

2.2.5 Boucles imbriquées et comportement logistique élémentaire.

2.2.5.1 Etude analytique

On suppose que Y représente la population d'un territoire isolé (sans émigration ni immigration). Y est une fonction continue dérivable en t , la dérivée de Y est proportionnelle à Y , on a la description de la **loi de Malthus** (1798).

$$\frac{dY}{dt} = cY \Rightarrow \frac{dY}{Y} = c \cdot dt \Rightarrow Y(t) = Y_0 \cdot e^{ct}$$

On suppose maintenant que c est une fonction affine de Y

$$c(Y) = a \cdot (b - Y) \quad a, b \in R^*$$

cela revient à dire que la croissance de la population dépend d'un facteur bloquant

On a alors l'équation différentielle

$$\dot{Y} = a \cdot Y \cdot (b - Y)$$

On peut essayer de donner une interprétation aux composantes

Y	= Population (individus)
$\dot{Y}$	= Taux de Naissance Net (individus/an)
a	= Norme Taux de Naissance Net (fraction/an)
$(b - Y)$	= Auxiliaire de Blocage (individus)
b	= Population maximale admissible (individus)

Dans l'état actuel des choses l'équation est mal dimensionnée car le membre de gauche est exprimé en *individus*² or en terme d'analyse dimensionnelle on doit avoir

$$choux = choux$$

Ainsi il faut redéfinir l'auxiliaire de blocage pour la rendre "sans dimension" on doit la mettre sous la forme

$$\frac{b - Y}{Y_0}$$

d'où l'équation différentielle traduisant la **loi de Verhulst** (1845).

$$\dot{Y} = \frac{a \cdot Y \cdot (b - Y)}{Y_0}$$

Recherche d'une solution. On pose

$$\alpha = \frac{a}{Y_0}$$

l'équation peut s'écrire

$$\frac{dY}{Y \cdot (b - Y)} = \alpha \cdot dt \Leftrightarrow \left[\frac{1}{Y} - \frac{1}{Y - b} \right] \cdot dY = \alpha b \cdot dt$$

↓

Intégration terme à terme

$$Y = \frac{bK \cdot e^{\alpha b t}}{1 + K \cdot e^{\alpha b t}} = \frac{bK}{K + e^{-\alpha b t}} = \frac{b}{1 + \left(\frac{1}{K}\right) \cdot e^{-\alpha b t}} = \frac{b}{1 + \lambda \cdot e^{-\alpha b t}} = \frac{b}{1 + \lambda \cdot e^{-\mu t}}$$

$$t = 0 \Rightarrow Y_0 = \frac{b}{1 + \lambda} \Rightarrow \lambda = \frac{b - Y_0}{Y_0}$$

$$Y(t) = \frac{b}{1 + \lambda \cdot e^{-\mu t}}$$

$\lambda = \frac{b - Y_0}{Y_0}$ (facteur d'échelle, nombre sans dimension.) $\mu = \alpha b = \frac{ab}{Y_0}$ (taux de croissance du phénomène)
--

$$t \rightarrow +\infty \Rightarrow Y \rightarrow \frac{b}{1 + \lambda \times 0} = b$$

$b \equiv$ population maximale atteinte dans le temps $\equiv M$ (pour maximum)

Fonction obtenue = fonction logistique

Graphe $\rightarrow$ " courbe logistique "
" courbe sigmoïde " (courbe en S)

↓

Décrit le comportement de tout phénomène borné par des facteurs bloquants.

Caractéristiques de la courbe.

$$\dot{Y}(t) = \frac{b\lambda\mu \cdot e^{-\mu t}}{(1 + \lambda \cdot e^{-\mu t})^2} > 0$$

$$\begin{aligned} t \rightarrow +\infty &\Rightarrow e^{-\mu t} \rightarrow 0^+ \Rightarrow \dot{Y}(t) \rightarrow 0^+ \\ t \rightarrow -\infty &\Rightarrow \dot{Y}(t) \cong \frac{b\mu}{\lambda \cdot e^{-\mu t}} \rightarrow 0^+ \end{aligned} \Rightarrow \exists \text{ point d'inflexion}$$

↓

$$\ddot{Y}(t) = \frac{b\lambda\mu^2 \cdot e^{-\mu t} (\lambda \cdot e^{-\mu t} - 1)}{(1 + \lambda \cdot e^{-\mu t})^3}$$

$$\ddot{Y}(t) \geq 0 \Rightarrow \lambda \cdot e^{-\mu t} \geq 1 \Rightarrow t \leq \frac{\ln \lambda}{\mu}$$

$$t = \frac{\ln \lambda}{\mu} \Rightarrow \begin{cases} \ddot{Y}(t) = 0 \\ \dot{Y}(t) = \frac{b\mu}{4} = \frac{ab^2}{4N_0} \\ Y(t) = \frac{b}{2} \end{cases}$$

↓
Point d'inflexion $\Omega \left(\frac{\ln \lambda}{\mu}, \frac{b}{2} \right)$ centre de symétrie de la courbe

NB

Le point d'inflexion n'apparaît que si

$$\lambda > 1 \Leftrightarrow \lambda = \frac{b - Y_0}{Y_0} > 1 \Rightarrow b > 2Y_0$$

Si

$$0 < \lambda < 1 \Leftrightarrow Y_0 < b < 2Y_0$$

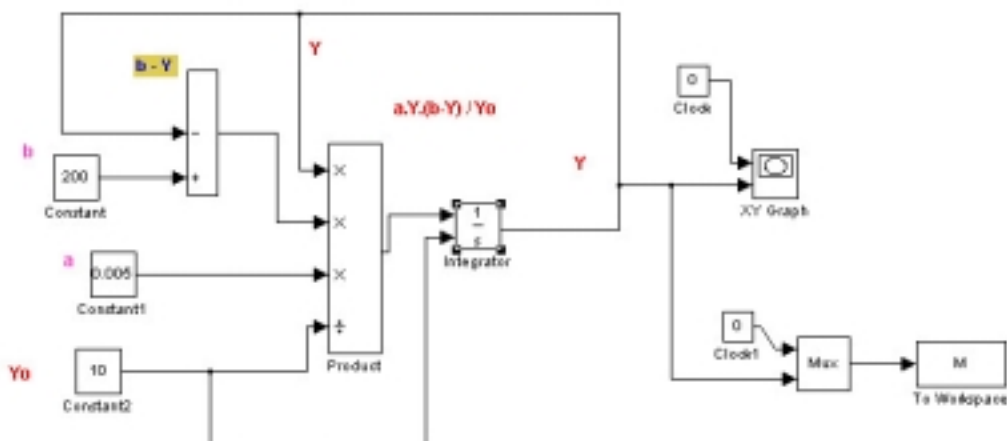
l'inflexion se fait pour des t négatifs
(phénomène logistique caché à l'observateur)

2.2.5.2 Simulation.

Pour modéliser sous Simulink on reprend BRNPOSP que l'on modifie quelque peu

- ajout de ports d'entrée dans le bloc product avec modification de signes,
- initialisation du bloc integrator par une source externe (Y_0).

On a ainsi le modèle " Verhulst "



Smlk049.

Les paramètres ont été fixés comme suit, on suppose que l'on a

$$b = 200 \quad Y_0 = 10$$

on veut que le taux de croissance μ soit égal à 0.1 on en déduit a

$$\mu = \alpha b = \frac{ab}{Y_0} = a \frac{200}{10} = 20a = 0.1 \Rightarrow a = 0.005$$

On réalise une simulation en prenant

$$t_{start} = 0 \quad t_{stop} = 100 \quad ode45$$

On envoie toujours la matrice M dans le Workspace, on passe les expressions

```
» t=M(:,1);
» Y=M(:,2);
```

Supposons que l'on veuille comparer le comportement simulé au comportement théorique. On a

$$\lambda = \frac{b - Y_0}{Y_0} = \frac{200 - 10}{10} = 19$$

d'où la fonction

$$Y = \frac{200}{1 + 19 \exp(-0.1t)}$$

Il nous faut engendrer un vecteur à 101 lignes donnant les valeurs de cette fonction sur $[0, 100]$ on écrit alors sous Matlab

```
» for i=1:101
V(i)=200/(1+19*exp(-0.1*(i-1)));
end
```

Explications.

On fait de la programmation sous Matlab tout comme on pourrait en faire dans une *M-file*. Le langage de programmation de Matlab (les fonctions utilisées) ressemble fort au bon vieux Basic.

Ici on veut décrire un vecteur à 101 lignes, l'indice de boucle i est aussi l'indice du vecteur. Il va nous servir pour décrire le temps qui va de 0 à 100. On ne peut prendre comme indice de boucle et de vecteur une valeur nulle sinon Matlab renvoie le message d'erreur

??? Index into matrix is negative or zero.

D'où "l'astuce" utilisé dans la formulation explicite de $V(i)$ qui revient à employer $(i - 1)$. On notera en outre que lorsque l'on décrit une boucle de la forme

```
for variable = expression
    énoncé
end
```

on tape Enter pour changer de ligne. Alors la boucle ne s'exécute qu'après le Enter suivant end. On voit d'ailleurs que le symbole ' » ' qui apparaît par défaut devant le for n'apparaît pas devant l'énoncé qui figure dans la boucle, ce qui veut dire que cet énoncé n'est pas exécutable tant que la boucle n'est pas fermée par un end. Si l'on a plusieurs boucles imbriquées du type

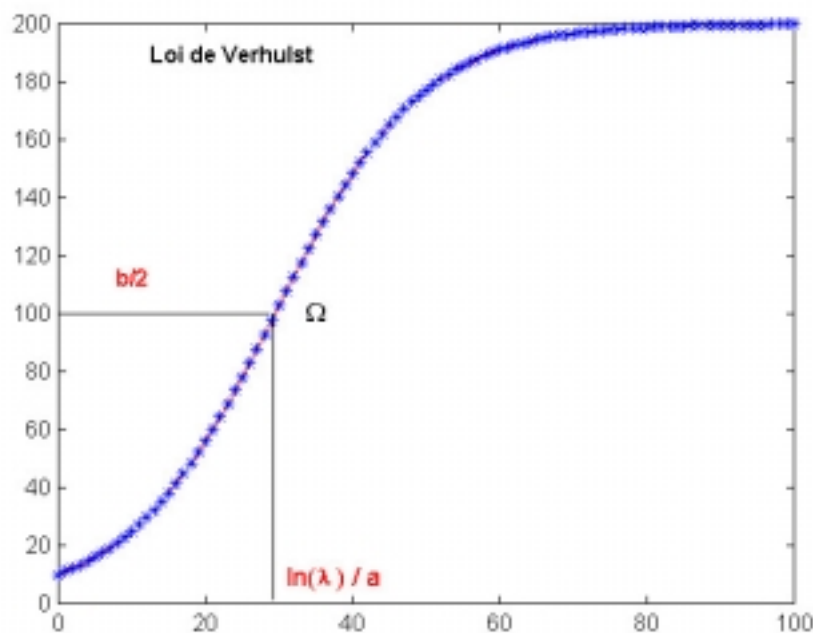
```
for i =
  for j =
    for k =
      u(i,j,k) =
    end
  end
end
```

elles ne s'exécutent qu'après la frappe du dernier end.

Cela étant on passe les expressions

```
» plot(t,Y,'-r',t,V,'*b')
» set(gcf,'color','w')
```

On obtient le graphe amélioré



Smlk050.

Graphiquement on constate que le solveur ode45 a bien fait son travail puisque la courbe en étoiles bleues recouvre la courbe continue rouge. Pour s'assurer de ce qu'il en est on va regarder la différence

$V'-Y$

Pourquoi V' ? Parce que tel qu'on la défini ce vecteur est un vecteur ligne alors que Y est un vecteur colonne. Si l'on prenait l'expression $V - Y$ on aurait le message d'erreur

??? Error using ==> -
Matrix dimensions must agree.

Pour s'éviter une longue liste de valeurs (la différence n'étant pas nulle) on prend

$$\gg \max(V'-Y), \min(V'-Y)$$

Noter la séparation des deux expressions par une virgule on obtient les réponses

ans =
7.558980996691389e-009
ans =
-1.032427547897896e-007

De fait comme l'on n'a pas écrit l'expression

toto = qchs

c'est à dire comme l'on n'a pas affecté de nom de variable à l'expression calculée, Matlab crée automatiquement la variable `ans` (= *answer*). On notera que les erreurs entre valeurs simulées et valeurs théoriques sont faibles

-1 dix millionième d'un côté
7 milliardième de l'autre

Cela étant on pourrait se poser la question de savoir ce qui se passe si jamais la valeur initiale est supérieure au maximum. En l'absence de réponse analytique il faudrait faire des simulations réitérées en modifiant les paramètres. Ici on peut apporter une réponse analytique qui est la suivante

Cas $0 < b < Y_0$

On note

$$\lambda = -\nu \in \mathbb{R}_+^*$$

↓

$$Y(t) = \frac{b}{1 - \nu \cdot e^{-\mu t}} \quad \nu = \frac{N_0 - b}{N_0} < 1$$

$$\mathcal{D} = \mathbb{R} - \{t^*\} \quad t^* = \frac{\ln \nu}{\mu} < 0$$

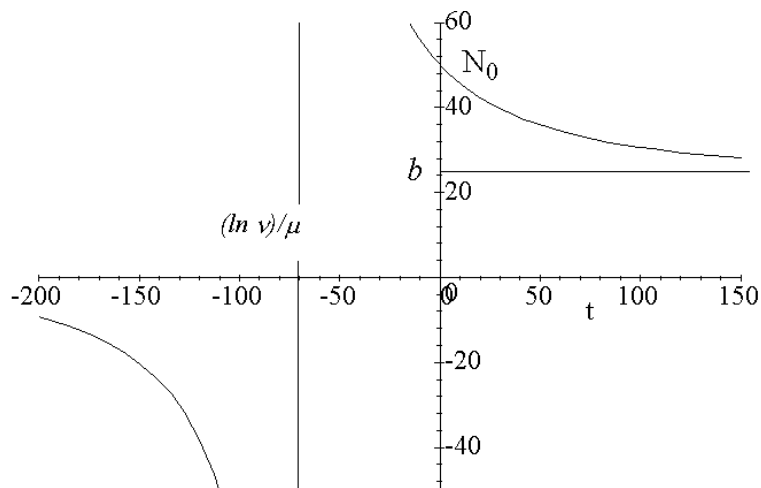
⇒ asymptote verticale en $t = t^*$

$$\begin{array}{c}
\boxed{
\begin{array}{l}
t \rightarrow t * +\varepsilon \Rightarrow N \rightarrow +\infty \\
t \rightarrow t * -\varepsilon \Rightarrow N \rightarrow -\infty
\end{array}
} \\
\boxed{
\begin{array}{l}
t \rightarrow +\infty \Rightarrow N \rightarrow \frac{b}{1-\varepsilon} \approx b(1+\varepsilon) \Rightarrow N \rightarrow b^+ \\
t \rightarrow -\infty \Rightarrow N \rightarrow \frac{b}{1-\infty} = 0^-
\end{array}
} \\
\Downarrow \\
2 \text{ asymptotes horizontales } \left\{ \begin{array}{l} N = b \text{ à droite} \\ N = 0 \text{ à gauche} \end{array} \right.
\end{array}$$

$$Y'(t) = \frac{-\mu \nu b \cdot e^{-\mu t}}{(1 - \nu \cdot e^{-\mu t})^2} < 0 \quad \forall t$$

$\Downarrow$
fonction décroissante sur $\mathcal{D}$
 $\downarrow$
comportement logistique “ *abâtardi* ”

$$\begin{aligned}
Y(t) &= \frac{b}{1 - \nu \cdot \exp(-\mu \times t)} & \nu &= \frac{N_0 - b}{N_0} & \mu &= \frac{a \times b}{N_0} \\
N_0 &= 50 & b &= 25 & a &= \frac{1}{50}
\end{aligned}$$



Tssd06.

2.2.6 Boucle de rétroaction négative du second ordre et comportement logistique.

Dans l'exemple précédent nous nous sommes contentés de traduire la loi de Verhulst ici nous allons voir comment on peut engendrer un tel comportement en considérant 2 variables d'état, la croissance de l'une étant freinée par l'épuisement de l'autre. On a ainsi

1 niveau principal Y
1 niveau secondaire Z

On peut considérer deux cas. Dans ces deux cas Y représente une population, dans un cas Z peut être assimilé à un stock de nourriture, dans l'autre à un espace occupé par la population.

