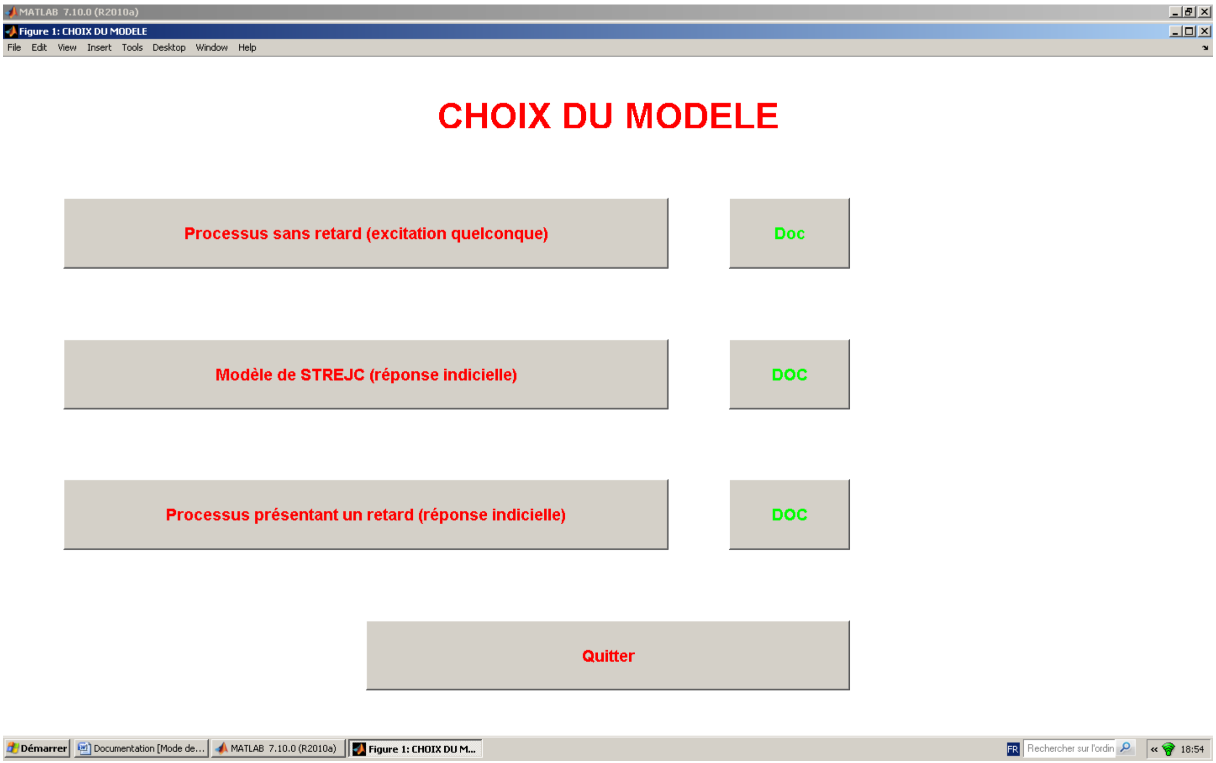
IDENTIFICATION

-1- Les possibilités du logiciel



-2- Identification d'un processus sans retard

L'excitation est quelconque, un échelon par exemple.
On cherche un modèle d'ordre n+k et de clase k :

$$T(p) = \frac{a_0 + a_1.p + ... + a_m.p^m}{p^k.\left(1 + b_1.p + ... + b_n.p^n\right)} \qquad (m \leq n+k)$$

Méthode d'identification : méthodes des moindres carrés
On suppose dans un premier temps que k = 0.

