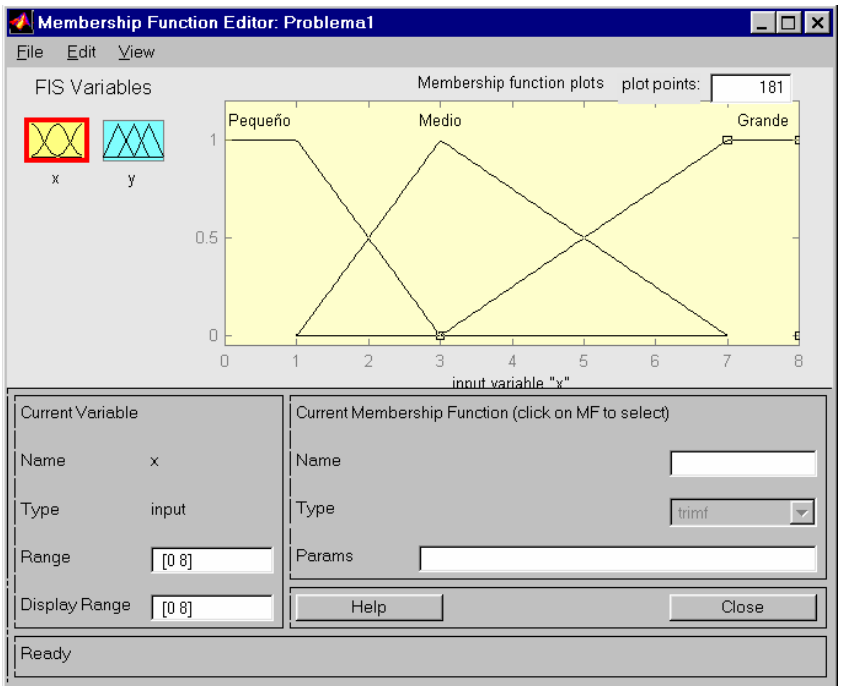
Para modificar el nombre o los parámetros de alguna función, se debe seleccionar primero dicha función con el ratón, apareciendo sus propiedades en el panel titulado *Current Membership Function*, donde pueden ser modificadas a través de los

campos correspondientes. También pueden modificarse arrastrando la función con el ratón directamente sobre su gráfica.

En nuestro caso, los nombres y parámetros de las funciones de pertenencia para la variable de entrada *x* aparecen en la tabla siguiente:

nombre	**tipo**	**parámetros**
Pequeño	trapmf	[0 0 1 3]
Medio	trapmf	[1 3 3 7]
Grande	trapmf	[3 7 8 8]

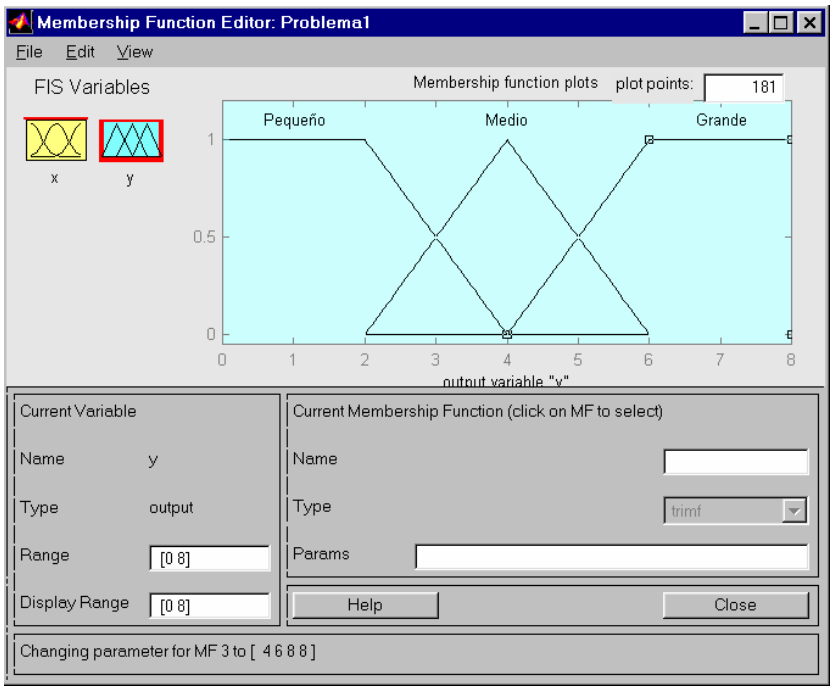
Las gráficas obtenidas deben ser las que aparecen a continuación:



