3.2. Matlab/Simulink

3.2.1. Généralités

Il s'agit d'un logiciel parfaitement dédié à la résolution de problèmes d'analyse numérique ou de traitement du signal. Il permet d'effectuer des calculs matriciels, de visualiser les résultats sous forme graphique. La formulation des problèmes s'apparente à la formulation mathématique des problèmes à résoudre. L'utilisation du logiciel consiste à lancer des lignes de commandes, qui peuvent le plus souvent s'apparenter à de la programmation en C. Associé à Simulink (commande lancée sous Matlab), il devient un outil graphique très simple d'utilisation pour la simulation de processus (programmation par copier/coller de blocs fonctionnels).

3.2.2. Travailler avec Matlab

Matlab dispose de plusieurs fenêtres (selon les versions) dont la principale est la **fenêtre de commande** (Command Window) associée à l'espace de travail (**Workspace**). Il s'agit de la première fenêtre ouverte dans laquelle seront tapées les différentes 'commandes' après le prompt '>>'. C'est également dans cet espace de travail que sont définies toutes les variables utilisées par Matlab. (cf. Figure 3.4, commande whos)

La syntaxe générale d'appel d'une fonction est :
[s1, s2, ..., sn] = nom_fonction(e1, e2, ..., ep)
où les ei sont les paramètres d'entrée de la fonction nom_fonction et les sj les paramètres de sortie. **Le point virgule ';'** facultatif dernière une commande empêche l'affichage du résultat de celle-ci (bien utile lorsque le résultat est un vecteur de grande taille).