Le processus est défini par sa réponse y(t) à une excitation x(t), ou plus exactement par N valeurs y(i) prises aux instants $t_i = (i-1).T_E$ (T_E = période d'échantillonnage).(le premier échantillon, i = 1, correspond à t = 0).

$$\frac{Y(p)}{X(p)} = T(p) \quad \Leftrightarrow \quad y + b_1.\frac{dy}{dt} + b_2.\frac{d^2y}{dt^2} + ... + b_n.\frac{d^ny}{dt^n} = a_0.x + a_1.\frac{dx}{dt} + ... + a_m.\frac{d^mx}{dt^m} \qquad \textbf{(E)}$$

<u>Problème</u> : à cause du bruit, il est très difficile d'estimer $\left.\frac{dy}{dt}\right|_{t=t_i}$ à partir des y(i), et à fortiori $\left.\frac{d^2y}{dt^2}\right|_{t=t_i}$, **...**

Posons :

$$y_1(t_i) = \int_0^t y(\theta).d\theta \qquad x_1(t) = \int_0^t x(\theta).d\theta$$

$$y_2(t_i) = \int_0^t y_1(\theta).d\theta \qquad x_2(t) = \int_0^t x_1(\theta).d\theta$$

$$\vdots \qquad\qquad \vdots$$

$$y_n(t) = \int_0^t y_{n-1}(\theta).d\theta \qquad x_n(t) = \int_0^t x_{n-1}(\theta).d\theta$$

$y_1, y_2, \ldots$ et $x_1, x_2, \ldots$ peuvent être facilement estimées en utilisant la méthode des trapèzes[1].
Les différentes intégrations permettent de minimiser l'influence du bruit.

Après n intégrations, l'équation **(E)** peut s'écrire (en supposant le système initialement au repos, c'est à dire $y(0^-) = 0$, $\left.\frac{dy}{dt}\right|_{t=0^-} = 0$, etc ...):

$$\mathbf{(E)} \Rightarrow y_n + b_1.y_{n-1} + \ldots + b_n.y = a_0.x_n + a_1.x_{n-1} + \ldots + a_m.x_{n-m}$$

$$\Rightarrow y_n = a_0.x_n + a_1.x_{n-1} + \ldots + a_m.x_{n-m} - b_n.y - b_{n-1}.y_1 - \ldots - b_1.y_{n-1}$$

Introduisons les vecteurs :

$$\mathbf{Y_k} = \begin{pmatrix} y_k(1) \\ y_k(2) \\ . \\ . \\ . \\ y_k(N) \end{pmatrix}$$

de même pour $\mathbf{X_n}$
Soit $\mathbf{Z_n}$ la valeur calculée de $\mathbf{Y_n}$ à partir des coefficients $a_0, a_1 \ldots a_m, b_1 \ldots b_n$.
Sous forme matricielle on a:

$$\mathbf{Z_n} = \mathbf{F} \times \mathbf{C}$$

où :

---

[1] La fonction Matlab ***cumtrapz*** réalise cette opération.

Exemple :
```
>> t=0:1:10;
>> y=cos(2*pi*t/10);
>> cumtrapz(t,y)
ans =
  Columns 1 through 8
0  9.0451e-001  1.4635e+000  1.4635e+000  9.0451e-001  1.1102e-016 -9.0451e-001 -1.4635e+000
  Columns 9 through 11
-1.4635e+000 -9.0451e-001 -4.4409e-016
```

$$\mathbf{C}=\begin{pmatrix} a_0 \\ \vdots \\ a_m \\ b_n \\ \vdots \\ b_1 \end{pmatrix} \quad \text{et} \quad \mathbf{F}=\begin{pmatrix} x_n(1) & x_{n-1}(1) & \dots & \dots & -y_1(1) & \dots & -y_{n-1}(1) \\ x_n(2) & x_{n-1}(2) & \dots & \dots & -y_1(2) & \dots & -y_{n-1}(2) \\ . & . & . & & . & & \\ . & . & . & \dots & . & & \\ . & . & . & & . & & \\ x_n(N) & x_{n-1}(N) & \dots & \dots & -y_1(N) & \dots & -y_{n-1}(N) \end{pmatrix}$$