2.2.6.1 Premier cas. Epuisement du niveau secondaire à raison du niveau principal.

On a deux variables d'état et donc deux équations différentielles

$\dot{Y} = X1$		$\dot{Z} = (-X2)$
$X1 = a.Y.R$	$R = Z/Z_0$	$X2 = b.Y.R$

On note que le niveau Y se remplit par le flux $X1$ alors que le niveau Z se vide par le flux $X2$ fonction du niveau Y . On a une variable auxiliaire R qui est sans dimension ($Choux/Choux$). Le coefficient b a un statut spécial

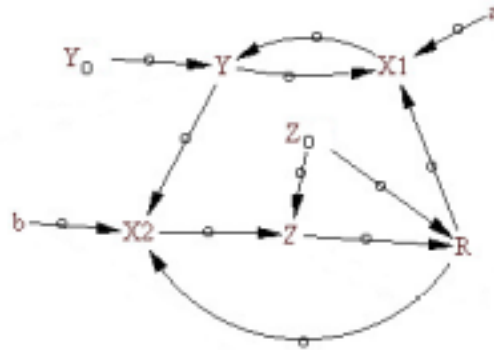
$b = \text{"coefficient de conversion"}$

$$\frac{\text{unités de } Z}{(\text{unités de } Y) \times (\text{unités de temps})}$$

$R = \text{auxiliaire de blocage}$

$\Rightarrow$ plus Z s'épuise plus la croissance de Y est ralentie,
l'épuisement de Z se faisant lui aussi de façon
moins que proportionnelle.

La logique du modèle peut être traduite par le diagramme de structure causale suivant



Smlk051.

Etude analytique.

On a les deux équations différentielles

$$\begin{aligned}\dot{Y} &= \frac{aYZ}{Z_0} \\ \dot{Z} &= -\frac{bYZ}{Z_0}\end{aligned}$$

On note alors que

$$\frac{\dot{Z}}{\dot{Y}} = -\frac{b}{a} \Rightarrow \dot{Z} = -\frac{b}{a} \cdot \dot{Y} \Rightarrow Z = -\frac{b}{a} \cdot Y + C$$

La dérivée de Z étant égale à celle de Y à une constante multiplicative près, Z se déduit de Y à une constante additive près après intégration. Par retour aux conditions initiales on a

$$C = Z_0 + \frac{b}{a}Y_0$$

En remplaçant Z par sa valeur en fonction de Y dans l'équation différentielle de Y il vient

$$\begin{aligned}\dot{Y} &= \frac{aY}{Z_0} \left[-\frac{b}{a} \cdot Y + Z_0 + \frac{b}{a} \cdot Y_0 \right] \\ &= -\frac{b}{Z_0}Y^2 + \left[a + \frac{b}{Z_0}Y_0 \right] \cdot Y\end{aligned}$$

soit

$$\dot{Y} = \alpha Y^2 + \beta Y \quad \alpha = -\frac{b}{Z_0} \quad \beta = a + \frac{b}{Z_0}Y_0$$

On note que

$$\frac{\dot{Y}}{Y^2} = \alpha + \frac{\beta}{Y}$$

on effectue alors le changement de variable $u = 1/Y$ et l'on obtient

$$-\dot{u} = \alpha + \beta u \Leftrightarrow -\dot{u} - \beta u = \alpha$$

qui est une équation différentielle du premier ordre non homogène que l'on sait intégrer. La méthode d'intégration en deux étapes valable dans ce cas, déjà vue antérieurement, conduit à la solution en u

$$u = -\frac{\alpha}{\beta} \cdot \left[1 + k \cdot e^{-\beta t} \right]$$

Comme $u = 1/Y$, il vient finalement

$$Y = \frac{-\beta/\alpha}{1 + k \cdot e^{-\beta t}}$$

On retrouve bien une équation de type logistique dans laquelle le maximum est $M = -\beta/\alpha$ et le facteur d'échelle est k . La détermination de ce facteur d'échelle se fait par retour aux conditions initiales. On a

$$Y_0 = \frac{-\beta/\alpha}{1 + k} \Rightarrow k = \frac{a}{b} \cdot \frac{Z_0}{Y_0}$$

La solution générale explicitée en fonction des conditions initiales s'obtient en remplaçant α , β et k par leurs valeurs

$$Y(t) = \frac{Y_0 + \frac{a}{b} \cdot Z_0}{1 + \frac{a}{b} \cdot \frac{Z_0}{Y_0} \cdot \exp \left[- \left(a + \frac{b}{Z_0} \cdot Y_0 \right) \cdot t \right]} \equiv \frac{M}{1 + \frac{1}{\lambda} \cdot e^{-\mu t}} \quad (1)$$

Pour obtenir l'équation de comportement temporel du niveau secondaire Z on considère les relations

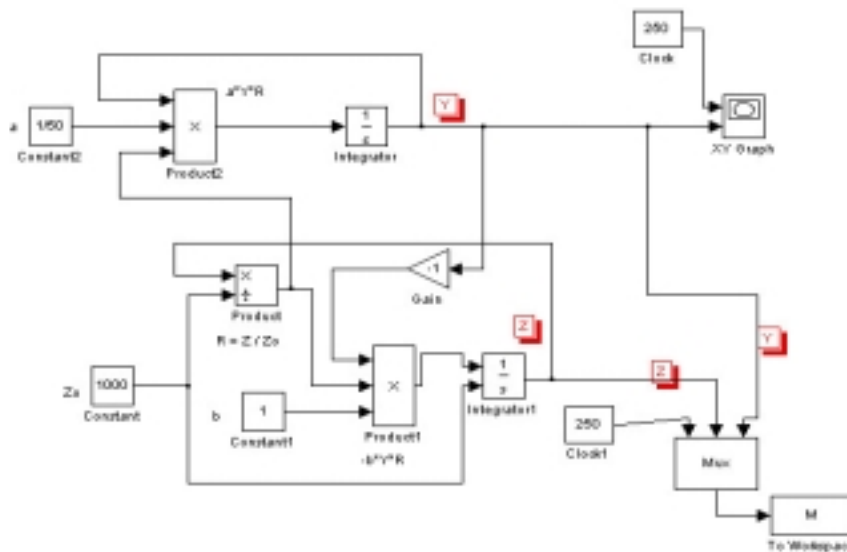
$$Z = -\frac{b}{a} \cdot Y + C \quad C = Z_0 + \frac{b}{a} \cdot Y_0$$

D'où l'on tire, après remplacement de Y par son équation

$$Z(t) = \frac{Z_0 + \frac{b}{a} \cdot Y_0}{1 + \frac{b}{a} \cdot \frac{Y_0}{Z_0} \cdot \exp \left[\left(a + \frac{b}{Z_0} \cdot Y_0 \right) \cdot t \right]} \equiv \frac{\frac{b}{a} \cdot M}{1 + \frac{1}{\lambda} \cdot e^{\mu t}} \quad (2)$$

Simulation.

Pour dessiner le modèle sous Simulink on repart à zéro. On réutilisera seulement les assemblages de sortie employés dans les autres modèles. Le modèle dénommé *Logis1* se présente comme suit.



Smlk052.

Nous ne nous attarderons pas sur sa construction, tous les éléments ont déjà été donnés. La seule chose qu'il faut savoir c'est que l'on n'arrive pas au dessin final du premier coup. De fait on importe les blocs, on trace les flèches, mais on est aussi amené à déplacer des éléments ou à les redimensionner.

Par exemple dans le schéma ci-dessus aucune valeur n'apparaît dans les "horloges", il suffit d'agrandir un peu ces blocs pour faire apparaître le temps final.

On notera que le bloc Mux a désormais 3 ports d'entrée un pour t , l'autre pour Z le troisième pour Y (attention à l'**ordre d'entrée** qui est celui du schéma quand on le lit de gauche à droite !). On notera aussi que l'on a conservé le bloc XY Graph, pourquoi alors que l'on fait des graphes améliorés sous Matlab ? Ce bloc est utile à titre d'avertisseur. Il permet d'observer très vite le comportement obtenu, en outre, en sa présence, l'ordinateur émet un "beep" qui permet de savoir que la simulation est finie.

Notons aussi que l'initialisation de Z se fait à l'aide d'une source externe, la constante Z_0 qui entre aussi dans le calcul de R .

Les valeurs paramétriques utilisées sont les suivantes

$a = 1/50$	$b = 1$
$Y_0 = 2$	$Z_0 = 1000$

Le paramètre a est pris sous forme fractionnaire pour bien montrer qu'il a la dimension

$$\frac{\text{fraction}}{\text{unité de temps}}$$

cela veut dire aussi que si Y croissait sans être entravé par Z son temps de doublement (à taux constant) serait égal à 50 unités de temps.

On va faire une simulation avec $t \in [0, 250]$, pour fixer les ordonnées du bloc **XY Graph** on note que théoriquement on a

$$\max(Y) = Y_0 + \frac{a}{b}Z_0 = 2 + \frac{1}{50}1000 = 22$$

c'est pourquoi dans le bloc **XY Graph** on peut prendre $y \in [0, 25]$.

Si on réalise une simulation, on voit apparaître un graphique de Y qui est celui attendu. Cependant on a envoyé une matrice $M = [t \ Z \ Y]_{(251 \times 3)}$ dans le Workspace. On passe alors les expressions

```
» t=M(:,1);
» Z=M(:,2);
» Y=M(:,3);
```

Si l'on fait

```
» plot(t,Y,t,Z)
» set(gcf,'color','w')
```

on a la surprise (qui ne devrait pas en être une) d'avoir un graphe fort peu agréable de fait Z démarre à 1000 alors que Y s'arrête à 21.136... Aussi le graphe de Y est quasi invisible. On pourrait utiliser la solution consistant à écrire

```
» plot(t,50*Y,t,Z)
```

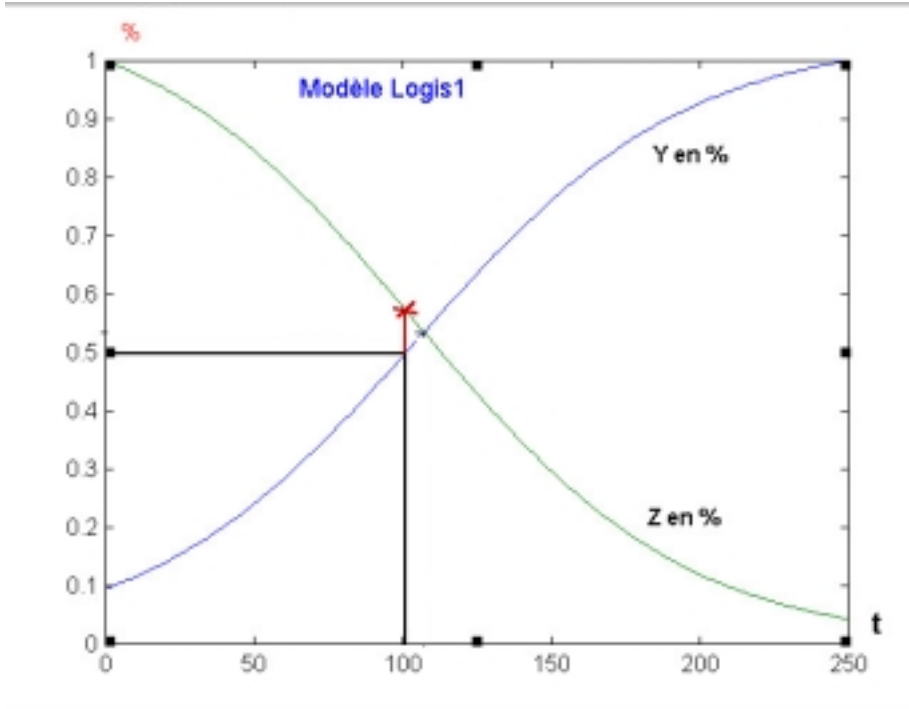
qui nous donne deux courbes quasi-symétriques. Cependant il existe une solution plus judicieuse, dans le cas présent, qui permet de mettre à jour un phénomène intéressant. On peut représenter l'évolution de Y et de Z sur un même graphique en raisonnant en pourcentages. Pour Y on considère l'évolution de Y/M , pour Z celle de Z/Z_0 de sorte que le maximum de chaque courbe est à 100%. On écrit les expressions

```
» Yp=Y/max(Y);
» Zp=Z/max(Z);
```

et on fait

```
» plot(t,Yp,t,Zp)
```

On obtient le graphe amélioré



Smlk053.

Revenons aux équations décrivant le comportement de Y et celui de Z

$$Y(t) = \frac{Y_0 + \frac{a}{b} \cdot Z_0}{1 + \frac{a}{b} \cdot \frac{Z_0}{Y_0} \cdot \exp \left[- \left(a + \frac{b}{Z_0} \cdot Y_0 \right) \cdot t \right]} \equiv \frac{M}{1 + \lambda \cdot e^{-\mu t}} \quad (1)$$

$$Z(t) = \frac{Z_0 + \frac{b}{a} \cdot Y_0}{1 + \frac{b}{a} \cdot \frac{Y_0}{Z_0} \cdot \exp \left[\left(a + \frac{b}{Z_0} \cdot Y_0 \right) \cdot t \right]} \equiv \frac{\frac{b}{a} \cdot M}{1 + \frac{1}{\lambda} \cdot e^{\mu t}} \quad (2)$$

→ Dans (1) comme dans (2) l'inflexion se produit en $t = \frac{\ln \lambda}{\mu}$

→ On a $M = Y_0 + \frac{a}{b} \cdot Z_0 = \varphi(Y_0, Z_0, a, b)$

. Dans (1) on sait qu'à l'inflexion on a $Y = M/2$, alors $\lambda = e^{\mu t}$. Si on porte cette valeur de λ dans (2) on obtient

$$Z = \frac{(b/a) \cdot M}{1 + 1} = \frac{b}{a} \cdot \frac{M}{2} = \frac{Z_0}{2} + \frac{b}{a} \cdot \frac{Y_0}{2} > \frac{Z_0}{2}$$

Ainsi quand Y est à 50% de sa valeur maximale, Z est au dessus des 50% de sa valeur initiale. Tout se passe comme s'il y avait une "anticipation" du niveau Y . Celui-ci s'infléchit à la moitié de sa valeur maximale, alors que le niveau secondaire, qui borne son évolution, n'a pas encore été épuisé de moitié. C'est ce qu'on voit bien dans le graphe antérieur.

Cela étant il est possible de montrer que les deux courbes du graphe ci-dessus se coupent en dessus de la valeur 50%. Dans ce qui suit on posera

$$x = e^{\mu t} \quad k = \frac{b}{a} \quad M = Y_0 + \frac{a}{b} \cdot Z_0 \quad \nu = \frac{kM}{Z_0} = 1 + k \frac{Y_0}{Z_0} = 1 + c$$

La résolution de $Y/M = Z/Z_0$ conduit à l'équation du second degré

$$x^2 + \lambda(1 - \nu)x - \lambda^2\nu = 0$$

dont le discriminant est

$$\Delta = \lambda^2(1 + \nu)^2$$

D'où les deux racines

$$x = \lambda\nu \text{ et } x = -\lambda$$

La seconde racine ne convient pas puisque par définition on a $x > 0$. Ainsi l'on a $e^{\mu t} = \lambda\nu$. Alors

$$\frac{Y}{M} = \frac{Z}{Z_0} = \frac{\nu}{1 + \nu} = \frac{1}{1 + \frac{1}{\nu}} = \frac{1}{1 + \frac{1}{1+c}}$$

On a $c > 0$, la fonction $1/(1 + c)$ est alors strictement inférieure à 1 et tend vers 0 par valeurs supérieures quand c augmente indéfiniment. La valeur de coupure VC des deux courbes est donc telle que

$$VC \approx \frac{1}{1 + (1 - \varepsilon)} > \frac{1}{2}$$

2.2.6.2 Deuxième cas. Epuisement du niveau secondaire à raison de l'accroissement du niveau principal.

Dans ce deuxième cas la seule modification apportée à l'ensemble des équations l'est pour l'équation décrivant X_2 qui s'écrit désormais

$$X_2 = (aYR) \cdot b = X_1b$$

Solution analytique.