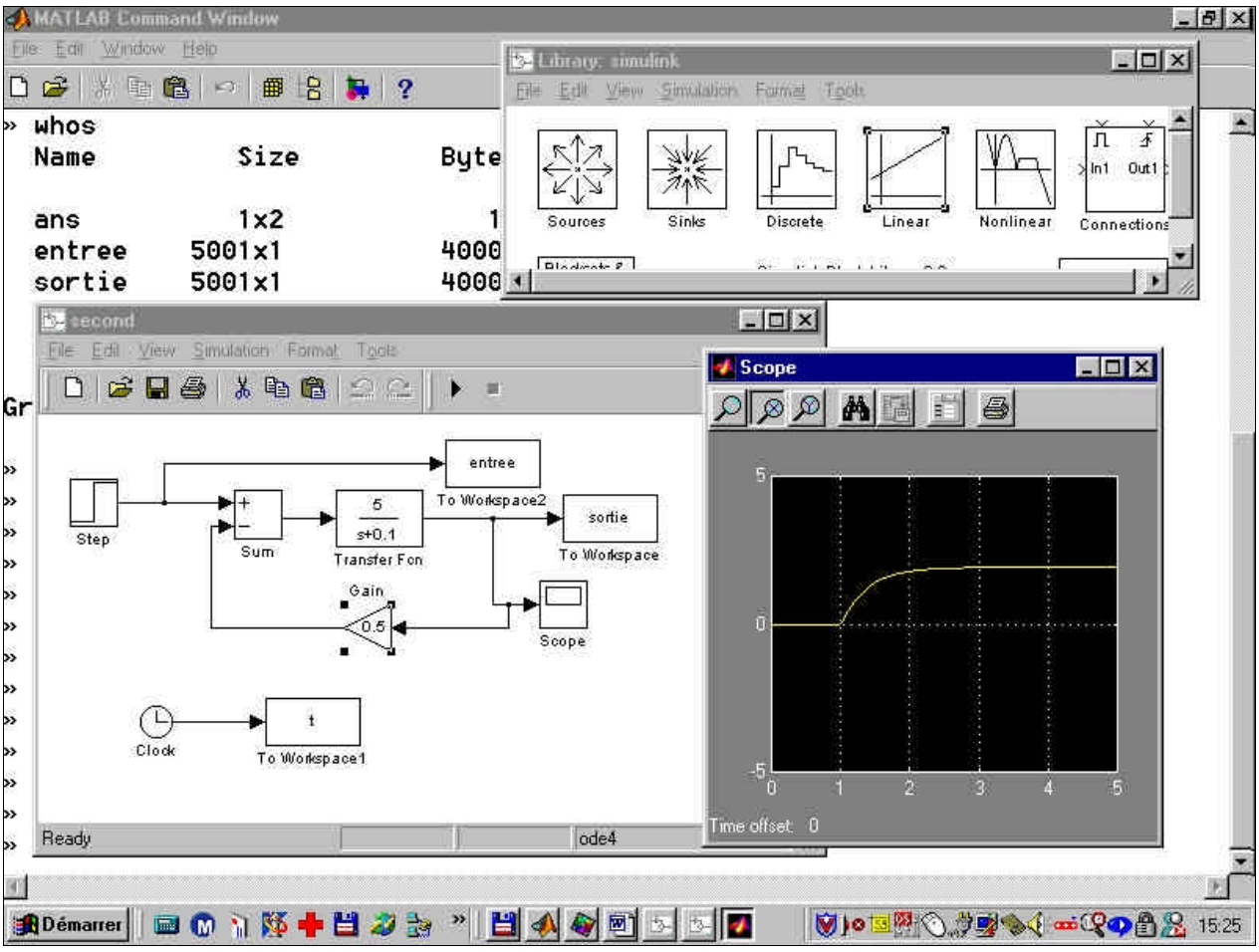

Figure 3.4 : Matlab v.5.2.1.1420 – Fenêtres de Commande, Simulink, Scope, second.mdl

COURBE.M

Fichier Edition Rechercher ?

```
% Programme exemple pour tracer sur un même graphe
% les courbes :          y=cos(t)   et  z=sin(y)
% sur l'intervalle :    [-2.pi, 2.pi]

t=-2*pi:0.1:2*pi;
y=cos(t);
z=sin(2.5*y);

figure(1);
clg;
plot(t,y,'r');
hold on;
plot(t,z,'g');
axis([-2*pi 2*pi -1 1]);

xlabel('t'); ylabel('y et z'); title('courbes pour exemple');
grid;
gtext('y=cos(t) -->');
gtext('<-- z=sin(3.cos(t))');
gtext('WN avril 1996');
```

Calcul des points des courbes

Sélection de la figure 1 et effacement

Tracé de la première courbe

Tracé de la 2de courbe sur le même graphe

Modification des échelles des axes

Etiquettes des axes, titres, ...

Figure 3.5 : Matlab v3 – Fichier programme, courbe.m

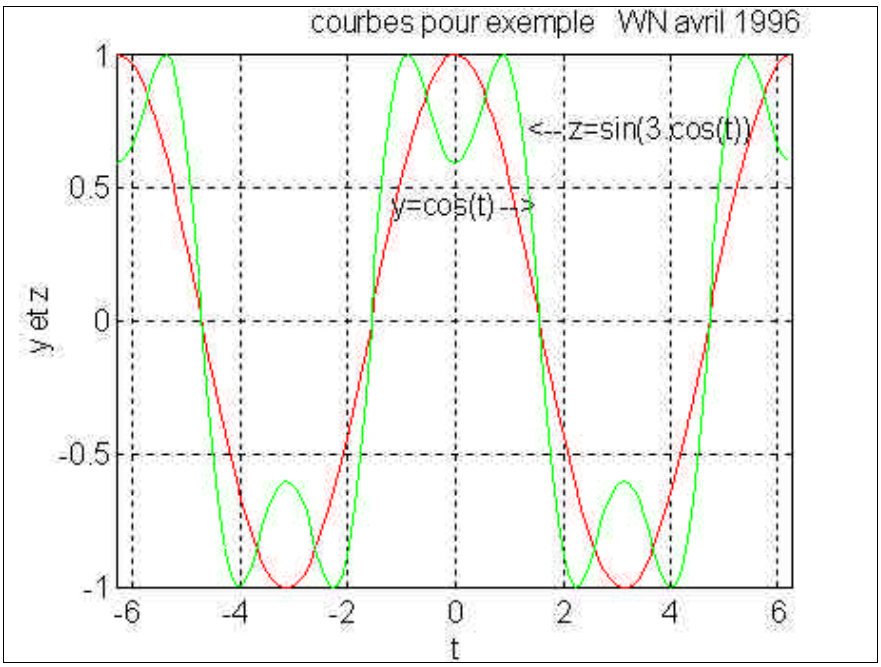

Figure 3.6 : Matlab v3 – Fenêtre graphique résultat du programme courbe.m (figure 3.5)

L'intérêt de Matlab est de pouvoir travailler sur des fichiers d'extension « .m » contenant des suites de commandes (créés avec un simple éditeur de texte) et lancé par leur nom (sans extension) comme une commande. Un exemple est donné sur la Figure 3.5, avec son résultat sur la Figure 3.6.