On va chercher a_0, a_1, a_2, ..., a_m, b_1,…, b_n de façon à minimiser la somme :

$$S=\sum_{i=1}^{N}\left[y_n(i)-z_n(i)\right]^2 \qquad \text{(méthode des moindres carrés)}$$

c'est à dire tels que :

$$\frac{\partial S}{\partial a_0}=0\ ;...;\ \frac{\partial S}{\partial a_m}=0\ ;\ ...\ ;\frac{\partial S}{\partial b_1};...;\frac{\partial S}{\partial b_n}=0$$

$$\frac{\partial S}{\partial a_k}=\sum_{i=1}^{N}2.\left[y_n(i)-z_n(i)\right].x_{n-k}(i)=0 \Leftrightarrow \boxed{\sum_{i=1}^{N}y_n(i).x_{n-k}(i)=\sum_{i=1}^{N}z_n(i).x_{n-k}(i)}$$

$$\frac{\partial S}{\partial b_k}=\sum_{i=1}^{N}-2.\left[y_n(i)-z_n(i)\right].y_{n-k}(i)=0 \Leftrightarrow \boxed{\sum_{i=1}^{N}y_n(i).y_{n-k}(i)=\sum_{i=1}^{N}z_n(i).y_{n-k}(i)}$$

Ces n+m conditions peuvent s'écrire sous forme matricielle :

$$\mathbf{F^t}\times\mathbf{Y_n}=\mathbf{F^t}\times\mathbf{Z_n} \quad \Leftrightarrow \quad \mathbf{F^t}\times\mathbf{Y_n}=\mathbf{F^t}\times\mathbf{F}\times\mathbf{C}$$

$$\Leftrightarrow \quad \boxed{\mathbf{C}=\left(\mathbf{F^t}\times\mathbf{F}\right)^{-1}\times\mathbf{F^t}\times\mathbf{Y_n}}$$

$\mathbf{F^t}$ désigne la matrice transposée de la matrice $\mathbf{F}$:

$$\mathbf{F^t}=\begin{pmatrix} x_n(1) & x_n(2) & & \dots & x_n(N) \\ x_{n-1}(1) & x_{n-1}(2) & & \dots & x_{n-1}(N) \\ \vdots & \vdots & \vdots & & \vdots \\ -y_1(1) & -y_1(2) & . & \dots & -y_1(N) \\ . & . & . & & . \\ -y_{n-1}(1) & -y_{n-1}(2) & & \dots & -y_{n-1}(N) \end{pmatrix}$$

Si k n'est pas nul, on cherchera à identifier le processus d'entrée $x_{int}(t)$ et de sortie y(t) par un modèle de fonction de transfert

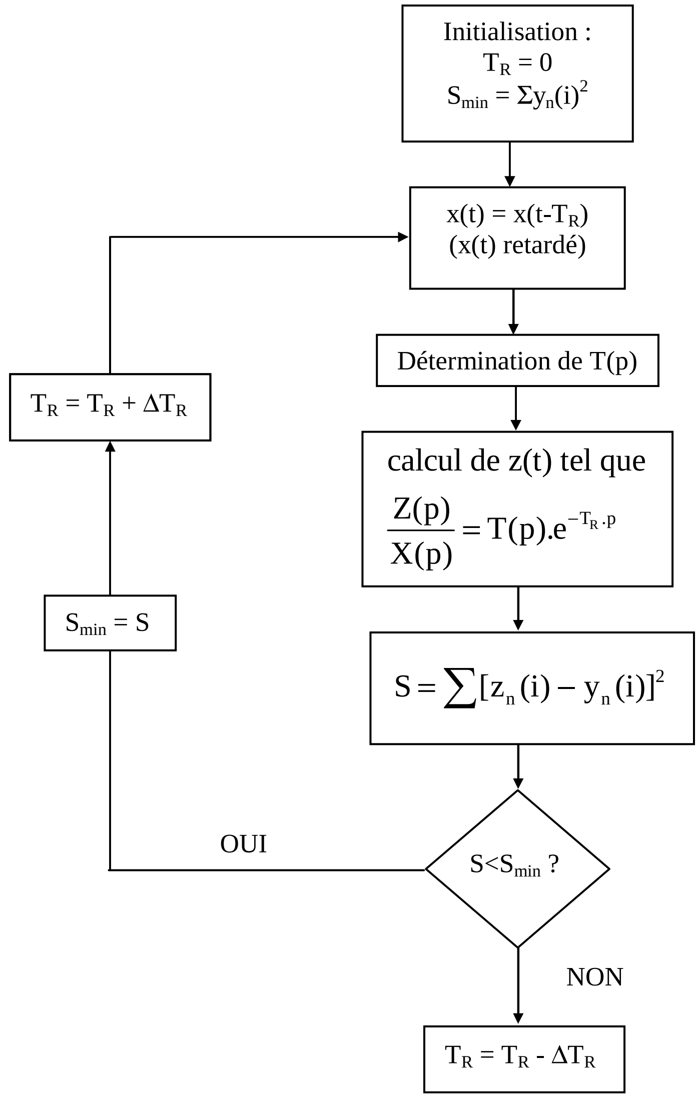
$$T'(p)=\frac{a_0+a_1.p+...+a_m.p^m}{\left(1+b_1.p+...+b_n.p^n\right)}$$

