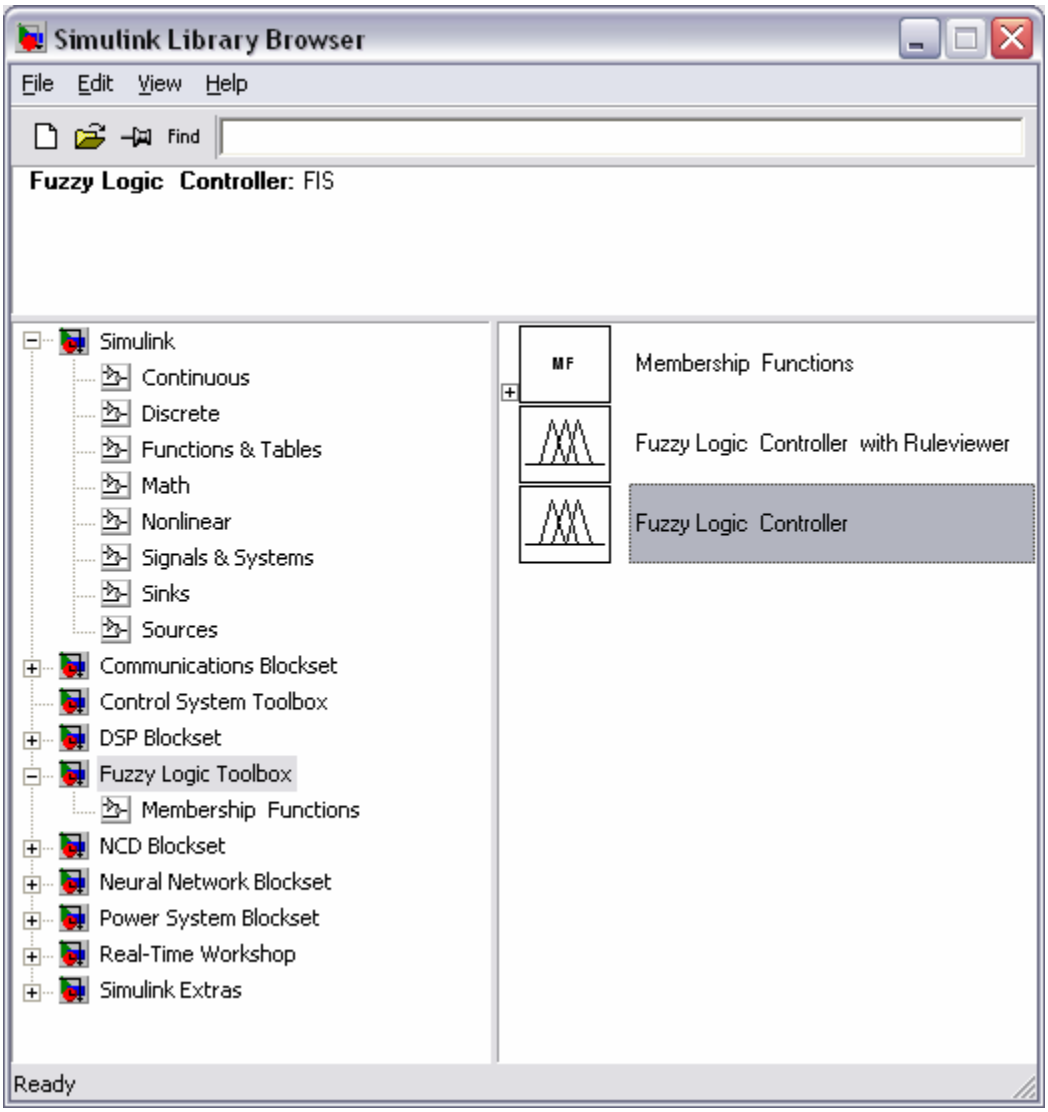
Utilisation de la logique floue: contrôleur en Simulink

Le développement d'un contrôleur en logique floue pour intégration dans Simulink est très simple si on possède la boîte à outils *Fuzzy Logic* de Matlab. Cette boîte à outils ajoute les composantes nécessaires dans Simulink.