Il est impératif avant d'utiliser une commande de regarder sa syntaxe, c'est le rôle de la commande :

help <commande_cherchée>	lorsque la commande est connue
lookfor <mot clef anglais>	permet une recherche par mot clef (ou une partie)

Toutes les **commandes unix** sont utilisables dans le workspace, il suffit de faire précéder celles-ci d'un point d'exclamation. Par exemple, « ! ls –la) (= dir).

On notera que Matlab permet de manipuler facilement les fonctions de Transfert (cf. cours d'Automatique, cf. cours de Mathématiques (transformée de Laplace)) pour l'étude du processus. Associé à Simulink, cela en fait un outil très pratique.

3.2.3. Travailler avec Simulink

Simulink est un logiciel de simulation de systèmes dynamiques muni d'une interface graphique pilotée par souris, ce qui facilite les deux phases d'utilisation du logiciel : saisie du modèle et analyse du modèle. Il s'agit d'une extension du logiciel de calcul Matlab qui propose de nombreuses bibliothèques de modèles intéressant l'ingénieur (modèles de type fonctions de transfert (linéaire ou échantillonnées...)) car immédiatement exploitables. La figure 3.6 présente la fenêtre de bibliothèque (Library) ainsi qu'un exemple de fichier de simulation (second.mdl).

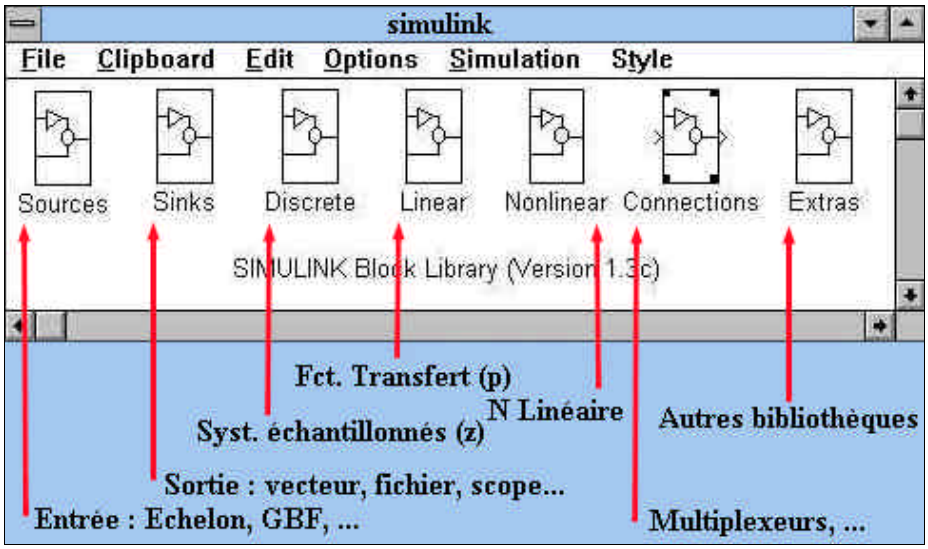

Figure 3.7 : Simulink v1.3c – Fenêtre bibliothèque de modèles

Pour accéder aux modèles qu'elles proposent, il suffit de cliquer deux fois (bouton de gauche) sur l'icône correspondante. Les différentes bibliothèques sont (cf. figure 3.7) :

Sources	permet de créer des signaux d'entrées (ex. échelon, sinus, bruits...)
Sinks	permet de récupérer les signaux de sorties (ex. oscilloscope, fichier,...)
Discrete	propose des modèles de fonctions de transfert numériques
Linear	propose des modèles de fonctions de transfert analogiques
Nonlinear	modèles non linéaires
Connections	divers outils permettant de connecter différents blocs (multiplexeur)
Extra	contient différentes bibliothèques spécifiques

Pour travailler, il convient de **créer un nouveau fichier** en ouvrant le menu déroulant FILE et en sélectionnant : NEW. Vous créez ainsi un espace de travail Simulink qui vous permettra de créer vos propres modèles. N'oubliez pas de sauvegarder votre fichier. Pour utiliser un modèle existant il suffit de sélectionner le modèle à l'aide de la souris, dans le menu EDIT de copier celui-ci (COPY) puis de le coller (PASTE) dans son espace de travail (fichier .mdl). L'utilisation des raccourcis Ctrl-C et Ctrl-V (copier/coller) est possible.