Où $x_{int}(t)$ correspond à x(t) intégré k fois :

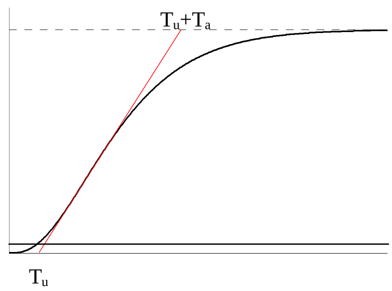
$$T(p)=\frac{Y(p)}{X(p)}=\frac{Y(p)}{X_{int}(p)}.\frac{X_{int}(p)}{X(p)}=T'(p).\frac{1}{p^k}$$

-3- Identification d'un processus présentant un retard

Soit T_R le retard. On désigne par ΔT_R le pas sur le retard ($\Delta T_R = 10.T_E$ dans un premier temps, puis $\Delta T_R = T_E$).On peut résumer le principe de l'identification par le schéma ci-dessous :



-4- Modèle de Strejc



On cherche ici un modèle de la forme :

$$T(p) = \frac{A}{(1+\tau.p)^n}.e^{-T_R \cdot p}$$

Pour cela, il faut tracer la tangente au point d'inflexion.

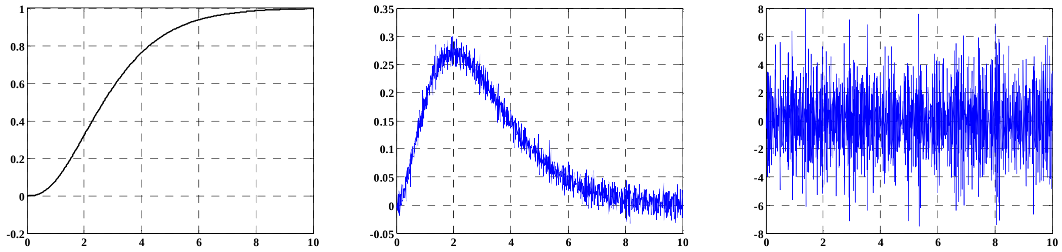
On peut montrer que pour un système d'ordre n sans retard :

$$T_a = \tau.\frac{(n-1)!}{(n-1)^{n-1}}.e^{(n-1)}$$

$$\frac{T_u}{T_a} = -1 + e^{-(n-1)}.\left[\frac{(n-1)^n}{(n-1)!} + \frac{(n-1)^{n-1}}{(n-1)!} + \frac{(n-1)^{n-2}}{(n-2)!} + ... + 1\right]$$

Le problème consiste donc à tracer cette tangente, et donc en premier lieu de déterminer le point d'inflexion, défini par $\frac{d^2y}{dt^2} = 0$. Pour des problèmes de bruit, on ne peut pas utiliser l'instruction ***diff*** :

y dy/dt=diff(y) d^2y/dt^2=diff(diff(y))



La première étape consistera donc à « approximer » la courbe y(t) par un polynôme de degré 5 *: P =* ***polyfit(t,y,5).***

On pourra alors déterminer la dérivée seconde du polynôme, puis chercher pour quelle valeur de t ce dernier polynôme s'annule :

DP = polyder(P) ; D2P = polyder(DP) ; t_inf=roots(D2P) ;

On pourra ainsi déterminer les valeurs de T_u et T_a, puis en déduire les valeurs de n, τ et T_R.

Mais ceci nécessite de connaître la valeur en régime établi. On sera donc amené à faire 2 enregistrements : le régime transitoire, puis le régime permanent.