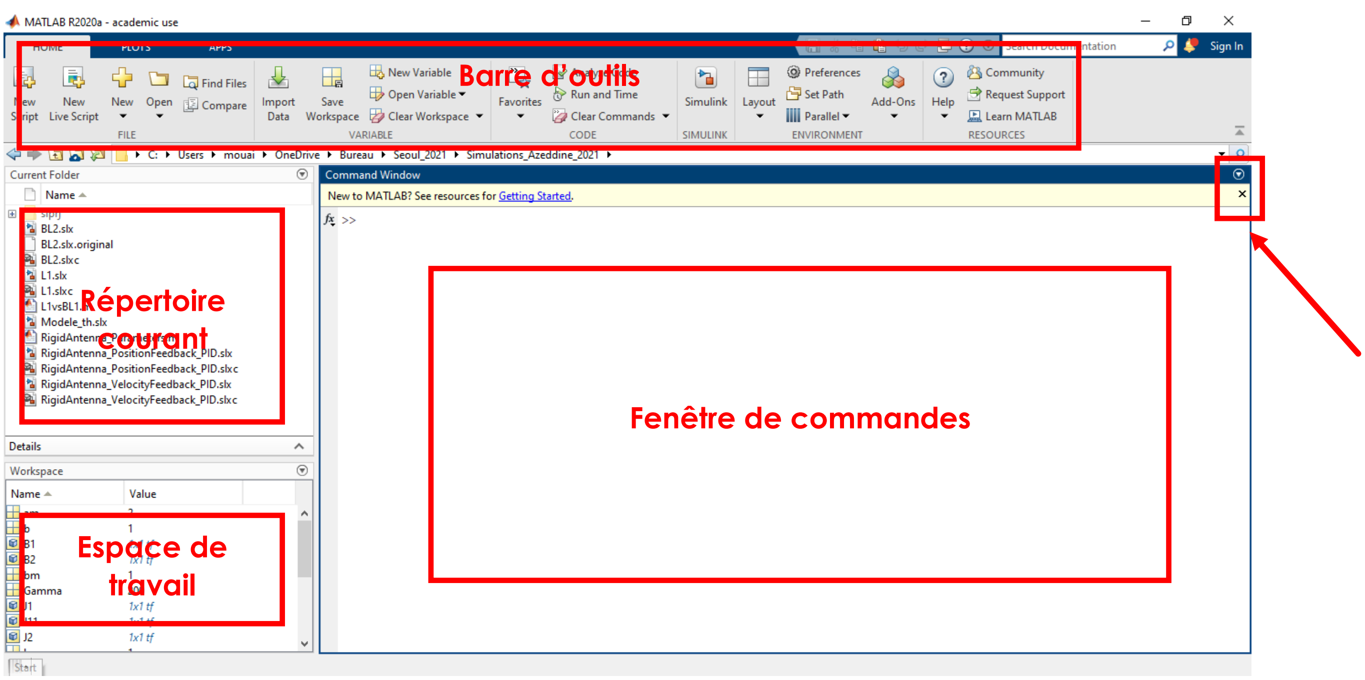
ANNEXE : OUTILS DE SIMULATION

Dans cette annexe, nous présentons les outils de simulation suivants : MATLAB, SIMULINK et GEOGEBRA. Chacun de ces outils offre au concepteur des illustrations convaincantes et souvent complémentaires du comportement des systèmes de rétroaction.

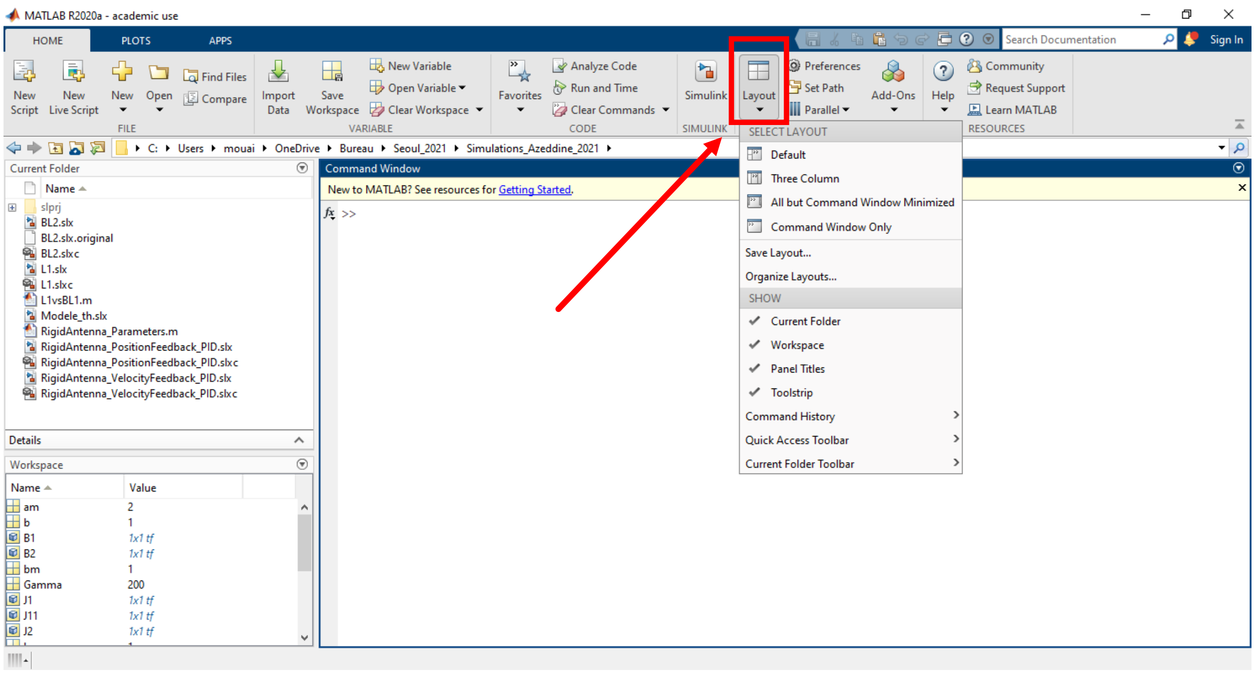
1- MATLAB (*Matrix Laboratory*)

1-1 Interface principale

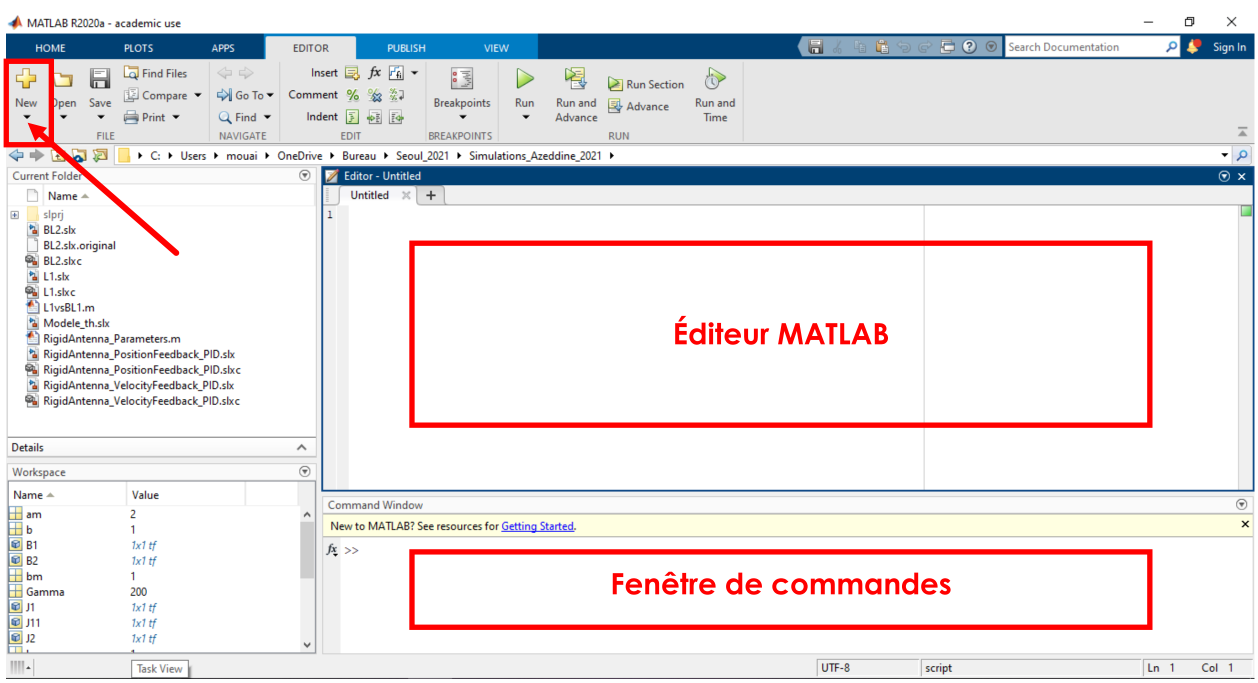
Par défaut, l'interface principale contient une barre d'outils et trois fenêtres: Command Window (fenêtre de commandes en cours), Current directory (répertoire courant) et Workspace (espace de travail). La fenêtre de commandes est utilisée pour communiquer avec le programme MATLAB, en exécutant des instructions de différents types appelées commandes en mode interactif. Dans cette fenêtre, MATLAB affiche l'invité >> (en haut à gauche) pour indiquer qu'il est prêt à recevoir des instructions. Les instructions sont exécutées en tapant sur la touche Enter. La fenêtre Répertoire courant est utilisée pour gérer et/ou accéder aux fichiers ayant l'extension .m qui sont les fichiers exécutables d'un programme particulier. La fenêtre Espace de travail affiche les variables créées dans la fenêtre de commande et leurs dimensions.



Pour éliminer une fenêtre, il suffit de cliquer sur le bouton x dans le coin supérieur droit de la fenêtre. D'autres options existent en cliquant sur la flèche située au-dessus du bouton **x**. L'interface peut être personnalisée pour contenir d'autres fenêtres, en cliquant sur le bouton **Layout** et en choisissant d'afficher et/ou de cacher des fenêtres. Pour restaurer la configuration par défaut, on sélectionne **Default**. Visualisez ces possibilités en appuyant sur le bouton **Layout**.



Un programme est constitué de nombreuses commandes. Les programmes sont écrits dans l'éditeur MATLAB. La fenêtre de l'éditeur apparait en cliquant sur le bouton **New Script**. Les programmes édités sont enregistrés dans des fichiers avec l'extension **.m**. Des commentaires peuvent être ajoutés pour faciliter la compréhension des lignes du programme. Ces commentaires sont précédés de % et ne sont pas exécutés. Par défaut, la couleur du texte des commentaires est verte.



Exemples :

```
% initiation à MATLAB
x=2; % assigner une valeur 2 à une variable x
y=x+1 % assigner une valeur x+1 à une variable y.

% calcul le cosinus de la racine carrée
x = sqrt(13:3:25); % calcul la racine carrée de x qui varie de
                   % 13 à 25 avec un pas de 3
y = cos(x) % calcul le cosinus de x
```

1-2 Opérations sur les scalaires

Symbole	**Opération**	**Forme MATLAB**
^	Puissance : a^b	a^b
*	Multiplication : ab	a*b
/	Division droite : $a/b = \frac{a}{b}$	a/b
\	Division gauche : $a\backslash b = \frac{b}{a}$	a\b
+	Addition : $a + b$	a+b
-	Soustraction : $a - b$	a-b

Les opérations mathématiques suivent un ensemble de règles appelées priorités.

Priorité	**Opération**
1	Parenthèses : évaluées en commençant par la paire la plus interne
2	Puissance : évaluée de gauche à droite
3	Multiplication et division avec une priorité égale, évaluées à partir de la gauche vers la droite
4	Addition et soustraction avec une priorité égale, évaluées à partir de la gauche vers la droite

Autres fonctions :

Fonction	Forme MATLAB
Exponentiel : e^x	exp(x)
Racine carrée : $\sqrt{x}$	sqrt(x)
Logarithme base e : lnx	log(x)
Logarithme base 10 : $logx$	log10(x)
Cosinus : $cosx$	cos(x)
Sinus : $sinx$	sin(x)
Tangente : $tanx$	tan(x)
Cosinus inverse : $cos^{-1}x$	acos(x)
Sinus inverse : $sin^{-1}x$	asin(x)
Tangente inverse : $tan^{-1}x$	atan(x)

Les fonctions trigonométriques utilisent x en radians. Pour utiliser x en degrés les fonctions trigonométriques deviennent sind(x), cosd(x) et tand(x). Les fonctions inverses acosd(x), asind(x) et atan(x) renvoient à des valeurs en degrés.

