

Identification de systèmes non entiers non linéaires par réseaux de neurones

François Benoît-Marand, Laurent Signac,
Thierry Poinot, Jean-Claude Trigeassou

Présentation du sujet
Réseaux de neurones
Simulation des systèmes non entiers
Méthode d'identification par erreur de sortie
Résultats
Conclusion
Méthode d'identification combinée

PRESENTATION DU SUJET



Processus physiques étudiés

Les processus physiques étudiés sont définis par:

$$\frac{d^n \underline{y}(t)}{dt^n} = \underline{f}(\underline{u}(t), \underline{y}(t))$$

avec

\underline{f} : fonction non linéaire,

$\underline{u}(t)$: signal d'entrée,

$\underline{y}(t)$: signal de sortie,

n : ordre de dérivation (entier ou non).

Les domaines d'application sont variés :

- Pour l'ordre entier : de l'amplificateur de puissance aux procédés de cinétique chimique
- Pour l'ordre non entier : l'ensemble des systèmes diffusifs (ex : transfert de chaleur)

Les limites du modèle :

- Limites intrinsèques : processus d'hystérésis, processus régis par une représentation d'états non linéaire
- Avancement des travaux : en particulier dans le cas du non entier qui se limite à l'identification en simulation de systèmes mono variable :

$$\frac{d^n y(t)}{dt^n} = f(u(t), y(t))$$

Par expertise on définit une base de connaissance composée de modèles physiques ou semi physiques :

$$\mathcal{B}_d = \{h_1, \dots, h_d\}$$

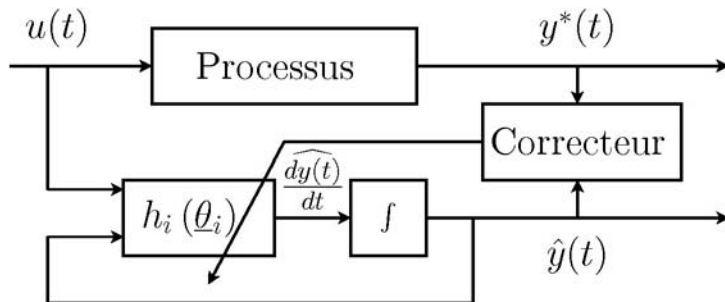
h_i : une fonction candidate
(non linéaire paramétrée),

$\underline{\theta}_i$: les paramètres associés à h_i .

L'objectif est de déterminer le couple $(h_i, \underline{\theta}_i^*)$ qui approche le mieux f .

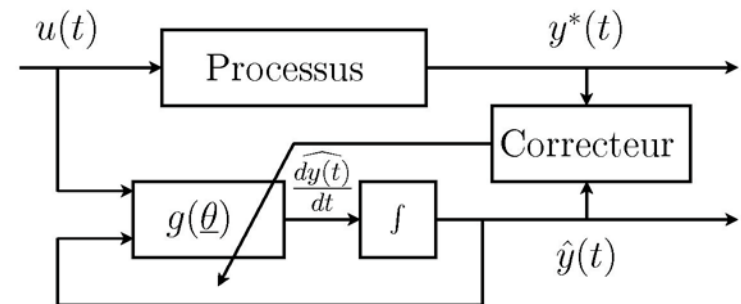
Méthode classique

Pour toutes fonctions de la base de connaissance

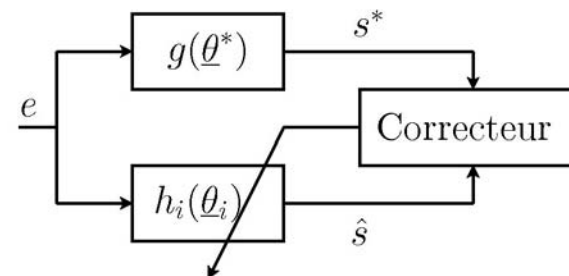


Extraction de la partie statique

Pour un approximateur universel g



Puis pour toutes fonctions de la base de connaissance



LES RESEAUX DE NEURONES STATIQUES



Les réseaux de neurones sont des approximateurs universels [Hornick 89] :

Soit f une fonction continue d'un compact C_1 de \mathbb{R}^n vers un compact C_2 de \mathbb{R}^m et $\epsilon > 0$, alors il existe F appartenant à l'ensemble des fonctions représentables par un réseau de neurones MLP telle que :

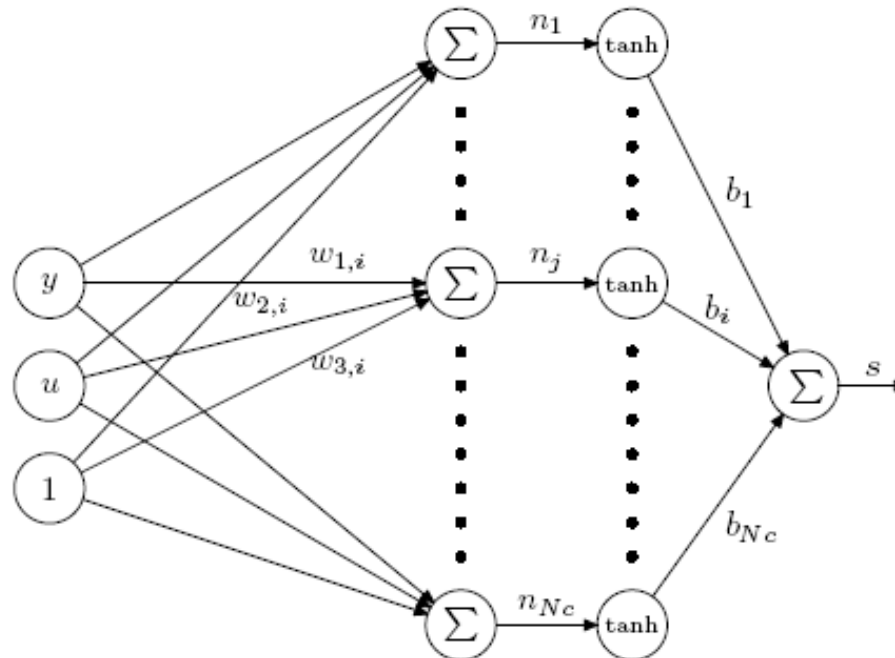
$$\forall x \in C_1, \|f(x) - F(x)\| < \epsilon$$

