

I.1.Introduction de MATLAB:

Matlab est un logiciel de calcul numérique commercialisé par la société MathWorks1. Il a été initialement développé à la fin des années 70 par Cleve Moler, professeur de mathématique à l'université du Nouveau-Mexique puis à Stanford, pour permettre aux étudiants de travailler à partir d'un outil de programmation de haut niveau et sans apprendre le Fortran ou le C [02].

Matlab signifie Matrix laboratory. Il est un langage pour le calcul scientifique, l'analyse de données, leur visualisation, le développement d'algorithmes. Son interface propose, d'une part, une fenêtre interactive type console pour l'exécution de commandes, et d'autre part, un environnement de développement intégré (IDE) pour la programmation d'applications[02].

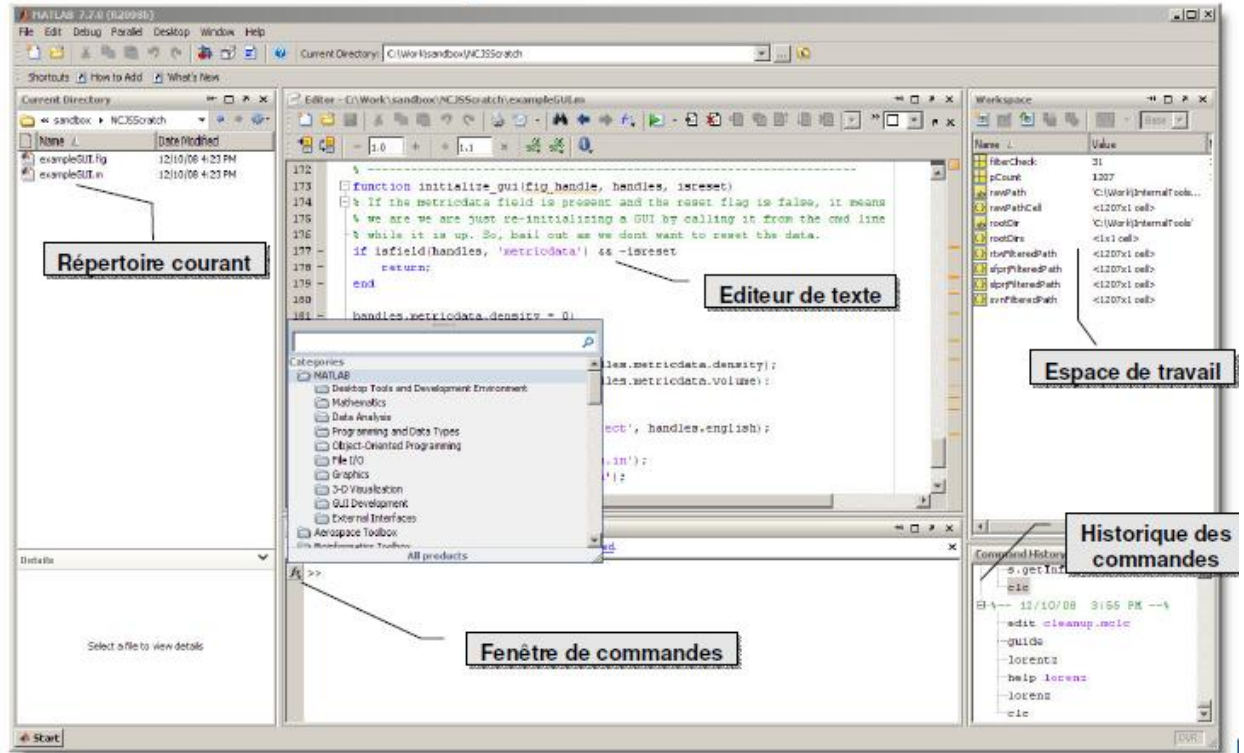
Matlab trouve ses applications dans de nombreuses disciplines. Il constitue un outil numérique puissant pour la modélisation de systèmes physiques, la simulation de modèles mathématiques, la conception et la validation (tests en simulation et expérimentation) d'applications. Le logiciel de base peut être complété par de multiples toolboxes, c'est-à-dire des boîtes à outils. Celles-ci sont des bibliothèques de fonctions dédiées à des domaines particuliers. Nous pouvons citer par exemple : l'Automatique, le traitement du signal, l'analyse statistique, l'optimisation...[02]

Voici une liste non exhaustive (loin de là) et en vrac de toolboxes, montrant la diversité des fonctionnalités de Matlab :