MATLAB octroie la réponse la plus récente d'un calcul après le mot **ans**, qui est l'abréviation anglaise du mot réponse. Une variable dans MATLAB est un symbole utilisé pour lui attribuer une valeur. Le résultat suivant la variable **ans** peut être utilisé pour d'autres calculs. Des variables peuvent être choisies pour écrire des expressions mathématiques plutôt que d'utiliser la variable par défaut **ans**. Pour exécuter sans afficher la réponse, l'instruction est terminée par un point-virgule.

Exemples :

Dans la fenêtre de commandes, ces opérations peuvent être exécutées:

```
>>8 + 3*5
ans =
23
>>x=(8 + 3)*5
x =
55
>>4^2 - 12 - 8/4*2
ans =
0
```

```
>>y=4^2 - 12 - 8/(4*2)
y =
3
>>3*4^2 + 5
ans =
53
>>(3*4)^2 + 5
ans =
149
>>27^(1/3) + 32^(0.2)
ans =
5
>>27^(1/3) + 32^0.2
ans =
5
>>z=27^1/3 + 32^0.2;
>>
```

1-3 Opérations sur les nombres complexes

MATLAB gère automatiquement l'algèbre des nombres complexes. L'addition, la soustraction, la multiplication et la division de nombres complexes sont facile à faire.

Exemples:

```
>>s = 3+7i;w = 5-9i;
>>w+s
ans =
8.0000 - 2.0000i
>>w*s
ans =
78.0000 + 8.0000i
>>w/s
ans =
-0.8276 - 1.0690i
```

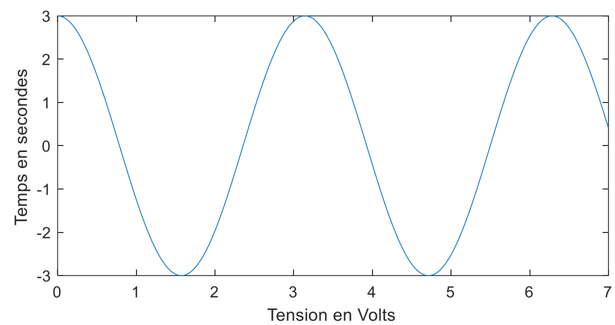
1-4 Traçage de courbes

MATLAB contient de nombreuses fonctions pour créer des tracés de plusieurs types, tels que les tracés rectilignes, logarithmiques, de surface et de contour. La fonction **plot (x,y)** génère un graphique avec les valeurs x sur l'axe horizontal et les valeurs y sur l'axe vertical. Pour spécifier les axes x et y, par exemple tension en Volts vs temps en secondes, ajouter les fonctions **xlabel** et **ylabel**.

Exemple:

L'exemple qui suit, représente le graphe d'une fonction y qui dépend de la variable x dont les valeurs varient entre 0 et 7 avec un pas de 0.01.

```
% Exemple plot
x = 0:0.01:7;
y = 3*cos(2*x);
plot(x,y)
xlabel('Tension en Volts')
ylabel('Temps en secondes')
```

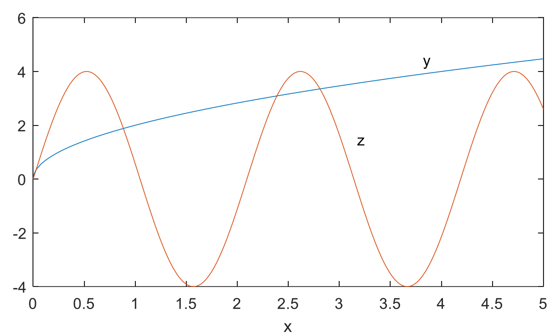


MATLAB peut créer plusieurs tracés superposés sur le même graphe. Pour définir les tracés, la fonction **gtext('x')** est ajoutée.

Exemple:

L'exemple qui suit, représente le graphe des fonctions y et z dont les valeurs de x varient entre 0 et 5 avec un pas de 0.01.

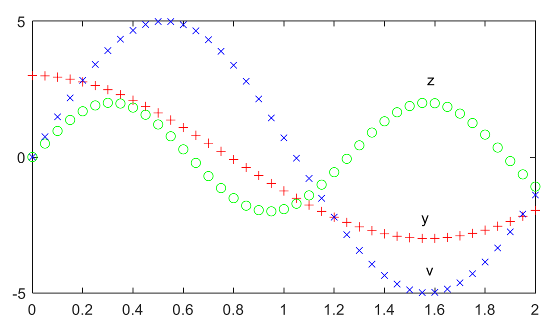
```
% Exemple plot supperposée
x = 0:0.01:5;
y = 2*sqrt(x);
z = 4*sin(3*x);
plot(x,y,x,z),xlabel('x'),gtext('y'),gtext('z')
```



La fonction **plot** peut être personnalisée par plusieurs paramètres. La commande **help** plot figurant dans la fenêtre de commandes, donne plus de détails.

Exemple :

L'exemple qui suit, représente le graphe des fonctions y, z et v dont les valeurs de x varient entre 0 et 2 avec un pas de 0.05.



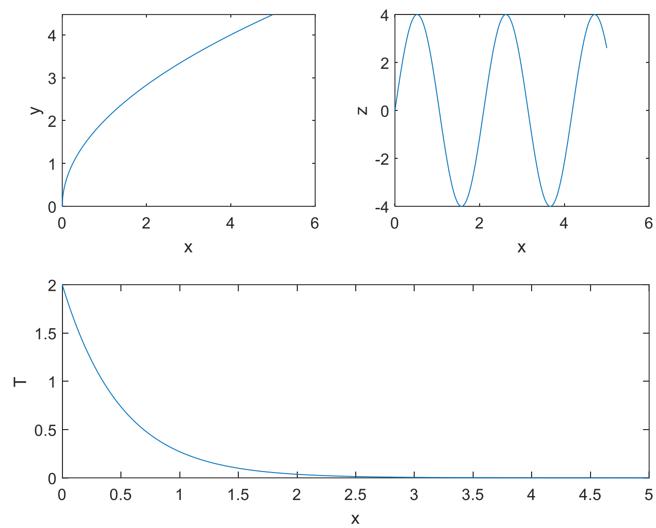

```
% Exemple plot paramétrée
x = 0:0.05:2;
y = 3*cos(2*x);
z = 2*sin(5*x);
v = 5*sin(3*x);
plot(x,y,'r+',x,z,'go',x,v,'bx'),xlabel('x'),gtext('y'),gtext('z'),gtext('v')
```

MATLAB peut créer plusieurs tracés adjacents, en incluant la fonction **subplot**.

Exemple :

L'exemple qui suit, représente le graphe des fonctions y, z et v dont les valeurs de x varient entre 0 et 5 avec un pas de 0.01.

```
% Exemple courbes adjacentes
x = 0:0.01:5;
y = 2*sqrt(x);
z = 4*sin(3*x);
v = 2*exp(-2*x);
subplot (2,2,1), plot(x,y),xlabel('x'), ylabel('y')
subplot (2,2,2), plot(x,z),xlabel('x'), ylabel('z')
subplot (2,2,3:4), plot(x,v),xlabel('x'), ylabel('v')
```

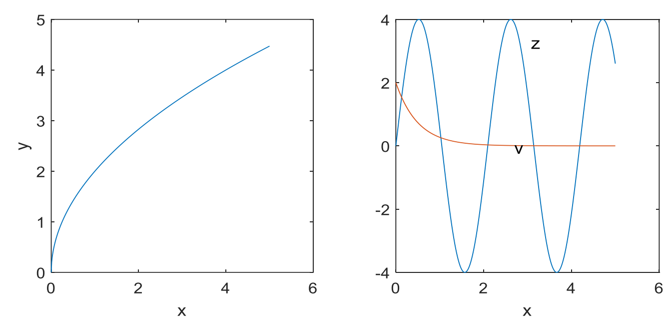


Au sein d'un **subplot**, il est possible d'introduire plusieurs graphes superposés.

Exemple :

L'exemple qui suit, représente le graphe des fonctions y, z et v dont les valeurs de x varient entre 0 et 5 avec un pas de 0.01.

```
% Exemple courbes adjacentes et superposées
x = 0:0.01:5;
y = 2*sqrt(x);
z = 4*sin(3*x);
v = 2*exp(-2*x);
subplot (1,1,1), plot(x,y),xlabel('x'), ylabel('y')
subplot (1,1,2), plot(x,z,x,v),xlabel('x'),
gtext('z'),gtext('v')
```



1-5 Opérations sur les éléments de vecteurs et de matrices

Pour créer un vecteur ligne dans MATLAB, il suffit de définir les éléments à l'intérieur d'une paire de crochets, en les séparant par un espace ou une virgule. Pour créer un vecteur colonne, séparer les éléments par des points-virgules. Pour créer un vecteur ligne, puis utiliser la notation de transposition ', qui convertit le vecteur ligne en vecteur colonne, ou vice versa.

Exemples:

```
>>g = [3;7;9]
g =
```

```
3
7
9
>>h = [3,7,9]
h =
3 7 9
>>g' = [3;7;9]'
g' =
3 7 9
>>h' = [3,7,9]'
h' =
3
7
9
```

Pour créer une matrice, il suffit de séparer les éléments d'une même ligne par des espaces ou des virgules et de séparer les lignes successives avec des points-virgules.

Exemple :

```
>>A = [2,4,10;16,3,7]
A =
   2   4   10
  16   3    7
```

Un vecteur peut être généré en regroupant un vecteur ligne à un autre vecteur ligne pour créer un troisième vecteur ligne. Une matrice peut être générée à partir de deux vecteurs (les deux vecteurs doivent avoir le même nombre de colonnes).

Exemples:

```
>>a = [1,3,5];
>>b = [7,9,11];
>>c = [a,b]
c =
1 3 5 7 9 11
>> d = [a;b]
d =
1 3 5
7 9 11
```

L'opération **transpose** intervertit les lignes et les colonnes. Dans les formules mathématiques, cette opération est dénotée par l'exposant **T**. Pour une matrice A de dimensions (mxn), avec m lignes et n colonnes, A^T (lire «A transpose») est une matrice de dimensions (nxm).

Exemples:

```
A=[1 3 5 ; 7 9 11]
A_transpose=A'
```

```
A =
    1    3    5
    7    9   11
A_transpose =
    1    7
    3    9
    5   11
```

Sélection des éléments d'un vecteur ou d'une matrice

MATLAB peut sélectionner des éléments d'un vecteur ou d'une matrice. Les notations suivantes sont utilisées.

- v(:) représente tous les éléments de ligne ou de colonne du vecteur v.
- v(2:5) représente du deuxième au cinquième élément; c'est-à-dire v(2), v(3), v(4), v(5)
- A(1,3) représente l'élément de la ligne 1 et la colonne 3.
- A(:,3) désigne tous les éléments de la troisième colonne de la matrice A.
- A(3,:) désigne tous les éléments de la troisième ligne de A.
- A(:,2:5) désigne tous les éléments de la deuxième à la cinquième colonne de A.
- A(2:3,1:3) désigne les éléments de la deuxième à la troisième ligne qui se retrouvent dans les éléments de la première à la troisième colonne.
- v=A(:) crée un vecteur v constitué de toutes les colonnes de A empilées du premier au dernier.
- A(end,:) désigne la dernière ligne de A et A(:,end) désigne la dernière colonne.

Exemples:

```
>>B = [2,4,10,13;16,3,7,18;8,4,9,25;3,12,15,17];
>>C = B(2:3,1:3);
B =  2    4   10   13
    16    3    7   18
     8    4    9   25
     3   12   15   17
C =
    16    3    7
     8    4    9
>>D = B(end,:);
D =
   3    12    15    17
>>H = B(:,end);
H =
    13
    18
    25
    17
>>B(1,3);
```

```
ans =
   10
```

Pour faire des opérations élément par élément, ces notations sont utilisées. Pour préciser qu'il s'agit d'opération entre éléments de matrices, un point est ajouté avant l'opérateur (*, /, ^) comme suit :

Opération	**Forme**	**Exemple**
Addition d'un scalaire	A+b	[6,3]+2=[8,5]
Soustraction d'un scalaire	A-b	[8,3]-5=[3,-2]
Addition de deux matrices	A+B	[6,5]+[4,8]+[10,13]
Soustraction de deux matrices	A-B	[6,5]-[4,8]=[2,-3]
Multiplier deux matrices AB	A.*B	[3,5].*[4,8]=[12,40]
Division droite AB^{-1}	A./B	[2,5]./[4,8]=[2/4,5/8]
Division à gauche $B^{-1}A$	A.\B	[2,5].\[4,8]=[2\4,5\8]
Puissance A^b	A.^b	[3,5].^2=[3^2,5^2]
		2.^[3,5]=[2^3,2^5]
		[3,5].^[2,4]=[3^2,5^4]

Racines polynomiales

Un polynôme peut être défini dans MATLAB par un tableau dont les éléments sont les coefficients du polynôme, en commençant par le coefficient de la puissance la plus élevée de x. Les racines du polynôme sont les valeurs de x qui annulent la valeur du polynôme. Les racines polynomiales sont calculées en utilisant la fonction **roots**.