Ce guide sert seulement pour l'intégration de la logique floue dans Simulink; ce n'est pas un guide sur le design de contrôleurs PID.

Pour commencer Simulink, il suffit de taper la commande **simulink** à la ligne de commandes de Matlab. Le contrôleur à logique floue est situé sous les options *Fuzzy Logic Toolbox*, comme montré à la figure ci-dessous:



La composante *Fuzzy Logic Controller* est celle utilisée. La composante *Fuzzy Logic Controller with Ruleviewer* est presque la même chose, sauf que l'éditeur qui permet de voir l'inférence et la défuzzification s'ouvre lors d'une simulation.

Pour mieux illustrer le concept d'un contrôleur flou, on comparera la performance d'un contrôleur PID classique avec le contrôleur flou. Le système sous étude est le suivant:

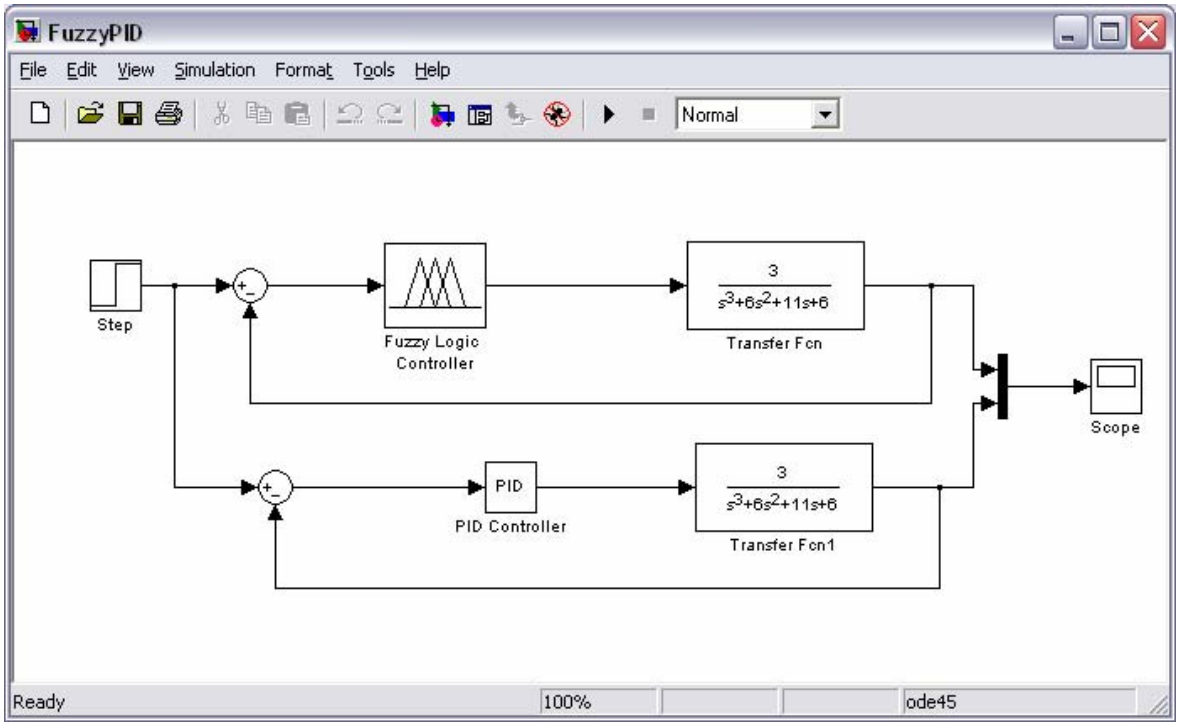
$$G(s) = \frac{3}{(s+1)(s+2)(s+3)}$$

En utilisant les méthodes classiques, on trouve que le contrôleur PID idéal a les valeurs suivantes:

K_P	12
K_I	12
K_D	3

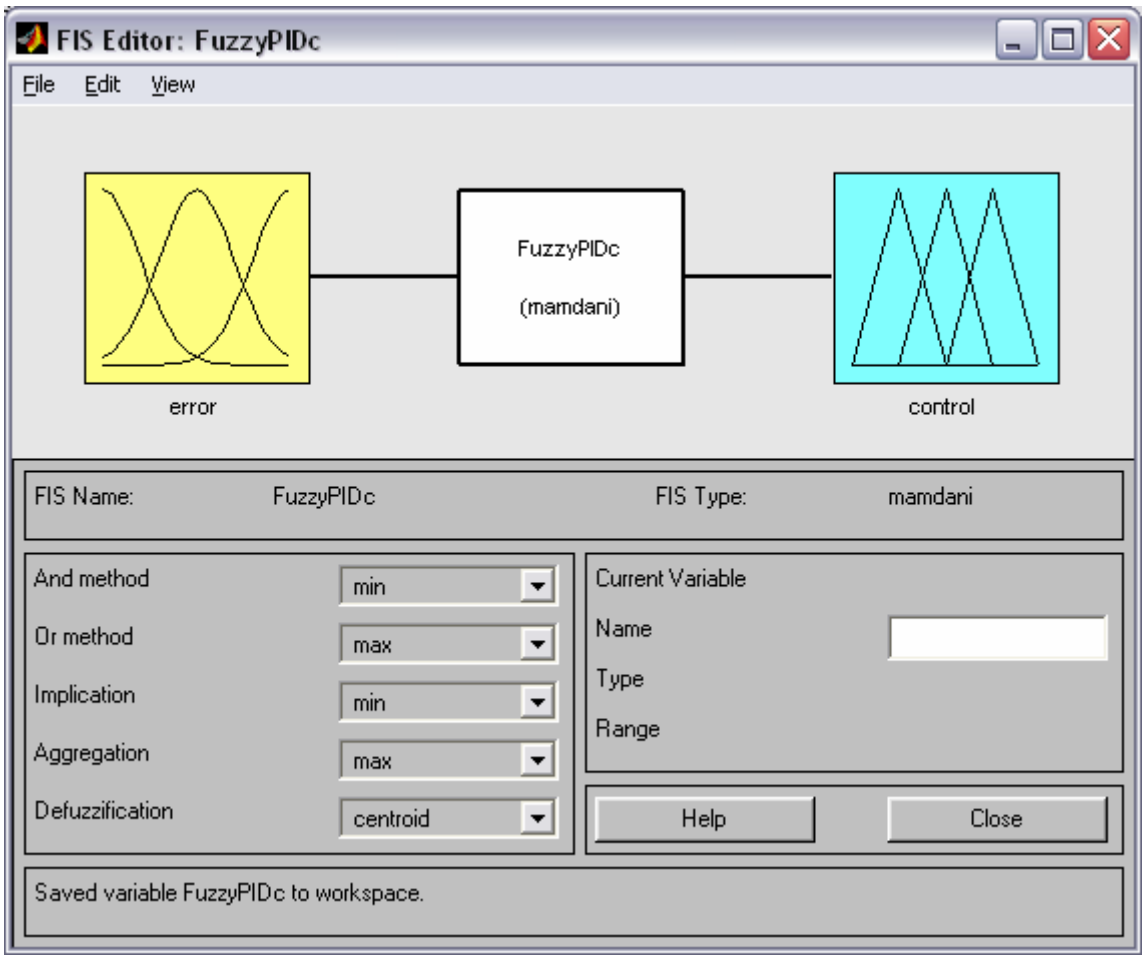
On utilisera la réponse à une entrée échelon du système avec PID pour comparer la performance.

Le système avec contrôleur flou et contrôleur PID devrait être le suivant:



Le design du contrôleur flou se fait de la même façon que tous les systèmes flous avec Matlab. On commence en tapant **fuzzy** à la ligne de commande. On obtient alors le *FIS Editor* de Matlab, montré à la figure suivante.

