TP – REGULATEUR PI FLOUE

1. Présentation

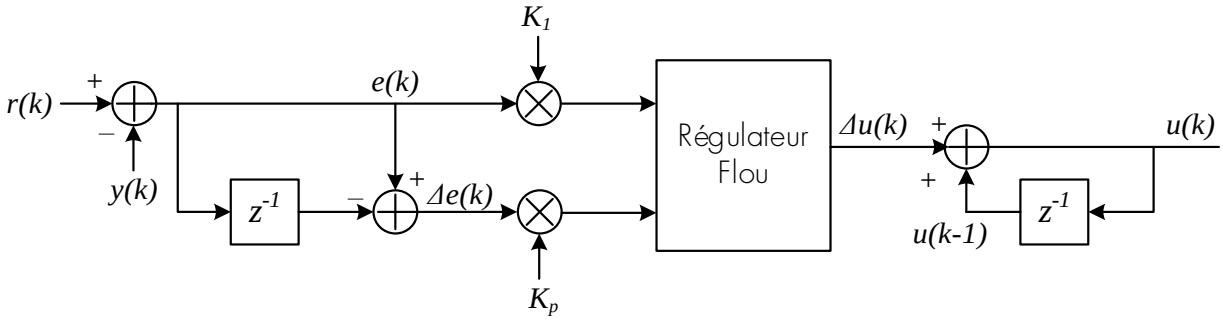
On se propose dans cette TP d'implémenter un régulateur PI Floue. La fonction de transfert d'un régulateur PI est donnée par l'équation : $C(p) = k_p + \frac{k_i}{s}$; En appliquant la transformation bilinéaire ($p = \frac{2}{T} \cdot \frac{1-z^{-1}}{1+z^{-1}}$)

$$C(z) = \frac{U(z)}{E(z)} = k_p + \frac{k_i T}{2} \cdot \frac{1+z^{-1}}{1-z^{-1}} = k_p - \frac{k_i T}{2} + \frac{k_i T}{1-z^{-1}} = K_p + \frac{K_i}{1-z^{-1}} \quad \text{avec} \quad K_p = k_p - \frac{k_i T}{2} \quad et \quad K_i = k_i T$$

$U(z) = z^{-1}U(z) + K_p(E(z) - z^{-1}E(z)) + K_i E(z)$ on en déduit l'équation récurrente :

$u(k) = u(k-1) + K_p(e(k) - e(k-1)) + K_i e(k) \quad \Rightarrow \Delta u(k) = u(k) - u(k-1) = K_p \Delta e(k) + K_i e(k)$ avec $e(k) = r(k) - y(k)$ et $\Delta e(k) = e(k) - e(k-1)$

Le régulateur flou a pour entrée $e(k)$ et $\Delta e(k)$ et comme sortie $\Delta u(k)$ dont on ajoute $u(k-1)$ pour avoir la sortie finale du régulateur $u(k)$.

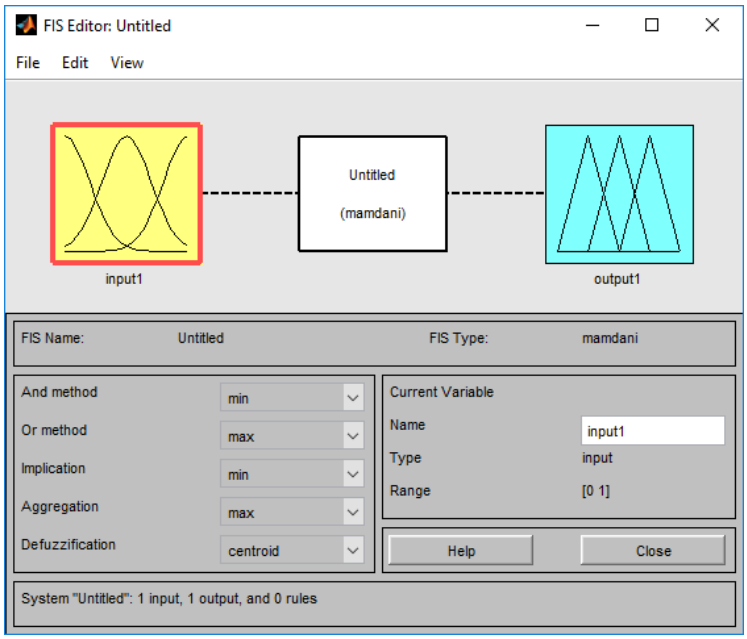


2. L'interface graphique de la boîte à outils Fuzzy Logic TOOLBOX

La commande **fuzzy** permet d'ouvrir l'interface graphique **FIS Editor** dans laquelle on peut définir complètement le système flou. Par défaut, l'interface propose une entrée et une sortie avec la méthode de **Mamdani**. Les opérateurs **ET** et **OU** sont réalisés respectivement par le **min** et le **max**, l'implication se fait par le **min**, l'agrégation des règles par le **max** et la défuzzification par la méthode du centre de gravité (centroid).