On a les deux équations différentielles

$$\dot{Y} = \frac{aYZ}{Z_0} \quad \dot{Z} = -\frac{abYZ}{Z_0}$$

Leurs solutions s'obtiennent aisément en remplaçant b par ab dans les solutions données antérieurement. Il vient ainsi

$$Y(t) = \frac{M}{1 + \lambda \cdot e^{-\mu t}} \quad Z(t) = \frac{bM}{1 + \frac{1}{\lambda} \cdot e^{\mu t}}$$

avec

$$M = Y_0 + \frac{1}{b} \cdot Z_0 = \varphi(Y_0, Z_0, b) \quad \lambda = \frac{1}{b} \cdot \frac{Z_0}{Y_0} \quad \mu = a + \frac{ab}{Z_0} \cdot Y_0$$

Le type de comportement évoqué ici est en quelque sorte un comportement de type “ proie-prédateur ” un peu particulier que l'on peut illustrer

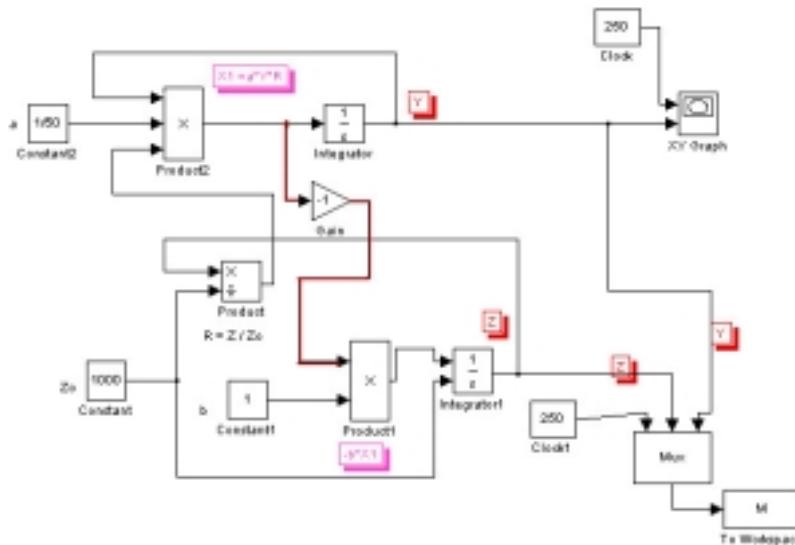
par une analogie. Le niveau Z représente la superficie d'une plaque métallique. Le niveau Y quant à lui est la superficie d'une tâche de rouille qui apparaît aléatoirement sur la plaque métallique. La tâche de rouille se développe à raison du rapport Z/Z_0 , qui représente, à tout instant, le rapport

$$\frac{\text{Surface restante non oxydée}}{\text{Surface métallique initiale}}$$

Cela étant lorsque l'on parle d'acier inoxydable il ne faut pas croire que cet acier ne puisse pas être attaqué par la rouille. Il le sera un jour ou l'autre. Tout ce qu'on fait les métallurgistes c'est de trouver des procédés qui permettent de reculer le plus loin possible, dans le temps, le point d'inflexion de la logistique représentative de l'évolution de la tâche de rouille. En outre pour ce qui est des modèles "proies-prédateurs" des exemples seront donnés montrant qu'il y a d'autres modes de comportement, oscillatoires notamment.

Simulation.

On peut utiliser le modèle *Logis1* et le modifier pour aboutir à *Logis2*. On a le schéma



Smlk054.

On comparera avec profit les diagrammes *Smlk052* et *Smlk054*. Désormais le flux agissant sur Y agit aussi sur Z . On est dans le cas où il y a correspondance directe entre deux flux à l'aide d'un coefficient de conversion. Ce coefficient de conversion b a pour dimension

$$\frac{\text{Unités de } Z}{\text{Unités de } Y}$$

Si l'on conserve les valeurs paramétriques antérieures

$a = 1/50$	$b = 1$
$Y_0 = 2$	$Z_0 = 1000$

on note que

$$M = Y_0 + \frac{1}{b} \cdot Z_0 = 1002$$

Il faut donc modifier les paramètres du bloc XY Graph si on veut avoir un affichage correct. Si on réalise la simulation le graphe s'affiche et on se rend compte qu'un t_{stop} de 250 est insuffisant. Il faut le passer à 500 tout à la fois dans les paramètres de simulation et dans ceux du bloc XY Graph.

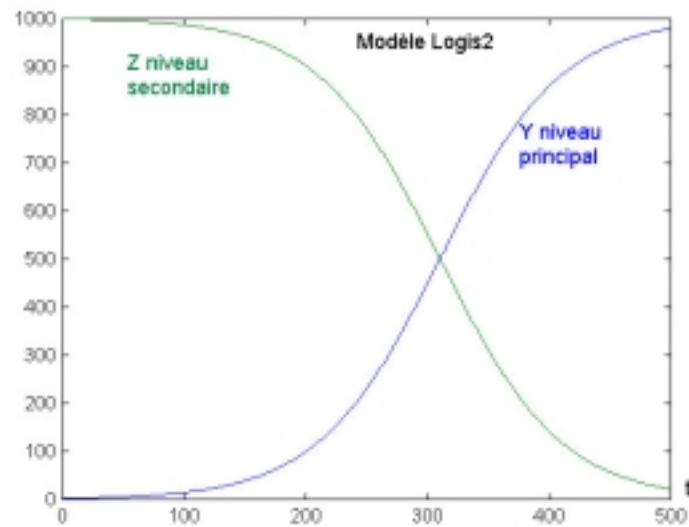
On repasse les expressions

```
» t=M(:,1);
» Z=M(:,2);
» Y=M(:,3);
```

Compte tenu des valeurs prises par Y et Z on peut écrire

```
» plot(t,Y,t,Z)
» set(gcf,'color','w')
```

d'où le graphe amélioré



Smlk055.

2.2.7 Boucle de rétroaction du second ordre et comportement oscillatoire.

Le modèle que nous allons traiter maintenant est censé engendrer un comportement oscillatoire. Il est d'ailleurs présenté comme tel par J.W. FORRESTER. Mais ainsi que nous le verrons le comportement engendré dépend

des valeurs paramétriques mises en jeu. Le comportement oscillatoire n'est pas nécessairement assuré même si c'est le cas le plus probable.

2.2.7.1 Etude analytique

Ce modèle est décrit par l'ensemble d'équations suivant.

$$\begin{aligned}\dot{Y}_1 &= X_1 & X_1 &= Y_2/D_1 \\ \dot{Y}_2 &= X_2 - X_1 & X_2 &= (S - Y_1)/D_2\end{aligned}$$

Pour illustrer les choses on peut dire que Y_1 est un stock de marchandises dont on désire qu'il atteigne un seuil désiré S . D'où la politique de commande décrite par X_2 , en fonction d'un écart sur stock, et d'un certain délai de temps D_2 , (*laps de temps nécessaire à taux constant pour combler l'écart sur stock enregistré*). Cela étant on suppose que les marchandises commandées n'entrent pas immédiatement en stock. Elles ne le font qu'au bout d'un certain laps de temps représenté par le délai D_1 (*délai de livraison*). Le flux X_1 représente alors le flux d'entrée en stock, alors que le niveau Y_2 représente les marchandises en commande.

On a les deux équations différentielles

$$\dot{Y}_1 = \frac{Y_2}{D_1} \quad (1)$$

$$\dot{Y}_2 = -\frac{Y_2}{D_1} + \frac{S - Y_1}{D_2} \quad (2)$$

L'équation (1) nous donne

$$Y_2 = D_1 \cdot \dot{Y}_1 \Rightarrow \dot{Y}_2 = D_1 \cdot \ddot{Y}_1$$

D'où en reportant ces deux valeurs dans (2) et en divisant par D_1

$$\ddot{Y}_1 + \frac{\dot{Y}_1}{D_1} + \frac{Y_1}{D_1 \cdot D_2} = \frac{S}{D_1 \cdot D_2} \quad (3)$$

On obtient ainsi une équation différentielle du second ordre, à coefficients constants, avec second membre. De fait l'équation différentielle représentative du comportement de Y_1 est du second ordre puisque que Y_1 est inclus dans une boucle comprenant deux délais de temps du premier ordre. A l'opposé l'équation donnant le comportement de Y_2 à Y_1 connu n'est que du premier ordre.

On sait que pour une équation comme (3) la solution générale avec second membre est égale à la solution générale sans second membre augmentée d'une solution particulière avec second membre, soit l'égalité mnémotechnique

$$SGASM = SGSSM + SPASM$$

On a donc une procédure de résolution en deux temps : recherche d'une solution générale suivie de la recherche d'une solution particulière

L'**équation caractéristique** associée à l'équation différentielle s'écrit

$$r^2 + \frac{r}{D_1} + \frac{1}{D_1 \cdot D_2} = 0$$

son discriminant est

$$\Delta = \frac{1}{D_1^2} \cdot \left[1 - \frac{4D_1}{D_2} \right]$$

On a ainsi trois cas possibles selon le signe du déterminant

$$\Delta < 0 \Leftrightarrow D_1 > D_2/4$$

$$\Delta = 0 \Leftrightarrow D_1 = D_2/4$$

$$\Delta > 0 \Leftrightarrow D_1 < D_2/4$$

Le premier est le plus courant en ce sens que c'est toujours celui présenté dans la littérature SSD. Dans l'exemple d'un modèle de stock il correspondrait à l'hypothèse où le délai de livraison est supérieur au quart du laps de temps nécessaire pour combler l'écart sur stock, à taux constant. Cela étant on voit bien qu'il existe deux autres cas possibles qui ne sont pas à exclure.

Etude du cas où le discriminant est négatif.

Recherche d'une solution générale.

La théorie mathématique nous dit que dans ce cas la SGSSM est de la forme

$$Y_1 = C_1 \cdot e^{r_1 t} + C_2 \cdot e^{r_2 t}$$

où r_1 et r_2 sont les deux racines complexes conjuguées de l'équation caractéristique, soit

$$\begin{aligned} r_1 &= -\frac{1}{2D_1} + \frac{i}{2D_1} \sqrt{\frac{4D_1}{D_2} - 1} = \alpha + i\beta \\ r_2 &= -\frac{1}{2D_1} - \frac{i}{2D_1} \sqrt{\frac{4D_1}{D_2} - 1} = \alpha - i\beta \end{aligned}$$

Alors on peut écrire

$$Y_1 = e^{\alpha t} \cdot [C_1 \cdot e^{i\beta t} + C_2 \cdot e^{-i\beta t}]$$

Or on sait que

$$e^{i\beta t} = \cos \beta t + i \sin \beta t \quad e^{-i\beta t} = \cos \beta t - i \sin \beta t$$

donc

$$Y_1 = e^{\alpha t} \cdot [(C_1 + C_2) \cdot \cos \beta t + i(C_1 - C_2) \cdot \sin \beta t]$$

Puisque C_1 et C_2 sont des constantes quelconques on peut les prendre complexes conjuguées soit

$$C_1 = \lambda - \mu i \quad C_2 = \lambda + \mu i$$

D'où il vient

$$\begin{aligned} Y_1 &= e^{\alpha t} \cdot [2\lambda \cdot \cos \beta t + 2\mu \cdot \sin \beta t] \\ &= e^{\alpha t} \cdot [A \cdot \cos \beta t + B \cdot \sin \beta t] \\ &= C \cdot e^{\alpha t} \cdot \cos(\beta t - \varphi) \end{aligned}$$

avec

$$A = C \cdot \cos \varphi \quad B = C \cdot \sin \varphi$$

Au total on peut donc écrire

$$Y_1(t) = C \cdot e^{-t/2D_1} \cdot \cos \left[\frac{t}{2D_1} \sqrt{\frac{4D_1}{D_2} - 1} - \varphi \right]$$

Obtention d'une solution particulière.

Les racines caractéristiques étant différentes de zéro (elles le seraient dans le cas précédent si D_1 tendait vers l'infini), le second membre étant une constante (notons la K) on montre que dans ce cas quel que soit Δ une solution particulière est

$$Y_1^* = \frac{K}{r_1 \cdot r_2}$$

soit

$$Y_1^* = \frac{K}{\alpha^2 + \beta^2} \quad \text{si} \quad \Delta < 0$$

Ici l'on a

$$Y_1^* = \frac{S}{D_1 \cdot D_2} \cdot \frac{1}{\alpha^2 + \beta^2} = S \quad \text{car} \quad \alpha^2 + \beta^2 = \frac{1}{D_1 \cdot D_2}$$

Solution générale au cas du discriminant négatif.

Au total on peut écrire

$$\begin{aligned} Y_1(t) &= C \cdot e^{-t/2D_1} \cdot \cos \left[\frac{t}{2D_1} \sqrt{\frac{4D_1}{D_2} - 1} - \varphi \right] + S \\ &= e^{\alpha t} \cdot [A \cos \beta t + B \sin \beta t] + S \end{aligned}$$

avec

$$C = \sqrt{A^2 + B^2} \quad tg\varphi = B/A \Leftrightarrow \varphi = \text{Arctg}(B/A) \quad -\pi/2 < \varphi < \pi/2$$

On détermine A et B , et donc C et φ , par retour aux conditions initiales.

On a en posant

$$\alpha = \frac{1}{2D_1}$$

$$\begin{aligned} Y_1(t) &= e^{-\alpha t} \cdot [A \cos \beta t + B \sin \beta t] + S \\ \Rightarrow \dot{Y}_1(t) &= e^{-\alpha t} \cdot [(\beta B - \alpha A) \cos \beta t - (\alpha B + \beta A) \sin \beta t] \end{aligned}$$

Alors

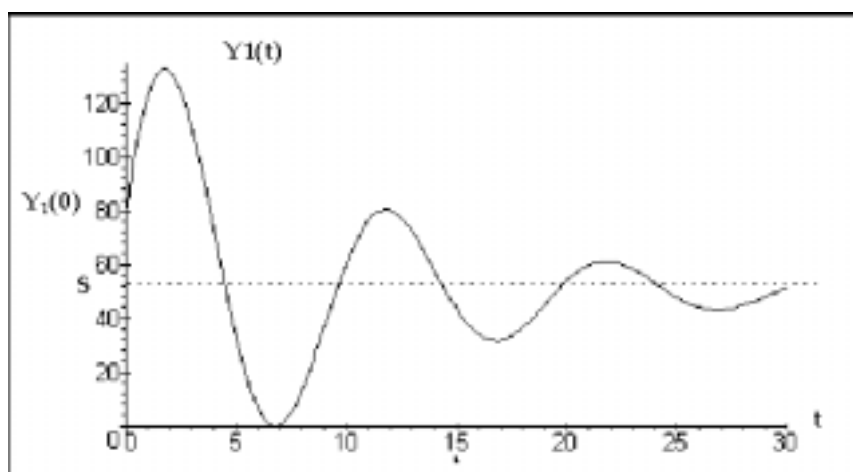
$$\begin{aligned} Y_1(0) &= A + S \\ \dot{Y}_1(0) &= \beta B - \frac{A}{2D_1} = \frac{Y_2(0)}{D_1} = X_1(0) \end{aligned}$$

d'où l'on tire

$$\begin{aligned} A &= Y_1(0) - S \\ B &= \frac{2Y_2(0) + Y_1(0) - S}{\sqrt{(4D_1/D_2) - 1}} \end{aligned}$$

On peut alors calculer C et φ .

L'équation de comportement de Y_1 n'est autre qu'une fonction sinusoïdale amortie, il en va de même pour Y_2 , puisque $Y_2 = D_1 \cdot \dot{Y}_1$. D'où le graphe de comportement temporel de principe de Y_1 donné ci-après.



Tssd07 : Mode de comportement oscillatoire de la boucle de rétroaction négative du second ordre.

On notera que ce mode de comportement est caractéristique d'une sorte de politique de "*stop and go*". Lorsqu'on est au dessus du seuil on vide le niveau, mais compte tenu des délais de réaction on passe en dessous du seuil. Lorsqu'on s'en aperçoit, avec retard, on remplit à nouveau le niveau, mais là encore les délais font que l'on passe au dessus du seuil. Le mouvement se poursuit la convergence vers le seuil se faisant progressivement.

Cas d'un discriminant nul.

Dans ce cas on a une racine double

$$r = -\frac{1}{2D_1}$$

Alors la SGSSM s'écrit

$$Y_1(t) = C_1 \cdot e^{rt} + C_2 \cdot t \cdot e^{rt}$$

Comme r est différent de 0, une solution particulière est

$$\frac{K}{r^2} = S$$

D'où la solution générale

$$Y_1(t) = (C_1 + C_2 \cdot t) \cdot e^{-t/2D_1} + S$$

On détermine les constantes par retour aux conditions initiales

$$\begin{aligned} Y_1(0) &= C_1 + S \\ \dot{Y}_1(0) &= C_2 - \frac{C_1}{2D_1} = \frac{Y_2(0)}{D_1} \end{aligned}$$

D'où l'on tire

$$\begin{aligned} C_1 &= Y_1(0) - S \\ C_2 &= \frac{2Y_2(0) + Y_1(0) - S}{2D_1} \end{aligned}$$

Soit au total

$$Y_1(t) = \left[Y_1(0) - S + \frac{2Y_2(0) + Y_1(0) - S}{2D_1} \cdot t \right] \cdot e^{-t/2D_1} + S$$

Au plan analytique on sait que les initialisations des niveaux et le seuil sont positifs. Par ailleurs on peut prendre comme unité de compte des quantités l'initialisation $Y_1(0)$. Il existe alors une valeur critique inhérente à la boucle étudiée, elle correspond au cas où $Y_2(0)$ est égal à la moitié de

l'écart, en valeur absolue, entre la valeur désirée de Y_1 et sa valeur initiale. Dans ce cas la constante C_2 est nulle. Cette valeur critique est donnée par

$$Y_2(0) = \frac{|S - Y_1(0)|}{2}$$

On pose alors

$$\begin{aligned} S &= p \cdot Y_1(0) & p \in R_+^* - \{1\} \\ Y_2(0) &= q \cdot \frac{|S - Y_1(0)|}{2} = q \cdot Y_1(0) \cdot \frac{|p - 1|}{2} & q \in R^+ - \{1\} \end{aligned}$$

L'étude analytique à laquelle on peut se livrer revient à étudier les déformations du comportement temporel de Y_1 compte tenu des valeurs prises par les différents paramètres et notamment q fournissant $Y_2(0)$ en fonction de $Y_1(0)$. On peut s'en tenir au cas où $Y_1(0) < S$ ($p > 1$). L'équation de comportement général de Y_1 s'écrit alors

$$\begin{aligned} Y_1(t) &= Y_1(0) [(1-p) \cdot [1 + (1-q)at] \cdot e^{-at} + p] & a = 1/2D_1 \\ &= f[Y_1(0), p, q, t, a] \end{aligned}$$

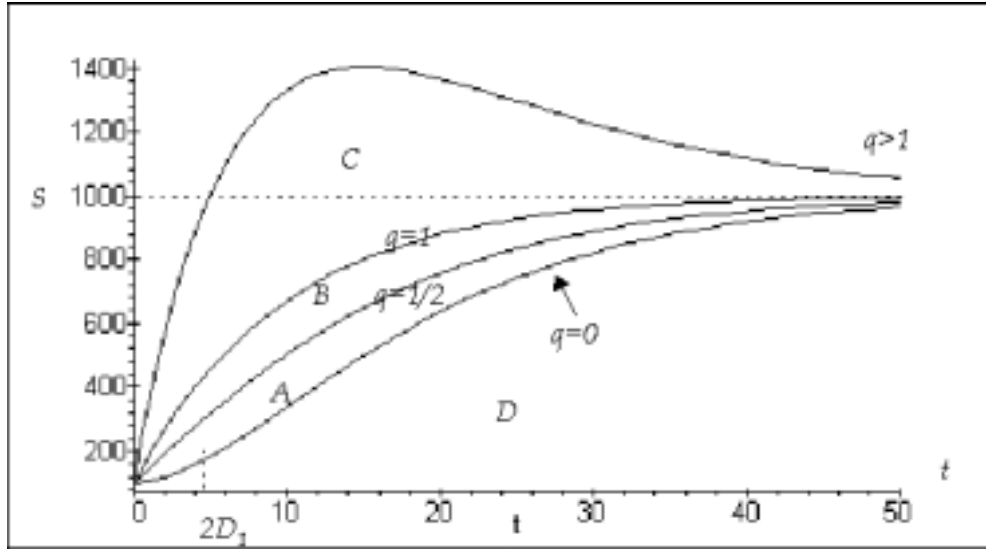
Selon les valeurs prises par q on a plusieurs types de comportement.