Pour le système simple sous étude, on utilise que l'erreur comme variable de contrôle. On a donc un système très simple, avec une entrée, une sortie. Dans plusieurs systèmes flous, on utilise aussi la dérivée de l'erreur, et on obtient alors un système à deux entrées, une sortie.



Il faut utiliser un peu de jugement dans le design du contrôleur flou. Il ne faut pas simplement créer des fonctions d'appartenance de toute façon. Il faut quand même étudier le système en détail.

Une chose importante à remarquer est la valeur finale du système en boucle ouverte. Ceci déterminera la sortie du contrôleur quand l'erreur est nulle. Pour le système sous étude, la valeur finale due à une entrée échelon unitaire est:

$$y(\infty) = \lim_{s\to 0} sR(s)G_0(s) = \lim_{s\to 0} s \cdot \frac{1}{s} \cdot \frac{3}{(s+1)(s+2)(s+3)} = \frac{1}{2}$$

La valeur finale est donc 0.5. Le contrôleur devra donc avoir un gain de

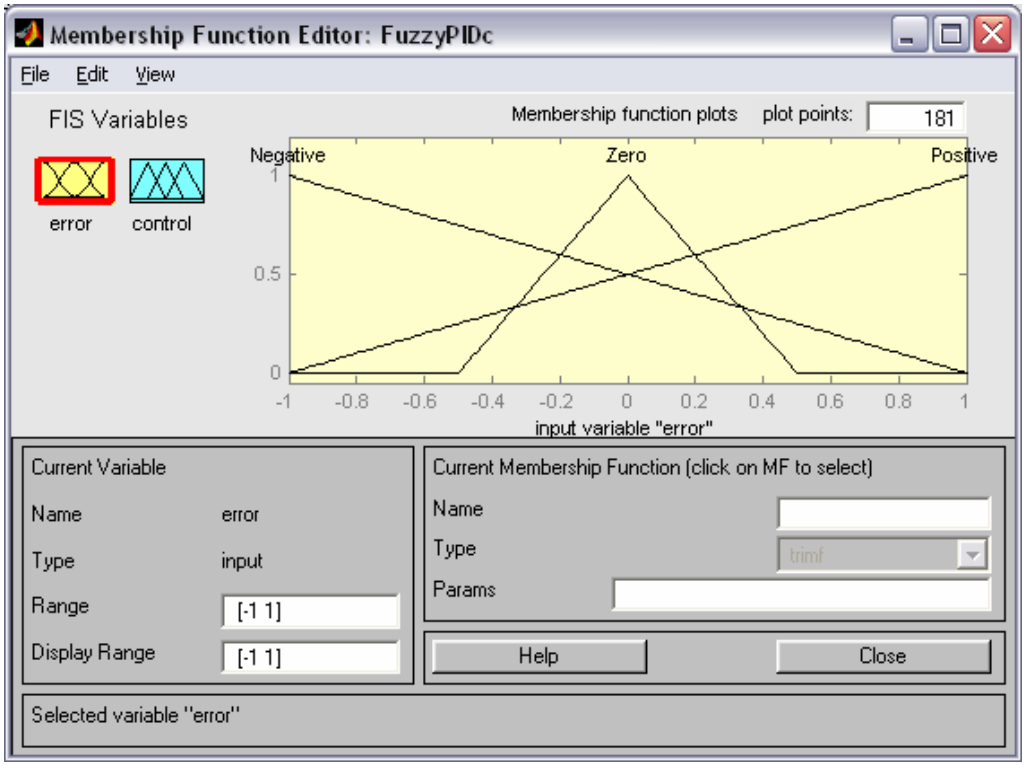
$$K_c = \frac{1}{y(\infty)} = 2$$

C'est-à-dire que le gain du contrôleur flou lorsque l'erreur (l'entrée) est 0 est 2.