Los nombres y parámetros de las funciones para la variable de salida *y* aparecen en la tabla siguiente:

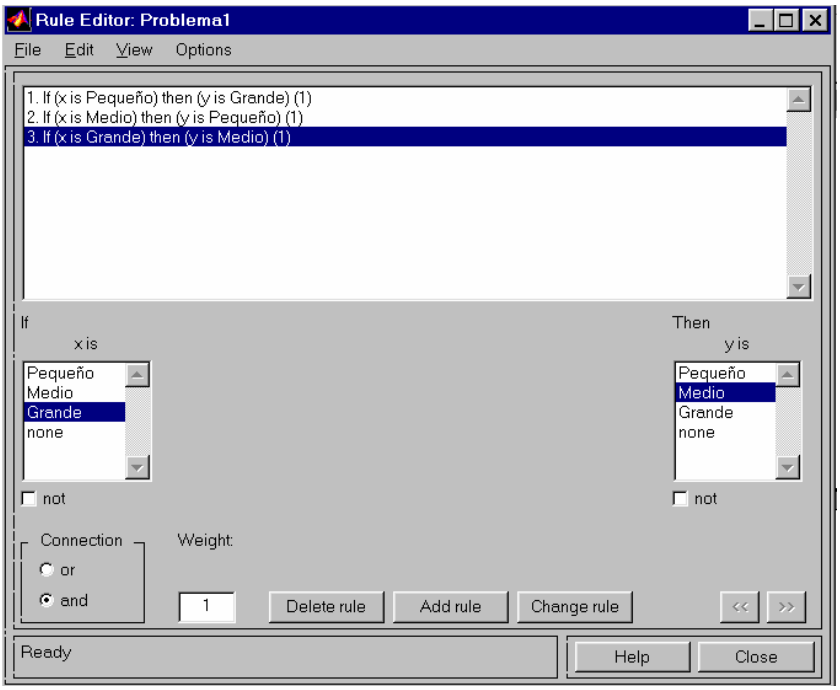
nombre	**tipo**	**parámetros**
Pequeño	trapmf	[0 0 2 4]
Medio	trapmf	[2 4 4 6]
Grande	trapmf	[4 6 8 8]

Las gráficas obtenidas deben ser las que aparecen a continuación:



3.4 El editor de reglas

Por último se editan las reglas utilizando el editor correspondiente que se abre mediante la opción *View->Edit Rules* o haciendo una doble pulsación sobre el icono que representa la base de reglas en la ventana principal. La ventana que se abre, es simplemente un panel de edición donde se editan las reglas siguiendo un formato prefijado. **En nuestro ejemplo, introduciremos las tres reglas que aparecen en la figura siguiente.**



Si está seleccionado el formato de la regla *verbose* y el idioma inglés (en *Options*), el formato de la regla es :