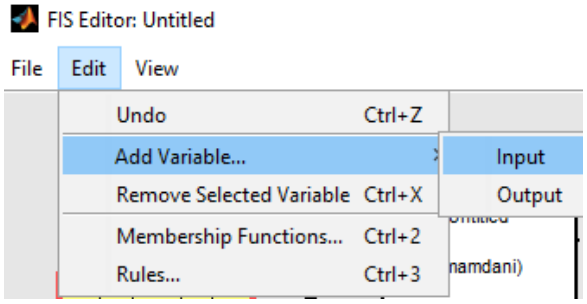
Tapez dans la fenêtre de commande : `>> fuzzy`

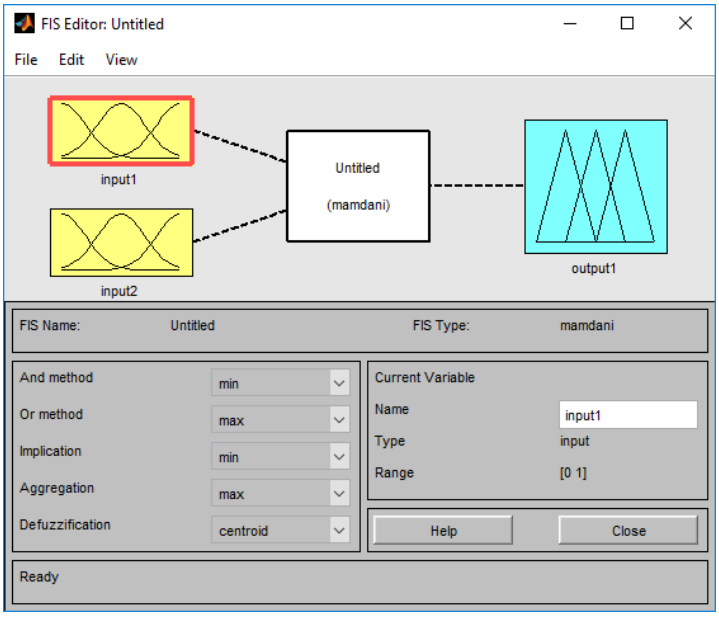
Une nouvelle fenêtre du FIS editor s'ouvre, il présente par défaut une entrée et une sortie.



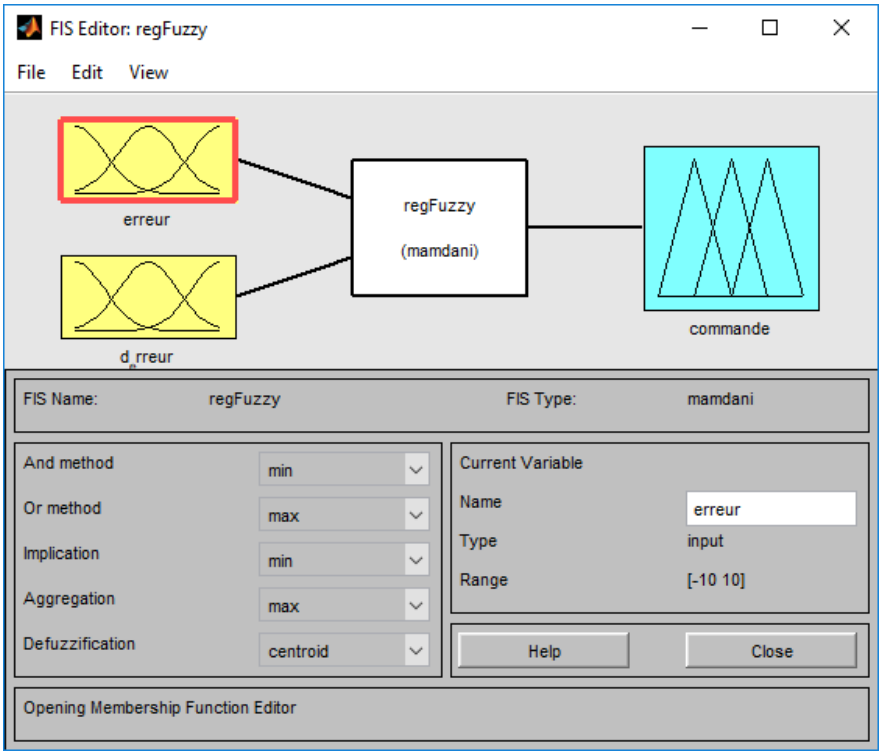
Notre régulateur comporte deux entrées $e(k)$ et $\Delta e(k)$ qu'on note ici erreur et d_erreur.