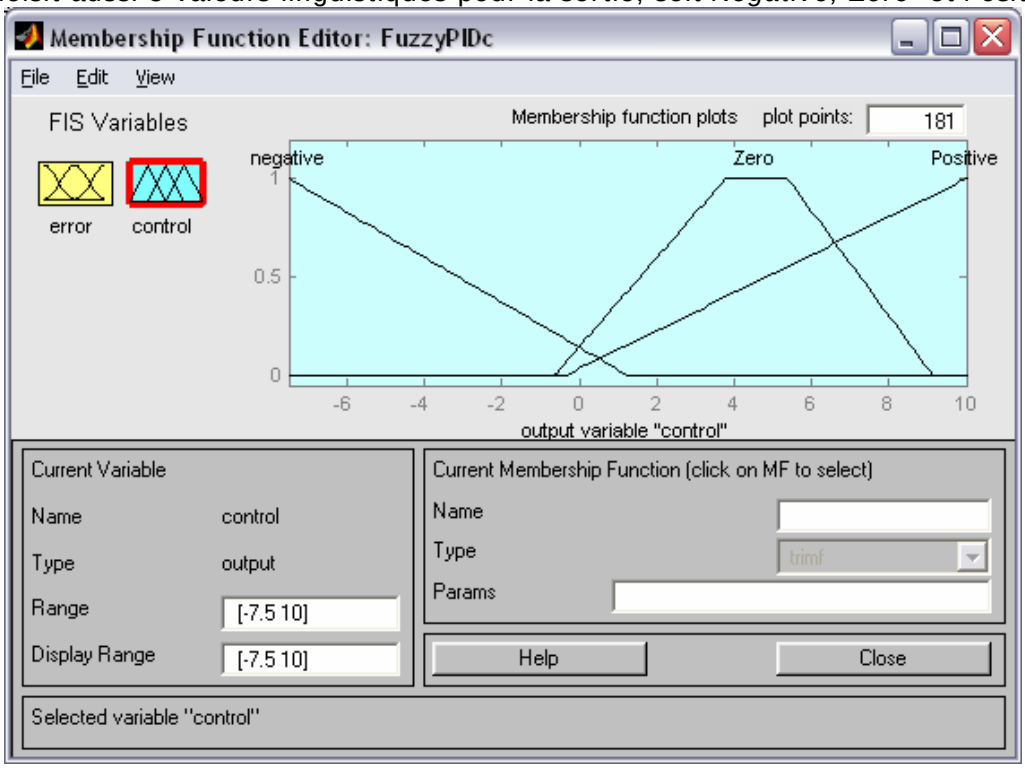
Il faut aussi bien choisir les limites du système (limites de l'entrée et de la sortie). Dans ce cas-ci, l'entrée utilisée est une entrée échelon unitaire. On choisit donc de développer les fonctions d'appartenance du contrôleur flou pour l'entrée entre -1 et +1.

On a choisit que 3 valeurs linguistiques possibles, soit *Negative, Zero* et *Positive*. Il est généralement plus facile de commencer avec un nombre restreint de valeurs

linguistiques possibles, pour simplifier le design initial. Par après, si la précision demandée est plus grande, on peut ajouter des valeurs linguistiques.



On choisit aussi 3 valeurs linguistiques pour la sortie, soit *Negative, Zero* et *Positive.*