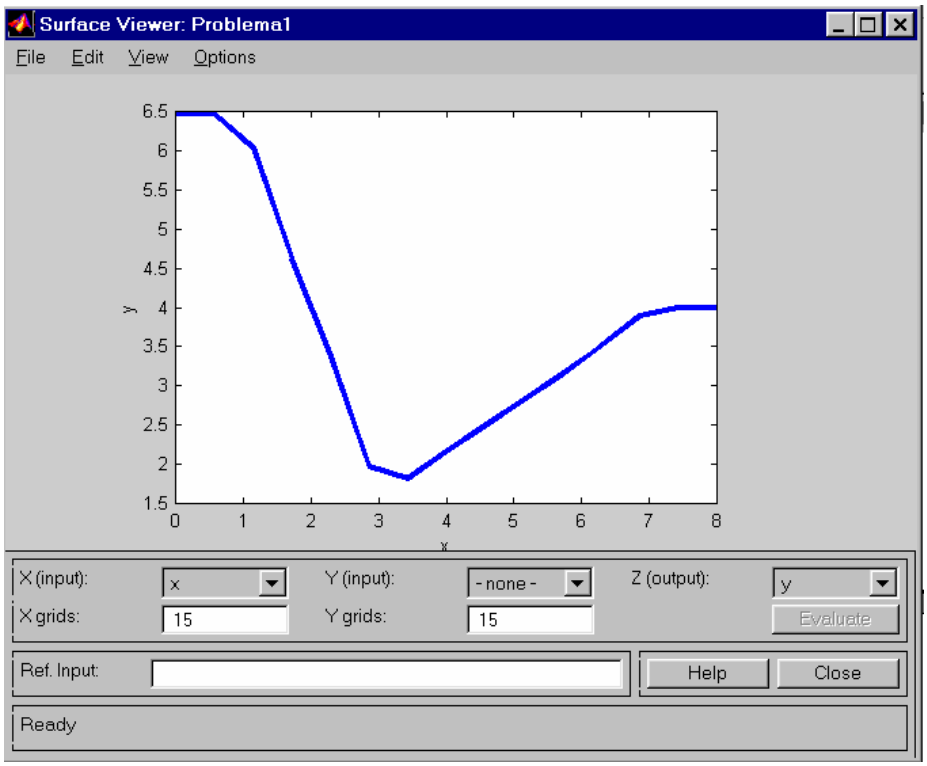
If (Nombre_Var **is** nombre_conjunto) {**and/or** (...)} **then** (Nom_Var **is** nom_conjunto) (Confianza_regla)

donde *Confianza_regla* es un valor del intervalo [0,1] que refleja la confianza que se tiene sobre esa regla de control, que se refleja en un peso de la misma sobre el valor de salida del regulador.

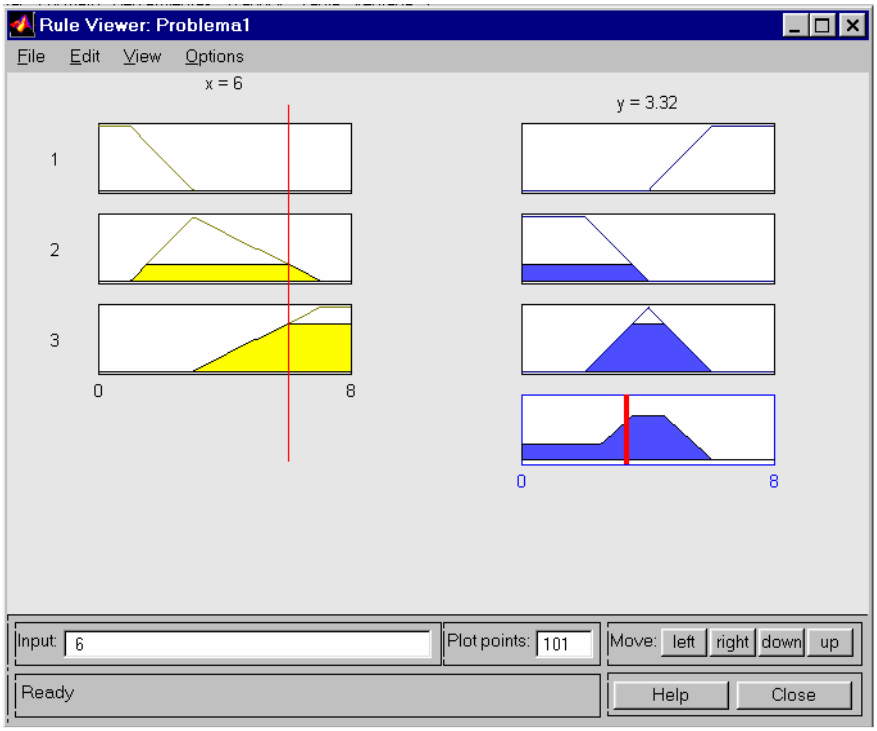
Puede probarse cómo quedan las reglas con distintos formatos. En el formato *Indexed*, el número que aparece representa el conjunto borroso que está en el lugar *i*-ésimo numerando de izquierda a derecha los conjuntos que se han definido en cada variable. Obsérvese que es posible escribir reglas en forma no canónica, mediante la s-norma “or” como conexión.