Control System Toolbox	Symbolic Math Toolbox	Signal Processing Toolbox
Neural Network Toolbox	Optimization Toolbox	Parallel Computing Toolbox
Statistics Toolbox	Fuzzy Logic Toolbox	Image Processing Toolbox
Aerospace Toolbox	Data Acquisition Toolbox	Bioinformatics Toolbox
MATLAB Compiler	Vehicle Network Toolbox	Model-Based Calibration Toolbox
Financial Toolbox	RF Toolbox	System Identification Toolbox

1.2. Interface principale de MATLAB

Après l'exécution de Matlab, vous pouvez constater que l'écran se compose de Cinque fenêtres.[2]



Le logiciel propose un véritable environnement de travail composé de multiples fenêtres. Nous pouvons distinguer quatre blocs :

Command window(console d'exécution) : à l'invite de commande « >> », l'utilisateur peut entrer les instructions à exécuter. Il s'agit de la fenêtre principale de l'interface. [2]

Current directory(répertoire courant) : permet de naviguer et de visualiser le contenu du répertoire courant de l'utilisateur. Les programmes de l'utilisateur doivent être situés dans ce répertoire pour être visible et donc exécutable. [2]

Workspace (espace de travail) : permet de visualiser les variables définies, leur type, la taille occupée en mémoire. [2]

Command history: historique des commandes que l'utilisateur a exécutées. Il est possible de faire glisser ces commandes vers la fenêtre de commande..[2]

I.3 - Interface graphique user interface de MATLAB

I.3.1.GUI Layout :

GUIDE, l'utilisateur graphique environnement de développement d'interface MATLAB, fournit un ensemble d'outils pour créer des interfaces utilisateur graphiques (GUI). Ces outils simplifient le processus de l'aménagement et la programmation des interfaces graphiques. Utilisation du guide Layout Editor, vous pouvez remplir une interface graphique en cliquant et en faisant glisser l'interface graphique des composants tels que des haches, des panneaux, des boutons, des champs de texte, curseurs, etc.-dans la zone de mise en page. Vous pouvez également créer des menus et des menus contextuels pour l'interface graphique. De l'éditeur de mise, vous pouvez la taille de l'interface utilisateur, modifier le composant look and feel, aligner les composants, définir l'ordre de tabulation, afficher une liste hiérarchique des objets de composant, et définir les options de l'interface graphique[03].

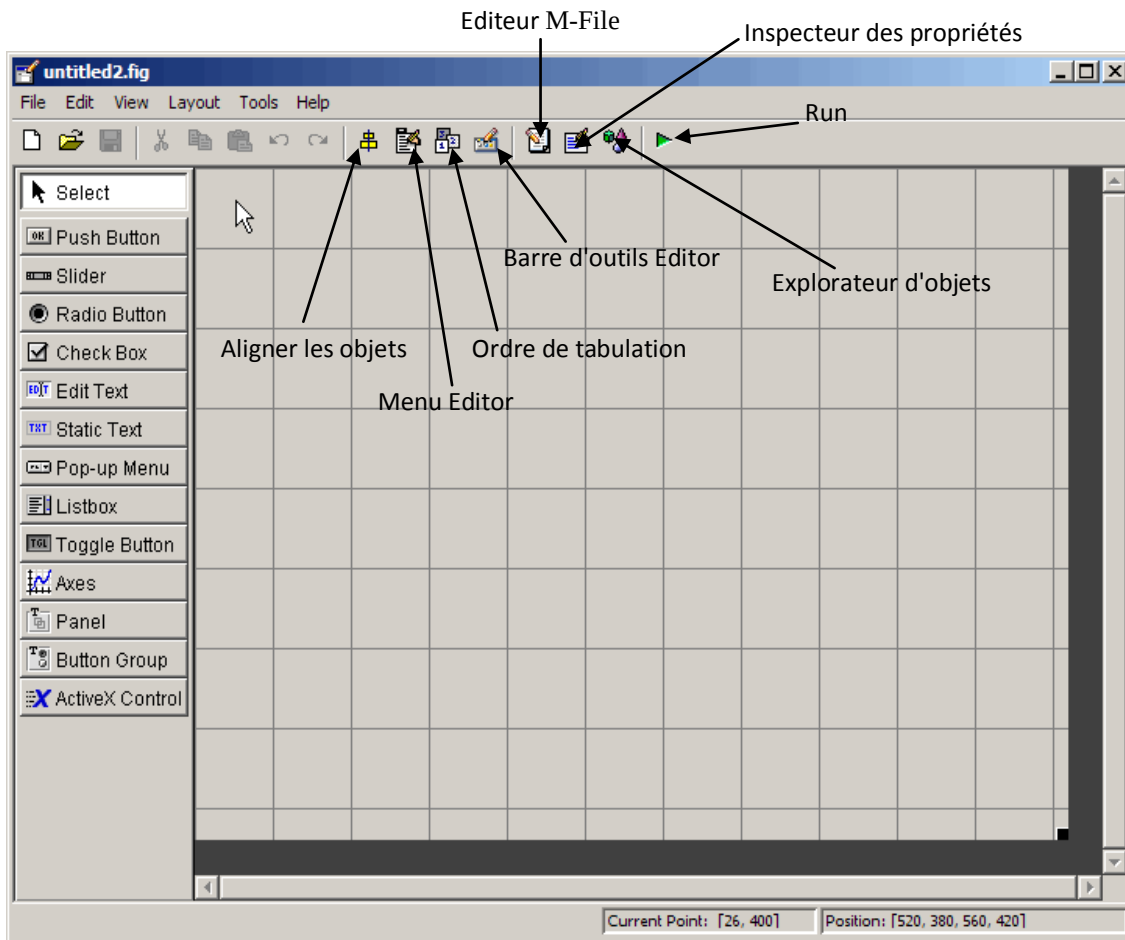
I.3.2.Programmation GUI :