Sur la Figure 3.4, un exemple de fichier de simulation d'un système du premier ordre avec retour de la sortie est donné ainsi que le résultat de la simulation dans la fenêtre Scope (réponse indicielle).

On note que pour **définir les paramètres du modèle copié**, il suffit de cliquer deux fois sur celui-ci. Une fenêtre de dialogue s'ouvre alors (cf. Figure 3.8). Vous devez remplir les informations nécessaires en respectant la syntaxe proposée. Il est très intéressant d'introduire des paramètres formels dans les blocs utilisés. L'intérêt est une plus grande souplesse de travail lorsque l'on choisit de modifier les paramètres. Les variables utilisées (ou créées) par Simulink sont définies (ou récupérées) dans l'espace de travail de Matlab. Ainsi, on créera un programme Matlab qui définira toutes les variables nécessaires (par exemple numérateur et dénominateur des fonctions de transfert).

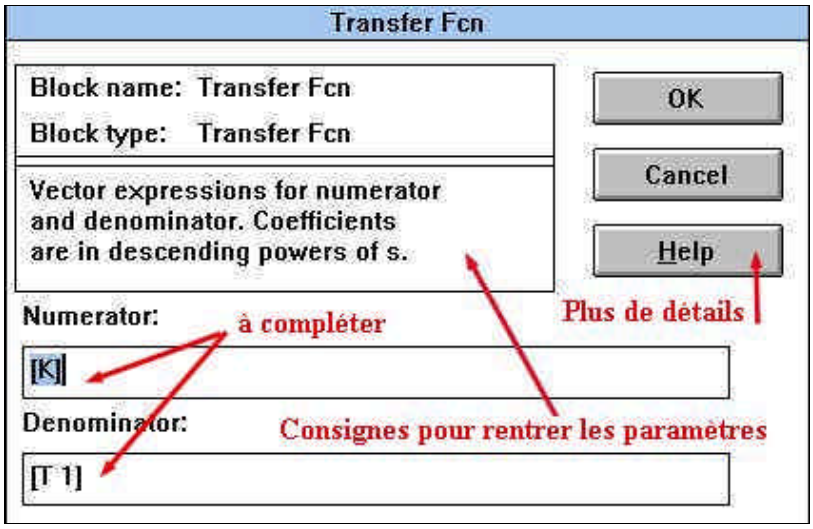

Figure 3.8 : Simulink v1.3c – Exemple de fenêtre de dialogue pour le bloc « Transfer Fcn »

Les connections entre les différents blocs se réalisent à la souris en cliquant sur la sortie d'un bloc et en tirant le fil ainsi créé jusqu'à l'entrée du bloc suivant. Pour modifier la position d'un fil, il suffit de le sélectionner en cliquant avec le bouton de gauche de la souris (la sélection est visualisée par un cercle).

Les signaux de sortie sont récupérés dans une variable du Workspace, dans un fichier d'extension « .mat » ou visualisés à l'écran selon la « SINKS » choisie (cf. Figure 3.10

Avant de simuler, assurez-vous que votre schéma comporte un signal d'entrée (échelon par exemple pour une réponse indicielle), un signal de sortie et que tous les paramètres des blocs sont définis. Afin de récupérer le temps de simulation, il est prudent d'ajouter une horloge (Clock) et d'en sauvegarder la valeur de sortie dans un vecteur.