Exemple :

Trouver les racines de $x^3 - 7x^2 + 40x - 34 = 0$.

```
% Exemple racine de polynôme
f = [1,-7,40,-34];
roots(f)
ans =
3.0000 + 5.000i
3.0000 - 5.000i
1.0000
```

Les deux commandes auraient pu être combinées en une seule commande :

```
roots([1,-7,40,-34])
```

1-6 Opérations sur les vecteurs et les matrices

L'addition et la soustraction de matrice sont identiques à celles utilisées en additionnant ou en soustrayant élément par élément. Cela est possible si les matrices ont des dimensions égales.

La multiplication de deux matrices est possible si le nombre de colonnes de la première matrice au nombre de rangées de la seconde. Ainsi, le résultat de la multiplication d'une matrice mxn par une matrice nxp est une matrice mxp.

Exemple :

```
% multiplier deux matrices
A = [6,-2;10,3;4,7];
B = [9,8;-5,12];
A*B
ans =
   64    24
   75   116
    1   116
```

Matrices particulières :

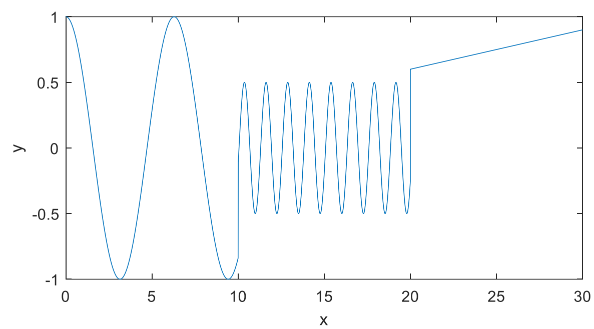
Fonction	Forme MATLAB
Matrice identité nxn	eye(n)
Matrice identité de même dimension que la matrice A	eye(size(A))
Matrice nxn avec tous les éléments égaux à 1	ones(n)
Matrice mxn avec tous les éléments égaux à 1	ones(m,n)
Matrice de même dimension que la matrice A avec tous les éléments égaux à 1	ones(size(A))
Matrice nxn avec tous les éléments égaux à 0	zeros(n)
Matrice mxn avec tous les éléments égaux à 0	zeros(m,n)
Matrice de même dimension que la matrice A avec tous les éléments égaux à 0	zeros(size(A))
Matrice diagonale ayant les valeurs d'un vecteur V	diag(V)

Exemple :

Tracer la courbe de la fonction suivante :

$$f(x) = \begin{cases} f_1(x) = \cos x & 0 \leq x \leq 2 \\ f_2(x) = sin2x & 2 \leq x \leq 5 \\ f_3(x) = e^{-3x} & 5 \leq x \leq 10 \end{cases}$$

```
% Multiplier deux matrices f(x)*x
x1 = 0:0.01:10;
f1 = cos(x1).*ones(size(x1));
x2 = 10.01:0.01:20;
f2 = 0.5*sin(5*x2).*ones(size(x2));
x3 = 20:0.01:30;
f3 = 0.03.*x3.*ones(size(x3));
f = [f1, f2, f3];
x = [x1, x2, x3];
plot(x,f),xlabel('x'),ylabel('y')
```



Équations algébriques linéaires

L'opérateur de division gauche \ est utilisé dans MATLAB pour résoudre un système d'équations algébriques.

Exemple :

$$\begin{cases} 6x + 12y + 4z = 70 \\ 7x - 2y + 3z = 70 \\ 2x + 8y - 9z = 70 \end{cases}$$

$X = [x; y; z] = A^{-1}b, b == [70; 70; 70]$

```
% Exemple solution système équations A*X=b, X=[x;y;z]= A\b =inv(A)*b
A = [6,12,4;7,-2,3;2,8,-9];
b = [70;5;64];
Solution = A\b
Solution =
3
5
-2
```

Puissance d'une matrice

Élever une matrice à une puissance équivaut à multiplier à plusieurs reprises la matrice par elle-même. Ce processus est possible si la matrice est carrée. MATLAB utilise le symbole A^ pour représenter l'exponentiation de la matrice.

Autres fonctions :

Opération	**Forme**
Addition A+B	A+B
Soustraction A-B	A-B
Matrice inverse A^{-1}	inv(A)
Matrice conjuguée A^*	conj(A)
Pseudo-inverse de A : $A^\dagger$	pinv(A)
Exponentiel d'une matrice e^A	expm(A)
Racine carrée d'une matrice $\sqrt{A}$	sqrtm(A)
Concaténation horizontale de deux matrices $[A\ B]$	[A,B]
Concaténation verticale de deux matrices $\begin{bmatrix} A \\ B \end{bmatrix}$	[A;B]
Déterminant d'une matrice A	det(A)
Valeurs propres et vecteurs propres d'une matrice A	eig(A)
Rang d'une matrice A	rank(A)
Trace d'une matrice A	trace(A)
Valeurs singulières d'une matrice A	svd(A)
La norme de la matrice A	norm(A)
Polynôme caractéristique d'une matrice A	poly(A)

Autres fonctions de sélection d'éléments

Fonction	**Forme**
La taille d'un vecteur V	length(V)
La valeur maximale d'un vecteur V	max(V)
La valeur minimale d'un vecteur V	min(V)
La valeur moyenne d'un vecteur	mean(V)
La somme des éléments d'un vecteur V	sum(V)

Le produit des éléments d'un vecteur V	prod(A)
Rangement des éléments d'un vecteur V dans l'ordre croissant	sort(V)
Les dimensions d'une matrice A	size(A)
Extraction de la matrice triangulaire supérieure de la matrice A	triu(A)
Extraction de la matrice triangulaire inférieure de la matrice A	tril(A)

Exemples :

```
% Transposé d'une matrice. Les éléments de A et de B vérifient : a_ij = b_ji
A = [16   2    3    13;
     5   11   10     8;
     9    7    6    12;
     4   14   15     1];
B=A'
B =

    16     5     9     4
     2    11     7    14
     3    10     6    15
    13     8    12     1


% Déterminent non nul d'une matrice
A = [1     3     7;
     2     4     8;
     5     0     6];
det(A)
inv(A)
ans =
   -32
ans =
   -0.7500    0.5625    0.1250
   -0.8750    0.9063   -0.1875
    0.6250   -0.4688    0.0625


% Déterminent nul d'une matrice
A = [ 16     1.5     3    1.5;
     5    5    10     5;
     9     3     6    3;
     4    0.5    15     0.5];
det(A)
inv(A)
```

```
ans =
    0
ans =
  Inf   Inf   Inf
  Inf   Inf   Inf
  Inf   Inf   Inf
```

% les valeurs propres λ_i et les vecteurs propres e_i d'une matrice vérifient
$\lambda_i I - A = 0$ et $(\lambda_i I - A)e_i = 0$
% Deux représentations sont possibles : e=eig(A) et [M,D] = eig(A)
% e : vecteur des valeurs propres, M : matrice de similarité
% D : matrice diagonale des valeurs propres
% La matrice de similarité
% Dans le cas de valeurs propres distincts, la matrice de similarité M % % % % définie par les vecteurs propres vérifie $M^{-1}AM = \Lambda$ ou Λ représente la % % % % matrice diagonale des valeurs propres.

```
A = [1.0000   0.5000    0.3333    0.2500;
    0.5000    1.0000    0.6667    0.5000;
    0.3333    0.6667    1.0000    0.7500;
    0.2500    0.5000    0.7500    1.0000]

e=eig(A)
e =
    0.2078
    0.4078
    0.8482
    2.5362

[M,D] = eig(A)

M =
    0.0694   -0.4422   -0.8105    0.3778
   -0.3619    0.7420   -0.1877    0.5322
    0.7694    0.0487    0.3010    0.5614
   -0.5218   -0.5015    0.4661    0.5088
D =
    0.2078         0         0         0
         0    0.4078         0         0
         0         0    0.8482         0
         0         0         0    2.5362
```

Décomposition en valeurs singulières

Considérons une matrice A représentant la relation existant être n entrées et m sorties. Cette matrice peut être décomposée ainsi : $A = U\Sigma V^A$

Tel que :

Σ : est la matrice de valeurs singulières de A, soit la racine (positive) des valeurs propres de la matrice $A^A A$.

V : est la matrice orthogonale des vecteurs propres de $A^A A$.

U : est la matrice orthogonale des vecteurs propres de AA^A et vérifie : $HV = U\Sigma$.

U et V sont des matrices orthogonales. Cette décomposition par valeurs singulières est utilisée dans la technologie MIMO.

```
% Exemple: Valeurs singulières
% La valeur singulière d'une matrice σ(A) est définie par la racine carrée
% positive de la valeur propre de la matrice (A^T)*A

A = [1    3    7;
    2    4    8;
    5    0    6;
    2    5    8];
svd(A)
ans =
  16.6221
   4.4365
   1.0116
```

Inverse d'une matrice non carrée

Si la matrice A n'est pas carrée, le pseudo-inverse $A^\dagger$ est utilisé. $A^\dagger = (A^A A)^{-1} A^A$. Si $A^A = A$, la matrice A est hermitienne. Une matrice A est dite hermitienne si elle est égale au conjugué de sa transposé. Si A est une matrice réelle, la matrice hermitienne est une matrice symétrique réelle.

Si A est de dimension (nxm), alors :

$A^\dagger = A^{-1}$ si $n = m$

$$A^\dagger = \left(A^T A\right)^{-1} A^T \text{ si } n < m$$

$$A^\dagger = A^T \left(AA^T\right)^{-1} \text{ si } n > m$$

La fonction **pinv** calcule le pseudo inverse pour une matrice non carrée.

```
% inverse d'une matrice non carrée
A = [1    3    7;
    2    4    8;
    5    0    6;
    2    5    8];
pinv(A)
```

```
ans =
   -0.4755   -0.0356    0.1838    0.3137
   -0.4679    0.0194   -0.1003    0.4652
    0.3912    0.0406    0.0124   -0.2672
```

Résolution d'équations linéaires à plusieurs inconnues :

La fonction ***solve*** permet une telle résolution.

Exemples :

Une inconnue :

```
syms x y
x=solve(2*x==5,x)
y=solve(3*y^2==25,y)
x =
5/2
y =
 -(5*3^(1/2))/3
  (5*3^(1/2))/3
```

Deux inconnues (ou plus):

```
syms a b
[a,b]=solve(a+b==15,a-b==2,a,b)
a = 17/2
b = 13/2
```

1-7 Opérateurs algébriques et logiques

Les opérateurs comparent deux variables dans une expression mathématique. Si l'expression est vraie, la valeur 1 lui est attribuée; autrement, la valeur 0 lui est attribuée. Ci-suivent des opérateurs algébriques utiles.

Opération algébrique	**Forme**
Moins que	<
Inférieur ou égal à	<=
Plus grand que	>
Supérieur ou égal à	>=
Égal à	==
Pas égal à	~=

Ci-suivent également des opérateurs logiques utiles.

Opération algébrique	Forme
Non	~
Et	&
Ou	|

Exemples:

Comparer les valeurs de deux variables A et B

```
>> A = 3;
>> B = 5;
>> A > B
ans =
 0

>> A < = B
ans =
  1
>> (A <=5) & (B >=1)
ans =
  1
```

1-8 Les déclarations d'expressions et des boucles

Comme les phrases en langues naturelles, une déclaration d'expressions est une unité complète d'exécution en terminant par un point-virgule (;).