→ $0 < q < 1/2$: Existence d'un point d'inflexion apparaissant entre $t = 0$ et $t = 2D_1$. Courbe "pseudo logistique".

→ $1/2 < q < 1$: Existence d'un point d'inflexion "virtuel" (existant pour $t < 0$). Lorsque $q \rightarrow 1^-$ ce point est rejeté à $-\infty$. La courbe de comportement correspond alors à la partie à croissance moins que proportionnelle d'une pseudo logistique convergeant vers S par valeurs inférieures.

→ $q > 1$: Existence d'un maximum apparaissant à un instant d'autant plus proche (resp. éloigné) de $2D_1$ que q est élevé (resp. proche de 1). Lorsque $q \rightarrow 1^+$ ce maximum est rejeté à $+\infty$. Existence par ailleurs d'un point d'inflexion apparaissant à un instant d'autant plus proche (resp. éloigné) de $4D_1$ que q est élevé (resp. proche de 1). Quand $q \rightarrow 1^+$ ce point d'inflexion est rejeté à l'infini. Le comportement de Y_1 est alors donné par une oscillatoire "dégénérée" convergeant vers S par valeurs supérieures.

On a alors le graphe de comportement de principe donné ci-après décrivant tous les cas possibles à l'aide de "courbes frontières".



Tssd08 : Modes de comportement au cas d'un discriminant nul.

On notera l'existence de 4 zones dans le graphe précédent

→ A: Zone de comportement pseudo logistique.

→ B : Zone où le comportement s'apparente à celui d'une boucle de rétroaction du premier ordre à seuil positif,

→ C : Zone de comportement oscillatoire dégénéré.

→ D : Zone correspondant aux initialisations négatives de Y_2 , exclues par hypothèse.

Cas d'un discriminant positif.

On a alors deux racines réelles distinctes

$$r_1 = -\frac{1}{2D_1} + \frac{1}{2D_1} \sqrt{1 - \frac{4D_1}{D_2}}$$

$$r_2 = -\frac{1}{2D_1} - \frac{1}{2D_1} \sqrt{1 - \frac{4D_1}{D_2}}$$

L'équation de comportement de Y_1 est de la forme

$$Y_1(t) = A \cdot e^{r_1 \cdot t} + B \cdot e^{r_2 \cdot t} + S$$

Les comportements enregistrés sont de type pseudo logistique ou oscillatoire dégénéré.

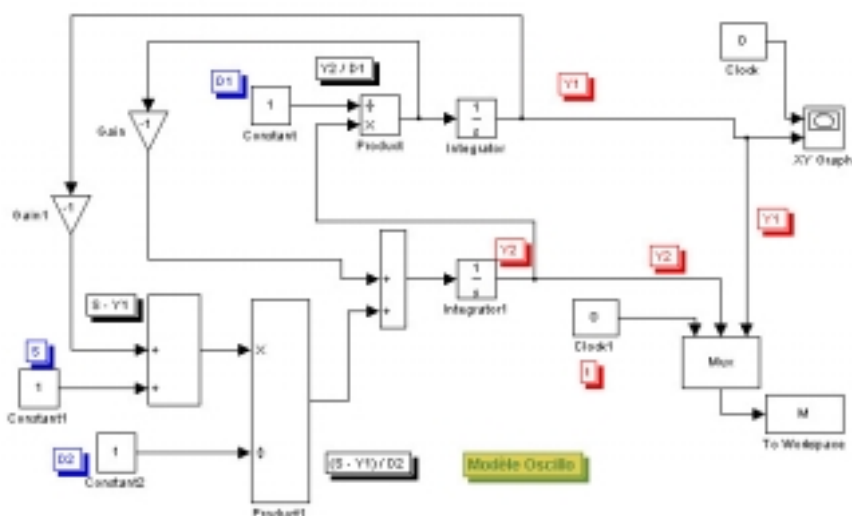
Pour conclure sur ce modèle on notera que fondamentalement on a

$$Y_1(t) = f[t, D_1, D_2, Y_1(0), Y_2(0), S]$$

Compte tenu du nombre de paramètres mis en jeu une étude analytique complète revient à envisager une trentaine de cas. On est alors loin de

2.2.7.2 Simulation.

Ici encore il faut élaborer le modèle sous SIMULINK en partant de zéro. Ce modèle est le suivant.



Smlk056.

Il s'agit du modèle "brut" dans lequel les paramètres n'ont pas été fixés. Supposons que l'on veuille recréer un comportement de Y_1 similaire à celui du graphe de principe *Tssd07 (oscillations amorties)*. Ce graphe a été créé en ajustant les divers paramètres de l'équation analytique sans tenir compte des liens éventuels qui les lient. On a l'équation de comportement suivante pour Y_1

$$Y_1(t) = C \cdot e^{-t/2D_1} \cdot \cos \left[\frac{t}{2D_1} \sqrt{\frac{4D_1}{D_2} - 1} - \varphi \right] + S$$

Supposons que l'on prenne

$$\begin{array}{rcl} D_1 & = & 2 \quad D_2 = 1 \\ Y_1(0) & = & 80 \quad Y_2(0) = 250 \quad S = 50 \end{array}$$

On impose un comportement bien précis puisque

$$C = \sqrt{A^2 + B^2} \quad \varphi = \arctan(B/A)$$

avec

$$\begin{aligned} A &= Y_1(0) - S \\ B &= \frac{2Y_2(0) + Y_1(0) - S}{\sqrt{(4D_1/D_2) - 1}} \end{aligned}$$

soit ici

$$\begin{aligned} A &= 80 - 50 = 30 \\ B &= \frac{2 \times 250 + 80 - 50}{\sqrt{(4 \times 2/1) - 1}} = \frac{530}{7} \sqrt{7} \end{aligned}$$

d'où

$$\begin{aligned} C &= \sqrt{30^2 + \left(\frac{530}{7} \sqrt{7}\right)^2} = \frac{20}{7} \sqrt{5026} = 202.55... \\ \varphi &= \arctan\left(\frac{\frac{530}{7} \sqrt{7}}{30}\right) = \arctan \frac{53}{21} \sqrt{7} = 1.422... \end{aligned}$$

Comme on a une expression de la forme

$$\cos(\omega t - \varphi) \Rightarrow \omega = \frac{1}{2D_1} \sqrt{\frac{4D_1}{D_2} - 1} = \frac{1}{4} \sqrt{7}$$

on peut dire que la période entre deux extrema est égale à

$$T = \frac{2\pi}{\omega} = \frac{2\pi}{\frac{1}{4}\sqrt{7}} = \frac{8}{7}\pi\sqrt{7} \approx 9.5$$

Introduisons les valeurs paramétriques sélectionnées dans notre modèle. On prend en outre comme paramètres de simulation

$$t_{start} = 0 \quad t_{stop} = 50 \quad ode45$$

et les paramètres de XY Graph sont

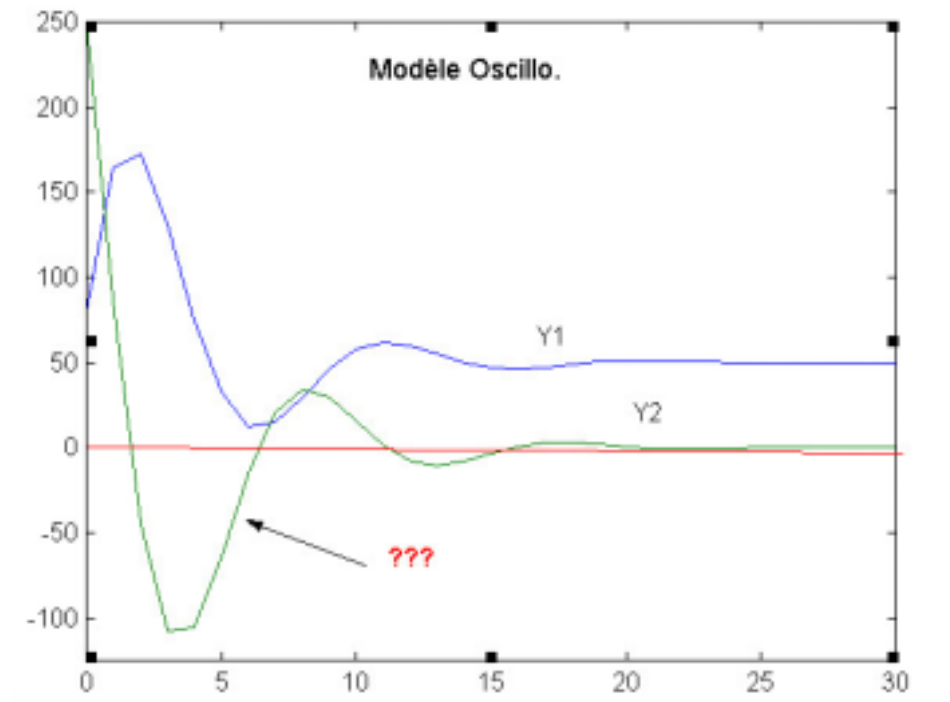
$$x \in [0, 50] \quad y \in [0, 200]$$

Sur le XY Plot on obtient un comportement de Y_1 qui est bien oscillatoire amorti, mais contrairement au graphe de principe on se rend compte que les oscillations s'amortissent rapidement.

On passe les expressions suivantes sous Matlab

```
» t=M(:,1);
» Y2=M(:,2);
» Y1=M(:,3);
» plot(t,Y1,t,Y2)
» set(gcf,'color','w')
» axis([0 30 -125 250])
```

La dernière expression a été écrite après un premier essai de tracé. On a le graphe suivant.



Smlk057.

Il est bien évident que, au vu de ce graphe, on est amené à se poser certaines questions. De fait que signifie un niveau Y_2 qui devient négatif ? Pour en revenir à l'interprétation du modèle que représente des marchandises en commandes négatives ? Si le modèle est mathématiquement valable il souffre d'une certaine carence pour ce qui est d'une représentation adéquate de la réalité physique. Il nous faut donc comprendre ce qui se passe et pallier les défauts existants.

Un problème majeur est que l'on a une politique de commande de marchandises fondée sur un écart sur stock

$$X_2 = \frac{S - Y_1}{D_2}$$

Si le stock existant (Y_1) dépasse le seuil désiré (S) cette quantité est négative, on a des **commandes négatives** ! On pourrait l'interpréter comme un retour des marchandises aux fournisseurs. Mais il est irréaliste de penser que le supermarché *Carchamp* peut renvoyer ses palettes d'eau gazeuse *Badrier* s'il en a trop !

Dès lors que faire ? On peut introduire une condition du type

Si $X_2 < 0$
alors
ne rien faire
sinon
commander
Fin Si

Si cela est possible *soulignons bien le fait que nous quittons le domaine des mathématiques pures pour rentrer dans celui de la vraie simulation*. On n'a jamais su résoudre des équations différentielles soumises à des conditions de ce type !

Il est possible d'introduire une condition en utilisant le bloc Relational Operator [∈ librairie Math]. Ce bloc accomplit une opération relationnelle sur ses deux intrants et produit un extrant du type *TRUE* ou *FALSE* égal en fait à 1 ou 0. Si, par exemple, on utilise l'opérateur

$$> =$$

si on a les 2 intrants I_1 et I_2 on a le tableau

	Etat	Output
$I_1 \geq I_2$	True	1
$I_1 < I_2$	False	0

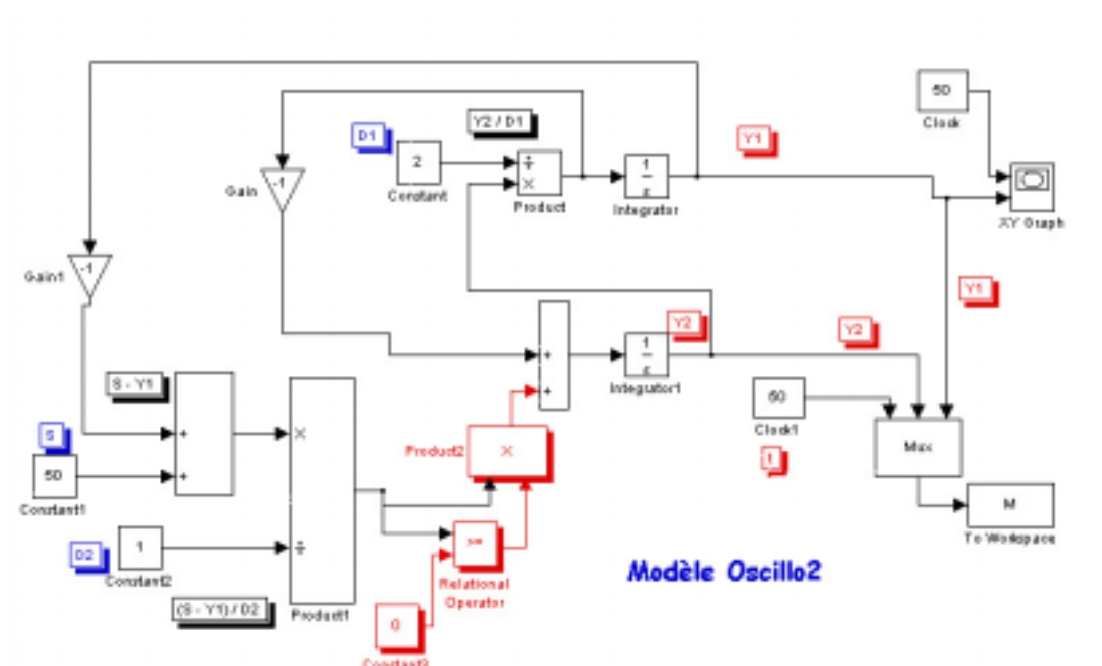
Les opérateurs possibles sont

$=$	égal à
$\neq$	non égal à
$<$	inférieur à
$\leq$	inférieur ou égal à
$\geq$	supérieur ou égal à
$>$	supérieur à

En ce qui nous concerne ici le 0 peut être assimilé à " *ne rien faire* " et le 1 à " *commander* ". Si on compare la quantité X_2 à la valeur 0 on a l'algorithme

Si $X_2 \geq 0$
alors
$1 \Rightarrow$ je commande
sinon
$0 \Rightarrow$ je ne fais rien
Fin si

D'où le modèle *Oscillo2* qui est une version de *Oscillo* intégrant le bloc Relational Operator et les modifications consécutives.



Smlk058.

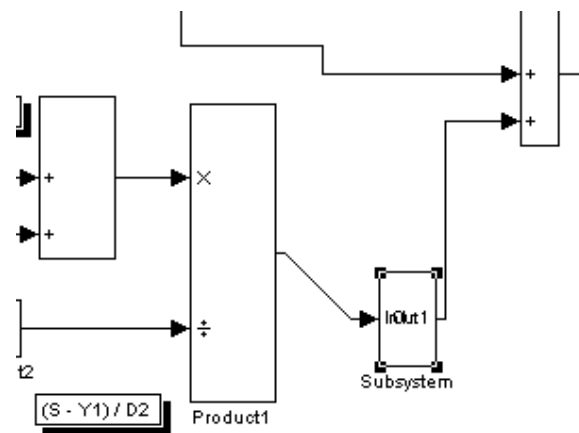
Les blocs ajoutés figurent en bas, en rouge et ombrés. Pour le **rouge** faire **Clic droit + Foreground Color + Red**, pour l'**ombre** faire **Clic droit + Format + Show Drop Shadow**. La logique de l'assemblage est la suivante.