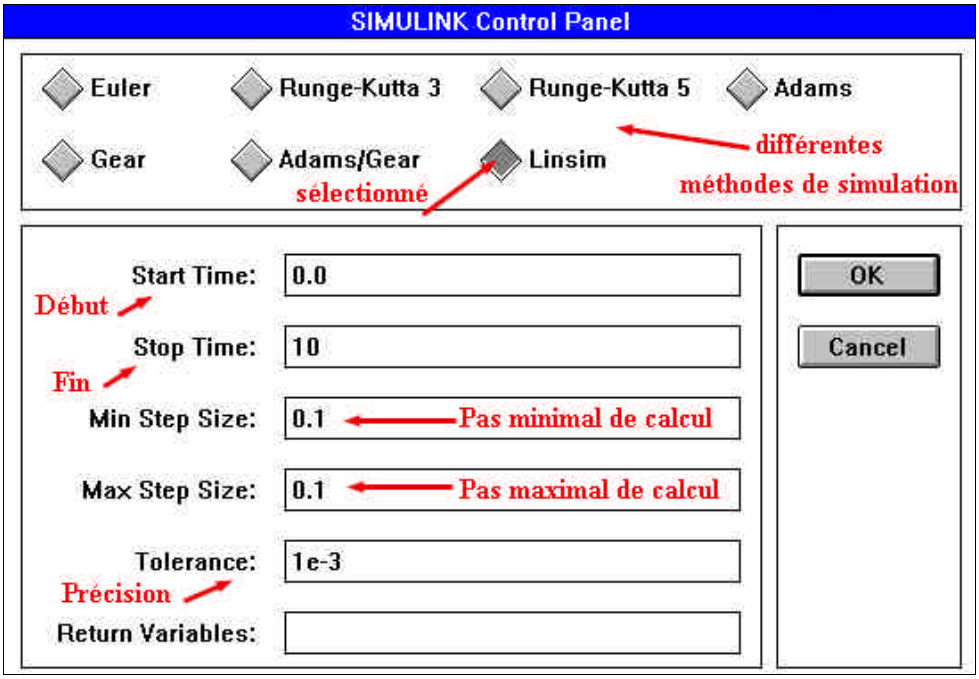

Figure 3.9 : simulink v1.3c – Exemple de fenêtre de dialogue pour le bloc « Transfer Fcn »

Enfin, il est impératif de configurer les paramètres de simulation (cf. Figure 3.9) en choisissant :

Algorithme de simulation	linsim (si vous travaillez en linéaire), runge-kutta etc.
L'instant de départ	Start time généralement 0 seconde
L'instant de fin de simulation	Stop time, pour atteindre le régime permanent
Le pas de calcul minimal	Min step size (cf. période d'échantillonnage)
Le pas de calcul maximal	Max step size

Le pas de calcul varie du pas min au pas max. Ceci permet au logiciel d'augmenter la vitesse de simulation cependant ceci peut empêcher l'observation de certains phénomènes transitoires. Pour éviter ce problème, on peut dans un premier temps choisir les deux pas égaux.

Pour un **système du premier ordre** de constante de temps T, on sait que le temps de réponse est de l'ordre de 4 à 5 fois cette constante de temps. Il suffit donc de choisir la durée de simulation en conséquence soit Stop time >5T. Par ailleurs, on prend généralement au moins 10 points par constante de temps, ce qui conduit à un pas de calcul
Step size < T/1. Pour les **systèmes d'ordre n** de constantes de temps T1, T2, ..., Tn, on peut faire les choix suivants : Stop time > 5.max(T1,..., Tn), Step size < min(T1,..., Tn)/10.

On lance la simulation en cliquant sur Start du menu Simulation. Au bip, la simulation est terminée. Vous pouvez dès lors utiliser ces résultats. Attention, si vous simulez à nouveau, vous perdrez les anciens résultats. On peut également la lancer d'un fichier par : linsim('fichier_simulink',[start_time,stop_time],[ ],[tolerance, min_step_size,max_step_size]);

3.2.4. Commandes utiles de Matlab et modèles de Simulink

%	ligne de commentaire
whos	donne la liste des **variables** déjà définies et leur taille
clear, clear <var>	efface toutes les variables (ou l'une d'elle <var>)
help	donne la liste de toutes les commandes (bibliothèques)
lookfor mot	**cherche** les commandes avec 'mot' dans leur nom ou définition.
fprintf('chargement \n');	affiche un commentaire à l'écran
load <fchier_données>	chargement des données de fichier_données.mat
save <esultat> <var>	sauvegarde la variable var (si elle existe) dans résultat.mat