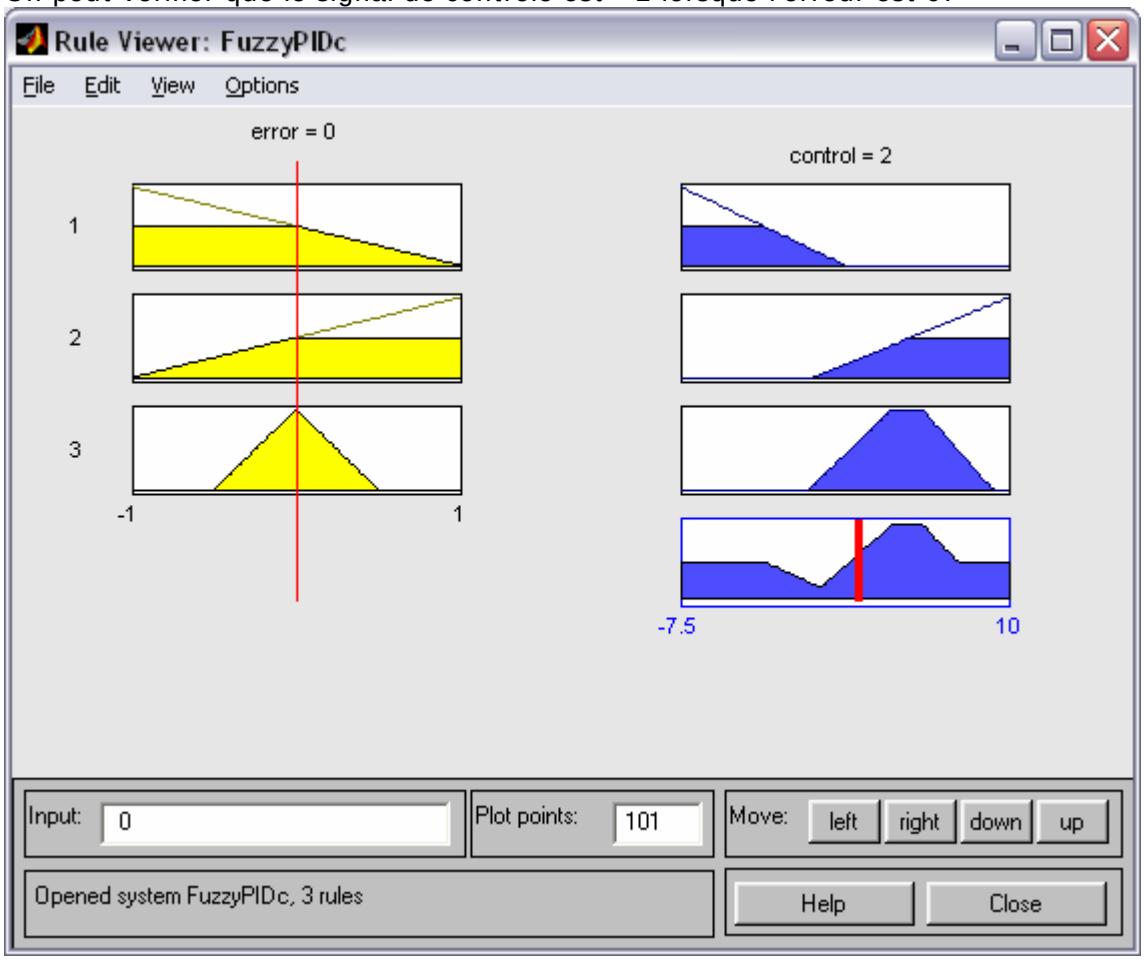


Remarquez que l'intervalle de la sortie est de -7.5 à +10. Le signal de sortie peut être bien plus grand que le signal d'entrée afin de rapidement corriger l'erreur. Cependant, si le signal de contrôle est trop élevé, il y aura un plus grand dépassement.

Les règles utilisées sont:

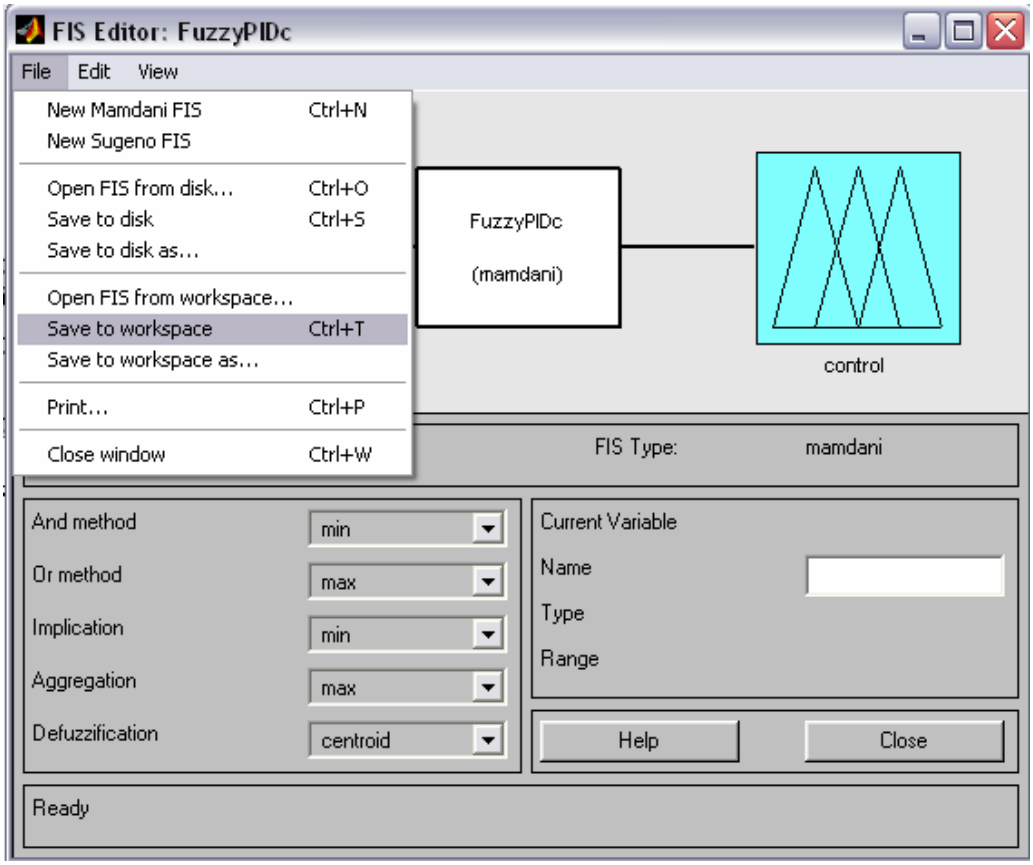
1. IF error is Negative THEN control is Negative
2. IF error is Zero THEN control is Zero
3. IF error is Positive THEN control is Positive

On peut vérifier que le signal de contrôle est +2 lorsque l'erreur est 0:

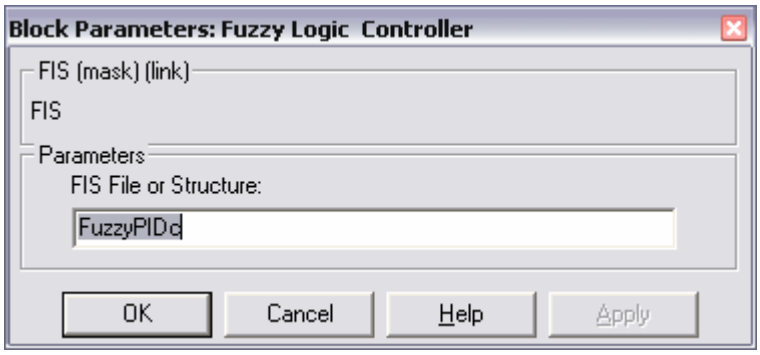


Il faut sauvegarder le système flou dans l'espace de travail afin de pouvoir l'utiliser dans Simulink.

On choisit l'option **File → Save to workspace**, ou CTRL+T.