Exemples :

La déclaration de l'expression attribuant une valeur à une variable:

```
A = 3;
```

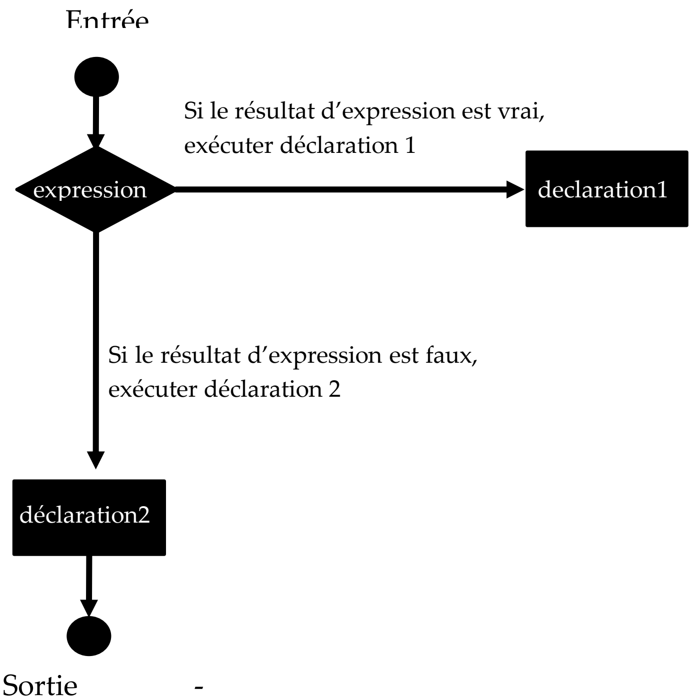
La déclaration de l'expression d'un scalaire:

```
A = A + 1;
```

La déclaration de l'expression d'un appel de fonction:

```
B = sqrt (A);
```

La déclaration de la prise de décision : if-else-end

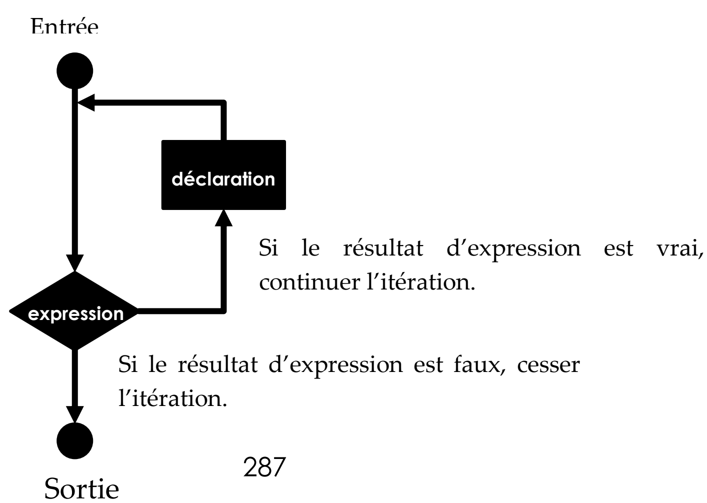


Exemple :

Comparer deux variables A et B.

```
A = 1;
B = 2;
if ( A >= B )
   fprintf('A is bigger.');
else
   fprintf('B is bigger.');
end
B is bigger.
```

La déclaration de boucle : while-end

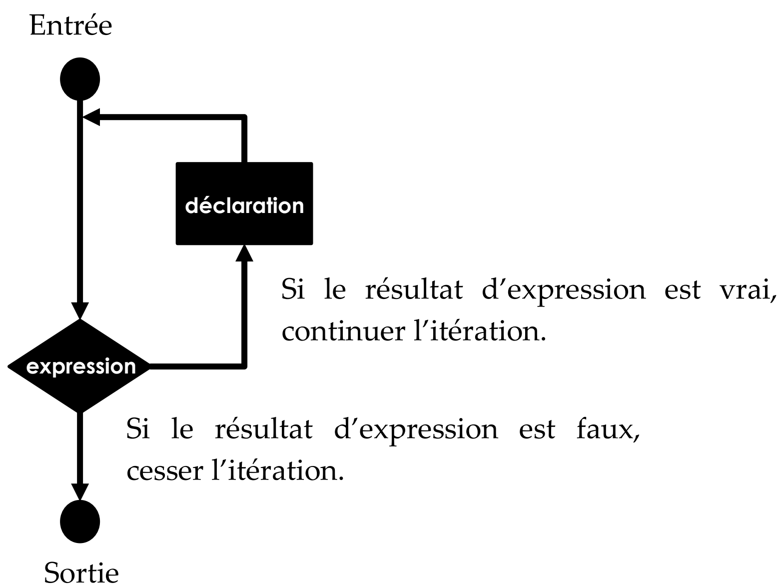


Exemple:

Identifier les nombres inférieurs ou égaux à 5

```
i = 1;
while ( i <= 5 )
   fprintf('value of i: %d\n', i);
   i = i+1;
end
value of i: 1
value of i: 2
value of i: 3
value of i: 4
value of i: 5
```

La déclaration de boucle: for-end



Exemple :

Identifier les nombres inférieurs ou égaux à 5.

```
for i = 1:5
   fprintf('value of i: %d\n', i);
end
value of i: 1
value of i: 2
value of i: 3
value of i: 4
value of i: 5
```

1-9 Fonction de transfert

Pour définir une fonction de transfert d'un système continu ou discret on utilise la fonction **tf** en spécifiant le polynôme du numérateur et celui du dénominateur.

Exemple :

Définir la fonction de transfert suivante:

$$G(s) = \frac{2s+1}{s^2+2s+1}$$

```
% Définir une fonction de transfert continue
G = tf([2 1], [1 2 1])
G =
    2 s + 1
  -------------
  s^2 + 2 s + 1
```

Représentation alternative :

```
% Définir une fonction de transfert continue
tf('s');
G=(2*s+1)/(s^2+2*s+1)
G =
    2 s + 1
  -------------
  s^2 + 2 s + 1
```

Exemple :

Définir la fonction de transfert discrète suivante en précisant le pas d'échantillonnage de 0.1s:

$$G(z) = \frac{6z+5}{z^2+8z+7}$$

```
% Définir une fonction de transfert discrète
G = tf([6 5], [1 8 7],0.1)
G =
    2 z + 1
  -------------
  z^2 + 2 z + 1
```

Une fonction de transfert peut être écrite en utilisant ses zéros, ses pôles et son gain.

Exemples :

```
% Définir une fonction de transfert continue avec zpk
G = tf([6 5], [1 8 7])
```

```
H = zpk(G)
G =
    6 s + 5
  -------------
  s^2 + 8 s + 7
H =
  6 (s+0.8333)
  ------------
  (s+7) (s+1)
```

```
% Définir une fonction de transfert discrète avec zpk
L = zpk([6 5], [1 8 7],1,0.1)
L =
   (z-2) (z-1)
  -------------
  (z-1)^2 (z-2)
```

Réponse temporelle d'une fonction de transfert G(s) ou G(z)

La fonction **impulse(G)** trace la réponse impulsionnelle d'une fonction de transfert G(s) et la fonction **step(G)** trace sa réponse indicielle.

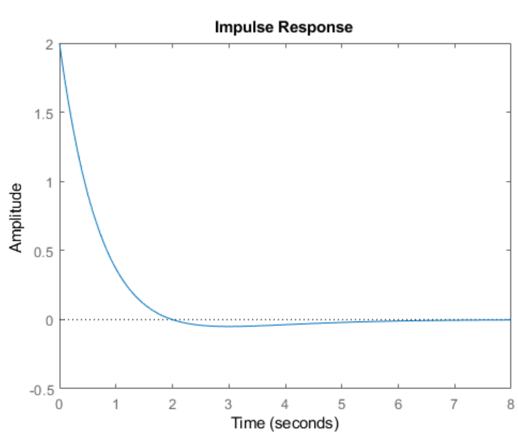
Pour :

$$G(s) = \frac{2s + 1}{s^2 + 2s + 1}$$

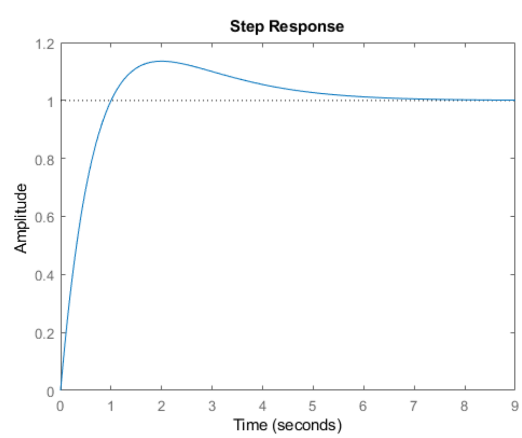
% Réponse impulsionnelle :l'entrée est une fonction de Dirac $\delta(t)$.

% Réponse indicielle :l'entrée Est une fonction échelon $u(t)$.

impulse(G)



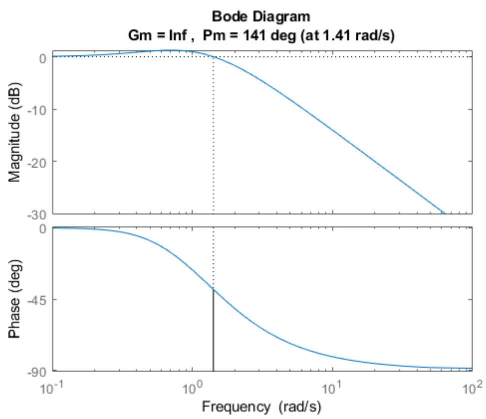
step(G)

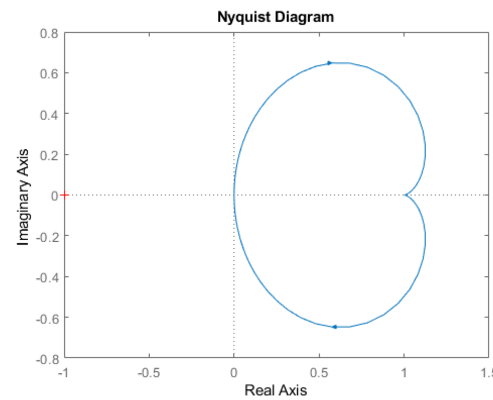


Réponse fréquentielle

Nous établissons une distinction entre la fonction de transfert $G(s)$ et la réponse en fréquence $G(j\omega)$ en fonction de la fréquence. Pour cette dernière, nous nous limitons à l'axe imaginaire s= $j\omega$ plutôt qu'à l'ensemble du plan complexe s. La fonction **bode (G)** trace l'amplitude $A = 20log|G(j\omega|$ en dB et argument $\varphi = \angle G(j\omega)$ en degrés. La fonction **margin(G)** spécifient la marge de gain et la marge de phase. La fonction **Nyquist (G)** trace Le diagramme de Nyquist qui représente la partie réelle et la partie imaginaire de $G(j\omega)$ lorsque ω varie de $-\infty$ à $+\infty$. Le diagramme de Nyquist du gain de boucle d'une boucle à rétroaction à gain unitaire sera utilisé pour étudier la stabilité de la boucle de rétroaction. Le diagramme de Nyquist étudie le tracé de la réponse en fréquence $G(j\omega)$. Toutefois, s'il existe un pôle sur l'axe imaginaire, il faudra le contourner.

Pour : $G(s) = \frac{2s+1}{s^2+2s+1}$



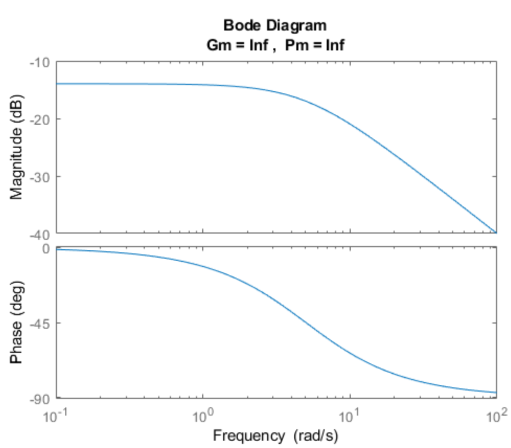


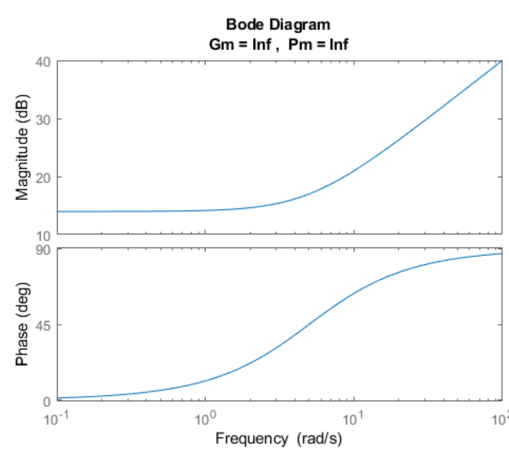
Autres exemples de diagramme de Bode:

Notez le profil de la phase des fonctions de transfert suivantes. Cette connaissance sera utile lors de la conception des contrôleurs PID (Proportionnel-Intégral-Différentiel).