GUIDE génère automatiquement un M-fichier qui contrôle la façon dont l'interface graphique fonctionne.

Ce M-fichier fournit le code pour initialiser l'interface graphique et contient un cadre pour l'interface graphique callbacks-les routines qui exécutent lorsqu'un utilisateur interagit avec un composant d'interface graphique. Utilisation de l'éditeur M-fichier, vous pouvez ajouter du code à callbacks pour exécuter les fonctions souhaitées[03].

I.3.3.GUIDE Outils Résumé :

Les outils de GUIDE sont disponibles à partir de l'éditeur de disposition indiqué dans la figure ci-dessous. Les outils sont appelés dans la figure et décrit brièvement ci dessous[03].



Utilisez cet outil...	À...
Layout Editor	Sélectionnez des composants à partir de la palette de composants, sur le côté gauche de l'éditeur de mise, et les disposer dans la zone de mise en page. [03]
Figure Resize Tab	Réglez la taille à laquelle l'interface graphique est initialement affiché lorsque vous exécuter. [03]
Menu Editor	Créer des menus et le contenu, à savoir, pop-up, menus. Voir "Création de menus"[03]
Aligner les objets	Aligner et distribuer des groupes de composants. Grids et dirigeants vous permettra également d'aligner les composants sur une grille avec une capacité optionnelle snap-to-grid. [03]
Ordre de tabulation Editor	Réglez l'onglet et l'ordre d'empilement des composants de votre disposition. [03]

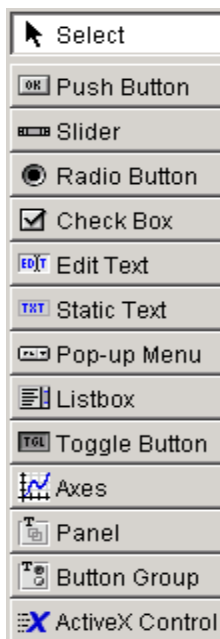
barre d'outils Editeur	Créer des barres d'outils contenant pression prédéfini et personnalisé boutons et boutons à bascule. [03]
icone Editor	Créer et modifier des icônes pour les outils dans une barre d'outils. [05]
Propriété Inspecteur	Définissez les propriétés des composants dans votre mise en page. Il fournit une liste de toutes les propriétés que vous pouvez définir et affiche leurs valeurs actuelles. [03]
Objet Navigateur	Afficher une liste hiérarchique des objets dans l'interface graphique. [05]
Run	Enregistrez et exécutez l'interface actuelle. [03]
Editor M-File	Affichage, dans votre éditeur par défaut, le M-fichier associé à la GUI. [03]
Position afficheurs	afficher en permanence la position du curseur de la souris et le les positions des objets sélectionnés[03]

Tableau (I.1): les outils en GUI

Vous pouvez également définir des préférences applicables à tous les gars à la création, et les options qui sont spécifiques à GUI. [05]

I.3.4.Afficher les noms dans la palette de composants :

Affiche les icônes et les noms dans la palette de composants, comme illustré ci-dessous. [05]