Les réseaux de neurones sont des approximateurs universels parcimonieux [Barron 93] :

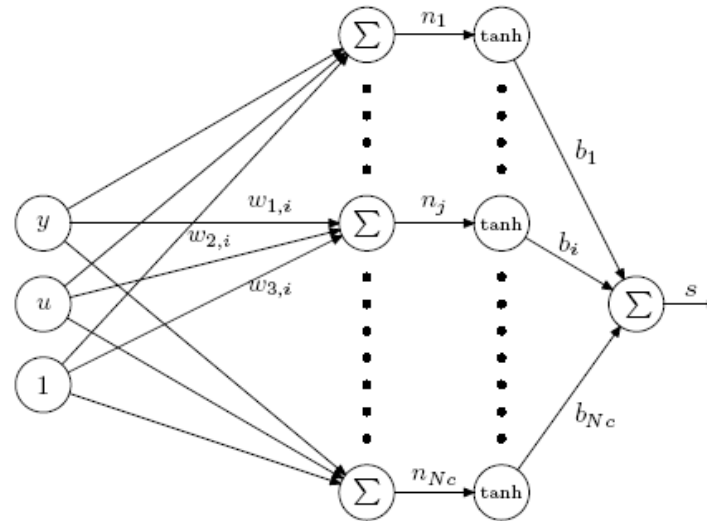
- Pour une précision donnée, en fonction du nombre de variables, le nombre de paramètres croit :
- exponentiellement pour un approximateur linéaire par rapport à ses paramètres (cas de la famille des polynômes)
 - linéairement pour un approximateur non linéaire par rapport à ses paramètres (cas des réseaux MLP)

Les réseaux de neurones MLP

Exemple d'un réseau de neurones MLP avec une couche cachée adapté au schéma d'identification précédemment présenté :



Calcul des fonctions de sensibilité



$$\frac{\partial s(k)}{\partial b_i} = \tanh(n_i(k))$$

$$\frac{\partial s(k)}{\partial w_{i,j}} = r_i(k) \cdot \tanh'(n_j(k)) \cdot b_j$$

$$\frac{\partial s(k)}{\partial w_{1,j}} = y(k) \cdot \tanh'(n_j(k)) \cdot b_j$$

$$\frac{\partial s(k)}{\partial w_{2,j}} = u(k) \cdot \tanh'(n_j(k)) \cdot b_j$$

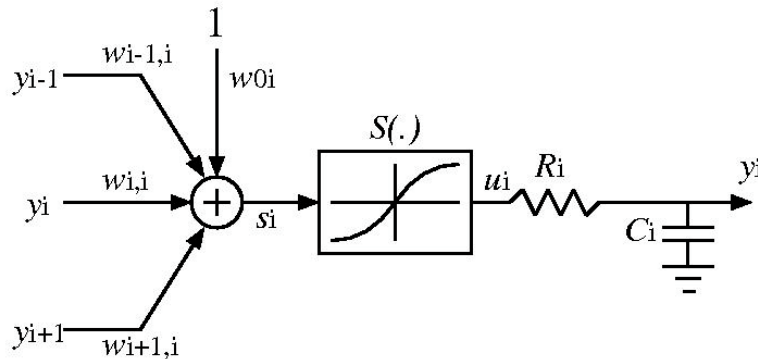
$$\frac{\partial s(k)}{\partial w_{3,j}} = \tanh'(n_j(k)) \cdot b_j$$

LES MODELES NEURONAUX DYNAMIQUES



Les modèles neuronaux à dynamique interne

- Représentation d'un neurone continu (ex de Hopfield) :

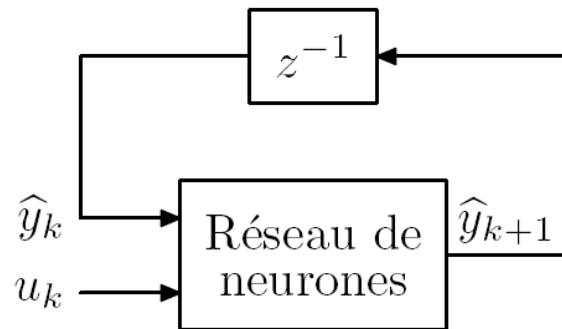


$$s_i = \sum_{j=0}^{N} w_{j,i} y_j, \quad u_i = S(s_i)$$

$$\frac{dy_i(t)}{dt} = \frac{1}{\tau_i} (u_i - y_i), \quad \tau_i = R_i C_i$$

- Ce type de réseau est dit totalement connecté.
- Des processus complexes peuvent être identifiés/contrôlés
- Limite : Les parties statiques et dynamiques du modèles sont non séparables

Représentation :



- **Avantage** : les parties dynamique et statique sont séparées
- **Limite** : Dans le cas d'un système linéaire où f est telle que

$$f(u(t), y(t)) = bu(t) + ay(t)$$

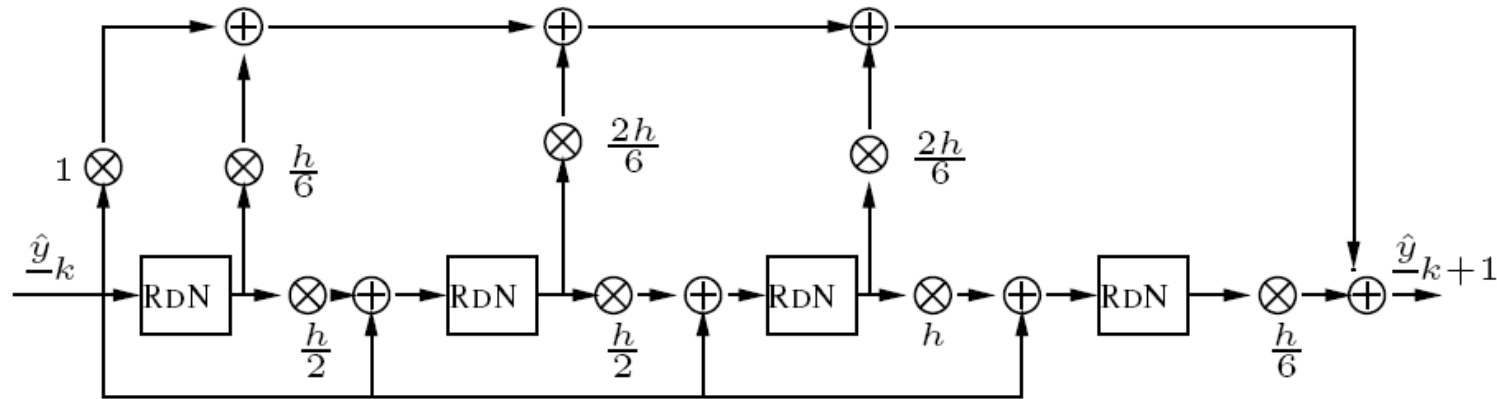
alors la fonction réalisée par le réseau (si $n=1$) est

$$g(u(t), y(t)) = e^{aT}y(t) + \frac{b}{a}(1 - e^{-aT})u(t)$$

$$g(u(t), y(t)) \neq f(u(t), y(t))$$

Les modèles neuronaux à dynamique externe et à temps discret (2)