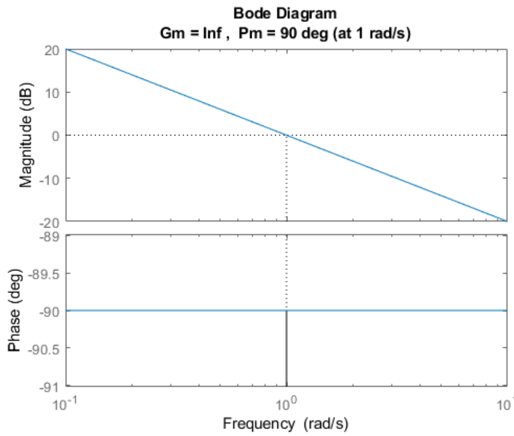
$$G(s) = \frac{1}{s+5}$$

$$G(s) = s + 5$$

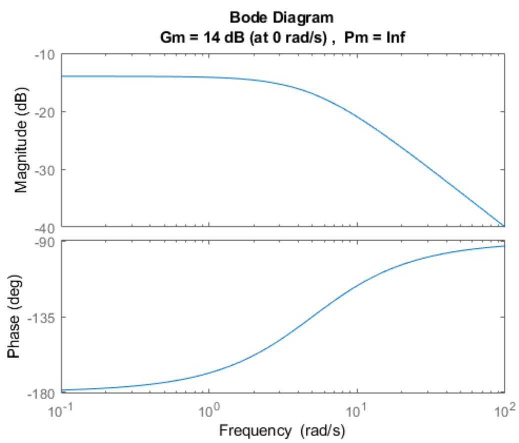




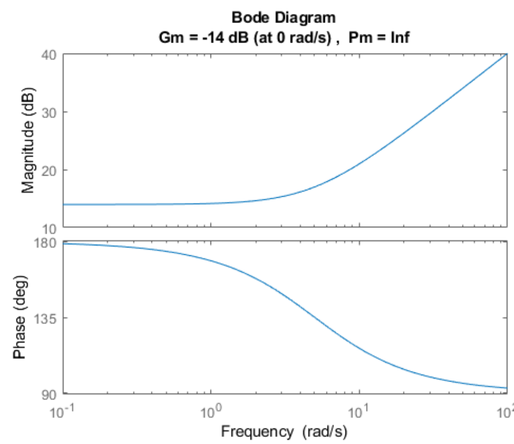
$$G(s) = \frac{1}{s}$$



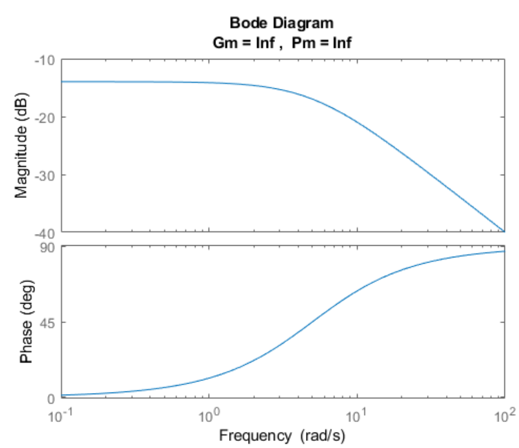
$$G(s) = \frac{1}{s-5}$$



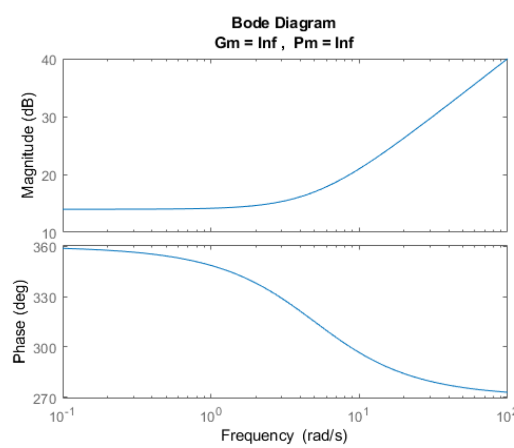
$$G(s) = s - 5$$



$$G(s) = \frac{1}{-s+5}$$



$$G(s) = -s + 5$$

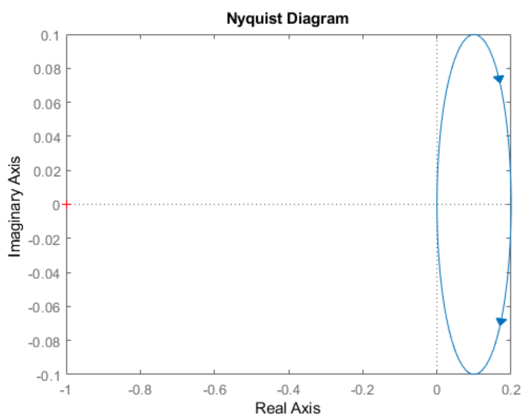


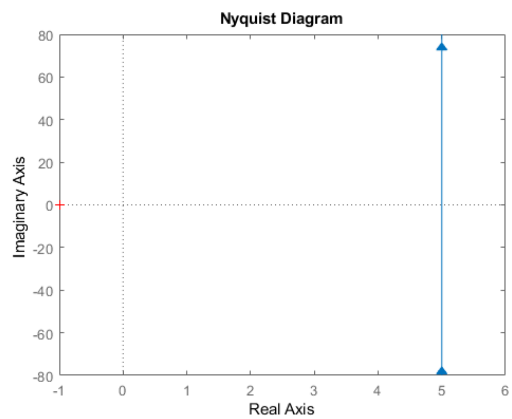
Autres exemples de diagrammes polaires:

Les diagrammes polaires représentent les coordonnées polaires de la fonction de transfert. Ce diagramme polaire est souvent dénoté plan de Nyquist. Rappelons que le critère de Nyquist ne s'applique que lors que la fonction de transfert représente le gain de boucle dans une boucle de rétroaction unitaire.

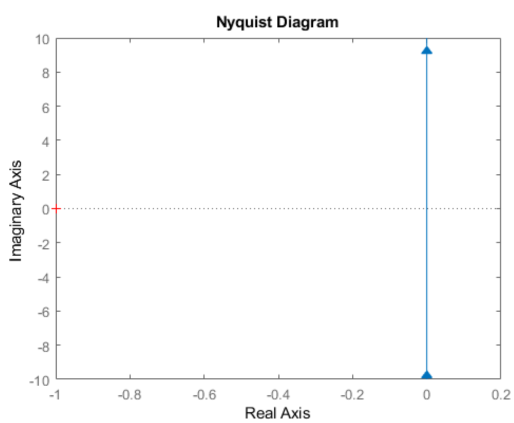
Précisons également que MATLAB ne peut fournir de l'information sur les encerclements à l'infini qui peuvent être présent lorsqu'il y a des intégrateurs en raison d'un problème d'échelle. Il incombe à l'utilisateur d'interpréter cette information manquante.

$$G(s) = \frac{1}{s+5} \qquad\qquad G(s) = s+5$$



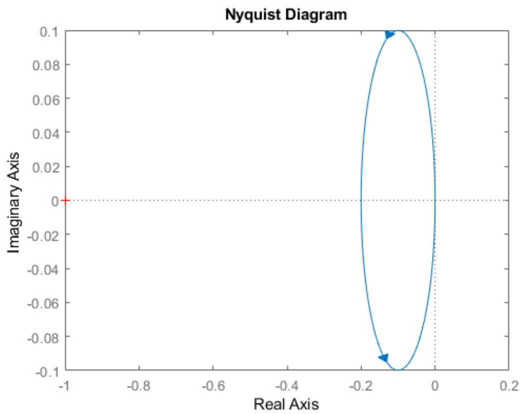


$$G(s) = \frac{1}{s}$$

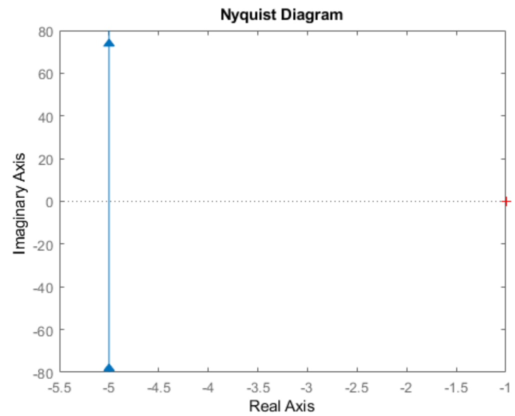


L'encerclement du demi-plan complexe de droite n'est pas figuré dans MATLAB.

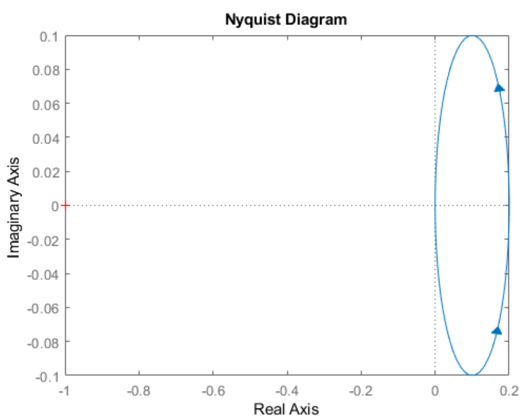
$$G(s) = \frac{1}{s-5}$$



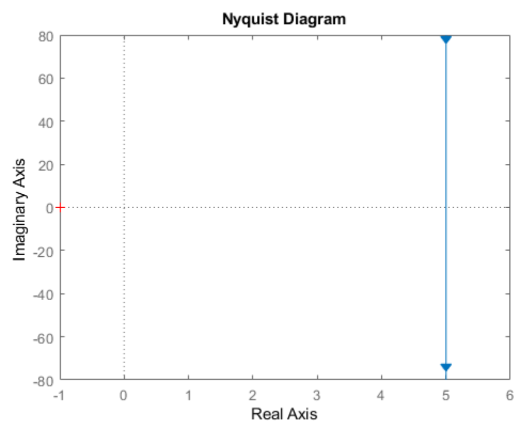
$$G(s) = s - 5$$



$$G(s) = \frac{1}{-s+5}$$



$$G(s) = -s + 5$$

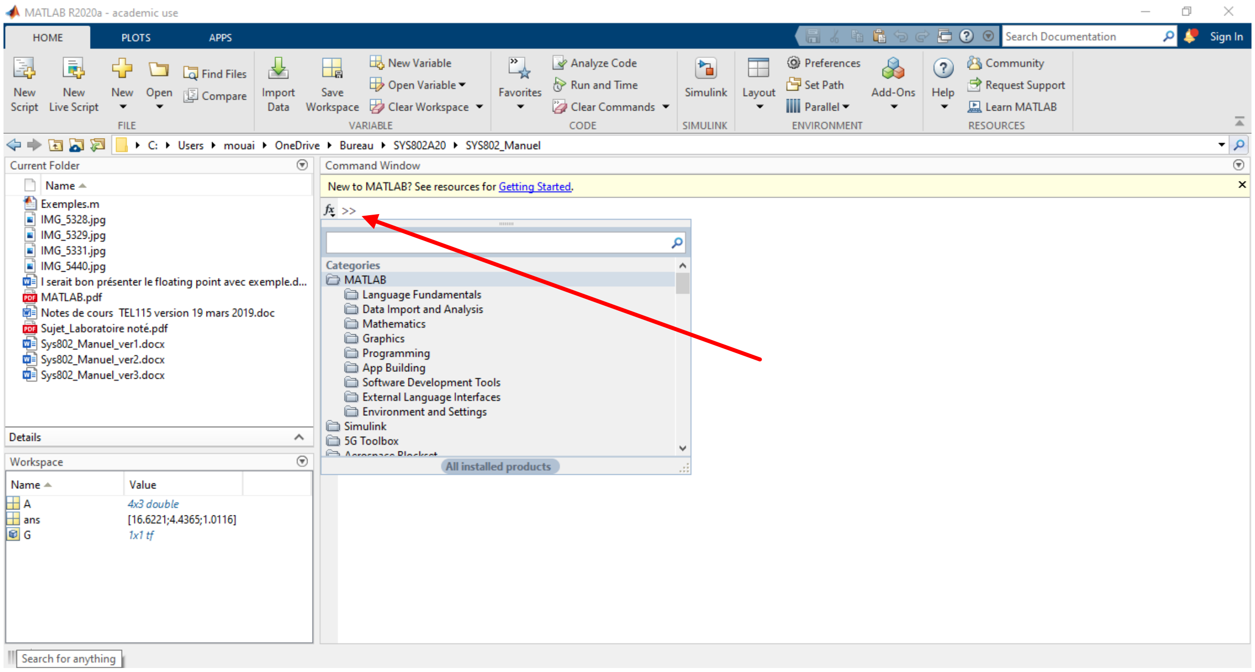


1-10 Le système d'aide de MATLAB

Pour explorer les fonctionnalités plus avancées de MATLAB, des options pour obtenir de l'aide sur l'utilisation des produits MathWorks sont fournies.