Palette de composants avec des noms



Palette de composants sans nomes

Composant	Icône	La description
Push Button		Boutons poussoirs génèrent une action lorsque vous cliquez dessus. [03]
Toggle Button		boutons à bascule génèrent une action et indiquer si elles sont activées ou désactivées. Lorsque vous cliquez sur un bouton à bascule, il apparaît enfoncé, montrant qu'il est allumé. Lorsque vous relâchez le bouton de la souris, le bouton bascule reste enfoncé jusqu'à ce que vous cliquez une deuxième fois. Lorsque vous le faites, le bouton retourne à l'état soulevé, montrant qu'il est hors tension. Utilisez un groupe de boutons pour gérer mutuellement exclusifs boutons à bascule. [03]
Radio Button		Les boutons radio sont similaires aux cases à cocher, mais les boutons radio sont généralement mutuellement exclusifs dans un groupe de boutons radio connexes. Autrement dit, lorsque vous sélectionnez un bouton sur le bouton sélectionné précédemment est désactivé. Pour activer un bouton radio, cliquez sur le bouton de la souris sur l'objet. L'affichage indique l'état de la touche. Utiliser un groupe de boutons pour gérer les boutons radio mutuellement exclusifs. [03]
Check Box		Cochez les cases peuvent générer une action lors d'un contrôle et d'indiquer leur état comme cochée ou non. Cochez les cases sont utiles lors de la fourniture à l'utilisateur un certain nombre de choix indépendants. [03]
Edit Text		Modifier les composants de texte sont des champs qui permettent aux utilisateurs d'entrer ou de modifier des chaînes de texte. Utilisez modifier le texte lorsque vous voulez que le texte en entrée. Les utilisateurs peuvent entrer des chiffres, mais vous devez les convertir en leurs équivalents numériques. [03]
Static Text		Le texte statique contrôle des lignes de texte d'affichage. Le texte statique est généralement utilisé pour étiqueter d'autres contrôles, donner des instructions à l'utilisateur, ou d'indiquer les valeurs associées à un curseur. Les utilisateurs ne peuvent pas modifier le

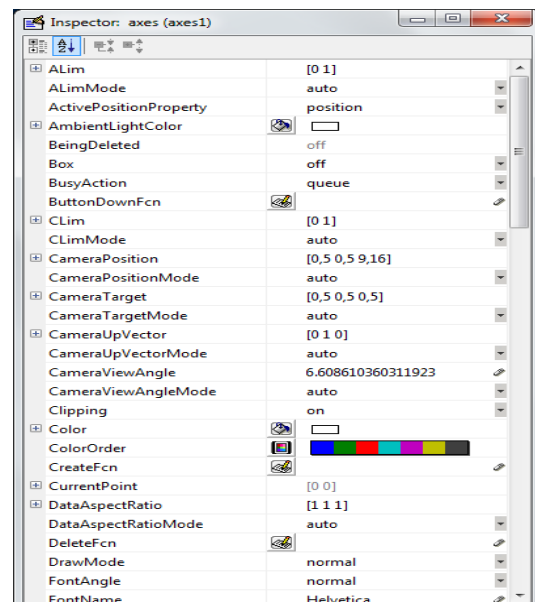
		texte statique interactif. [03]
Slider		curseur acceptent une entrée numérique dans une plage spécifiée en permettant à l'utilisateur de déplacer une barre coulissante, qui est appelé un curseur ou le pouce. Les utilisateurs se déplacent le curseur en cliquant sur le curseur et en le déplaçant, en cliquant dans le creux, ou en cliquant sur une flèche. L'emplacement du curseur indique la position relative à l'intérieur de la plage spécifiée. [03]
List Box		Les zones de liste affichent une liste d'éléments et permettent aux utilisateurs de sélectionner un ou plusieurs éléments. [03]
Pop-Up Menu		Pop-up menus ouverts pour afficher une liste de choix lorsque les utilisateurs cliquent sur la flèche. [03]
Axes		Haches permettent à votre interface graphique pour afficher des graphiques tels que des graphiques et des images. Comme tous les objets graphiques, les axes ont des propriétés que vous pouvez définir pour contrôler de nombreux aspects de son comportement et de l'apparence. Voir «Propriétés Haches» dans la documentation graphique MATLAB et des commandes telles que les suivantes pour plus d'informations sur les objets axes: terrain, surf, ligne, bar, polaire, tarte, contour et maille. Voir Fonctions - Par catégorie dans la documentation de référence Fonction MATLAB pour une liste complète. [03]
Panel		Panneaux organiser composants GUI en groupes. En regroupant visuellement les contrôles connexes, les panneaux peuvent rendre l'interface utilisateur plus facile à comprendre. Un panneau peut avoir un titre et plusieurs frontières. les enfants du panneau peuvent être contrôles de l'interface et des axes ainsi que des groupes de boutons et d'autres panneaux. La position de chaque composant dans un panneau est interprétée par rapport au panneau. Si vous déplacez le panneau, ses enfants se déplacent avec elle et maintiennent leurs positions sur le panneau. [03]
Button Group		groupes de boutons sont comme des panneaux, mais sont utilisés pour gérer le comportement de sélection exclusive pour les

		boutons radio et boutons à bascule. [05]
ActiveX Component		composants ActiveX vous permettent d'afficher les contrôles ActiveX dans votre interface graphique. Ils ne sont disponibles que sur la plate-forme Microsoft Windows. Un contrôle ActiveX peut être l'enfant que d'une figure, à savoir, de l'interface elle-même. Il ne peut pas être l'enfant d'un groupe de panneaux ou sur le bouton. [03]