3.5 Análisis entrada-salida del comportamiento del sistema

Para analizar el comportamiento del sistema borroso que se acaba de definir se dispone de dos ventanas. La primera permite ver gráficamente la transformación del espacio de entrada en el de salida. Ésta se abre mediante la opción *View->view surface*. En nuestro ejemplo el sistema sólo tiene una entrada y una salida, por lo que la superficie entrada-salida del sistema es una gráfica en dos dimensiones, como aparece en la figura siguiente.



La segunda forma de visualizar el comportamiento del sistema permite analizar cómo se obtiene la conclusión de una forma interactiva para distintos valores de entrada utilizando la opción *View->View rules*. Los cambios realizados en los editores correspondientes se reflejarán de forma automática en estas dos ventanas.



3.6 Gestión de los sistemas borrosos

La información editada en estas ventanas se puede almacenar en un fichero mediante la opción *File->save to disk* o *File->save to disk as...* La extensión de los ficheros es '.fis'. También se puede almacenar la información en una variable de matlab mediante las opciones *File->save to workspace*. En este último caso hay que recordar que si se sale de matlab se perderá la información editada, a no ser que se almacene en un fichero de alguna forma.

ATENCIÓN : La primera vez que se salva un sistema borroso nuevo hay que salvarlo con la opción *Save as*, pues de lo contrario se creará un fichero con el nombre *untitle.fis*

Para abrir y modificar sistemas ya editados se dispone de las opciones *File->Open from disk* o *File->open from workspace*.

Finalmente, si se quiere comenzar a editar un nuevo sistema se dispone de dos opciones :

- *File->New Mamdani FIS* que abre un sistema tipo ***Mamdani***, donde se definen funciones de pertenencia para las variables de salida.

- *File->New Sugeno FIS* para editar un sistema borroso tipo ***Sugeno***, que estudiaremos en próximas sesiones de teoría, donde los consecuentes son funciones lineales de las variables de entrada.

4. Ejercicio 1: Análisis del comportamiento de un sistema borroso.

En este ejercicio se pretende analizar las consecuencias que se producen al modificar distintos parámetros en los sistemas borrosos.

- Se pide como ejercicio comparar cómo se modifica el comportamiento del sistema borroso editado al cambiar los siguientes parámetros :

  1. Utilización de las funciones mínimo y producto como funciones de *implicación.*

  2. Utilización de las funciones máximo y suma como métodos de *agregación*.

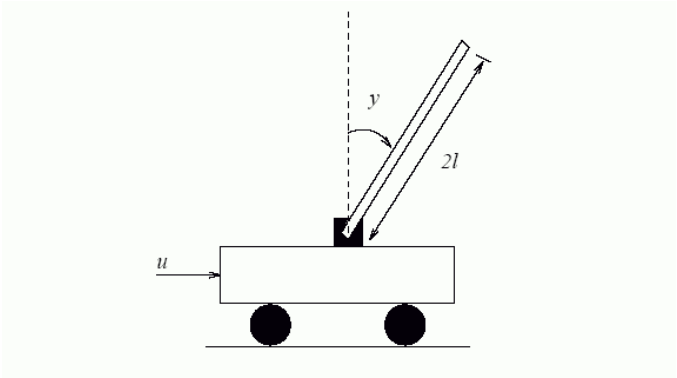
  3. Comparación de los resultados utilizando el centro de las áreas o la media ponderada de los centros como mecanismo de *desborrosificación.*