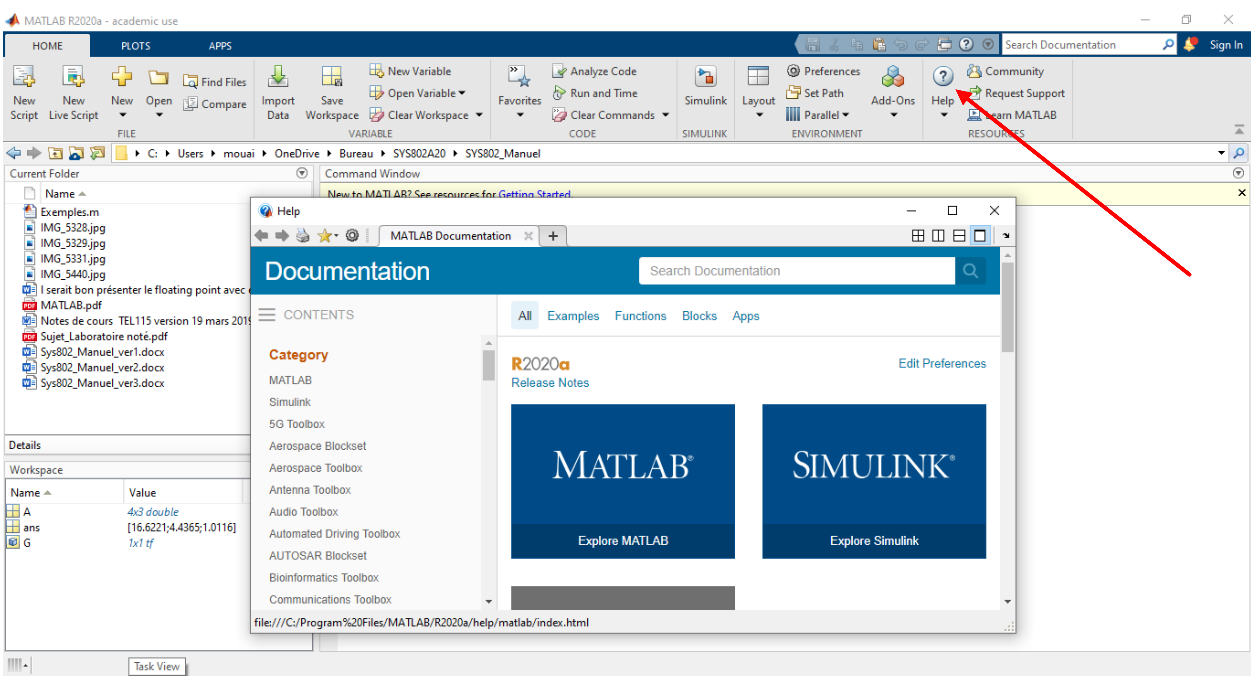
Le navigateur de fonctions :

Le navigateur de fonctions permet d'accéder rapidement à la documentation des fonctions MATLAB. Pour activer le navigateur de fonctions, sélectionner l'icône **fx** à gauche de l'invité MATLAB dans la fenêtre de commandes. Faire une recherche sur la fonction puis défiler pour voir toute la documentation de la fonction recherchée.



Le navigateur d'aide :

Cette interface utilisateur graphique aide à trouver des informations et à consulter la documentation en ligne de vos produits MathWorks. Pour ouvrir le navigateur d'aide, sélectionner **help** ou cliquer sur le bouton de point d'interrogation **?** dans la barre d'outils.



Les fonctions d'aide :

Les fonctions **help**, **lookfor** et **doc** peuvent être utilisées dans la fenêtre de commandes pour afficher les informations de syntaxe pour une fonction spécifiée.

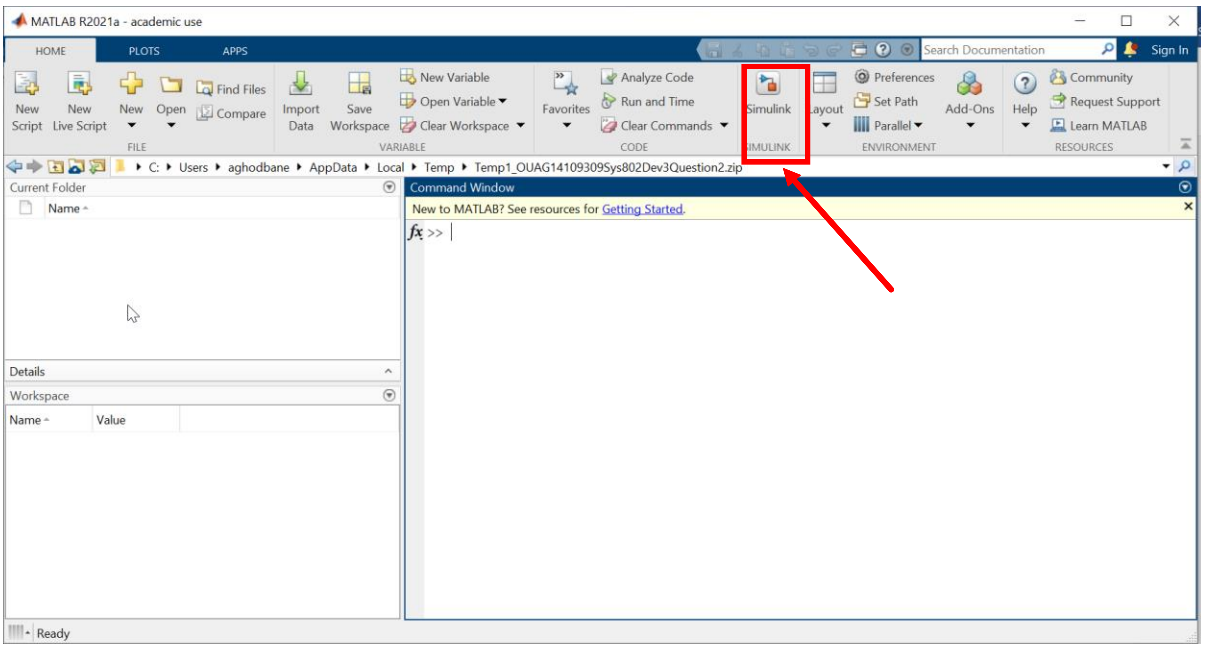
Le site Web MathWorks :

Le site de **The MathWorks Inc.**, la maison de MATLAB peut être accédé via internet : **http://www.mathworks.com**. Il est possible de poser des questions, de faire des suggestions et de signaler les bogues via courriel. Un moteur de recherche du site Web MathWorks est aussi accessible.

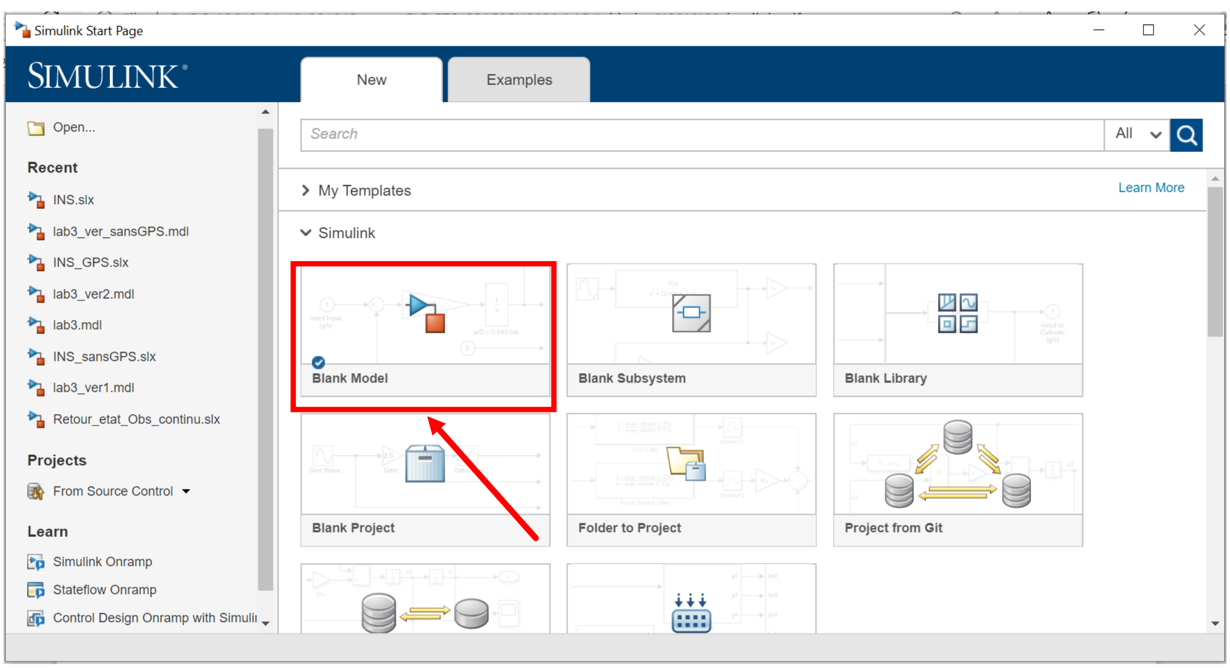
2- SIMULINK

2-1 Interface principale :

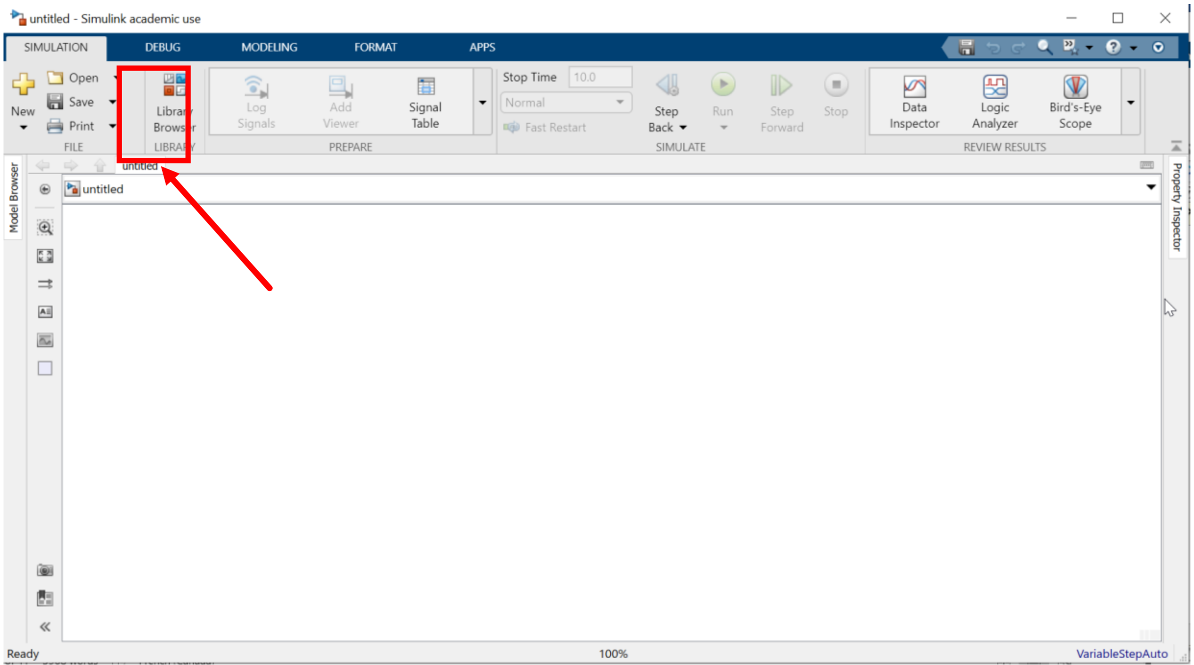
Simulink est une extension du logiciel MATLAB (*Matrix Laboratory*). C'est un logiciel de programmation graphique dédié principalement à la modélisation, la simulation et l'analyse des systèmes dynamiques. Les systèmes sont représentés par des blocs-diagramme. Pour accéder à Simulink, cliquer sur l'icône de Simulink présent sur la barre d'outils de MATLAB.