Vecteurs, matrices, variables

t=deb:pas:fin;	crée un vecteur de points espacés de pas sur [deb,fin]
a=2+3j	définie un nombre complexe (i tel que i*i=-1 est noté i ou j)
find(condition_sur_x)	renvoie les indices des points de x vérifiant la condition (x>100)
n=[2 1]; **A**=[1 2 ; 2 4]	définition de **vecteur** (polynôme) ou **matrice**
zeros(1,4),	crée un vecteur ligne (1x4) de zéros
ones(5,1)	crée un vecteur colonne (5x1) de 1
eye(4)	crée une matrice identité 4x4
diag(x)	crée une matrice diagonale avec les éléments de x
[y,m]=max(x)	renvoie le maximum de x (y) et son indice (m) si x est un vecteur

size(A)	renvoie le nombre de ligne et de colonne de A
A(i,j), A(:,2), A(1,:)	renvoie l'élément de la $i^{ème}$ ligne et de la $j^{ème}$ colonne de A , renvoie la deuxième colonne de A , la première ligne de A **inv(A)** calcul **l'inverse** de A
A'	calcul de la **transposée** de A
det(A)	calcule du **déterminant** de A

Tracés de courbes

Figure, figure(i)	crée une nouvelle figure, rend active la figure numéro i
w=logspace(-2,2,100);	crée une échelle logarithmique de 100 points entre 0.01 et 100
diff	**dérivée** numérique.
plot(t,y)	**tracé** de y en fonction de t
plot(t,y,'r',t,z,'g')	tracé de y (rouge) et z (vert) fonction de t sur le même graphe
title('titre') , grid	met le 'titre' en titre de la figure ; trace une grille sur le graphe
xlabel, ylabel	étiquettes des abscisses et des ordonnées
gtext	place un commentaire (à la souris) sur le graphe
hold on, hold off	permet de tracer plusieurs courbes sur le même graphe
zoom on, zoom off	permet de faire le zoom d'une zone à l'aide de la souris
close, close (i)	fermet la fenêtre active ou précisée (i)
subplot(231); subplot(232)	permet de diviser la fenêtre graphique en 6 zones (2x3), 2 lignes, 3 colonnes. Les différentes zones sont repérées de 1 à 6 (cf. tableau) colonnes puis lignes.

Statistiques…

mean(x) , std(x)	valeur **moyenne** de x et **écart-type** de x (standard deviation)
polyfit(x,y,n)	calcul les coefficient s d'un **polynôme** , de degré n, passant par les points spécifiés : M(x,y)
polyval(P,x)	permet le calcul des valeurs d'un polynôme P(x), defini par ses coefficients P=(p(i,j)), pour des valeurs de x.

Sur les systèmes dynamiques

n=[2 1]; d=conv(n,[1 1])	définition du **polynôme** d **convolution** de n=2x+1 et (1.x+1)
printsys(n,d)	permet d'afficher la fonction de transfert n/d
[a,b,c,d]=tf2ss(n,d)	renvoie la représentation d'état correspondante n/d (cas continu)
y=step(n,d,t);	réponse indicielle (échelon) du système n/d, échelle de temps : t
[nbf,dbf]=cloop(n,d,-1)	boucle fermée avec retour unitaire négatif
[nbf,dbf]=feedback(n,d,nc,dc,-1)	boucle fermée avec correcteur (nc/dc) et retour négatif
bode, margiN	Diagrammes de Bode
loglog, semilogx,	Tracé en échelle log/log ou semi-log
nyquist	Diagrammes de Nyquist
rlocus	lieu d'Evans
k=place(A,B,P)	calcul du gain k tel que les pôles de (A- Bk) soit P (vecteur)