Une version améliorée : discrétisation équivalente à la méthode d'intégration de Runge-Kutta 4 :

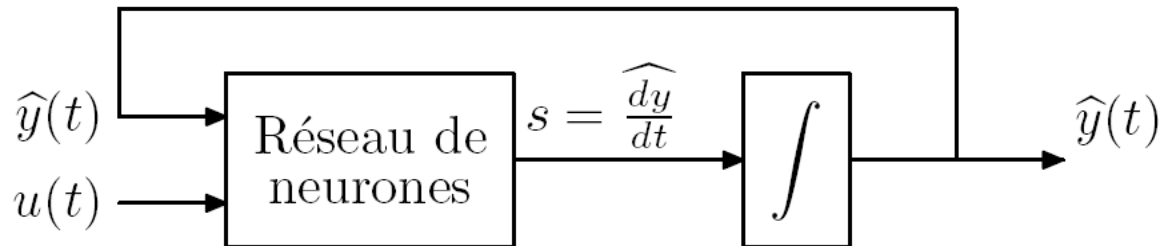


Avantage : dans ce cas le réseau est une approximation précise de la fonction f .

Limite : la structure du modèle et la méthode d'identification ont à être adaptées en fonction de la précision voulue et de la tâche à réaliser.

Extension en temps continu des modèles à dynamique externe

Nous proposons donc d'utiliser le schéma suivant :



Dans ce cas nous avons :

$$\frac{d\widehat{y}(t)}{dt} = g(u(t), \widehat{y}(t))$$

Ainsi, par définition, le réseau de neurone approche f avec la précision définie par l'opérateur d'intégration.

Une analyse de l'influence de la méthode d'intégration utilisée a été menée dans le cas des systèmes entiers.

SIMULATION DES SYSTEMES NON ENTIERS



Définition de l'opérateur d'intégration non entier

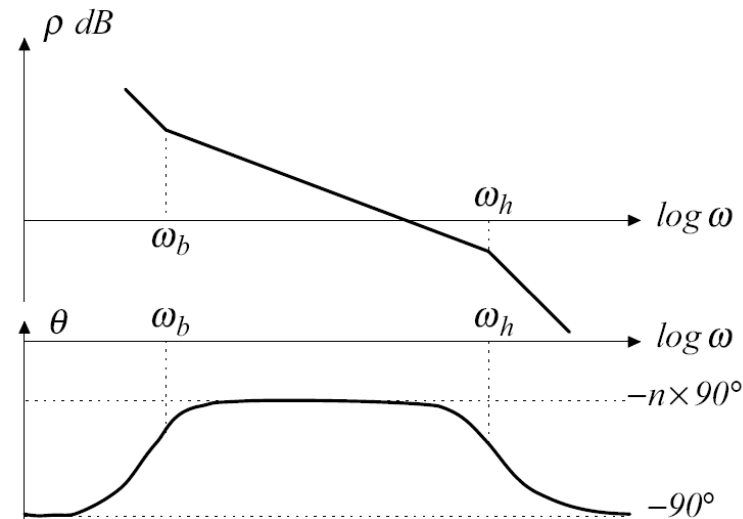
- L'opérateur idéal est $I_n(s) = \frac{1}{s^n}$
 - On utilise l'approximation suivante : $\hat{I}_n(s) = \frac{G}{s} \prod_{i=1}^N \frac{1 + \frac{s}{\omega'_i}}{1 + \frac{s}{\omega_i}}$
- complètement définie par :

$$\omega'_1 = \omega_b, \quad \omega_N = \omega_h, \quad \omega_u = \sqrt{\omega_b \omega_h}$$

$$\omega_i = \alpha \omega'_i, \quad \omega'_{i+1} = \eta \omega_i, \quad n = 1 - \frac{\ln \alpha}{\ln \alpha \eta}$$

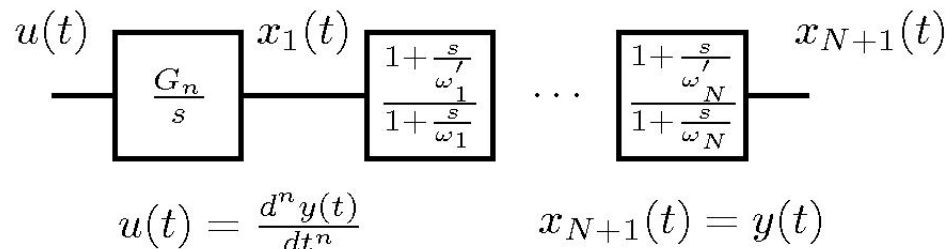
et avec G tel que :

$$\left| \left(\frac{1}{s} \right)^n \right|_{\omega_u} = \left| \hat{I}_n(s) \right|_{\omega_u}$$