Ajouter la deuxième entrée avec la commande *Edit> Add Variable ...>input*

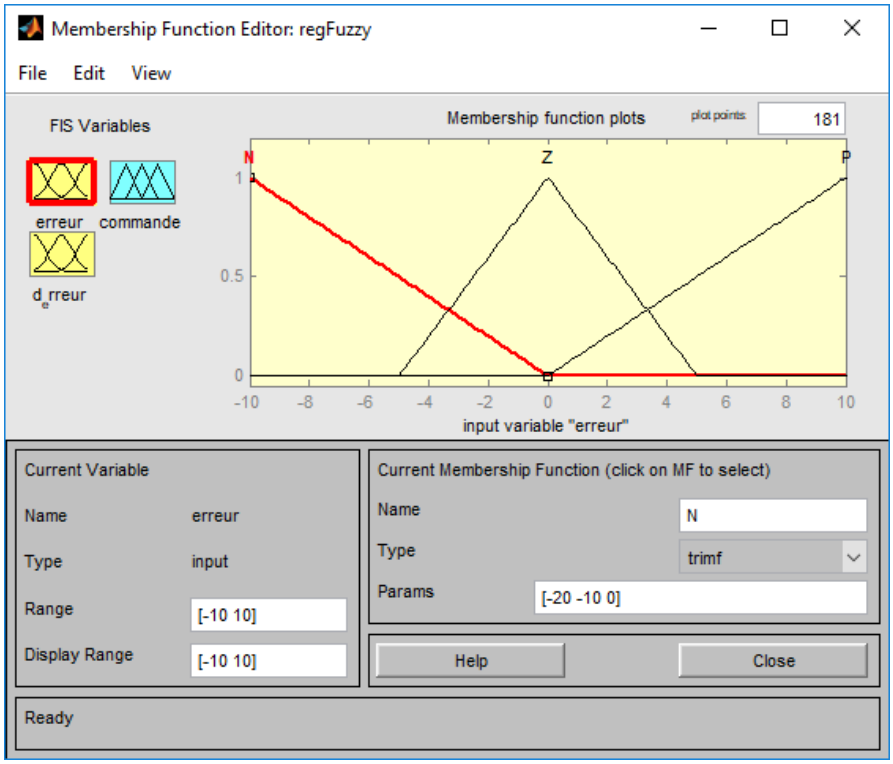




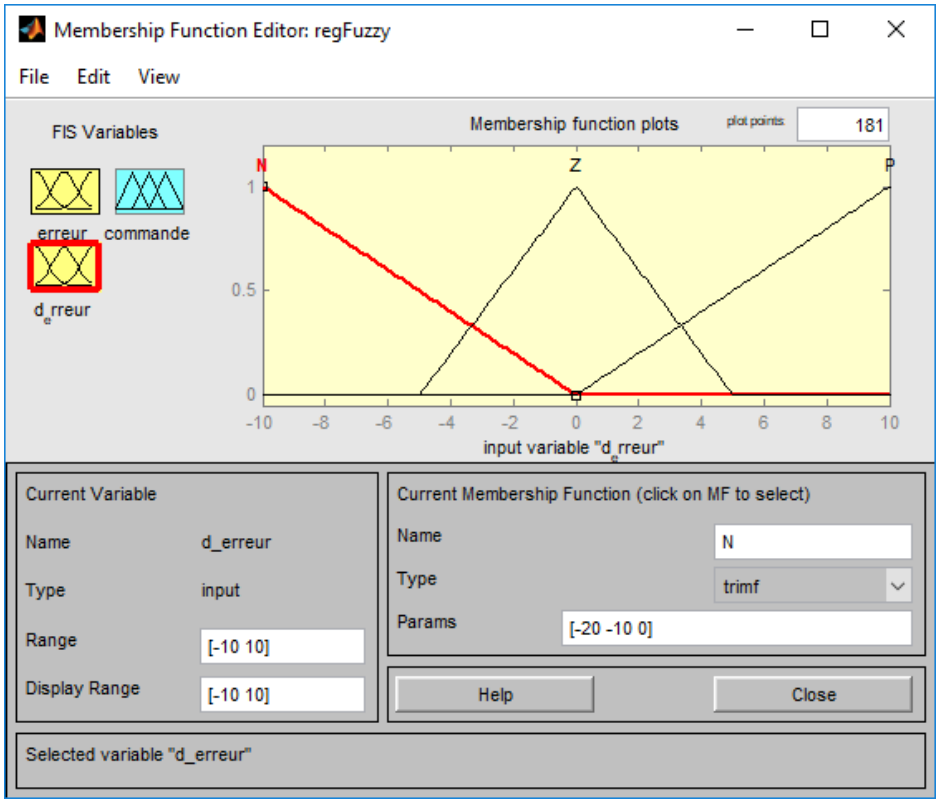
Par défaut, les noms des entrées input1, input2 et la sortie