Avec le bloc **Relational Operator** on compare $(S - Y_1) / D_2$ à 0 avec l'opérateur relationnel $\geq$, puis on multiplie l'extrant de ce bloc avec une branche de $(S - Y_1) / D_2$ (dans le bloc **Product2**). Si la quantité est positive ou nul l'extrant du bloc **Relational Operator** est égal à 1 donc $(S - Y_1) / D_2$ sera bien compris dans le flux intégré pour donner Y_2 . L'opposé se produit dans le cas contraire.

Notons que le schéma tend à se compliquer, aussi on pourrait transformer les blocs ajoutés en sous-système. Pour ce faire

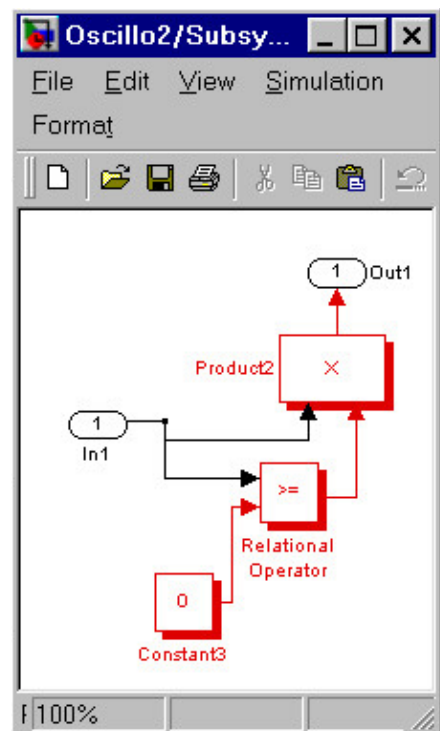
1 $\rightarrow$ On sélectionne les blocs et les lignes de connexion désirées. (Clic gauche + Etirer + Relâcher).

2 $\rightarrow$ Dans le menu **Edit** on choisit **Create Subsystem**. Simulink remplace les blocs sélectionnés par un bloc **Sous-Système**.



Smlk058b.

Si on fait un double clic gauche sur le sous-système Simulink fait apparaître une fenêtre montrant le contenu du sous-système.



Smlk058c.

On reprend les mêmes conditions paramétriques que précédemment et on réalise une simulation. On a une surprise à la sortie du graphe de Y_1 par le bloc XY Graph. On a un bout de courbe ascendante en haut à gauche du graphe $\Rightarrow$ modifier abscisses et ordonnées ainsi que les paramètres de

simulation. On procède par tâtonnement n'ayant plus de solution analytique pour nous guider ! On prend comme paramètres de simulation

$$t_{start} = 0 \quad t_{stop} = 100 \quad ode45$$

et les paramètres de XY Graph sont

$$x \in [0, 100] \quad y \in [0, 400]$$

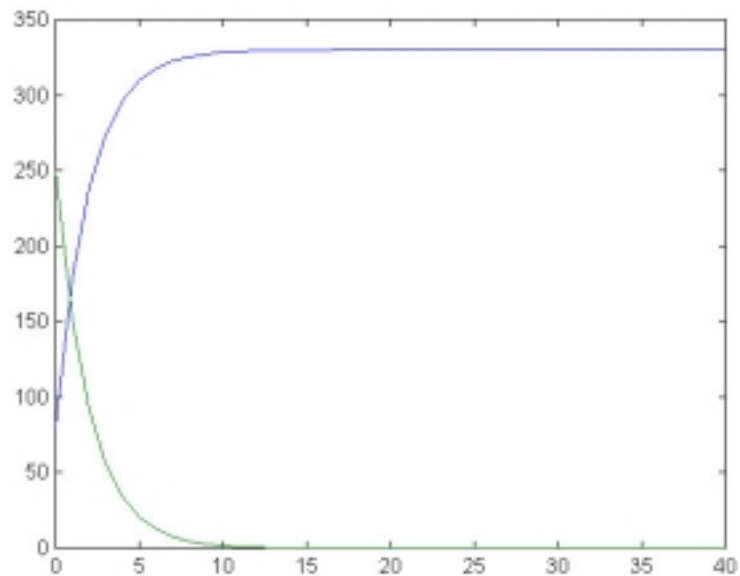
On obtient une courbe croissant vers un seuil. Soit avec les ordres usuels sous Matlab

```

» t=M(:,1);
» Y1=M(:,3);
» Y2=M(:,2);
» plot(t,Y1,t,Y2)
» set(gcf,'color','w')
» axis([0 40 0 350])

```

le graphe suivant



Smlk059.

Avant de nous affoler, ou de jouer les "Mozart du clavier" prenons le temps de réfléchir. Comment peut-on expliquer les évolutions enregistrées. Désormais il n'y a plus de fausses commandes négatives. A l'origine on a

$$S - Y_1 = 50 - 80 = -30 < 0$$

On a plus de marchandises que le seuil désiré, on ne commande pas. Pendant ce temps Y_2 se vide dans Y_1 . Ce dernier ne peut qu'augmenter sans se vider. La valeur final de Y_1 doit être ainsi égale à

$$Y_1|_{stop} = Y_1(0) + Y_2(0) = 330$$

on le vérifie en faisant `max(Y1)` sous Matlab la réponse étant 299.999...

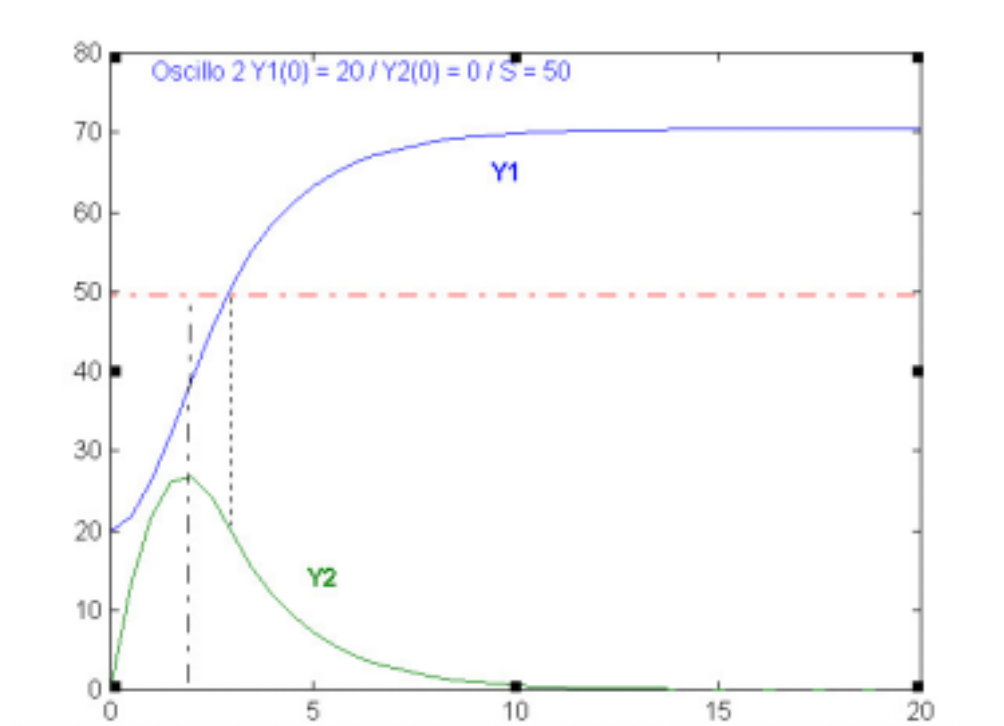
Modifions les paramètres en prenant

$$\begin{aligned} D_1 &= 2 & D_2 &= 1 \\ Y_1(0) &= 20 & Y_2(0) &= 0 & S &= 50 \end{aligned}$$

On fait sortir graphe comparatif sous Matlab après passage des expressions idoines, ce graphe n'étant pas assez lissé on modifie les paramètres de simulation et d'envoi au workspace.

$$\begin{aligned} t_{start} &= 0 & t_{stop} &= 20 & ode45 \\ Decimation &= 1 & SampleTime &= 0.5 \end{aligned}$$

On a le graphe suivant



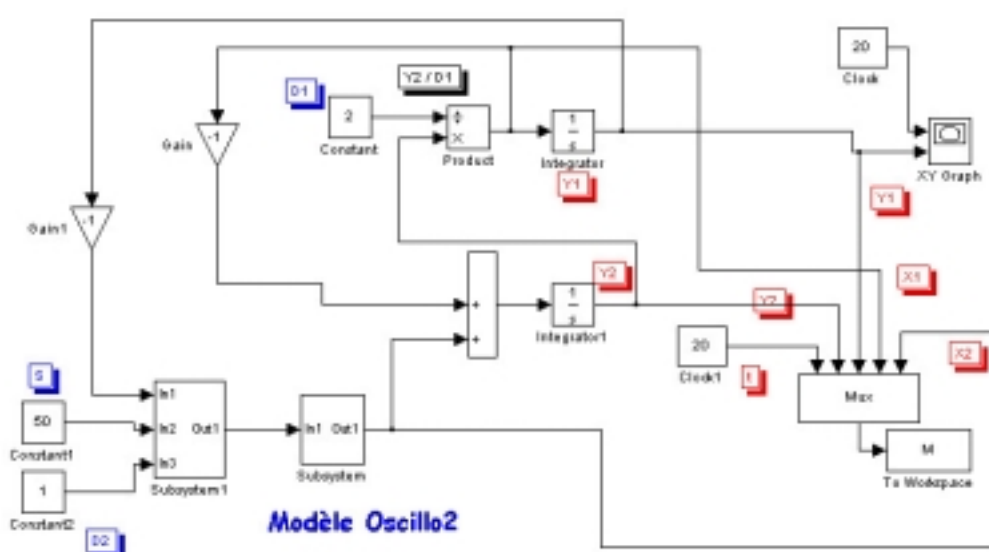
Smlk060.

Pour analyser cette simulation on va transformer quelque peu notre modèle de façon à connaître les valeurs des flux

$$X_1 = \frac{Y_2}{D_1}$$

$$X_2 = \frac{S - Y_1}{D_2}$$

On crée un nouveau sous-système et on envoie les flux X_1 et X_2 dans le bloc To Workspace.



Smlk061.

Il ne faut pas perdre de vue le fait que Y_1 se remplit par X_1 alors que Y_2 se remplit par X_2 et se vide par X_1 . Observons les premières valeurs numériques (tronquées à la première décimale)

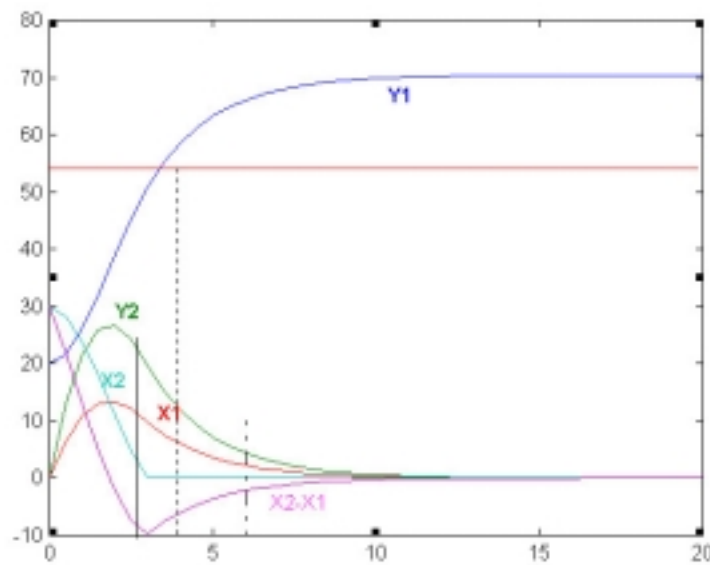
t	0	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0
Y_1	20.0	21.7	26.1	32.1	38.8	45.2	50.7	55.1	58.5
Y_2	0	12.9	21.6	26.0	26.6	24.1	19.6	15.3	11.9
$X_1 = Y_2/2$	0	6.4	10.8	13.0	13.3	12.0	9.8	7.6	5.9
$X_2 = (50 - Y_1)$	30.0	28.2	23.8	17.8	11.1	4.7	0	0	0
$X_2 - X_1$	30.0	21.8	13.0	4.8	-2.2	-7.3	-9.8	-7.6	-5.9

Tant que Y_1 est inférieur à $S = 50$ ($t \neq 3$) le flux de commande X_2 est positif. Il va en s'amenuisant lorsque $Y_1 \rightarrow S$. Ce flux remplit Y_2 . Dans le même temps Y_2 se vide par X_1 égal à la moitié de ce niveau. Compte tenu

du jeu combiné de ces deux flux Y_2 atteint un maximum en $t = 2$. Au delà Y_2 se vide à taux moins que proportionnel. A partir du moment où Y_1 dépasse le seuil S , il n'y a plus de flux de commande ($X_2 = 0$) mais Y_1 continue à se remplir, à taux moins que proportionnel par simple vidage de Y_2 . Dès que Y_2 est vide (il n'y a plus de commandes en cours), il n'y a plus aucun flux en action les niveaux restent stables.

$$Y_1 = 70.43 \quad Y_2 = 0$$

On peut résumer ce que nous venons de dire à l'aide du graphe suivant.



Smlk062.

Quelques remarques.

Le fait que l'on élimine les commandes négatives, fait que le comportement oscillant disparaît. Nous voulant plus réaliste nous aboutissons à un modèle qui ne fait plus ce qu'il était censé faire.

La notion de seuil désiré tend à être gommée puisque de fait Y_1 se stabilise à une valeur nettement supérieure. Ce n'est pas trop grave puisque en complexifiant le modèle on peut essayer de retrouver cette notion de valeur désirée. De fait, si on décide de réguler un niveau en pratiquant une politique d'écart sur stock, on sait (en théorie de la commande) que cette politique échoue à partir du moment où le niveau est perturbé par adjonction d'un flux d'entrée supplémentaire ou d'un flux de sortie. Par exemple dans notre cas on a omis un éventuel flux de marchandises vendues qui viendrait agir négativement sur Y_1 .

En ce qui nous concerne ici le problème n'est pas de savoir comment améliorer le modèle qui n'a qu'un caractère d'illustration. On a un problème

plus immédiat. On constate que Y_1 se stabilise à une certaine valeur, sans aucun doute fonction des valeurs initiales des niveaux et des délais de temps mis en jeu. Mais on est incapable de prévoir cette valeur de façon *a priori*, ne pouvant plus avoir recours à une solution analytique.

Nous allons voir maintenant comment pratiquer pour essayer d'entrevoir, via des simulations réitérées, vers quelles valeurs tend Y_1 .

2.3 Pratiques de la simulation.

Le problème posé est de savoir comment on peut réaliser un certain nombre de simulations afin de savoir comment évoluent les niveaux Y_1 et Y_2 en fonction de certains paramètres. Le but est d'automatiser aux maximum les choses en se servant de la puissance de calcul de Matlab.

Le modèle initial peut être défini à l'aide d'une M-file que l'on nommera *Osc*. C'est la suivante.

```
function dy = Osc(t,y)
sd=50; del1=2; del2=1;
dy = zeros(2,1); % on définit un vecteur colonne
flux1=y(2)/del1;
flux2=(sd-y(1))/del2;
dy(1) = flux1;
dy(2) = flux2-flux1;
```

Remarques.

Le symbole % utilisé dans une M-file sert à mettre des commentaires, tout ce qui suit le symbole n'est pas exécutable.