Représentation d'état de l'opérateur d'intégration non entier

- On utilise le schéma bloc suivant



- Au quel on associe la représentation d'état suivante :

$$\dot{\underline{x}} = A^* \underline{x} + \underline{B}^* v$$

$$A^* = M^{-1} A, \quad \underline{B}^* = M^{-1} \underline{B}$$

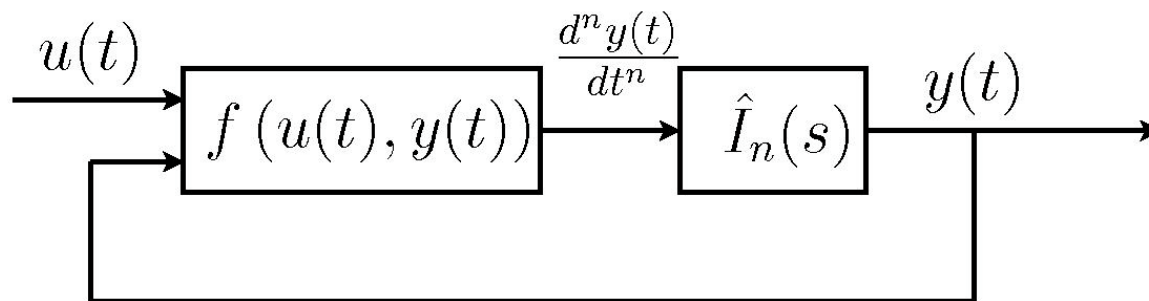
$$M = \begin{bmatrix} 1 & 0 & \dots & 0 \\ -\alpha & \ddots & & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & -\alpha & 1 \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 0 & \dots & 0 \\ \omega_1 & -\omega_1 & & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & \omega_N & -\omega_N \end{bmatrix}, \quad \underline{B}^T = [G_n \ 0 \ \dots \ 0]$$

- L'avantage de cet opérateur est que l'on peut simuler de la même manière des système linéaires et non linéaires.
- Supposons :

$$\frac{d^n y(t)}{dt^n} = f(u(t), y(t))$$

avec f linéaire ou non.

- Alors on simule le système ainsi :

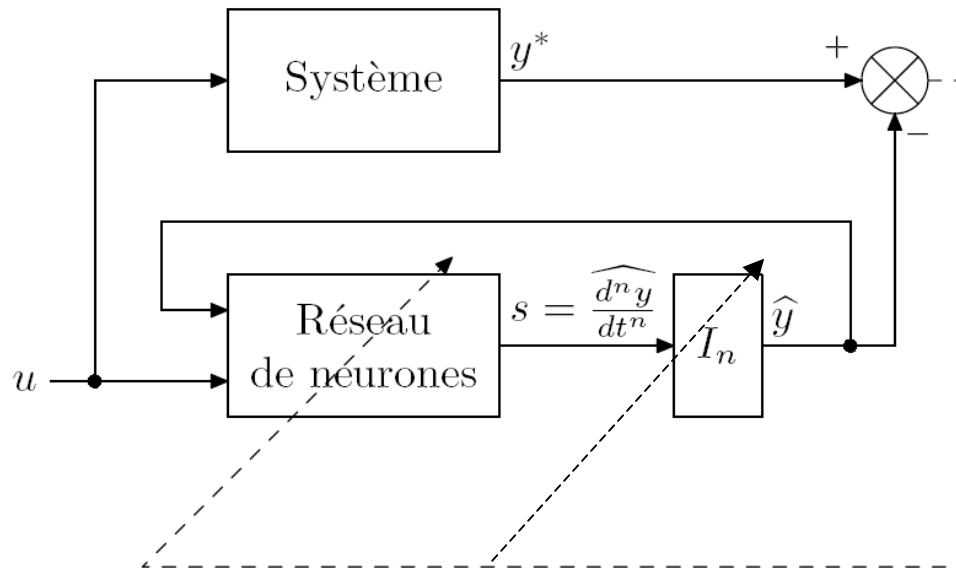


METHODE D'IDENTIFICATION PAR ERREUR DE SORTIE



Principe de l'identification par erreur de sortie

- On utilise le schéma d'identification suivant :



$$\frac{d^n \widehat{y}(t)}{dt^n} = s(u(t), \widehat{y}(t), \underline{\phi})$$

$$\underline{\theta}^T = [\underline{\phi}, \alpha]$$

- L'objectif est de minimiser le critère suivant :

$$J = \sum_{k=1}^K (y_k^* - \hat{y}_k)^2 = \sum_{k=1}^K e_k^2$$

- L'algorithme de LM est un processus itératif défini par

$$\underline{\theta}_{i+1} = \underline{\theta}_i + \Delta \underline{\theta} \text{ avec } \Delta \underline{\theta} = (\widetilde{H}_{\underline{\theta}} + \lambda I)^{-1} J'_{\underline{\theta}}$$

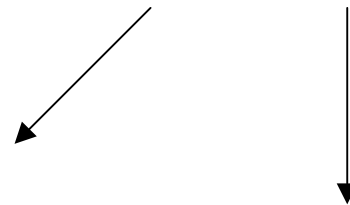
Avec $J'_{\underline{\theta}} = -2 \sum_{k=1}^K \left(e_k \cdot \frac{\partial \hat{y}_k}{\partial \underline{\theta}} \right)$, $[\widetilde{H}]_{j,l} = 2 \sum_{k=1}^K \left(\frac{\partial \hat{y}_k}{\partial \theta_j} \cdot \frac{\partial \hat{y}_k}{\partial \theta_l} \right)$

Il faut donc déterminer $\sigma_p(k) = \frac{\partial \hat{y}_k}{\partial \theta_p}$ pour tout p et tout k .

Recherche des paramètres du réseaux pour Ices

- Pour rechercher la valeur de $\sigma_p(k) = \frac{\partial \hat{y}_k}{\partial \phi_p}$ on écrit

$$\begin{aligned} \frac{\partial}{\partial \phi_p} \left(\frac{d^n \hat{y}}{dt^n} \right) &= \frac{d^n \sigma_p(t)}{dt^n} = \frac{\partial}{\partial \phi_p} \left(s(\underline{\phi}, u(t), \hat{y}(\underline{\phi}, t)) \right) \\ &= \frac{\partial s(t)}{\partial \phi_p} + \frac{\partial s(t)}{\partial \hat{y}(t)} \cdot \frac{\partial \hat{y}(t)}{\partial \phi_p} \\ &= \frac{\partial s(t)}{\partial \phi_p} + \frac{\partial s(t)}{\partial \hat{y}(t)} \cdot \sigma_p(t) \end{aligned}$$



$$\frac{\partial s(k)}{\partial b_i} = \tanh(n_i(k))$$

$$\frac{\partial s(k)}{\partial w_{i,j}} = r_i(k) \cdot \tanh'(n_j(k)) \cdot b_j$$

$$\frac{\partial s(t)}{\partial \hat{y}(t)} = \sum_{i=1}^{i=Nc} w_{1,i} \cdot \tanh'(n_i(t)) \cdot b_i$$