  4. Modificación de una o varias *reglas*.

  5. Modificación de las *funciones de pertenencia* : trapezoidal, triangular gaussiana.

5. Ejercicio 2: Control de un péndulo invertido

En este ejercicio se diseñará un controlador borroso para equilibrar un péndulo invertido en la posición deseada con la ayuda del Toolbox de Matlab y Simulink. La figura siguiente muestra el péndulo invertido sobre un carro, en la que *y* denota el ángulo que forma el péndulo con la vertical (en radianes, sentido positivo igual al de las agujas del reloj), *l* es la mitad de la longitud del péndulo (en metros) y *u* es la fuerza de entrada que mueve el carro (en Newtons).

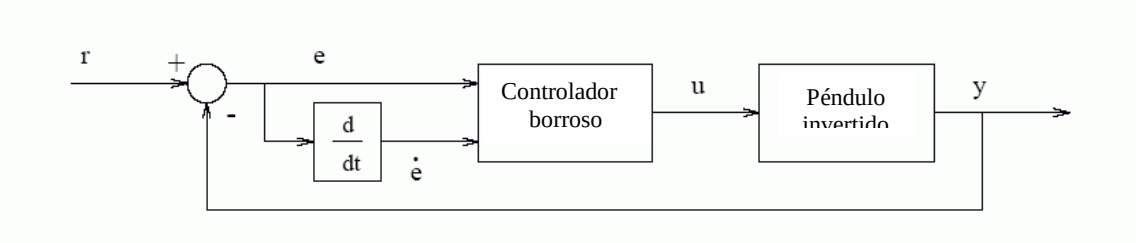


Denotaremos por *r* la posición angular deseada del péndulo. La dinámica del péndulo invertido viene dada por:

$$\ddot{y} = \frac{9.8\text{sen}(y) + \cos(y)\left[\dfrac{\bar{u} - 0.25\dot{y}^2\text{sen}(y)}{1.5}\right]}{0.5\left[\dfrac{4}{3} - \dfrac{1}{3}\cos^2(y)\right]}$$

$$\dot{\bar{u}} = -100\bar{u} + 100u$$

El sistema de control se muestra en la figura siguiente



El controlador borroso tendrá dos entradas: el error $e(t) = r(t) - y(t)$ y la derivada del error $\dot{e}(t)$ y una salida, $u(t)$. Consideraremos inicialmente que

$$e(t) \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right],$$

$$\dot{e}(t) \in \left[-\frac{\pi}{4}, \frac{\pi}{4}\right], \text{ y}$$

$$u(t) \in [-60, 60],$$

Debe elegirse un número de funciones de pertenencia para cada intervalo (al menos 5), escribiendo a continuación la matriz de reglas. Utilizar Simulink para obtener la respuesta del sistema para el controlador diseñado.

En la página web de la asignatura está disponible un modelo del sistema en Simulink así como un documento con más indicaciones sobre cómo realizar el diseño del controlador.

➢ En el informe de prácticas debe entregarse, al menos, lo siguiente:

- El modelo en Simulink utilizado incluyendo el regulador y el modelo del proceso.
- El fichero con extensión .fis con el regulador diseñado.
- Una breve explicación de los experimentos realizados hasta obtener el regulador final.
- Gráficas del ángulo de salida del péndulo y señal de control para las condiciones iniciales
  a) $y(0) = 0.1,\ \dot{y}(0) = 0$.
  b) $y(0) = \frac{\pi}{3},\ \dot{y}(0) = 0$.
  c) $y(0) = \frac{\pi}{6},\ \dot{y}(0) = \frac{\pi}{6}$
- Gráficas del ángulo de salida del péndulo y señal de control para las ganancias $g_0 = 1,\ g_1 = 0.1,\ h = 1$ (véase el documento anexo).
- Si para alguno de los casos a), b), c) anteriores no funciona correctamente el regulador diseñado, sintonizarlo por medio de las ganancias g_0, g_1, h, según se indica en el documento anexo.
- Una breve explicación de las reglas y los parámetros más significativos del regulador.