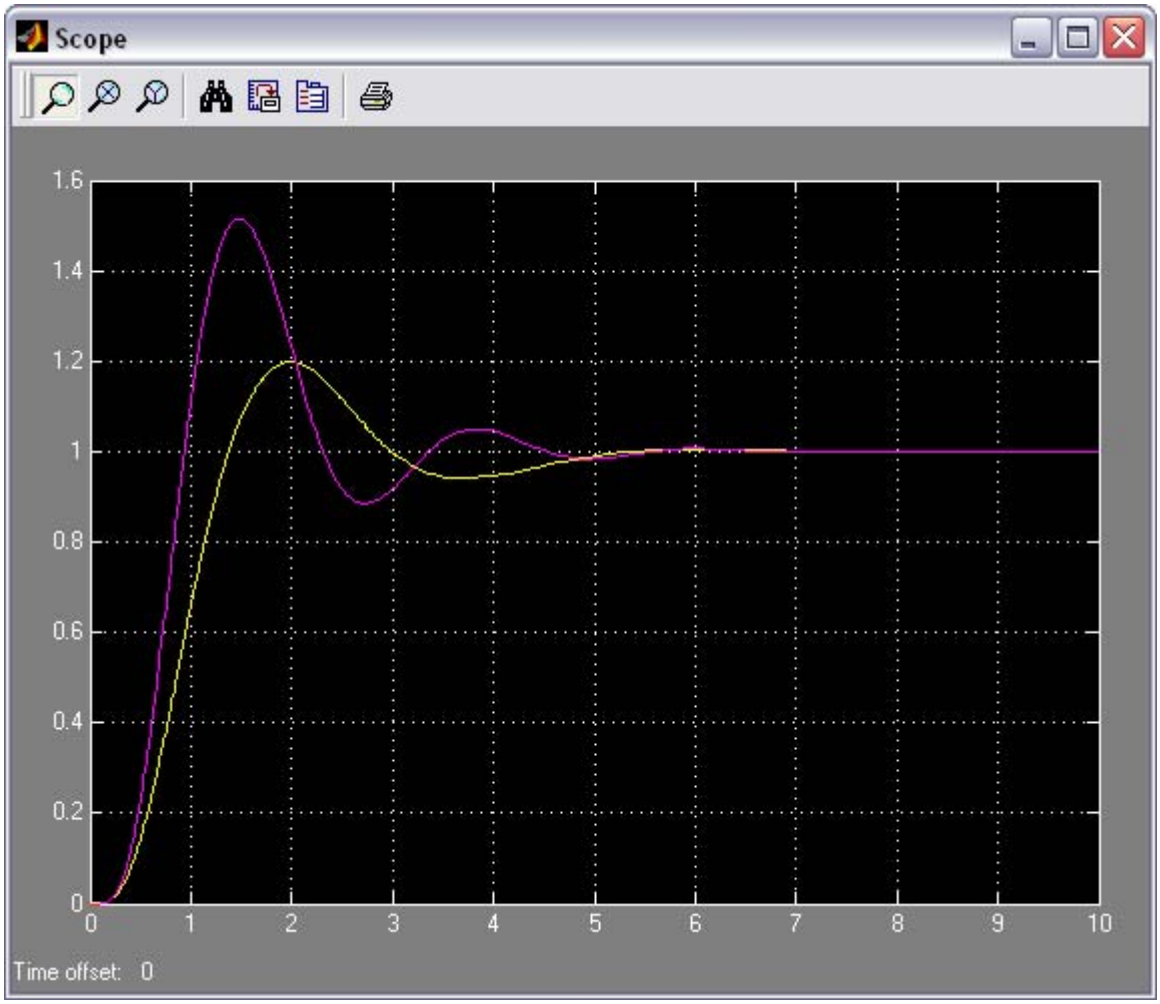


On peut maintenant utiliser le contrôleur flou dans Simulink. Aller sur l'icône *Fuzzy Logic Controller* dans la fenêtre Simulink, et double-cliquez dessus. Vous devriez avoir:



Entrez le nom du fichier (dans ce cas-ci, *FuzzyPIDc*).

Il ne reste qu'à simuler le système et comparer avec le PID.



La courbe en violet représente le PID, et la courbe en jaune le contrôleur flou. Les deux contrôleurs ont le même temps de stabilisation, mais le contrôleur flou a un plus petit dépassement maximal.

On pourrait encore modifier les fonctions d'appartenance du contrôleur flou pour obtenir une meilleur performance.