Recherche du paramètre α de l'intégrateur pour Ices

- On veut évaluer $\sigma_\alpha(k) = \frac{\partial \hat{y}(k)}{\partial \alpha}$
- On dérive par rapport à α la représentation d'état de l'opérateur non entier.
- En étendant cette RE avec $x_{(N+1+i)}(t) = \frac{\partial x_i(t)}{\partial \alpha}$ on obtient

$$\dot{x}_1 = G \circ s$$

$$-\alpha \dot{x}_{i-1} + \dot{x}_i = \omega_{i-1} x_{i-1} - \omega_{i-1} x_i$$

$$-\alpha \dot{x}_N + \dot{x}_{N+1} = \omega_N x_N - \omega_N x_{N+1}$$

$$\dot{x}_{N+2} = \frac{\partial G}{\partial \alpha} s + G \frac{\partial s}{\partial \alpha}$$

$$-\dot{x}_{i-1} - \alpha \dot{x}_{N+i} + \dot{x}_{N+i+1} = \bar{\omega}_{i-1} x_{i-1} - \bar{\omega}_{i-1} x_i + \omega_{i-1} x_{N+i} - \omega_{i-1} x_{N+i+1}$$

$$-\dot{x}_N - \alpha \dot{x}_{2N+1} + \dot{x}_{2(N+1)} = \bar{\omega}_N x_N - \bar{\omega}_N x_{N+1} + \omega_N x_{2N+1} - \omega_N x_{2(N+1)}$$

avec

$$\bar{\omega}_i = \frac{\partial \omega_i}{\partial \alpha}, \hat{y} = x_{N+1}, \sigma_\alpha = x_{2(N+1)}$$

Recherche du paramètre α de l'intégrateur pour Ices (2)

- Ceci peut s'exprimer sous la représentation d'états suivante

$$\dot{\underline{x}} = C^* \underline{x} + D^* \underline{v}$$

$$C^* = R^{-1}C \quad D^* = R^{-1}D$$

avec (M et A étant définis diapo 20)

$$D^T = \begin{bmatrix} \underline{D}_1 & 0 \\ 0 & \underline{D}_1 \end{bmatrix} \quad R = \begin{bmatrix} M & 0 \\ R_1 & M \end{bmatrix} \quad C = \begin{bmatrix} A & 0 \\ C_1 & A \end{bmatrix}$$

$$\underline{D}_1^T = [1 \quad 0 \quad \cdots], \quad R_1 = \begin{bmatrix} 0 & & & & \\ -1 & \ddots & & & \\ & \ddots & \ddots & & \\ & & \ddots & -1 & 0 \end{bmatrix}, \quad C_1 = \begin{bmatrix} 0 & & & & \\ \frac{\partial \omega_1}{\partial \alpha} & -\frac{\partial \omega_1}{\partial \alpha} & & & \\ & \ddots & \ddots & & \\ & & \ddots & \frac{\partial \omega_N}{\partial \alpha} & -\frac{\partial \omega_N}{\partial \alpha} \end{bmatrix}$$

et

$$\underline{v}^T = \left[f(u(t), \hat{y}(t), \underline{\theta}) \quad \frac{\partial G}{\partial \alpha} f + G \frac{\partial f}{\partial \hat{y}(t)} \sigma_\alpha(t) \right]$$

$$\hat{y}(t) = x_{(N+1)}(t) \quad \sigma_\alpha(t) = x_{(2N+2)}(t)$$

Quelques calculs (1)

- Pour calculer la valeur de $\frac{\partial \omega_i}{\partial \alpha}$ on utilise un procédé de calcul itératif :

$$\frac{\partial \omega_1}{\partial \alpha} = \omega_b$$

$$\frac{\partial \omega_i}{\partial \alpha} = \alpha \eta \frac{\partial \omega_{i-1}}{\partial \alpha} + \omega_{i-1} \left[\eta + \alpha \frac{\partial \eta}{\partial \alpha} \right]$$

avec

$$\frac{\partial \eta}{\partial \alpha} = \left[\frac{\omega_h}{\omega_b} \right]^{\frac{1}{N-1}} \left(\frac{N}{1-N} \right) \alpha^{\frac{2N-1}{1-N}}$$

Quelques calculs (2)

- Finalement il nous reste à calculer $\frac{\partial G}{\partial \alpha}$
- Dans le cas général on a :

$$\frac{\partial G}{\partial \alpha} = -\ln(\omega_u) \frac{\partial n}{\partial \alpha} G + \frac{1}{2} G \sum_{k=1}^{k=N} G_k^2 \frac{\partial}{\partial \alpha} \left(\frac{1}{G_k^2} \right)$$

$$\frac{\partial}{\partial \alpha} \left(\frac{1}{G_k^2} \right) = \{\omega_k^2 + \alpha^2 \omega_u^2\}^{-2} \left\{ 2\omega_k \frac{\partial \omega_k}{\partial \alpha} (\omega_k^2 + \alpha^2 \omega_u^2) - (\omega_k^2 + \omega_u^2) \left(2\omega_k \frac{\partial \omega_k}{\partial \alpha} + 2\alpha \omega_u^2 \right) \right\}$$

$$\frac{\partial n}{\partial \alpha} = \frac{(\eta + \alpha \frac{\partial \eta}{\partial \alpha}) \ln(\alpha) - \eta \ln(\alpha \eta)}{\alpha \eta [\ln(\alpha \eta)]^2}$$

- Et si $\omega_u=1$, alors on a :

$$\frac{\partial G}{\partial \alpha} = \frac{1}{2} G \sum_{k=1}^{k=N} G_k^2 \frac{\partial}{\partial \alpha} \left(\frac{1}{G_k^2} \right)$$

$$\frac{\partial}{\partial \alpha} \left(\frac{1}{G_k^2} \right) = \frac{2(\alpha^2 - 1) \left(\frac{\partial \omega_k}{\partial \alpha} \omega_k \right) - 2\alpha(\omega_k^2 + 1)}{(\omega_k^2 + \alpha^2)^2}$$

RESULTATS EN SIMULATION



- On a simulé le système suivant :