Tableau (I.2): les composants en GUI

I.3.5.La fenêtre Inspecteur des propriétés (Property Inspector) :

Utilisez la propriété Tag pour affecter chaque composant un identifiant de chaîne significative unique. (En double-cliquant sur l'élément ou ) [03]



I.4-simulation en matlab (Simulink):

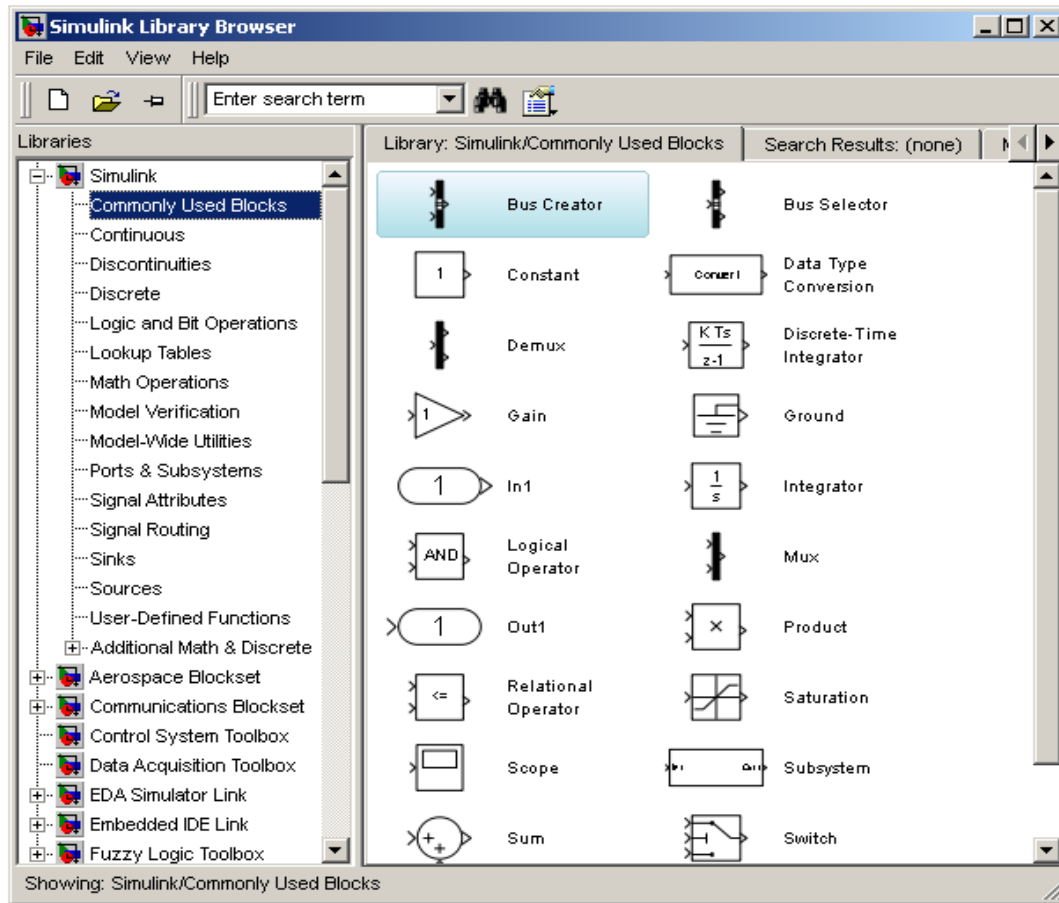
Simulink est un logiciel muni d'une interface graphique pour la modélisation, la simulation et l'analyse des systèmes dynamiques. Etant intégré à MATLAB, les deux environnements sont parfaitement compatibles et les différentes fonctionnalités de ce dernier sont alors directement accessibles. Simulink est basé sur une interface graphique qui permet une construction aisée et conviviale de schémas-blocs. Chaque bloc composant le système est sélectionné depuis un ensemble de bibliothèques prédéfinies[02].