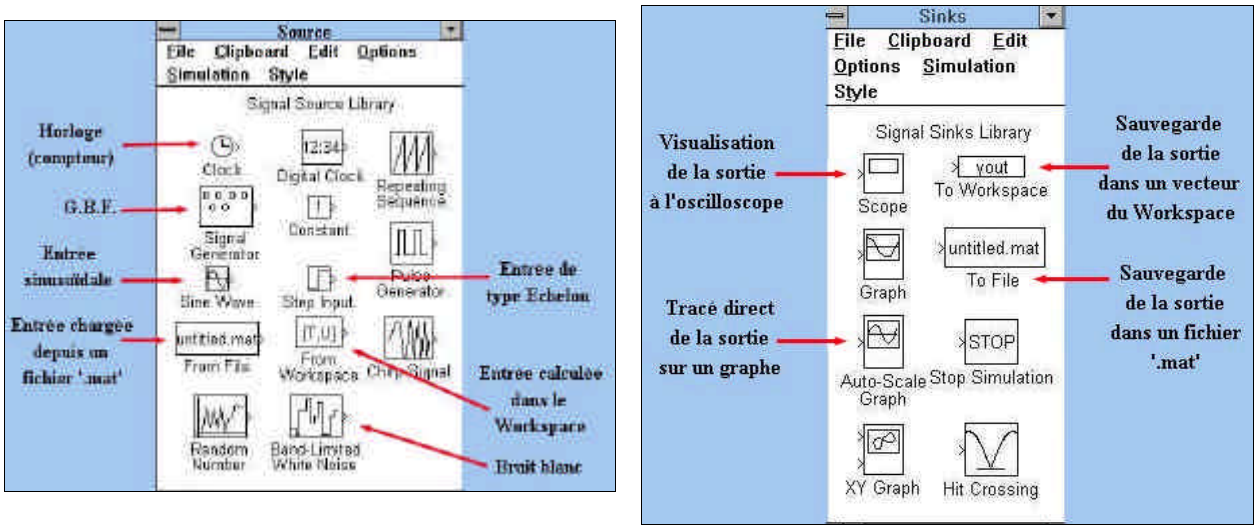

Figure 3.10 : Simulink v1.3c – Modèles « SOURCE » et « SINKS »

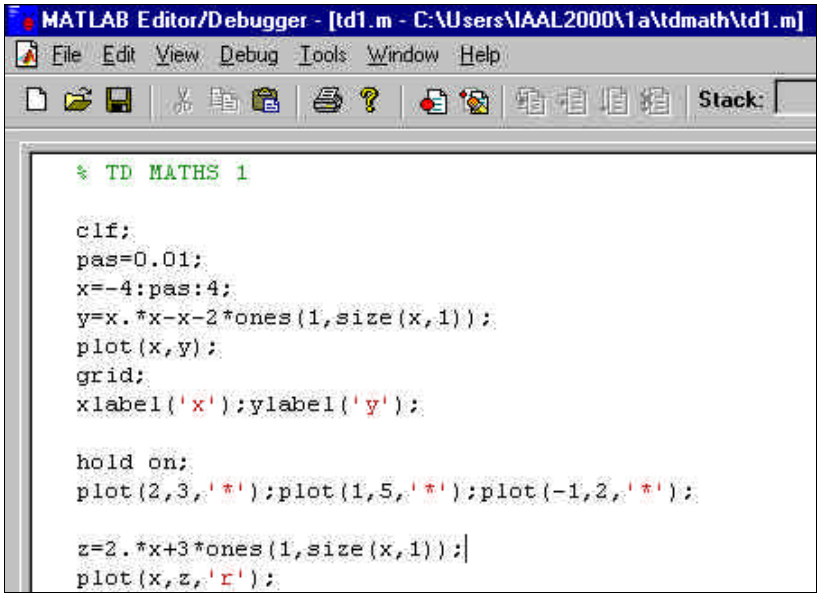

Figure 3.11 : Matlab v5 – Editeur/Debugger avec fichier td1.m ouvert.

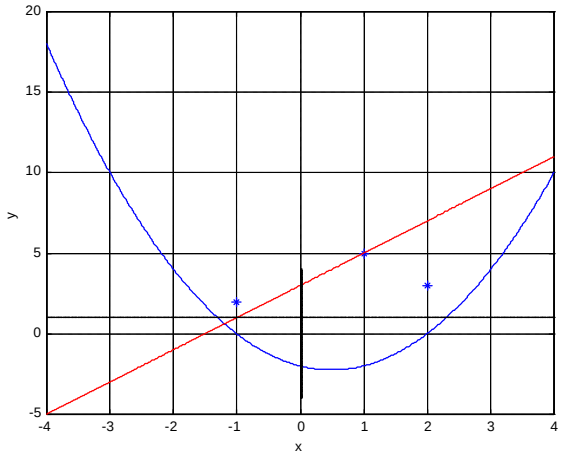

Figure 3.12 : Matlab v5 – Résultat du fichier td1.m (cf. Figure 3.11).