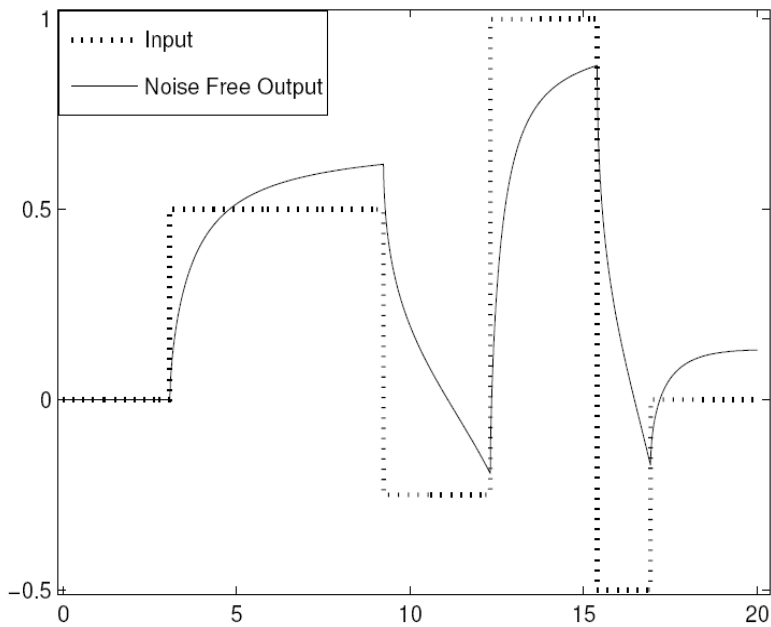
$$\frac{d^n y(t)}{dt^n} = au(t) + by(t)^2$$

avec $a = 1$, $b = -1$, $n = 0.6$, $\omega_b = 10^{-4}$ rad/s,
 $\omega_h = 10^4$ rad/s et $N = 20$.

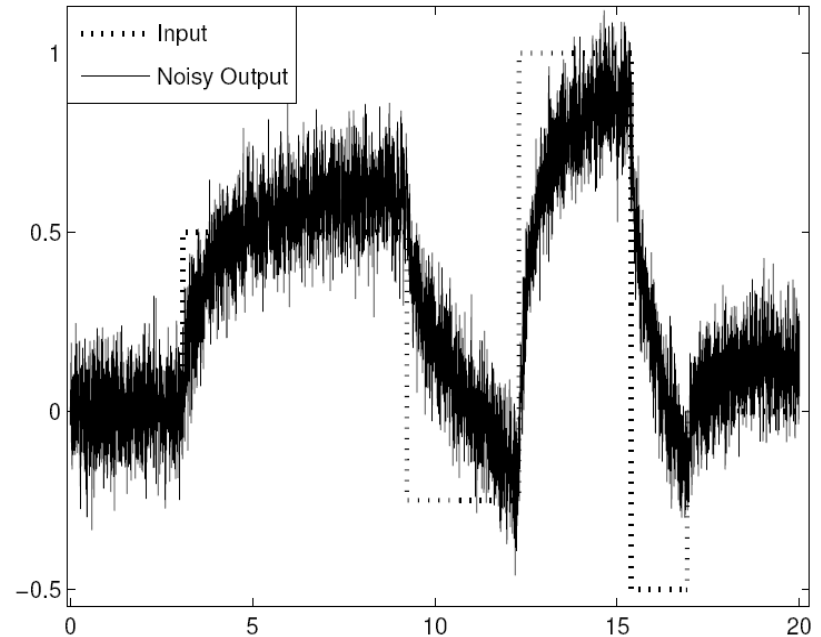
- Pour créer des données bruitées on a ajouté un bruit blanc tel que le rapport signal sur bruit soit égal à 10
- On a utilisé 3 neurones dans le réseau (initialisé par une heuristique simple)
- La valeur de l'ordre de l'intégrateur a été initialisée à 0.3