I.4.1.Création d'un modèle

Simulink peut être lancé depuis l'environnement de MATLAB

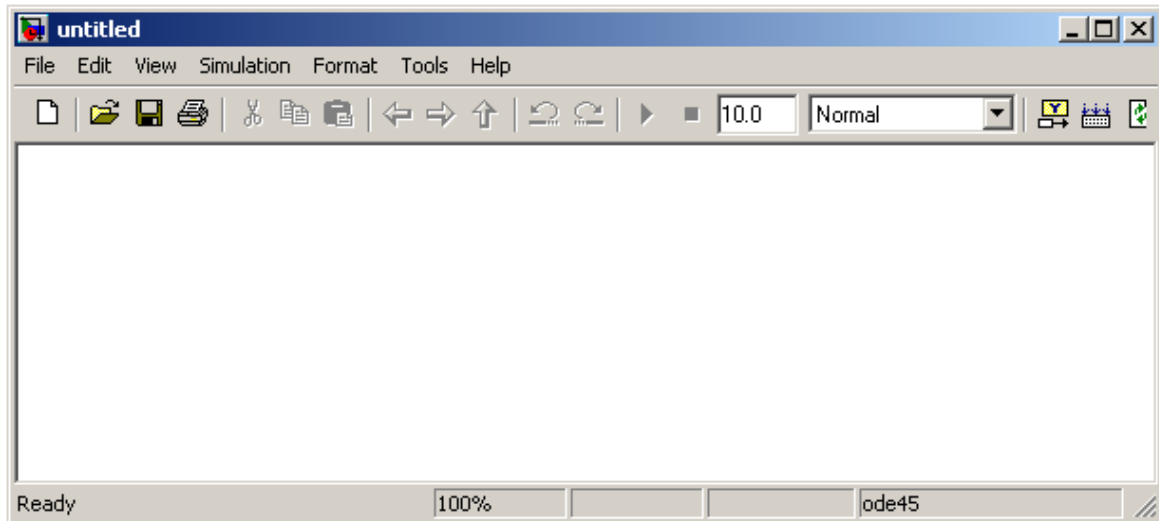
- en cliquant, dans la barre d'outils, sur le bouton 
- ou en tapant simulink