output1, renommer les **erreur**, **d_erreur** et **commande** pour la sortie.

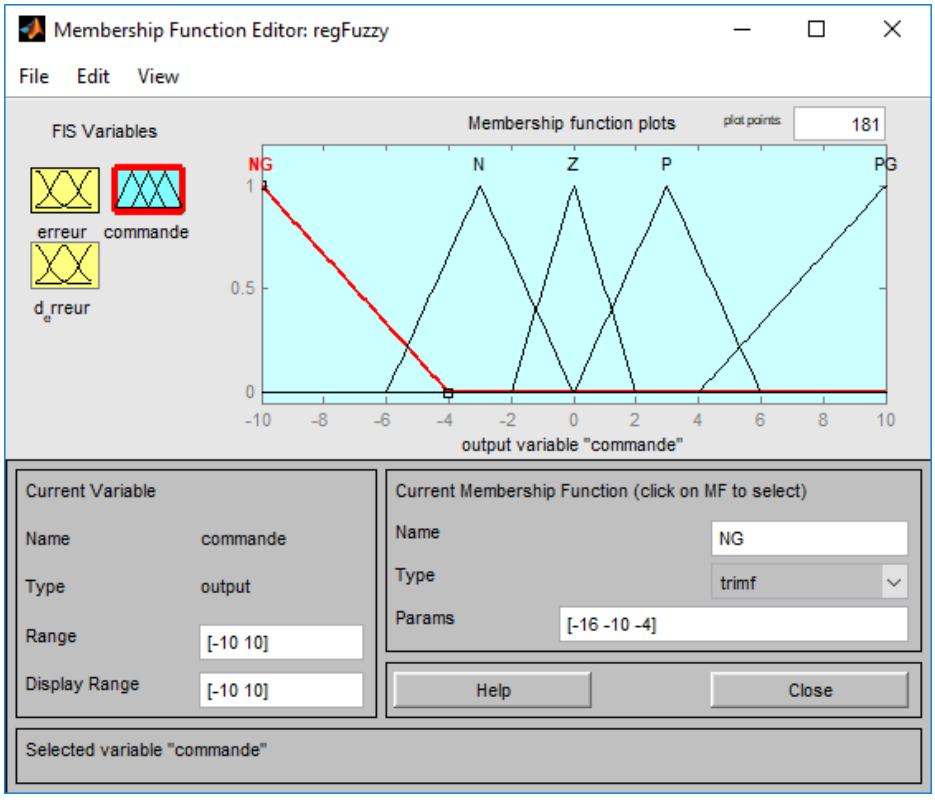


Cliquer deux fois sur le rectangle jaune de l'entrée **erreur** et configurer les fonctions membres comme suit :