Jeux de données utilisés

■ Jeu de données clair

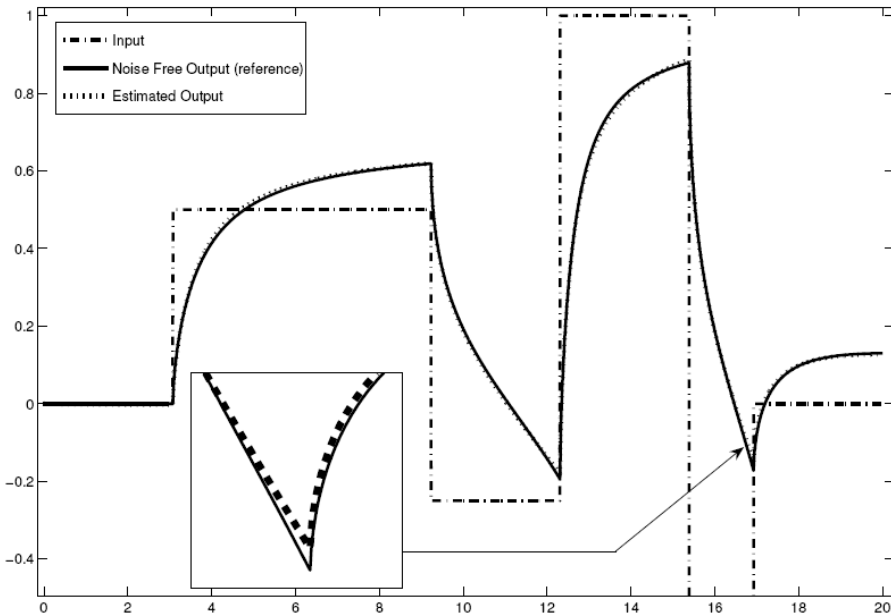


■ Jeu de données bruitées



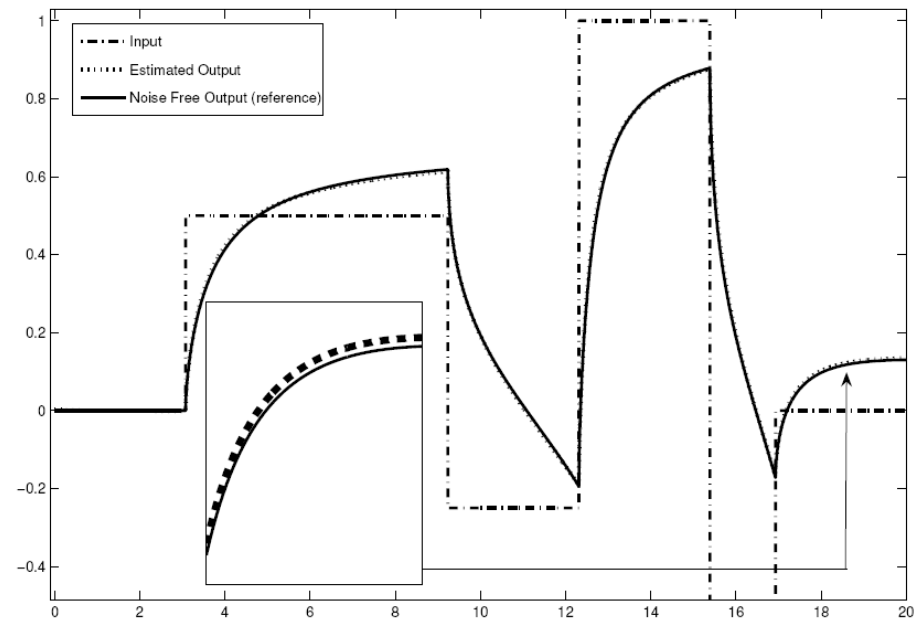
Résultats de l'identification

■ Sur le jeu de données clair



- Réduction de modèle :
 $a=1.0038$, $b=-1.0145$, $n=0.6074$

■ Sur le jeu de données bruitées



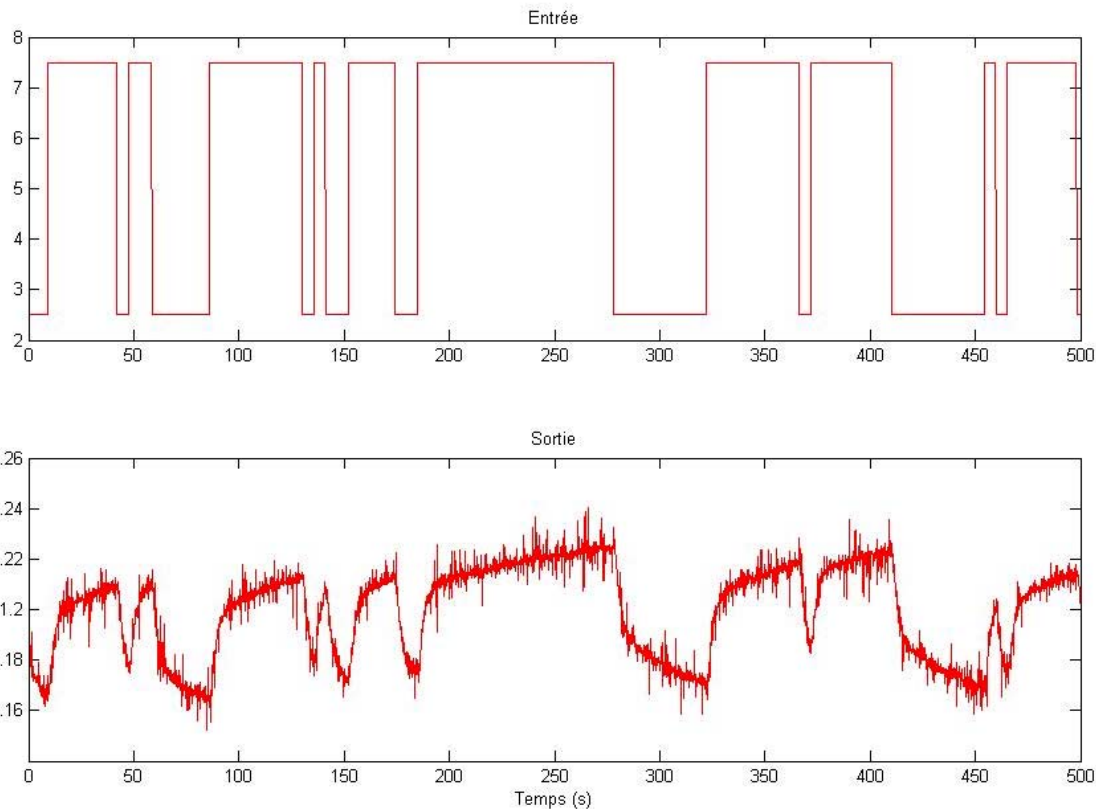
- Réduction de modèle :
 $a=0.9713$, $b=-0.9391$, $n=0.5859$