$dy = \text{zeros}(2,1)$ veut dire que initialement, avant toute affectation on a

$$dy = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Cette M-file étant sauvée on écrit sous Matlab

$$[t,y] = \text{ode45}('Osc',[0 \ 20],[10 \ 0]);$$

c'est à dire qu'on réalise une intégration sur l'intervalle de temps $[0, 20]$ avec les initialisations

$$Y_1(0) = 10 \quad Y_2(0) = 0$$

Attention ! le vecteur extrant y est formé de deux colonnes correspondant l'une à Y_1 et l'autre à Y_2 . D'ailleurs si l'on écrit

```
size(y)
```

l'application de la fonction `size` à y nous renvoie les dimensions

```
101 2
```

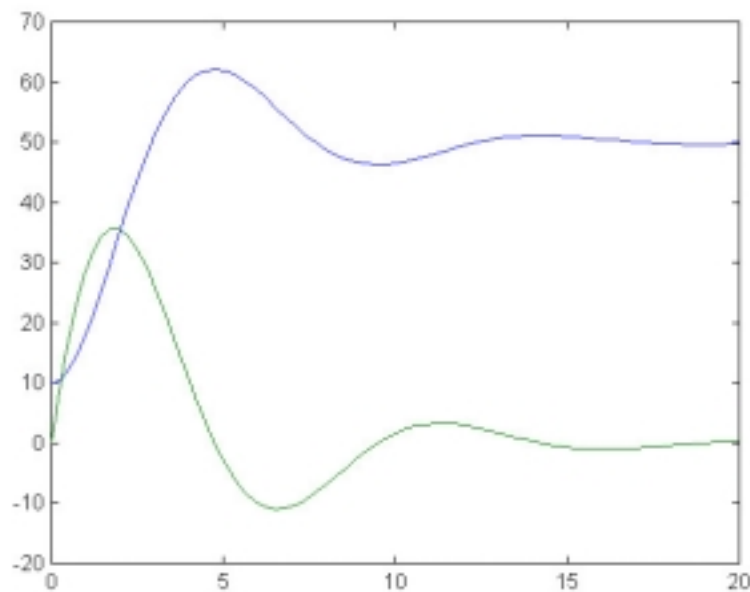
Si l'on veut visualiser les valeurs de chaque composante on tape

```
y(:,1) ou y(:,2)
```

L'expression

```
plot(t,y)
set(gcf,'color','w')
```

permet un tracé simultané des valeurs de Y_1 et Y_2 en fonction du temps.



Smlk063.

Comme on l'a vu on peut améliorer le modèle en évitant un faux flux de commande. On a la M-file *Oscb*

```
function dy = Oscb(t,y)
sd=50; del1=2; del2=1;
dy = zeros(2,1); % on définit un vecteur colonne
flux1=y(2)/del1;
```

```

flux2=(sd-y(1))/del2;
if (sd-y(1)) <= 0
    flux2 = 0;
end
dy(1) = flux1;
dy(2) = flux2-flux1;

```

Elle se distingue de la précédente par la boucle

```

if (sd-y(1)) <= 0
    flux2 = 0;
end

```

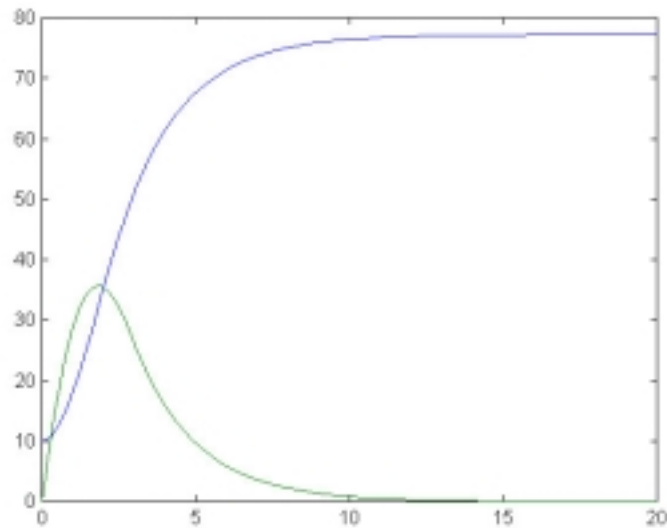
permettant de supprimer le flux de commande lorsque le niveau Y_1 est supérieur au seuil S . Cela étant on passe les expressions

```

[t,y] = ode45('Oscb',[0 20],[10 0]);
plot(t,y)

```

on obtient le graphe



Smlk064.

Notons au passage que l'on a

```

>> size(y)
ans =
89 2

```

La modification du modèle engendre une modification dans le nombre de pas effectués par le solveur!

Jusque là rien de bien révolutionnaire. Nous n'avons fait que refaire, peut être un peu plus rapidement ce que l'on a fait avec Simulink. On peut alors se demander si on ne pourrait pas automatiser les choses en créant par exemple une boucle de calcul du type

```
Pour SD allant de à
  Pour D1 allant de à
    Pour D2 allant de à
      exécuter Oscb
    Fin pour
  Fin pour
Fin pour
```

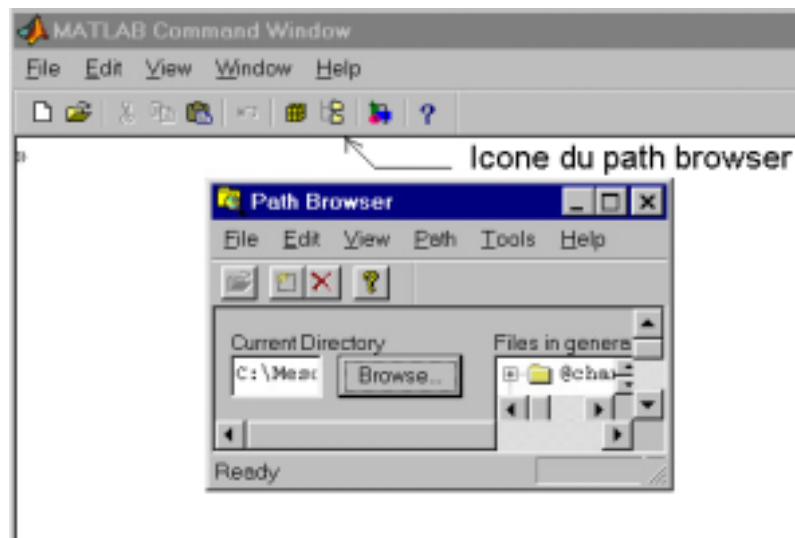
où Oscb prendrait la forme

```
function dy = Oscb(t,y)
dy = zeros(2,1);
flux1=y(2)/D1;
flux2=(S-y(1))/D2;
if (S-y(1))<=0
  flux2=0;
end
dy(1) = flux1;
dy(2) = flux2-flux1;
```

Hélas cela n'est pas réalisable sous Matlab on ne peut "paramétrer" une fonction définie par "*function*" ! Il faut modifier les paramètres manuellement et faire un stockage idoine des résultats.

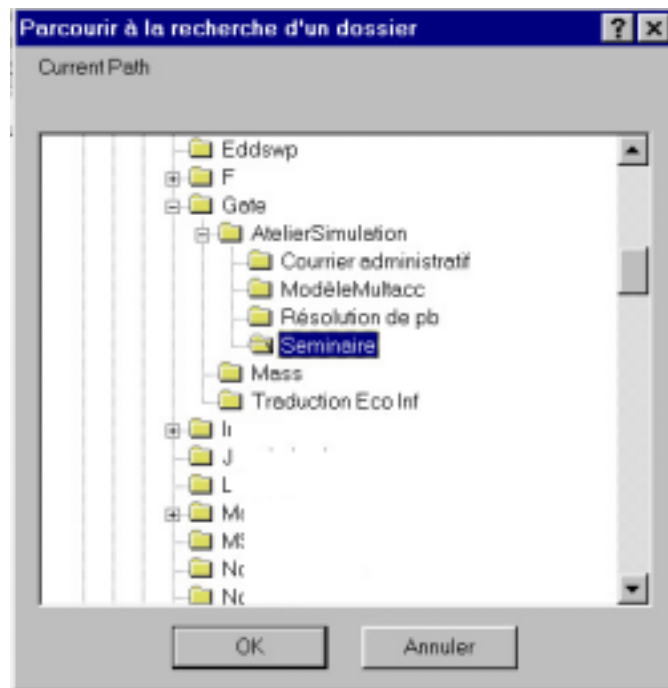
Cela dit une autre possibilité nous est offerte ici. Pour des valeurs données des paramètres on peut jouer sur les initialisations. On va voir comment faire et introduire par la même de nouvelles notions.

Avant de continuer il nous faut préciser un **élément fondamental** qui ne l'a pas été jusqu'ici. Au début de toute session de travail il est nécessaire de se placer dans un fichier où se trouvent les éléments sur lesquels on veut travailler (M-file ou modèles Simulink). On le fait à l'aide du Path Browser de la fenêtre de commande de Matlab. On clique sur l'icône correspondante pour faire apparaître la fenêtre "Path Browser".



Smlk065.

Dans cette fenêtre on clique sur Browse pour aller choisir un dossier dans l'arborescence



Smlk066.

Le dossier sélectionné, on clique sur OK, on revient à la fenêtre Path Browser que l'on ferme.

Cette précision a son importance car l'on va créer une M-file faisant appel à une autre M-file. Les deux fichiers doivent se trouver dans le même

dossier sélectionné si l'on veut que Matlab s'y retrouve. Notre M-file s'appelle *Boucle* et va nous permettre un certain nombre de développements.

La M-file *Boucle* est la suivante

```
%Boucle
M=zeros(100,10,10);
N=zeros(100,10,10);
T=zeros(100,10,10);
for i=1:10
    for j=1:10
        [t,y] = ode45('Oscb',[0 20],[i j]);
        q=size(y,1);
        M(1:q,i,j)=y(:,1);
        N(1:q,i,j)=y(:,2);
        T(1:q,i,j)=t;
    end
end
```

Le programme est court, les commentaires conséquents. Notons tout d'abord que une M-file peut être un "*script*" ou une "*function*". Ici nous avons affaire à **un script qui est un fichier contenant une suite d'énoncés de Matlab.**

Notion de boucle.

Nous voulons voir comment évoluent les variables d'état en fonction des initialisations. Ici pour simplifier les choses on fait varier $Y_1(0)$ et $Y_2(0)$ de 1 à 10. On considère les indices de boucle i (pour Y_1) et j (pour Y_2).

Une boucle se décrit à l'aide de la fonction `for` de syntaxe générale

```
for variable = expression
    énoncé
    ...
    énoncé
end
```

on note que la boucle se termine toujours par `end`. Lorsqu'on écrit la M-file sous le Matlab Editor/Debugger les symboles clés comme `for` et `end` sont mis en bleu, l'indentation est réalisé automatiquement. Bien évidemment on peut avoir des boucles imbriquées comme c'est le cas ici.

L'énoncé

```
for i=1:10
```

veut dire que i va aller de 1 à 10, implicitement par pas de 1. Mais on peut préciser un incrément, par exemple l'énoncé

```
for i=1:0.5:10
```

voudrait dire que i va aller de 1 à 10, par pas de 0.5.

Cela étant on a au coeur de la boucle l'énoncé

```
[t,y] = ode45('Oscb',[0 20],[i j]);
```

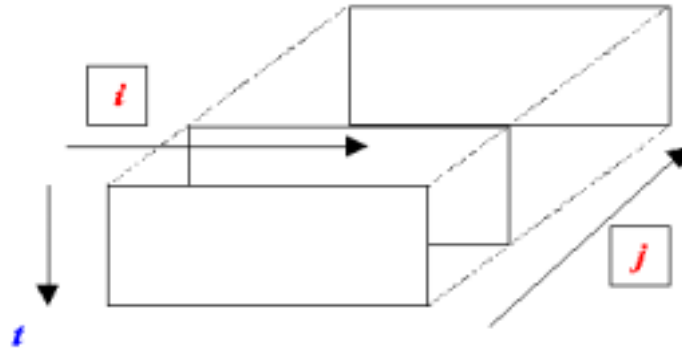
cela veut dire que l'on réalise l'intégration de la fonction (M-file) *Oscb* sur l'intervalle $[0, 20]$ pour des valeurs initiales i et j . L'intégration nous retourne 3 vecteurs

$$t \quad Y_1 \equiv y(1) \quad Y_2 \equiv y(2)$$

lesquels peuvent en outre être avec des lignes variables (selon le bon vouloir du solveur). Il faut stocker ces résultats quelque part si on ne veut pas les perdre.

Notion de tableau multi-dimensionnel.

Outre les matrices et les vecteurs Matlab peut manipuler des tableaux multi-dimensionnels (*Multidimensional Arrays*), on peut dire que ce sont des matrices à plusieurs dimensions. Ici nous allons utiliser des tableaux 3-D. Obéissant au schéma de principe suivant



Smlk067.

La première dimension (t) représente en fait les lignes d'une matrice située sur une feuille. La deuxième dimension (i) représente son nombre de colonnes. La troisième dimension (j) représente le nombre de feuilles successives.

On note M le tableau servant à récupérer les valeurs de Y_1 et N celui servant pour Y_2 . On peut rechercher différents éléments dans de tels tableaux.

$$\begin{aligned}
 M(1,3,4) &= \text{élément} \in \begin{cases} 1^{\text{ère}} \text{ ligne} \\ 3^{\text{ème}} \text{ colonne} \\ 4^{\text{ème}} \text{ feuille} \end{cases} \\
 M(:,3,4) &= \text{toutes les lignes de} \begin{cases} 3^{\text{ème}} \text{ colonne} \\ 4^{\text{ème}} \text{ feuille} \end{cases} \\
 M(2, :, 4) &= \text{toutes les colonnes de} \begin{cases} 2^{\text{ème}} \text{ ligne} \\ 4^{\text{ème}} \text{ feuille} \end{cases} \\
 M(:, :, n) &= \text{matrice de la } n^{\text{ème}} \text{ feuille}
 \end{aligned}$$

Cela étant on a un petit problème. Le solveur ode45 renvoie des vecteurs pouvant avoir un nombre de lignes différent et Matlab a horreur des incohérences. C'est pourquoi on commence par définir les tableaux M et N bourrés de 0, grâce à la fonction "zeros" en surdimensionnant leur nombre de lignes (ici 100).

```
M = zeros(100, 10, 10);
```

On pourrait dire que la première composant de y retournée à chaque étape va aller se placer dans la colonne correspondante de M en écrivant

```
M(:, i, j) = y(:, 1);
```

Si l'on pratiquait ainsi et si l'on exécutait le programme Matlab nous répondrait gentiment

```
» ??? Error using ==> run
Subscripted assignment dimension mismatch
```

En bon français : " *vous vous être trompé avec les indices, j'attends un vecteur de 100 lignes et j'en reçois un ayant un nombre différent de lignes* " !!!

C'est pourquoi il faut connaître à tout moment la dimension ligne du vecteur y retourné. Cela se fait par

```
q = size(y, 1);
```

En fait on demande la longueur du vecteur (t). Cette dimension connue on "injecte" dans M ou N des données sur un nombre q de lignes d'où les ordres du type

```
M(1 : q, i, j) = y(:, 1);
```

Et la Matlab ne proteste plus, il sait qu'il doit remplir la colonne i de la feuille j (du tableau M) de la ligne 1 à la ligne q , les lignes au delà restent à 0.

En outre puisque le vecteur est de longueur variable, bien, que commençant à 0 et finissant à 20 il faut aussi stocker tous les vecteurs t retournés par le programme. D'où la création du tableau

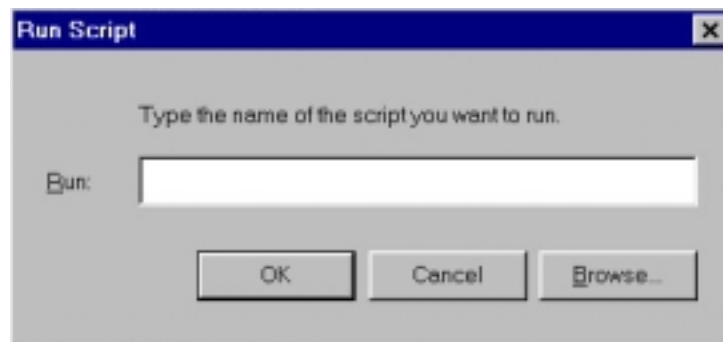
```
T = zeros(100, 10, 10);
```

et le stockage à chaque étape des valeurs

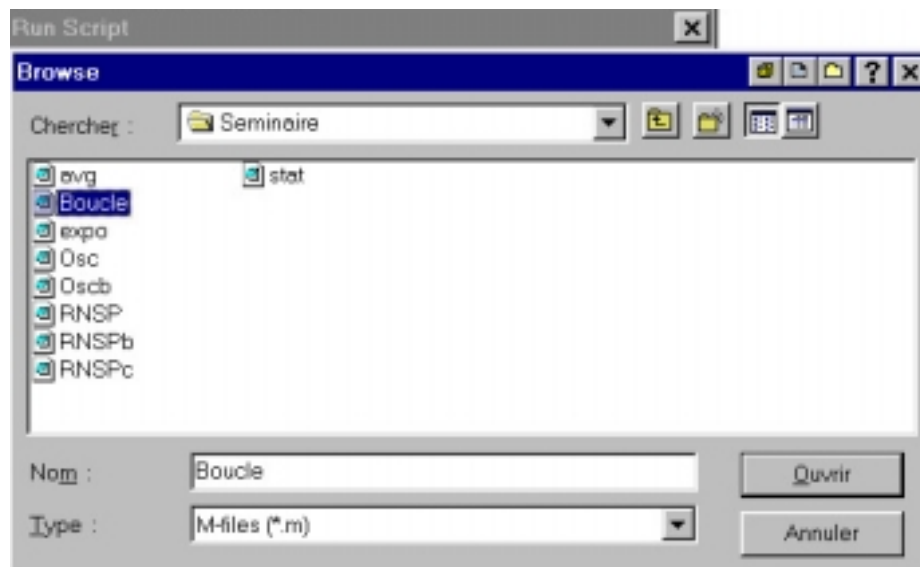
```
T(1 : q, i, j) = t;
```

Voilà tout est dit, et ça marche !