Une interface s'ouvre :

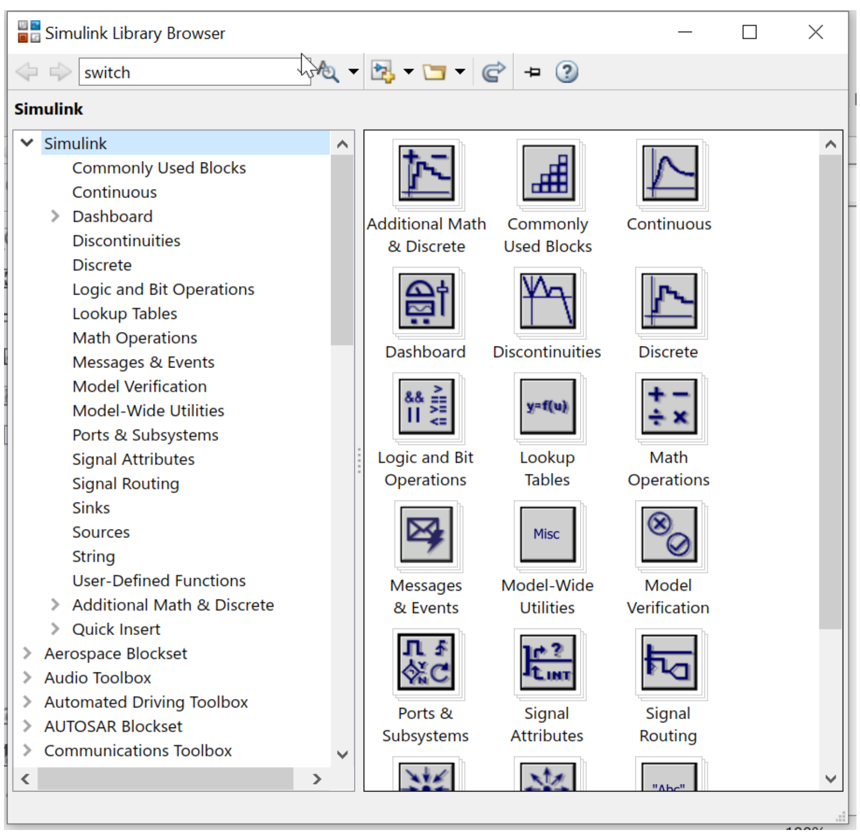


Cliquer sur l'icône **Blank Model** ; la fenêtre suivante s'ouvre :



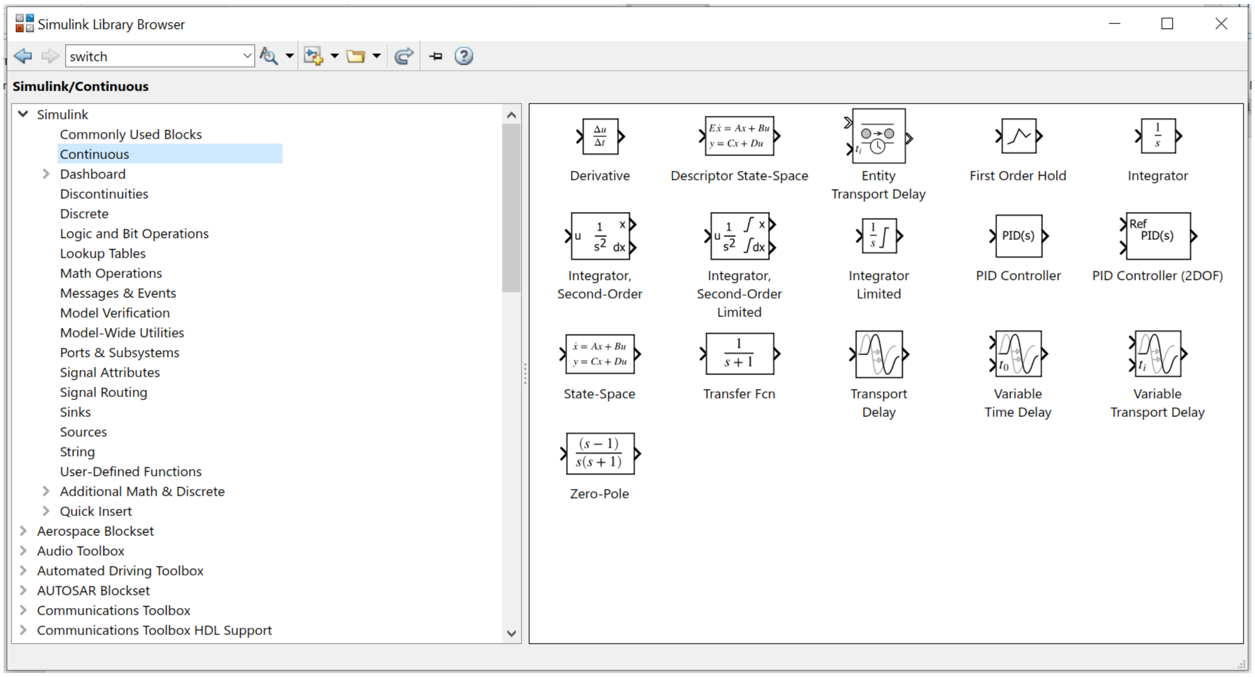
2-2 Librairies de Simulink :

Cliquer sur l'icône (Library Browser) pour accéder à la librairie de Simulink.

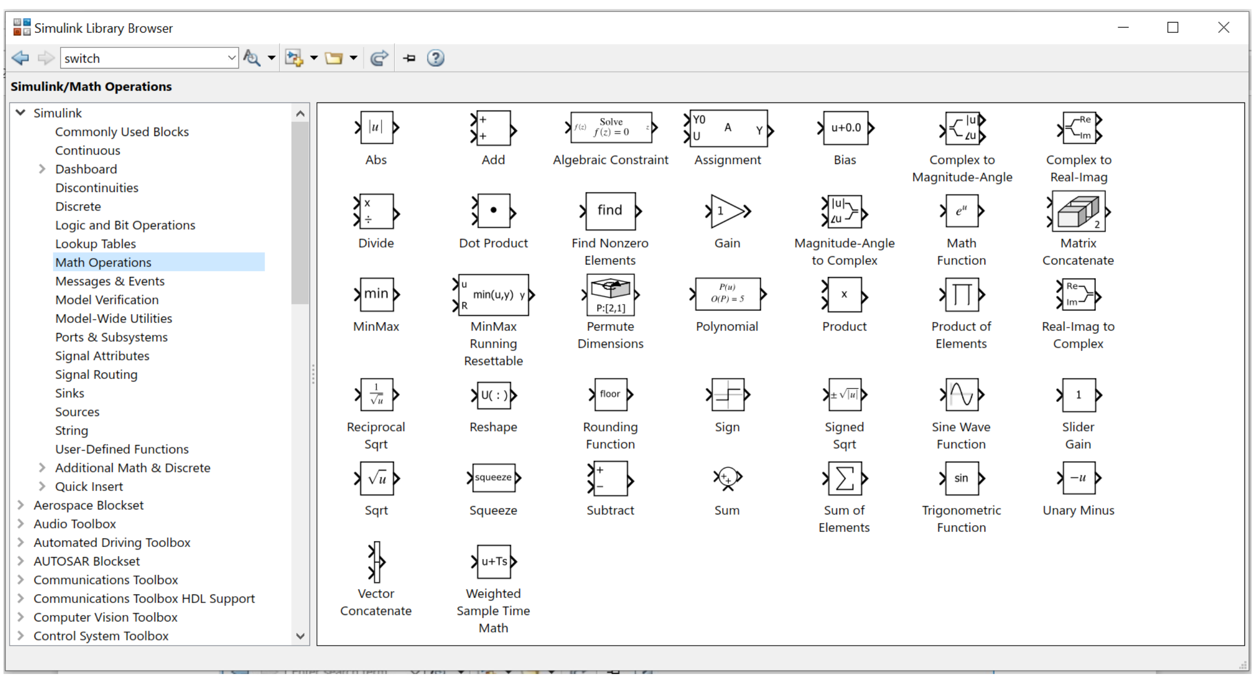


Pour afficher le contenu d'une librairie, il suffit de cliquer sur l'icône de cette dernière. Chaque librairie contient plusieurs blocs. Les principales librairies sont les suivantes :

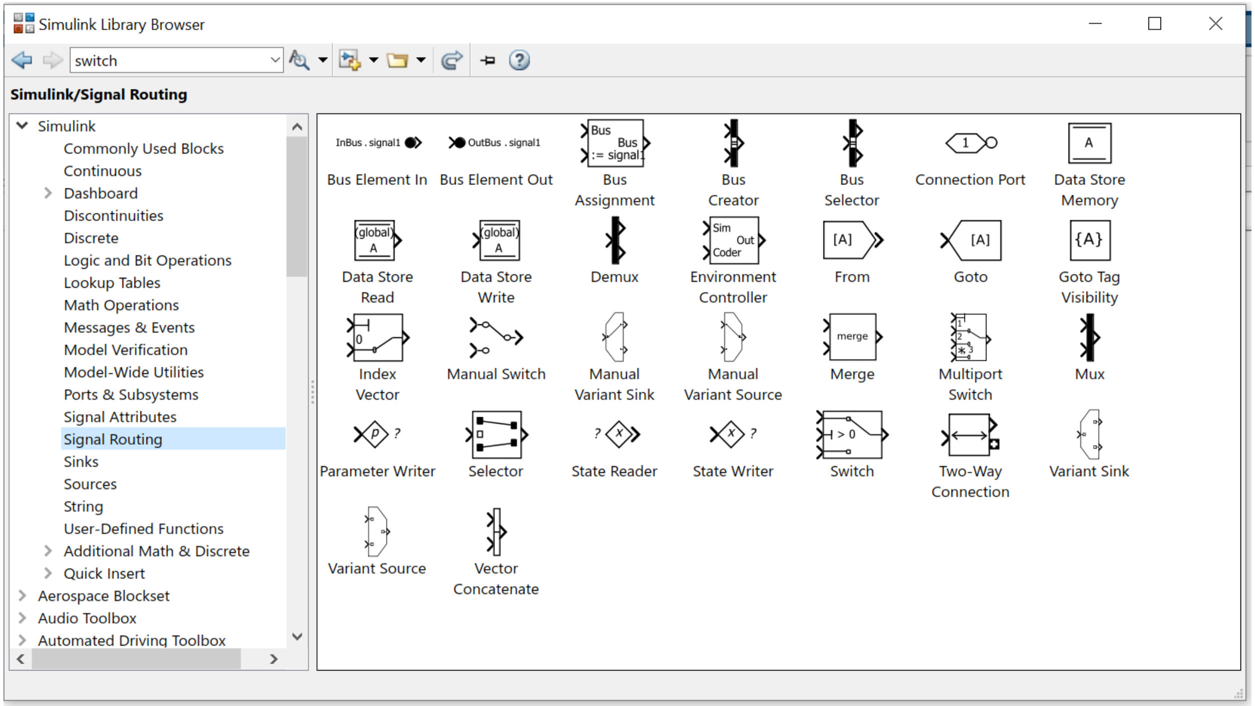
Continuous : Dans cette librairie les principaux blocs sont :