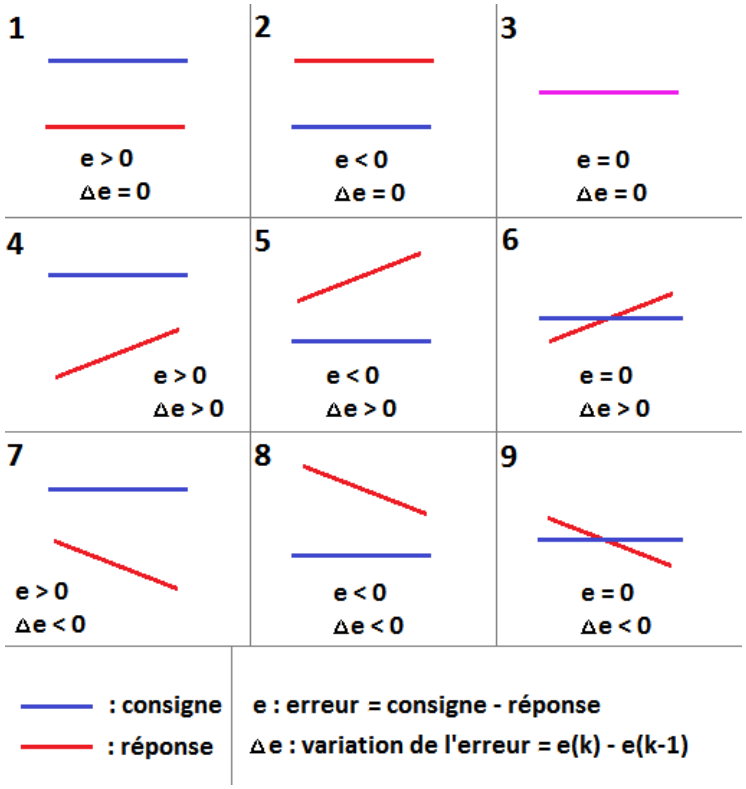


Faites la même chose pour **d_erreur** et la sortie **commande** .





On doit maintenant déterminer les divers règles à entrer au système. Comme il y a deux d'entrées à trois variables linguistiques chacune, il y aura donc au maximum 3*3 = 9 règles d'inférences. Ces neuf cas sont représentés schématiquement ci-dessous :



Cas un :

Dans ce premier cas, l'erreur est positive, ce qui veut dire que la réponse est inférieure à la consigne et la variation de l'erreur est nulle, ce qui veut dire que la réponse n'évolue pas. Dans ce cas, pour que la réponse atteigne la consigne, il faut augmenter la commande. La variation de la commande sera donc positive.

Cas deux :

A l'inverse, ici la réponse est supérieure à la consigne et n'évolue pas au cours du temps. Il faut donc diminuer la commande pour que la réponse atteigne la consigne. La variation de la commande est donc négative.

Cas trois :

Dans ce cas, la réponse n'évolue plus et vaut exactement la consigne. Dans ce cas-là, le système est asservit et il ne faut donc plus toucher à la commande. La variation de la commande est nulle.

Cas quatre :

Ici, la réponse est inférieure à la consigne, et continu à diminuer. Il faut donc diminuer fortement la variation de la commande.

Cas cinq :

Dans ce cas-là par contre, la réponse est supérieure à la consigne (erreur négative), mais elle en phase de diminution (variation de l'erreur positive) elle se rapproche de la consigne. Ici, Il ne faut donc pas toucher à la commande. La variation de la commande sera nulle.

Cas six :

L'erreur est nulle dans ce cas. Par contre, mais la réponse à tendance de diminuer. Il faut donc augmenter la commande. La variation de la commande est positive.

Cas sept :

Ici la réponse est bien en dessous de la consigne (erreur positive) et celle-ci augmente (variation de l'erreur négative). Il ne faut donc pas toucher à la commande. La variation de la commande sera nulle.