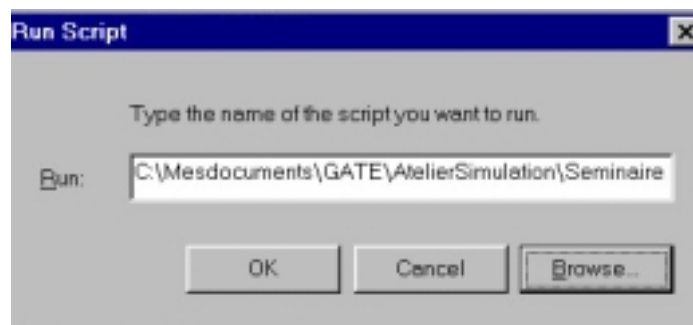
Pour exécuter le programme Boucle. Dans la fenêtre de commande de Matlab on fait File + Run Script. On fait apparaître la fenêtre correspondante

*Smlk068.*

On clique sur Browse pour aller choisir notre programme (*Script*) dans le dossier courant

*Smlk069.*

On fait un double clic gauche sur le script désiré (ici boucle) et on a la fenêtre Run Script.

*Smlk070.*

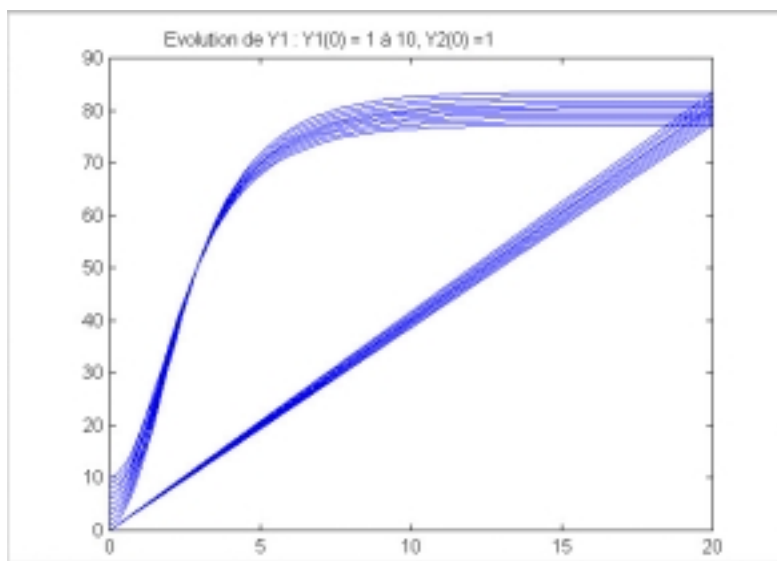
On clique sur OK le programme s'exécute. La fenêtre Run script reste à l'écran le temps de l'exécution qui ici est "relativement" longue (12 secondes chez moi).

On veut maintenant faire un graphe comparatif de l'évolution de Y_1 lorsque $Y_1(0)$ varie de 1 à 10, $Y_2(0)$ valant 1 (i.e. on est dans la première feuille du tableau M). On peut réaliser une petite boucle qui est la suivante

```
» for i=1:10
hold on
plot(T(:,i,1),M(:,i,1))
end
```

Attention. Le `hold on` à l'intérieur de la boucle est impératif si l'on ne veut pas se retrouver avec un seul graphe. Il permet de conserver chaque tracé courant et les propriétés des axes de sorte que les graphes subséquents s'ajoutent au(x) graphe(s) existant(s).

On obtient la figure suivante



Smlk071.

Cherchez l'erreur ! → les vecteurs T (comme les vecteurs composants de M) se terminent pas des zéros si bien que Matlab trace une droite partant du dernier point d'ordonnée non nulle pour aller vers l'origine.

Pour améliorer la présentation il suffit de rectifier le tableau T . Nous ne le ferons que pour la feuille 1. On peut créer une M-file, car taper une boucle dans la fenêtre de commande de Matlab finit par être lassant. La M-file se note *modifT*. C'est la suivante

```

% Modification de T
for i=2:101 % Attention on va ajouter une ligne
    for j=1:10
        if T(i-1,j,1)>=20
            T(i,j,1)=20; % Ne pas oublier le ;
        end
    end
end
for i=1:100
    for j=1:10
        T1(i,j,1)=T(i,j,1); % Ne pas oublier le ;
    end
end
end

```

Commentaires.

On notera qu'il y en a déjà dans le corps du programme.

La première boucle sert à mettre à 20 les "fins de vecteurs" qui sont actuellement nuls. L'indice de ligne doit aller de 2 à 101 parce qu'on va jouer sur la valeur d'une ligne antérieure et Matlab a horreur des indices de tableau (vecteurs ou matrices) nuls. Ce faisant on a dans T des vecteurs colonnes de format (101×1) alors que ceux de M ou N sont de format (100×1) . Attention à la différence de dimension.

Pour avoir des dimensions identiques on recopie $T(i,j,1)$ dans un tableau $T1(i,j,1)$ c'est l'objet de la seconde boucle.

A noter la remarque "ne pas oublier le point virgule" après l'expression d'affectation. Si on ne le fait pas on va afficher $100 \times 10 = 1000$ fois le tableau. Bonjour les dégâts à l'écran !!!

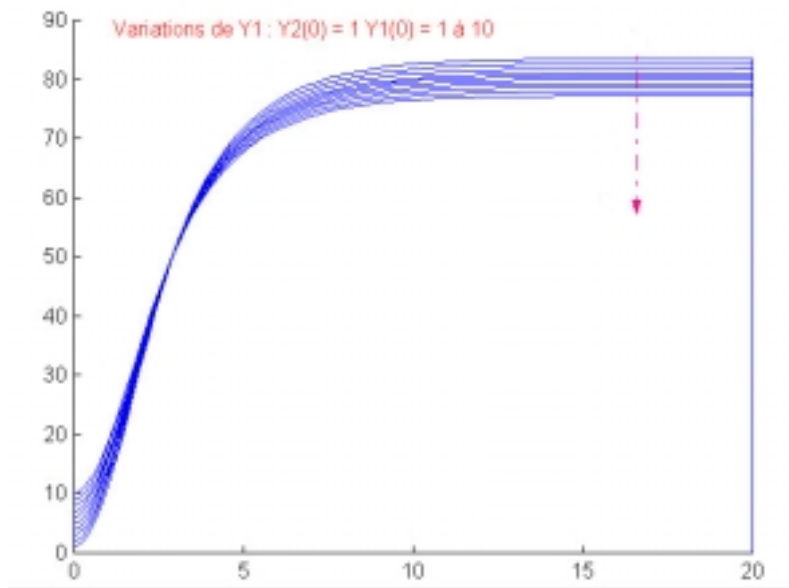
On peut maintenant refaire un tracé graphique. On crée une M-File pour ce faire, c'est la M-file *GraphM1*.

```

%Graph M(:, :, 1)
for i=1:10
    hold on
    plot(T1(:,i,1),M(:,i,1))
end
set(gcf,'color','w')

```

On obtient le graphe suivant (amélioré)



Smlk072.

Ce type de graphe permet de voir comment évolue une variable en fonction d'un paramètre.

On peut se demander maintenant comment on peut étudier la variation des valeurs finales de Y_1 en fonction de $Y_1(0)$ et $Y_2(0)$? Compte tenu de la forme en S des courbes temporelles il nous suffit de rechercher les maxima. Pour ce faire il existe la fonction `max`.

La fonction `max`.

Si $\mathbf{u}$ est un vecteur `max(u)` retourne la plus grande valeur de u .

Si $\mathbf{M}$ est une matrice `max(M)` traite les **colonnes** de $\mathbf{M}$ comme des vecteurs, et retourne un vecteur **ligne** contenant les éléments maximaux de chaque colonne.

Si $\mathbf{T}$ est un tableau multi-dimensionnel `max(T)` traite les valeurs le long de la première dimension comme des vecteurs, renvoyant la valeur maximale de chaque vecteur. Pour un tableau 3-D on travaille en fonction de la dimension ligne et ce feuille à feuille.

Si on utilise un programme (M-file) de la forme

```
v=zeros(10,10);
for i=1:10
v(i,:)=max(:,i);
end
```

Matlab nous répond gentiment

```
??? Error using ==> run
Error using ==> max
MAX with two matrices to compare and
a working dimension is not supported.
```

D'où l'idée du programme suivant dénommé *MaxM*

```
%Recherche de Max(M)
max(M)
u=zeros(10,10);
for i=1:10
    u(i,:)=ans(:,i);
end
```

En écrivant `max(M)` sans point virgule on obtient l'affichage des valeurs maxima de chaque feuille de M sous la forme

```
ans(:,i) =
avec  $i \in [1, 10]$ 
```

Chaque vecteur réponse " ans " apparaît sous la forme d'un vecteur ligne (1×10) son format Matlab étant en fait ($1 \times 10 \times 10$) comme le montre le *Workspace Browser*. D'où le stockage ligne à ligne de ces vecteur dans la matrice **u**.

Ce n'est peut être pas joli (on doit faire afficher les réponses à `max(M)`) mais ça marche.

Notons que dans **u** les colonnes représentent la variation en fonction de $Y_1(0)$ les lignes la variation en fonction de $Y_2(0)$.

Pourquoi ? Dans $M(t,i,j)$ ou $N(t,i,j)$ la dimension colonne (indice i) représente les variations de $Y_1(0)$ la dimension feuille (indice j) représente les variations de $Y_2(0)$. Quand on récupère un vecteur ligne par `Max` chaque colonne représente bien une variation en fonction de $Y_1(0)$. L'empilement de ces vecteur lignes dans **u**, fait que chaque ligne de **u** représente une variation en fonction de $Y_2(0)$.

Cela dit la matrice **u** est donnée ci-après.

83.48	82.64	81.77	81.03	80.53	79.96	79.35	78.69	77.99	77.25
83.48	82.70	82.08	81.37	80.71	79.91	79.42	78.79	78.13	77.44
83.44	82.65	82.19	81.57	80.89	80.17	79.41	78.62	77.80	76.96
83.65	82.90	82.09	81.25	80.37	79.82	79.29	78.70	78.06	77.39
83.12	82.66	82.11	81.50	80.83	80.12	79.38	78.67	78.08	77.38
83.58	82.86	82.13	81.37	80.82	80.14	79.47	78.76	78.06	77.54
83.62	82.82	82.23	81.56	80.83	80.28	79.67	79.01	78.33	77.61
83.57	82.97	82.30	81.47	81.05	80.43	79.76	79.05	78.32	77.55
83.71	83.04	82.28	81.82	81.19	80.50	79.78	79.01	78.22	77.40
83.78	83.15	82.59	81.95	81.24	80.49	79.70	78.88	78.04	77.52

Au vu de ce tableau un petit problème se fait jour. la tendance en fonction de $Y_1(0)$ est nette il y a une baisse. Pour $Y_2(0)$ c'est plus flou on enregistre des sortes d'oscillations. Peut être que cela est du à un horizon d'intégration insuffisant. Pour s'en assurer on recommence tout en prenant dans la M-file *boucle*

```
[t,y] = ode45('Oscb',[0 30],[i j]);
```

On fait un premier essai pour voir si la dimension ligne de 100 est suffisante. On fait afficher par exemple

```
T(:, :, 1) T(:, :, 5) T(:, :, 10)
```

Il semble que cela passe mais dans certains cas (selon les pas du solveur) les 100 lignes sont presque remplies.

Les nouvelles valeurs d'état stationnaire sont les suivantes. Nous avons fait figurer en colonne des flèches $\nearrow$ ou $\searrow$ pour bien voir les augmentations ou les diminution de valeur ligne à ligne.

	1	2	3	4	5
1	83.489	82.652	81.776	81.034	80.532
2	83.491 $\nearrow$	82.707 $\nearrow$	82.085 $\nearrow$	81.378 $\nearrow$	80.714 $\nearrow$
3	83.447 $\searrow$	82.656 $\searrow$	82.204 $\nearrow$	81.579 $\nearrow$	80.898 $\nearrow$
4	83.659 $\nearrow$	82.906 $\nearrow$	82.100 $\searrow$	81.254 $\searrow$	80.375 $\searrow$
5	83.133 $\searrow$	82.664 $\searrow$	82.118 $\nearrow$	81.508 $\nearrow$	80.842 $\nearrow$
6	83.591 $\nearrow$	82.865 $\nearrow$	82.133 $\nearrow$	81.381 $\searrow$	80.831 $\searrow$
7	83.623 $\nearrow$	82.830 $\searrow$	82.237 $\nearrow$	81.563 $\nearrow$	80.834 $\nearrow$
8	83.580 $\searrow$	82.973 $\nearrow$	82.307 $\nearrow$	81.474 $\searrow$	81.057 $\nearrow$
9	83.715 $\nearrow$	83.051 $\nearrow$	82.289 $\searrow$	81.830 $\nearrow$	81.194 $\nearrow$
10	83.783 $\nearrow$	83.153 $\nearrow$	82.603 $\nearrow$	81.952 $\nearrow$	81.247 $\nearrow$
	6	7	8	9	10
1	79.971	79.358	78.699	77.998	77.259
2	79.921 $\searrow$	79.426 $\nearrow$	78.801 $\nearrow$	78.139 $\nearrow$	77.449 $\nearrow$
3	80.175 $\nearrow$	79.415 $\searrow$	78.624 $\searrow$	77.806 $\searrow$	76.966 $\searrow$
4	79.831 $\searrow$	79.296 $\searrow$	78.708 $\nearrow$	78.073 $\nearrow$	77.395 $\nearrow$
5	80.127 $\nearrow$	79.394 $\nearrow$	78.678 $\searrow$	78.085 $\nearrow$	77.382 $\searrow$
6	80.144 $\nearrow$	79.478 $\nearrow$	78.766 $\nearrow$	78.072 $\searrow$	77.546 $\nearrow$
7	80.283 $\nearrow$	79.672 $\nearrow$	79.019 $\nearrow$	78.332 $\nearrow$	77.616 $\nearrow$
8	80.434 $\nearrow$	79.765 $\nearrow$	79.060 $\nearrow$	78.322 $\searrow$	77.555 $\searrow$
9	80.507 $\nearrow$	79.781 $\nearrow$	79.019 $\searrow$	78.227 $\searrow$	77.409 $\searrow$
10	80.496 $\searrow$	79.708 $\searrow$	78.888 $\searrow$	78.042 $\searrow$	77.527 $\nearrow$

Il ne semble pas y avoir de schéma général quant aux "oscillations" des valeurs d'état stationnaire en fonction de $Y_2(0)$. Tout au plus note-t-on

une tendance à l'augmentation quand $Y_2(0)$ augmente. On peut essayer de résumer les choses en calculant le % d'augmentation entre la première et la dernière ligne

	1	2	3	4	5	6	7	8	9	10
1	83.489	82.652	81.776	81.034	80.532	79.971	79.358	78.699	77.998	77.259
10	83.783	83.153	82.603	81.952	81.247	80.496	79.708	78.888	78.042	77.527

Plutôt que de faire les calculs à la main on se fait un petit programme sous Matlab dénommé *Pourcent*

```
% Pourcent
for i=1:10
    Prct(i)=100*(u(10,i)-u(1,i))/u(1,i);
end
```

On obtient les valeurs suivantes

$Y_1(0)$	1	2	3	4	5	6	7	8	9	10
%	0.3524	0.6058	1.0104	1.1333	0.8870	0.6562	0.4405	0.2407	0.0571	0.3473

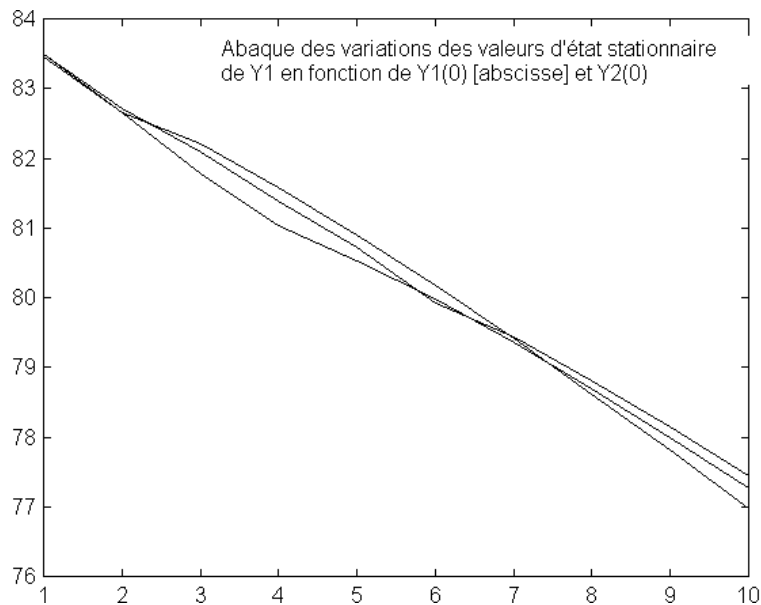
Il semblerait qu'il y ait d'abord une augmentation du pourcentage d'accroissement en fonction de la valeur initiale de Y_1 puis une baisse, encore que l'on note une bizarrerie pour les valeurs 9 et 10 ?