La fenêtre suivante apparaît

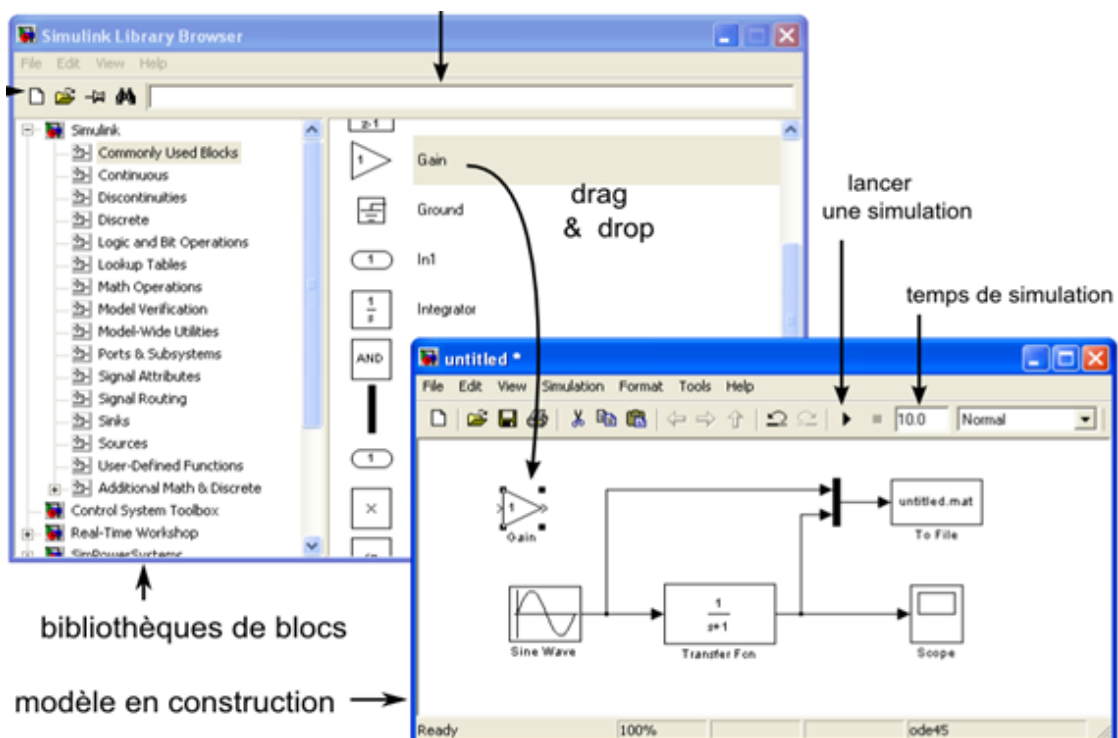


La colonne de gauche liste les bibliothèques disponibles. Celles-ci sont rangées par disciplines (Automatique, Aerospace, Communication, Traitement du signal, Système embarqués...) et regroupent un ensemble de blocs fonctionnels liés à une catégorie de fonctions particulières. A la sélection d'une bibliothèque, les blocs qui la composent sont affichés dans la partie de droite.

Un nouveau modèle est créé en cliquant sur l'icône « feuille blanche » (ou dans le menu File > New > Model). Une fenêtre vide apparaît, elle va servir de support pour construire votre modèle[02].



L'interface est intuitive et facile à utiliser, un modèle est construit à partir des blocs fonctionnels par glisser-déposé. Les blocs sont ensuite interconnectés par des flèches dessinées à l'aide de la souris (cliquer-maintenir sur une entrée ou une sortie puis relâcher sur une entrée ou une sortie). Chaque bloc peut être édité (réglages de ses paramètres) en double-cliquant dessus. Cette dernière action permet d'ouvrir une fenêtre de dialogue dans laquelle une description du bloc et des champs paramétrables sont donnés[02].

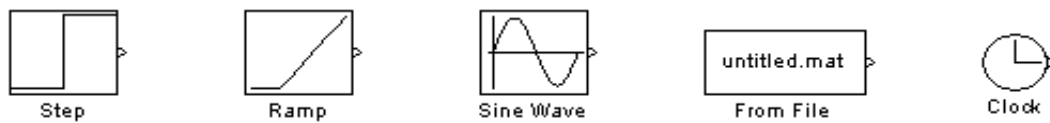


I.4.2. Quelques bibliothèques

Simulink possède de nombreuses bibliothèques, adaptées pour la modélisation d'une multitude de systèmes de différentes natures. De plus, l'ajout de toolboxes supplémentaires permet d'enrichir les librairies et les fonctionnalités du logiciel. Nous présentons dans ce paragraphe seulement quelques blocs simples et présents dans la version de base de Simulink. Nous allons voir quelques blocs standards, souvent utilisés pour la simulation de systèmes asservis LTI[02].

I.4.2.1. Sources

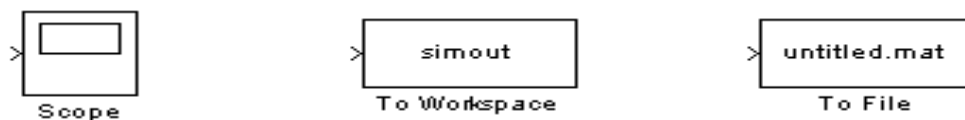
Les sources sont des blocs possédant une ou plusieurs sorties et aucune entrée. Ils sont utilisés pour la génération de signaux[02]



Signal échelon -- signal rampe -- signal sinusoïdal -- valeurs fournies par un fichier -- temps.

I.4.2.2. Sinks

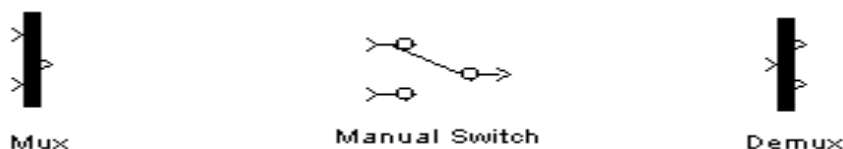
Les blocs de cette librairie, à l'inverse des sources, ne possèdent qu'une ou plusieurs entrées (sin signifie lavabo, c'est-à-dire qui collecte le flux d'information). Ils sont utilisés pour l'affichage (digital, oscilloscope) ou la mémorisation de signaux (vers une variable, un fichier) [02].



Affichage type oscilloscope -- stockage dans une variable--stockage dans un fichier.

I.4.2.3. Signal Routing

Ensemble de blocs utiles pour l'aiguillage de signaux ou la connexion de blocs.



Multiplexeur: combine deux entrées en une seule (vecteur) ; switch: permet de sélectionner manuellement une entrée; démultiplexeur: sépare une entrée (vecteur) en plusieurs composantes[02].