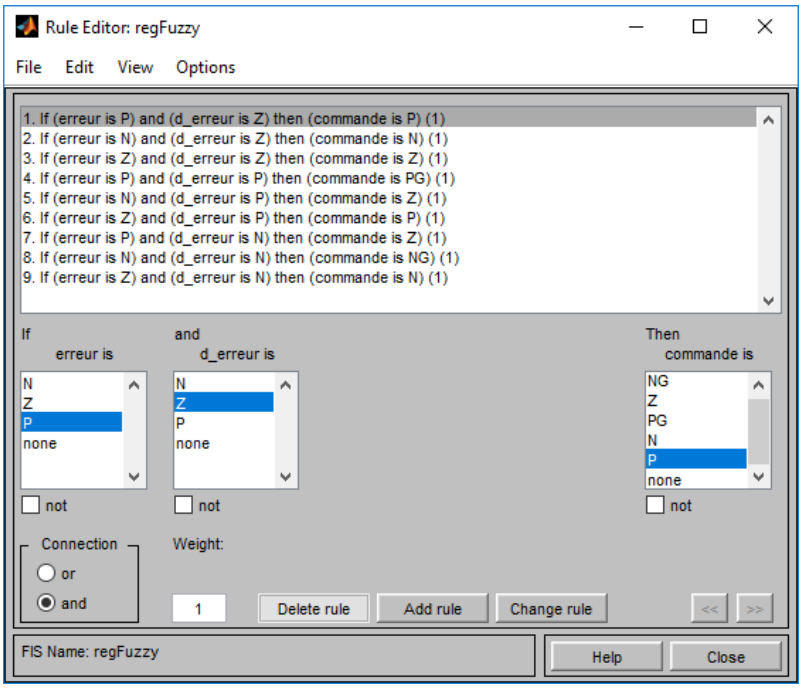
Cas huit :

Dans ce cas, la réponse est au-dessus de la consigne, et elle continue à augmenter. Il faut donc diminuer fortement la commande. La variation de la commande est négative.

Cas neuf :

Enfin, dernier cas. La réponse est égal à la consigne, mais elle tend à augmenter. Pour contrer cela, il faut diminuer la commande. La variation de la commande sera négative.

Cliquer deux fois sur le rectangle blanc pour ouvrir le fenêtre des règles (rules) et programmer les règles comme suit :

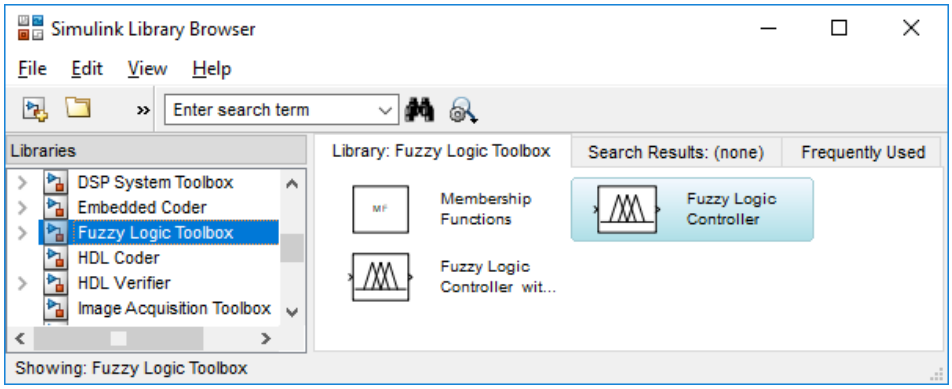


On va maintenant sauvegarder le système flou sous le nom *regFuzzy.fis* à travers le commande File>Export>To File...