Il est bien évident que l'on pourrait pousser l'analyse pour comprendre comment se passent les choses. Ce n'est pas le but premier de nos développements qui visent à montrer l'attitude à adopter face à une simulation donnée.

Dans cette optique comment peut-on visualiser le tableau des valeurs d'état stationnaire. On pourrait penser à des tracés successifs des différentes lignes de notre tableau sur un même graphe. Ecrivons par exemple

```
» plot(u(1,:))
» hold on
» plot(u(2,:))
» plot(u(3,:))
» set(gcf,'color','w')
```

Le résultat n'est guère parlant. C'est pourquoi nous avons seulement passé des expressions sous la fenêtre de commande de Matlab (plutôt que de faire une M-file) afin de voir ce que cela donnait.

*Smlk073.*

Le fait est que l'on ne distingue pas grand chose si ce n'est la décroissance en fonction de $Y_1(0)$.

Il existe un mode de représentation plus parlant obtenu à l'aide de la fonction `surf`.

La fonction `surf`.

On a les deux fonctions permettant le tracé de surfaces ombrées en 3-D

`surf`, `surfc`

Syntaxe. On a les différentes syntaxes

```
surf(Z)
surf(X,Y,Z)
surf(X,Y,Z,C)
surf(...,'PropertyName',PropertyValue)
surfc(...)
```

Description.

On utilise `surf` et `surfc` pour visualiser des fonctions sur une région rectangulaire. Ces fonctions créent des surfaces paramétriques colorées spécifiées par X, Y et Z , avec les couleurs spécifiées par Z ou C .

→ `surf(Z)` crée une surface ombrée en 3 dimensions à partir des composantes `z` d'une matrice `Z` en utilisant

$$x = 1:n \text{ et } y = 1:m$$

$$o = \text{size}(Z)$$

c'est à dire que ***`x` correspond aux colonnes de `Z` et `y` aux lignes (! attention !)***. La hauteur `Z` est une fonction simple définie sur une grille géométrique rectangulaire. `Z` spécifie la donnée des couleurs tout autant que la hauteur de la surface. Ainsi la couleur est proportionnelle à la hauteur de la surface.

→ `surf(X,Y,Z)` crée une surface ombrée en utilisant `Z` pour la donnée des couleurs tout autant que la hauteur de la surface. `X` et `Y` sont des vecteurs ou des matrices définissant les composantes `x` et `y` de la surface.

Si `X` et `Y` sont des vecteurs tels que

$$\text{length}(X) = n \quad \text{length}(Y) = m$$

$$o = \text{size}(Z)$$

Alors dans ce cas les sommets des faces de la surface sont les triplets

$$[X(j), Y(i), Z(i, j)]$$

→ `surf(X,Y,Z,C)` crée une surface ombrée où les couleurs sont définies par `C`. Matlab réalise une interpolation linéaire sur les données de `C` pour obtenir des couleurs à partir de la " colormap " courante.

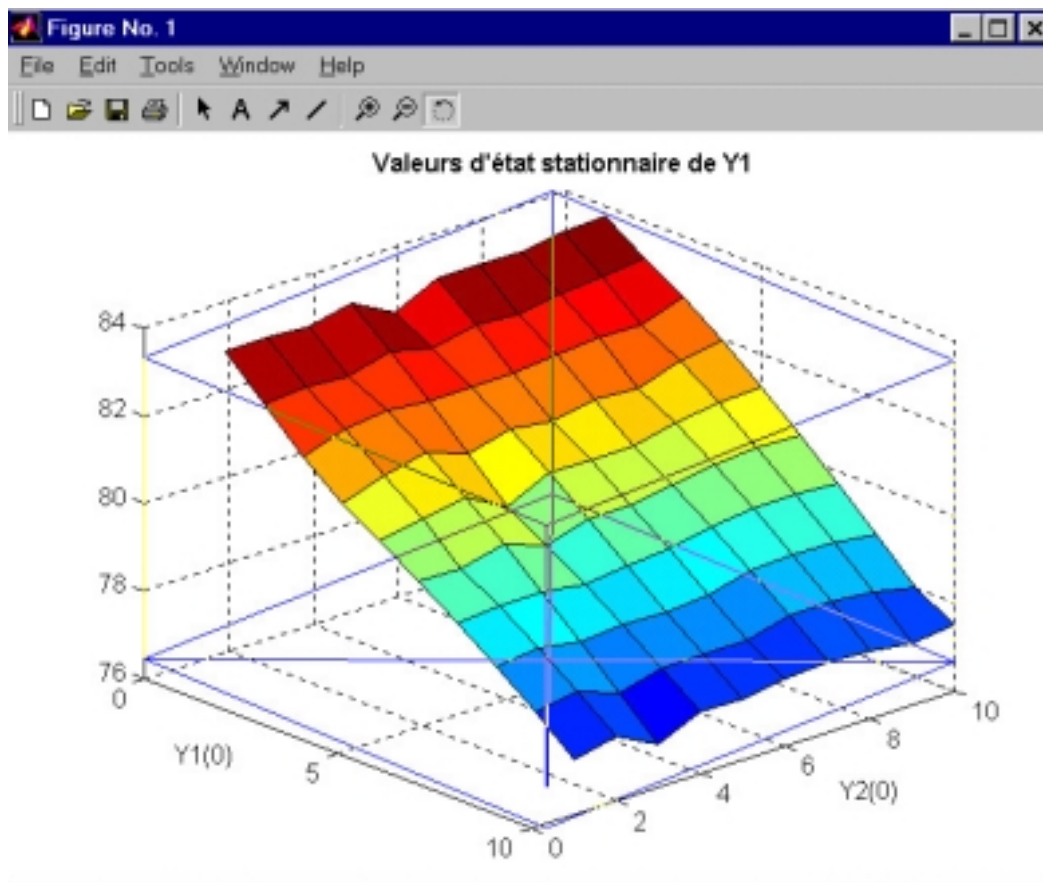
→ `surf(...)` trace un contour (courbes de niveaux) en dessous de la surface.

Dans le cas présent nous écrivons

$$\text{surf}(u)$$

et nous profitons des possibilités interactives offertes pour les graphes 3-D dans la dernière version de Matlab. On peut annoter le graphe en mettant un titre, en nommant les axes (***(se méfier du `x` = colonnes et `y` = lignes de la matrice)***).

En outre, la surface peut fort bien être tracée initialement sous un angle de vue qui ne facilite pas sa visualisation. Il n'est plus besoin d'avoir recours à la fonction `view` pour effectuer une rotation. Examinons la figure obtenue.



Smlk074.

Lorsque la figure est tracée elle est à l'état "brut" (sans annotations sans avoir été soumise à rotation). Pour faire évoluer la figure il suffit de cliquer sur l'icone  de la Barre d'outils. Puis on fait un clic gauche sur la figure un parallélépipède rectangle apparaît en surimpression il suffit alors de déplacer la souris à droite, à gauche, en haut ou en bas pour déplacer la figure.

Attention annotations et rotation de figure sont indépendantes. On peut avoir fait une rotation et annotés les axes. L'angle de vue ne nous plaisant plus si on change la position de la figure, on ne change pas pour autant la position des annotations d'axes.

Quelques remarques pour finir. Ne surtout pas essayer de faire `surf(M)`, `M` étant un tableau multi-dimensionnel Matlab nous répond

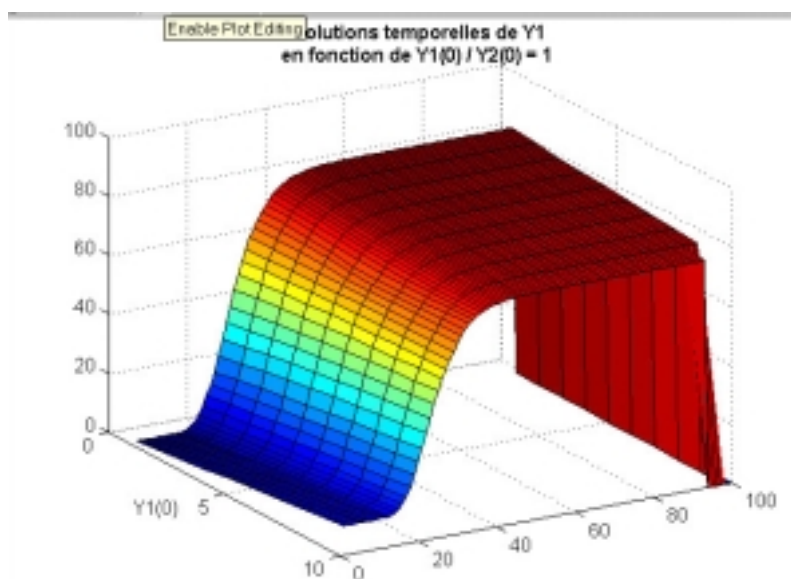
```
??? Error using ==> surface
CData must be an M-by-N matrix or M-by-N-by-3 array
+ un message technique
```

Pour tracer une surface représentative d'une feuille $N^{\circ}i$ de `M` (variations

temporelles de Y_1 en fonction de $Y_1(0)$ à $Y_2(0)$ donnée) il faut écrire

$$\text{Surf}(M(:, :, i))$$

On retrouve alors le même problème que celui visualisé dans la figure *Smlk071*. Les "fins" de vecteurs colonnes constitutifs de M étant à 0 on a une surface qui enregistre une chute généralisée en fin de parcours



Smlk075.

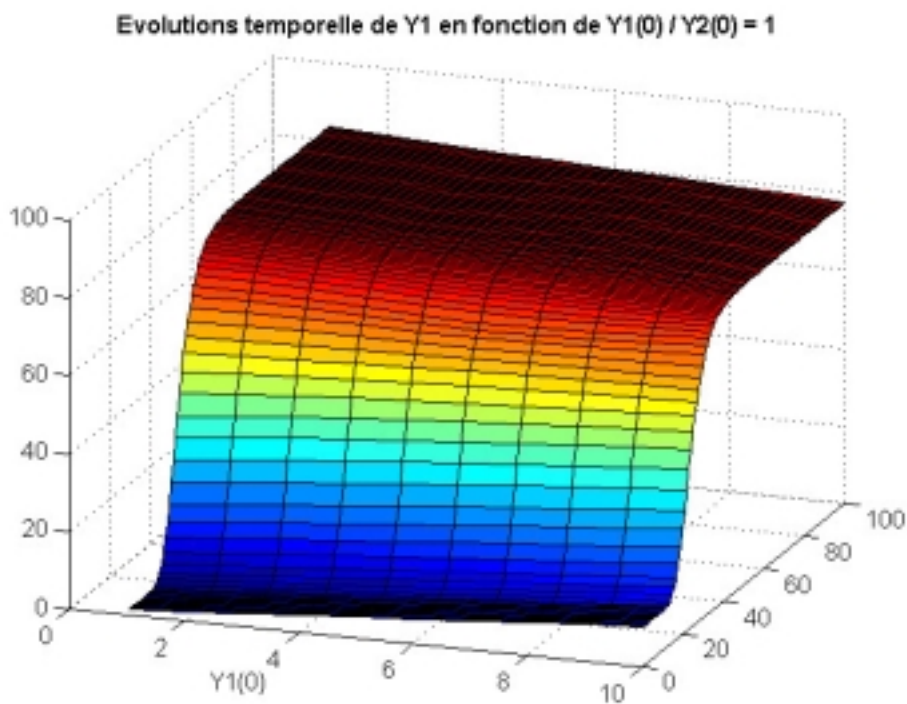
Pour éviter cet artefact il faut rectifier les feuilles de M comme on a rectifié la première feuille de T . Cependant dans le cas présent on ne connaît pas la valeur maximale atteinte chaque fois (pour T c'était 20) aussi il faut la rechercher dans chaque colonne. D'où le programme (M-file) dénommé *ModifM1*

```
% Modification de M1
for i=2:101 % Attention on va ajouter une ligne
    for j=1:10
        m(j)=max(M(:,j,1)); % Ne pas oublier le ;
        if M(i-1,j,1) >= m(j)
            M(i,j,1)=m(j); % Ne pas oublier le ;
        end
    end
end
for i=1:100
    for j=1:10
        M1(i,j,1)=M(i,j,1); % Ne pas oublier le ;
    end
end
```

On notera que ce programme permet de récupérer par la même occasion les valeurs maximales atteintes dans le temps par Y_1 (valeurs d'état stationnaire). Après exécution de ce programme on écrit

```
» surf(M1)
» set(gcf,'color','w')
```

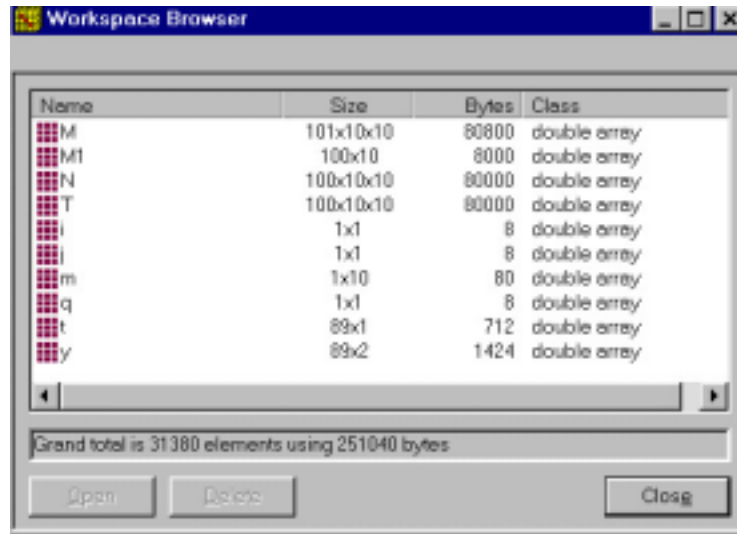
et on obtient la figure suivante (améliorée)



Smlk076.

L'angle de vision permet bien de voir que lorsque $Y_1(0)$ augmente la valeur d'état stationnaire diminue.

Cela étant jetons un coup d'oeil aux variables contenues dans l'espace de travail après exécution des deux programmes *Boucle.m* et *modifM1.m*. Pour ce faire on utilise le Workspace Browser.



Name	Size	Bytes	Class
M	101x10x10	80800	double array
M1	100x10	8000	double array
N	100x10x10	80000	double array
T	100x10x10	80000	double array
i	1x1	8	double array
j	1x1	8	double array
m	1x10	80	double array
q	1x1	8	double array
t	89x1	712	double array
y	89x2	1424	double array

Grand total is 31380 elements using 251040 bytes

Buttons: Open, Delete, Close

Smlk077.

Les matrices **M** et **N** contiennent toutes les informations sur les évolutions de Y_1 et Y_2 en fonction des valeurs initiales $Y_1(0)$ et $Y_2(0)$. On peut aisément retrouver ces informations en refaisant le programme *Boucle.m* après fermeture de Matlab et ouverture d'une nouvelle session. Mais on peut aussi sauvegarder ces informations en faisant dans la fenêtre de commande de Matlab **File + Save Workspace As**. Nous sauvegardons ainsi ces données sous le nom *Boucle.mat*.

Ainsi dans notre dossier courant on a deux fichiers *Boucle*, mais les **extensions** diffèrent l'une est ".m" c'est une *M-file*, l'autre est ".mat" ce sont des données Matlab.

On peut sans problème faire

```
» surf(M(:,:,1))
```

c'est à dire que l'on trace la surface correspondant à la matrice de la première feuille du tableau **M**. Si on veut maintenant utiliser l'information contenu dans la direction orthogonale, c'est à dire l'évolution temporelle de Y_1 pour une valeur donnée de $Y_1(0)$ avec variation de $Y_2(0)$ on a un petit problème. par exemple prenons l'expression

```
» surf(M(:,1,:))
```

Matlab nous répond

```
??? Error using ==> surf
Input argument must be a matrix
not a vector or a scalar
```

De fait lorsque l'on fait

$$M(:, :, 1)$$

Matlab nous fournit une réponse sous forme d'une matrice constituée de 10 vecteurs colonnes (100×1). Mais si l'on fait

$$M(:, 1, :)$$

il nous renvoie une réponse sous forme de 10 vecteurs colonnes (100×1) séparés. C'est à dire que **dès que l'on n'est plus dans une "feuille", le plan formé à l'aide des 2 premiers indices du tableau, il faut avoir recours à un stockage dans une matrice intermédiaire.**

Dans le cas présent si l'on veut pouvoir travailler sur les variations temporelles de Y_1 en fonction de $Y_2(0)$ sachant que $Y_1(0) = 1$ on doit passer par le petit programme dénommé *MY2.m*

```
%Matrice en Y2
for i=1:100
    for j=1:10
        MY2(i,j)=M(i,1,j);
    end
end
```

On peut alors effectuer sans problème

```
>> surf(MY2)
```

En possession de tous ses éléments de base on peut développer un modèle de SSD avec son traitement complet.