I.5.Fichiers Fonction :

Le principe d'une fonction est d'effectuer des opérations à partir d'une ou plusieurs entrées et fournir une ou plusieurs sorties (résultat). Les variables d'entrées sont des paramètres à spécifier en argument de la fonction, tandis que les variables de sorties sont des valeurs quelle renvoie. Un m-file fonction est tout à fait semblable aux fonctions intégrées de Matlab.

```
>> taille = length(tab);
```

Un m-file est défini comme une fonction en plaçant en tête du fichier le mot clé fonction suivi de son prototype. Un prototype est de la forme

```
[s1,s2,...] = nomfonction(e1,e2,...)
```

Le membre de gauche regroupe les sorties renvoyées par la fonction et les variables entre parenthèses sont les entrées. Le nom du fichier doit être identique au nom de la fonction. Ecrivons dans l'éditeur notre fonction qui permet de calculer l'aire et le volume d'une sphère pour un rayon donné[02].

I.6.Opérateurs relationnels et logiques

Comme dans tout langage de programmation, Matlab possède des opérateurs qui permettent d'établir des expressions renvoyant en résultat une valeur logique, c'est-à-dire 0 ou 1. Ces expressions logiques sont généralement utilisées dans les structures de contrôle présentées dans la prochaine section[02].

I.6.1.Opérateurs relationnels :

Ces opérateurs comparent deux opérandes de même dimension :

- == égal à
- ~= différent de
- >strictement supérieur à
- >= supérieur ou égal à
- <strictement inférieur à
- <= inférieur ou égal à[02]

I.6.2.Opérateurs logiques

Ces opérateurs effectuent un test logique entre deux variables logiques de même dimension[02]:

& et

| ou
~ non
xor ou exclusif
any(x) retourne 1 si un des éléments de x est non nul
all(x) retourne 1 si tous les éléments de x sont nuls

I.7.Structures de contrôle

Dans sa forme la plus simple, le déroulement d'un programme est linéaire dans le sens où les instructions qui le composent s'exécutent successivement. Les structures de contrôle sont des mécanismes qui permettent de modifier la séquence d'exécution des instructions. Plus précisément, lors de l'exécution, en fonction des conditions réalisées certaines parties précises du code seront exécutées. [02]

I.7.1.Branchement conditionnel (if ... elseif ... else)

Cette structure permet d'exécuter un bloc d'instructions en fonction de la valeur logique d'une expression. Sa syntaxe est :

```
If    expression  
      Instructions ...  
End
```

L'ensemble des instructions *instructionsest* exécuté seulement si *expressionest* vraie. Plusieurs tests exclusifs peuvent être combinés.

```
If    expression1  
      instructions1 ...  
elseif expression2  
      instructions2 ...  
else  
      instructions3 ...  
end
```

Plusieurs elseif peuvent être concaténés. Leur bloc est exécuté si l'expression correspondante est vraie et si toutes les conditions précédentes n'ont pas été satisfaites. Le bloc *instruction3* associé au else est quant à lui exécuté si aucune des conditions précédentes n'a été réalisée. [02]

I.7.2.Branchement multiple (switch ... case)

Dans cette structure, une expression numérique est comparée successivement à différentes valeurs. Dès qu'il y a identité, le bloc d'instructions correspondant est exécuté. Sa syntaxe est :

```
Switch expression
```

```
Case valeur1,  
                                instructions1 ...  
  
case valeur2,  
                                instructions2 ...  
  
case valeur3,  
                                instructions3 ...  
  
.....  
otherwise  
                                Instructions ...  
  
end
```

L'expression testée, *expression*, doit être un scalaire ou une chaîne de caractère. Une fois qu'un bloc instruction est exécuté, le flux d'exécution sort de la structure et reprend après le *end*. Si aucune case vérifie l'égalité, le bloc qui suit *otherwise* est exécuté. [02]

I.7.3. Boucle conditionnelle (**while...**)

Ce mécanisme permet de répéter une série d'instructions tant qu'une condition est vérifiée.

Sa syntaxe est :

```
While expression  
    Instructions ...  
End
```

Le terme *expression* est une expression logique. Si cette dernière est vraie, le bloc instructions est exécuté. Puis, *expression* est de nouveau testée. L'exécution du bloc est répétée tant que le test est vrai. [02]

I.7.4. Boucle itérative (**for ...**) :

Cette boucle exécute le bloc interne autant de fois que spécifié par une variable jouant un rôle de compteur. Sa syntaxe est :

```
For variable = debut:increment:fin  
    Instructions ...  
End
```

Le compteur *variable* est initialisé à la valeur *debut* et évolue jusqu'à la valeur *fin* par pas de *increment*. A chaque itération, le bloc instructions est exécuté. Généralement, *variable* est un scalaire, et souvent un entier. [02]