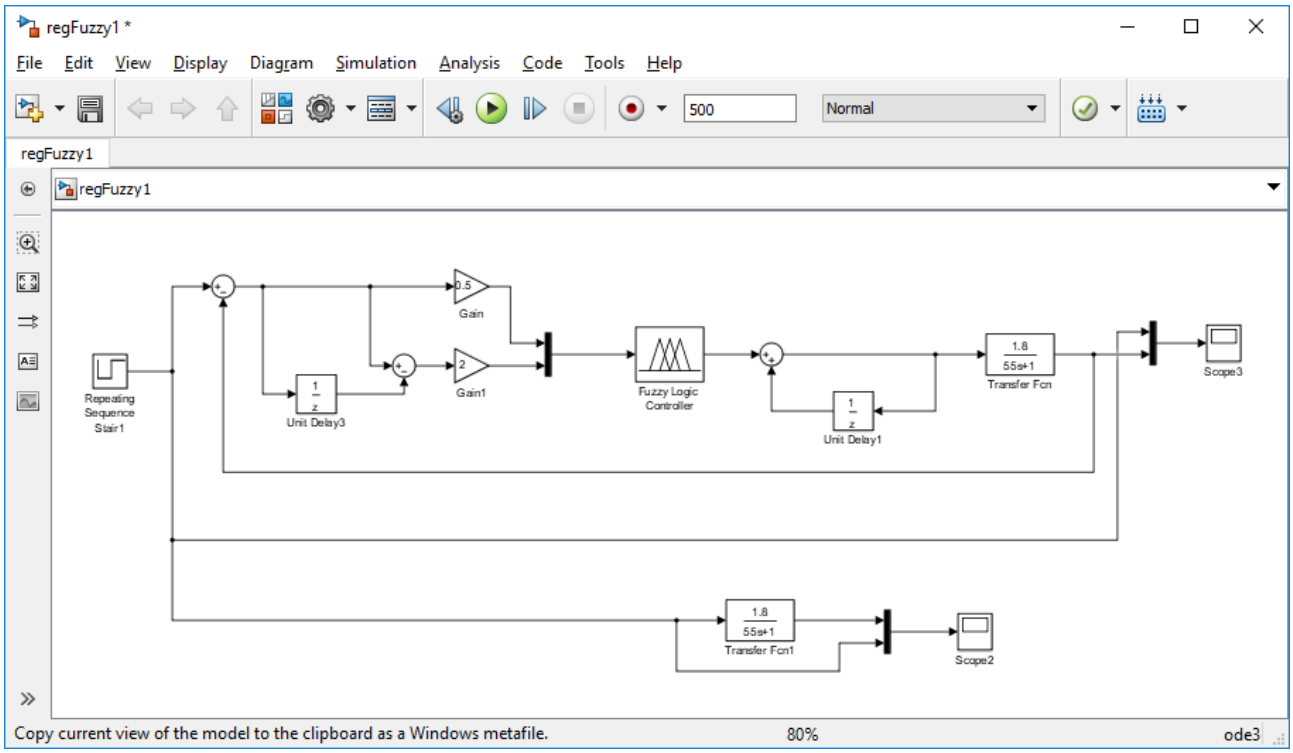
De même exporter le système au Workspace avec la commande File>Export>To Workspace....

3. Utilisation du régulateur Flou dans Matlab SIMULINK

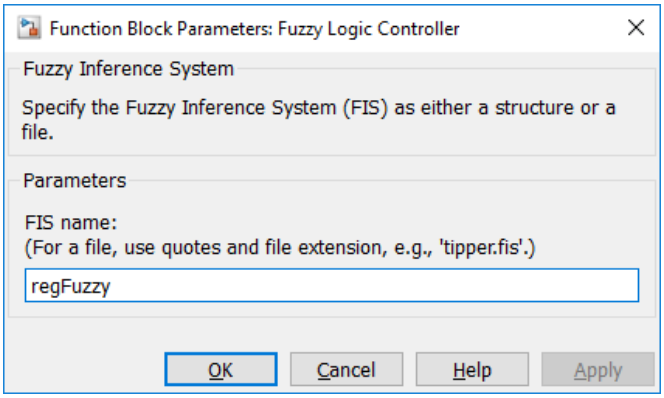
La boite à outils *Fuzzy Logic Toolbox* est un outil additionnel à **Simulink**.



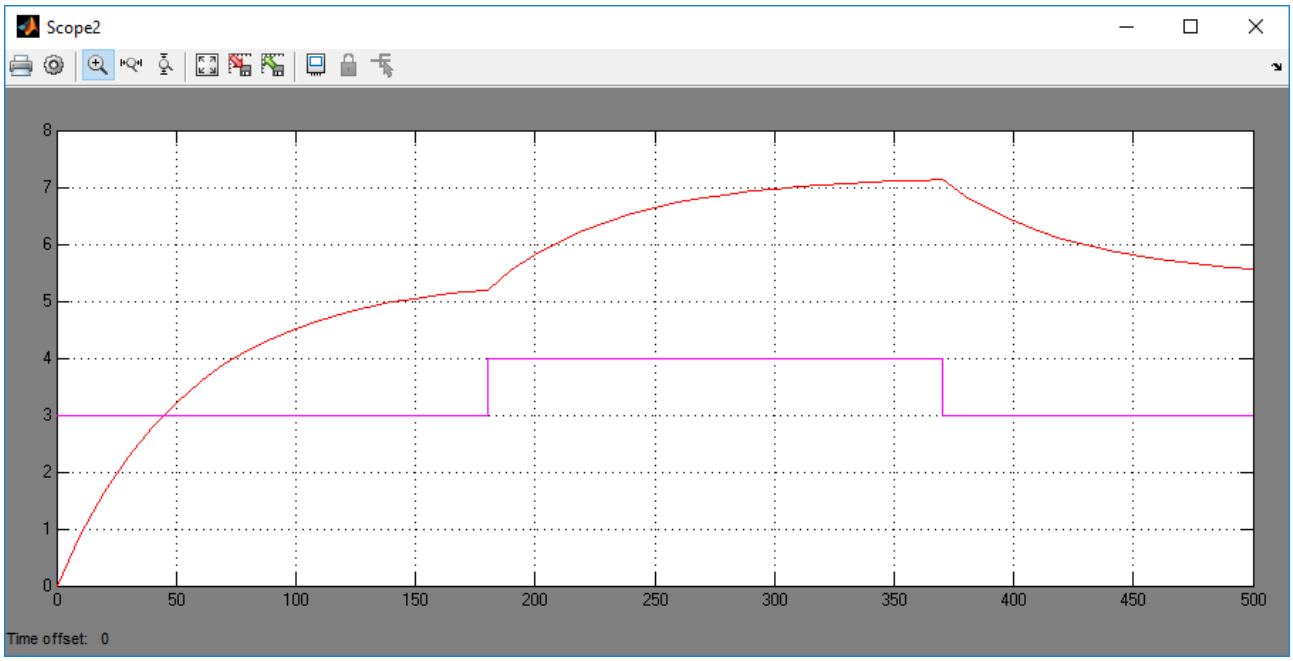
Créer un nouveau model simulink, puis représenter le schéma du régulateur flou suivant :