RESULTATS EXPERIMENTAUX



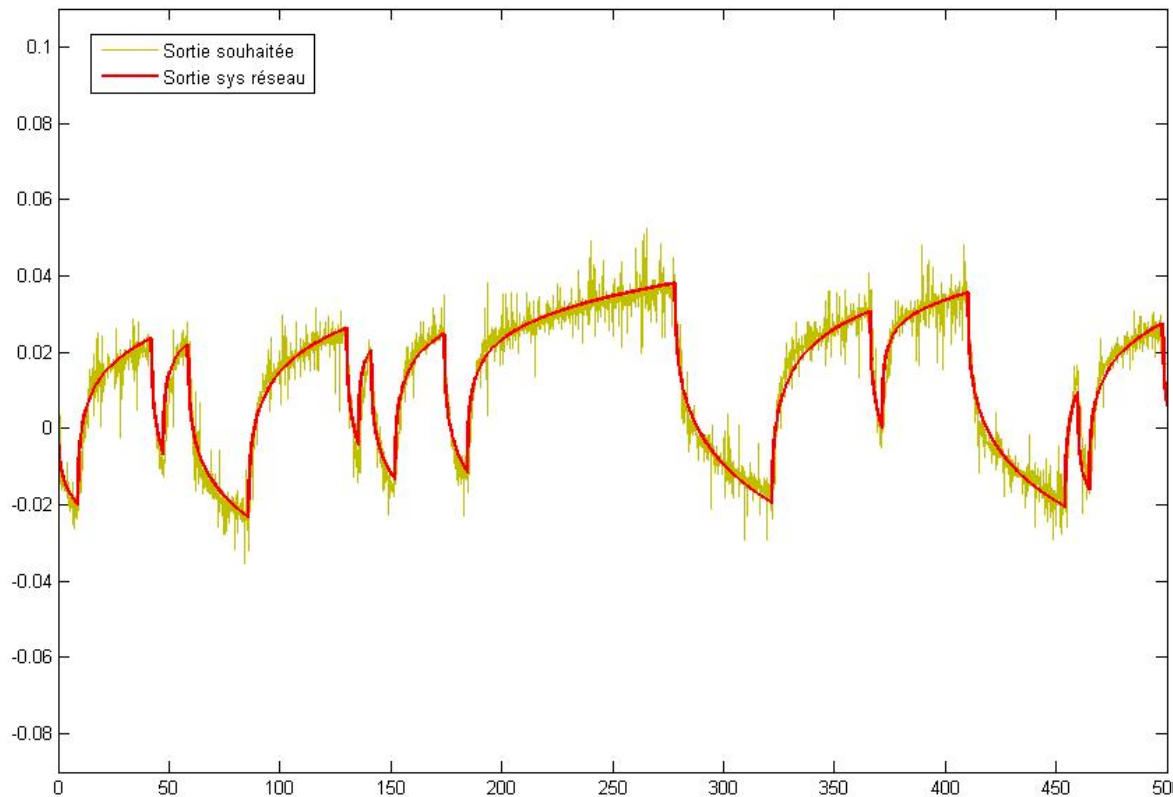
Processus étudié

- On a travaillé sur le processus de diffusion de chaleur présenté par Amel Benchellal (excité par une SBPA)



Résultats de l'identification

- L'ordre du système ainsi déterminé est 0.347232

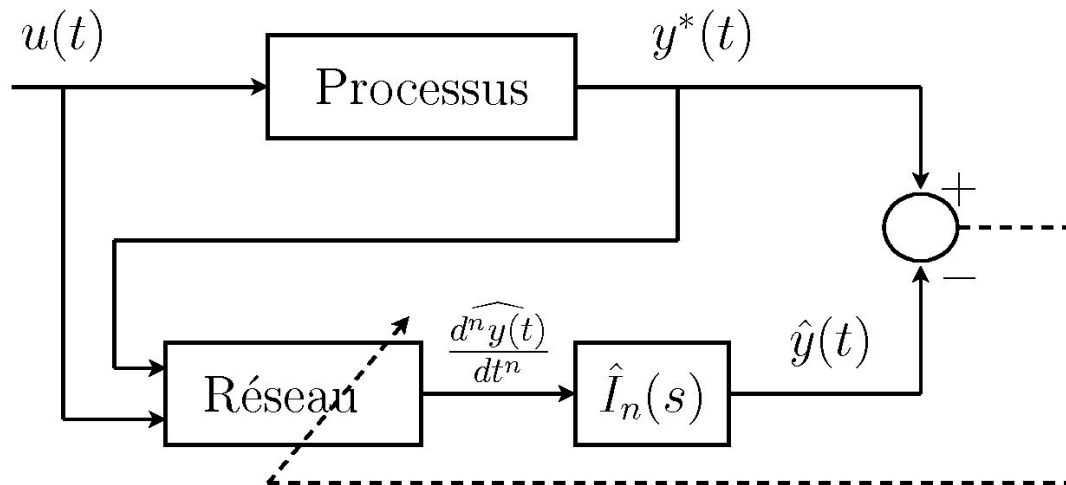


CONCLUSION

- Nous avons développé une méthode d'identification par réseaux de neurones des systèmes non entiers non linéaires, mono entrée mono sortie, qui fournit de bons résultats en simulation.
- Nos travaux à venir portent principalement sur :
 - La poursuite de l'identification du système réel de transfert de chaleur
 - L'extension de la méthode au cas de deux intégrateur afin de prendre en compte la géométrie du système
- Finalement, le principal inconvénient de cette méthode est le temps de calcul. Nous proposons dans une dernière partie de discuter de l'utilisation de la méthode d'identification par erreur d'équation avant d'appliquer la procédure présentée dans cette exposé. En effet, dans le cas des systèmes entiers nous avons montré qu'une telle approche permet un gain de temps extrêmement appréciable.

METHODE COMBINEE

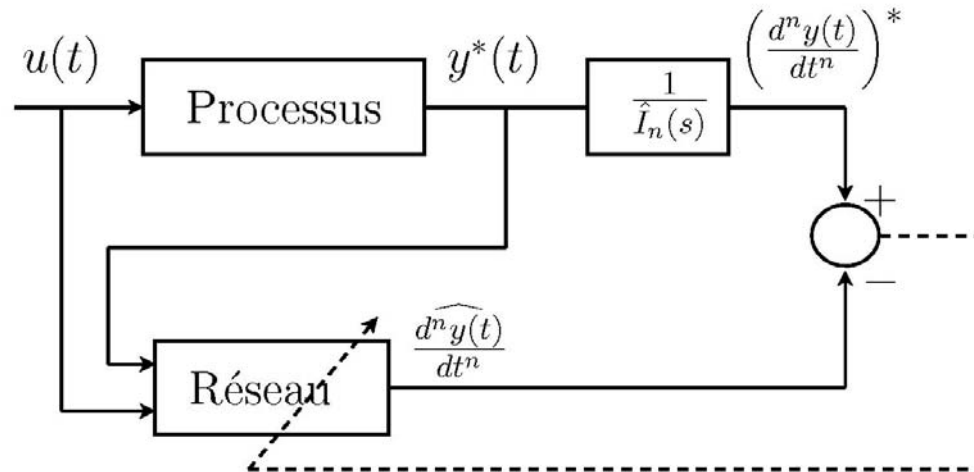
- On utilise le schéma d'identification suivant :



$$\frac{d^n \hat{y}(t)}{dt^n} = s(u(t), y^*(t), \underline{\phi})$$

- On ne sait pas encore identifier l'ordre par cette méthode

- Le principe d'identification est le suivant :
- On utilise un dérivateur non entier $\frac{1}{\hat{I}_n(s)}$ pour calculer les valeurs de $\left(\frac{d^n y(t)}{dt^n}\right)^*$
- On identifie ensuite les paramètres du réseau de manière statique
- Ceci peut être représenté par le schéma suivant :



Méthodologie de la méthode combinée

- Exciter le système avec de faibles variations autour d'un point de fonctionnement.
- Identifier à l'aide d'un modèle linéaire la valeur de l'ordre.
- Exciter le système classiquement.
- Identifier le système par erreur d'équation avec l'ordre précédemment déterminé.
- On obtient ainsi rapidement des valeurs initiales pour l'ordre du modèle et pour les paramètres du réseau.
- On peut alors lancer l'identification par erreur de sortie sur l'ordre et le réseau en partant de ces valeurs initiales.