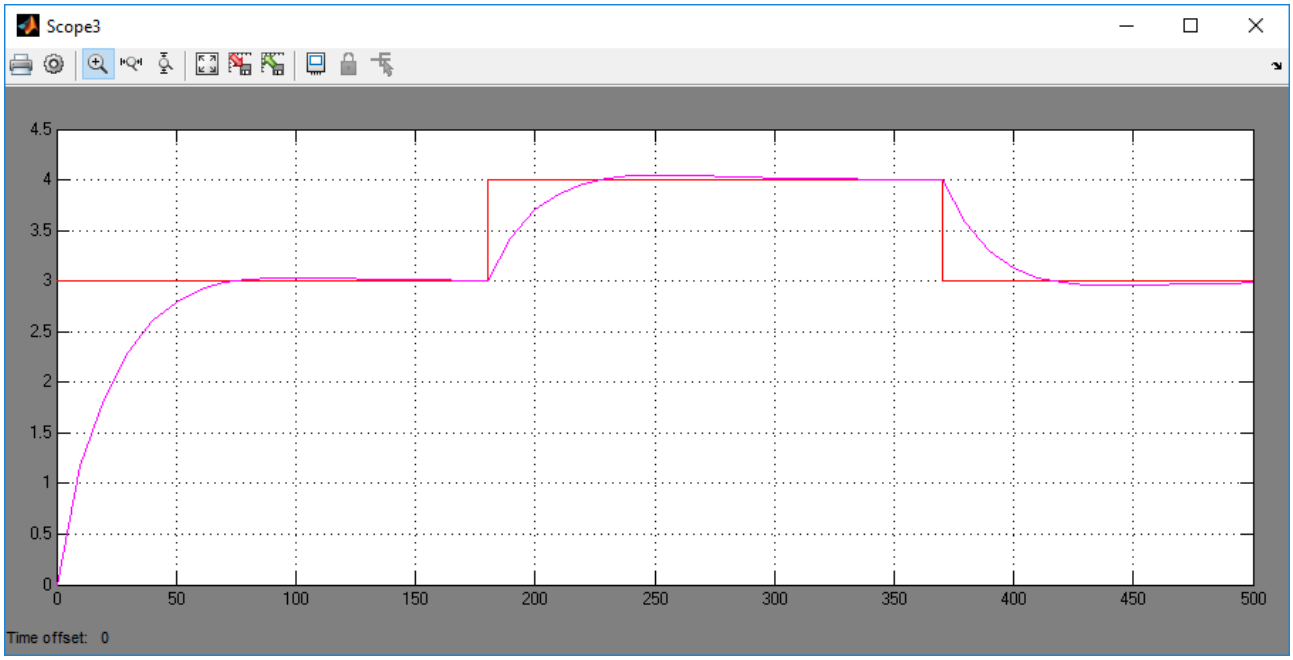
Double cliquer sur **Fuzzy Logic Controller** et insérer le nom du système fou « **regFuzzy** »



Il reste maintenant à faire la simulation du régulateur PI flou



Réponse du processus en boucle ouverte



Réponse du processus en boucle fermé