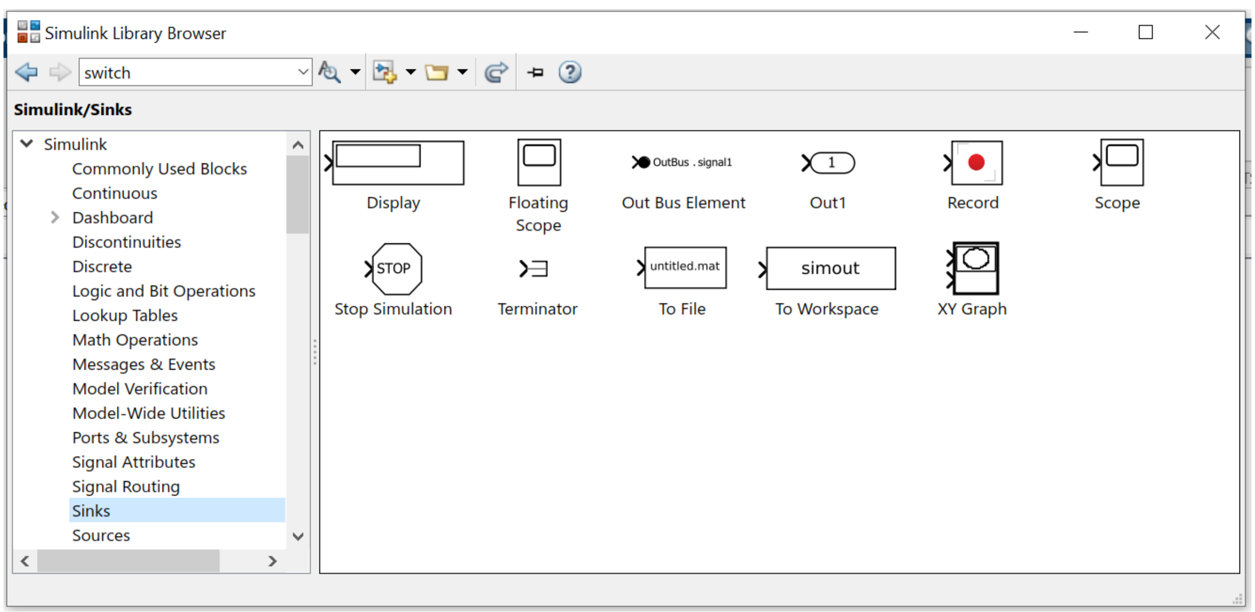
Math Operations : Dans cette librairie les principaux blocs sont :



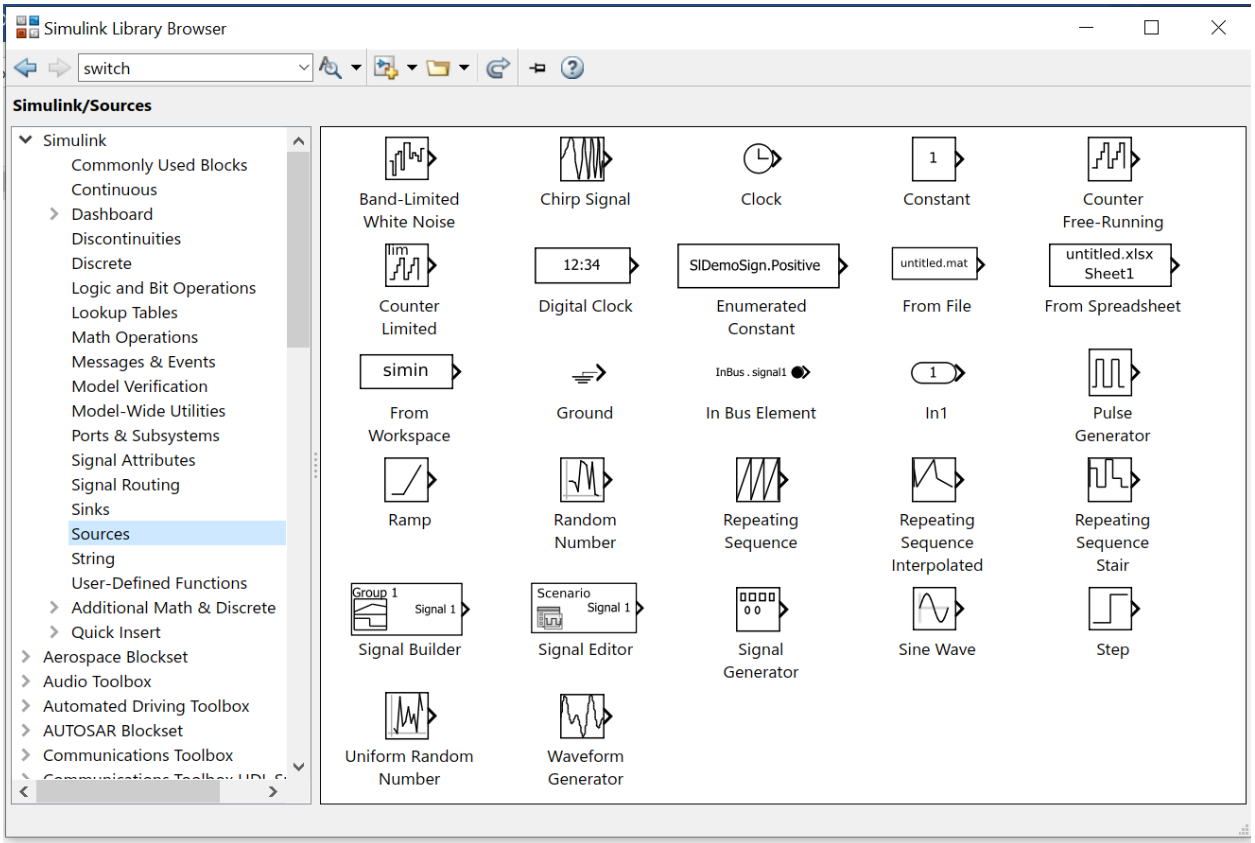
Signal Routing : Dans cette librairie les principaux blocs sont :



Sinks : Dans cette librairie les principaux blocs sont :



Sources : Dans cette librairie les principaux blocs sont :



Remarques :

- Chaque bloc d'une librairie de Simulink possède une entrée et/ou une sortie. Seules les connexions de type (*to*) sont possibles. L'information qui circule entre les connexions des blocs constitue donc un chemin orienté.
- Il est possible d'obtenir de l'aide à propos d'un bloc en cliquant sur ce dernier (clic droit), puis dans le menu qui s'ouvre en cliquant sur (*Help*).

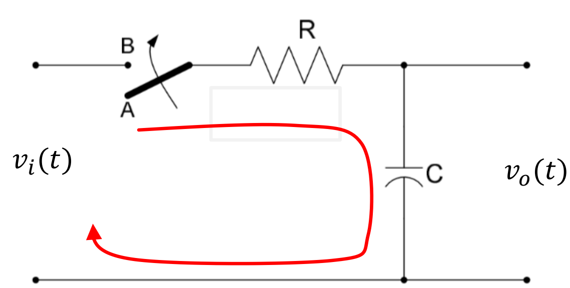
Utilisation de Simulink: Trois étapes sont nécessaires.

- La conception du système étudié.
- La configuration des paramètres de la simulation.
- L'exécution de la simulation.

2-3 Exemple 1 : Cas continu.

Système 1er ordre : Circuit RC

$R = 1K\Omega$, $C = 2mF$, l'entrée $v_i(t)$ est une fonction échelon $u(t)$ de 5V. Supposer une condition initiale $v_o(0) = 0$.



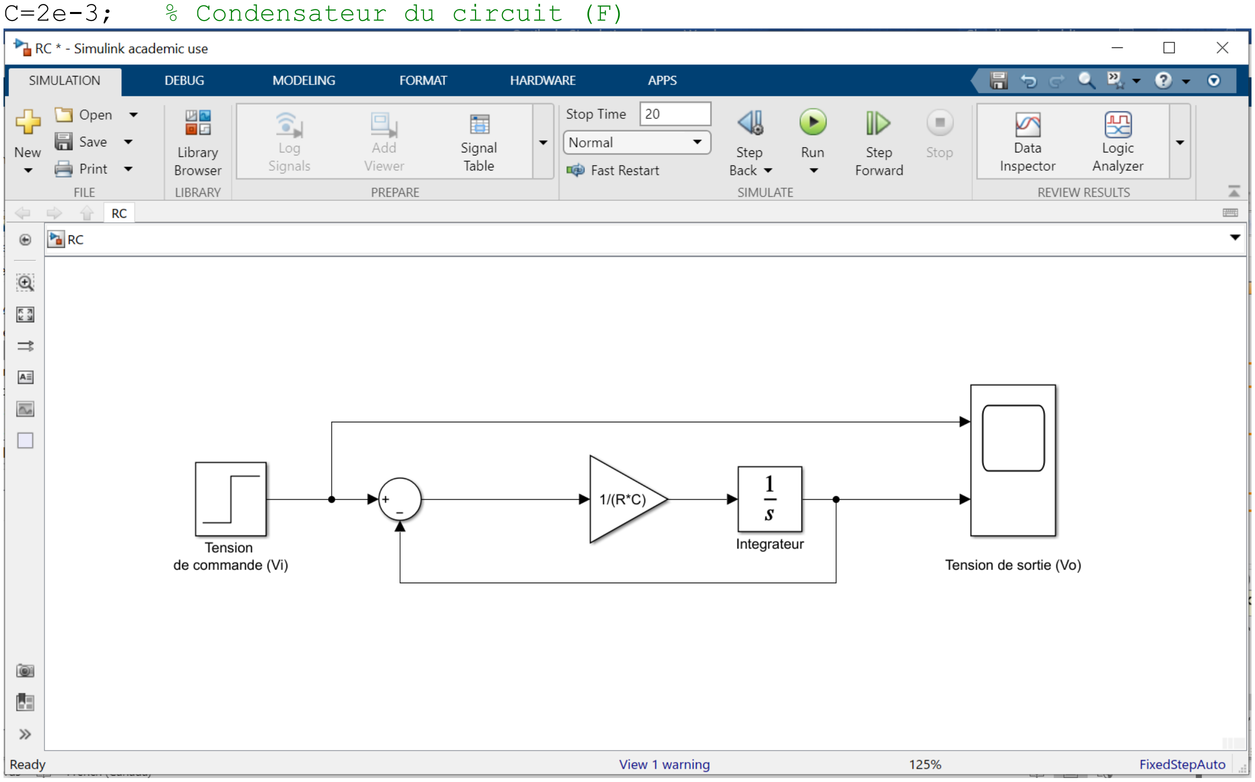
Modélisation par équation différentielle :

$$\frac{dv_o(t)}{dt} = \frac{1}{RC}(v_i(t) - v_o(t))$$

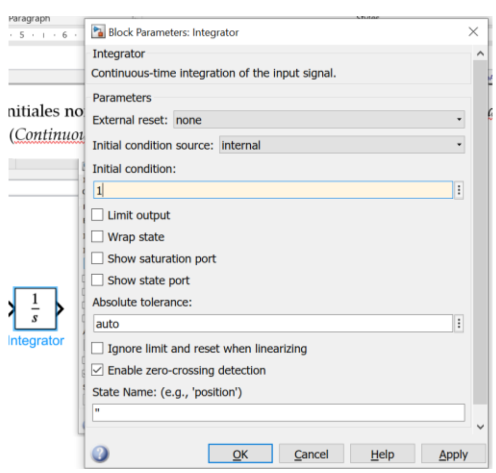
Conception par Simulink :

```
% Parametres du circuit RC

R=1e3;    % Resistance du circuit (Ohm)
C=2e-3;    % Condensateur du circuit (F)
```

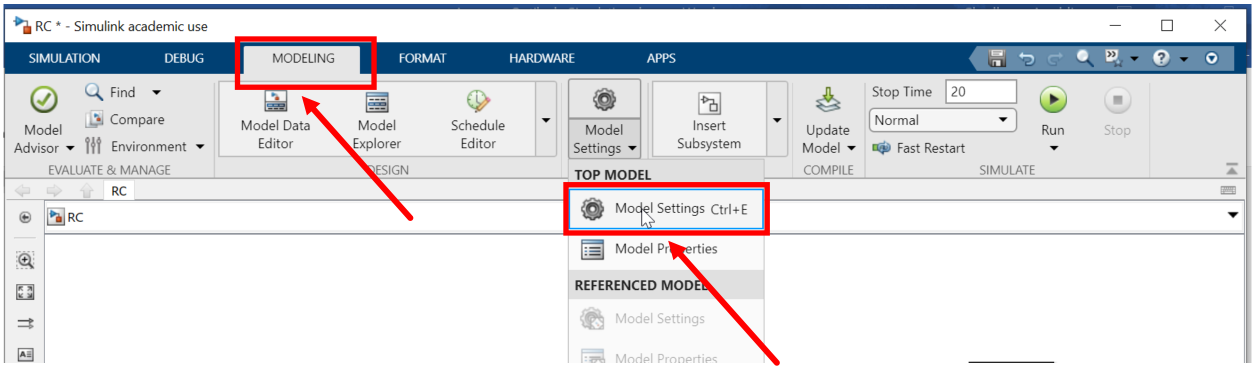


Remarque : Dans le cas de conditions initiales non nulles, Celles-ci sont configurées dans le bloc (*integrator*) disponible dans la librairie (*Continuous*). Dans l'exemple illustré ci-dessous, la valeur initiale est $v_o(0) = 1V$.

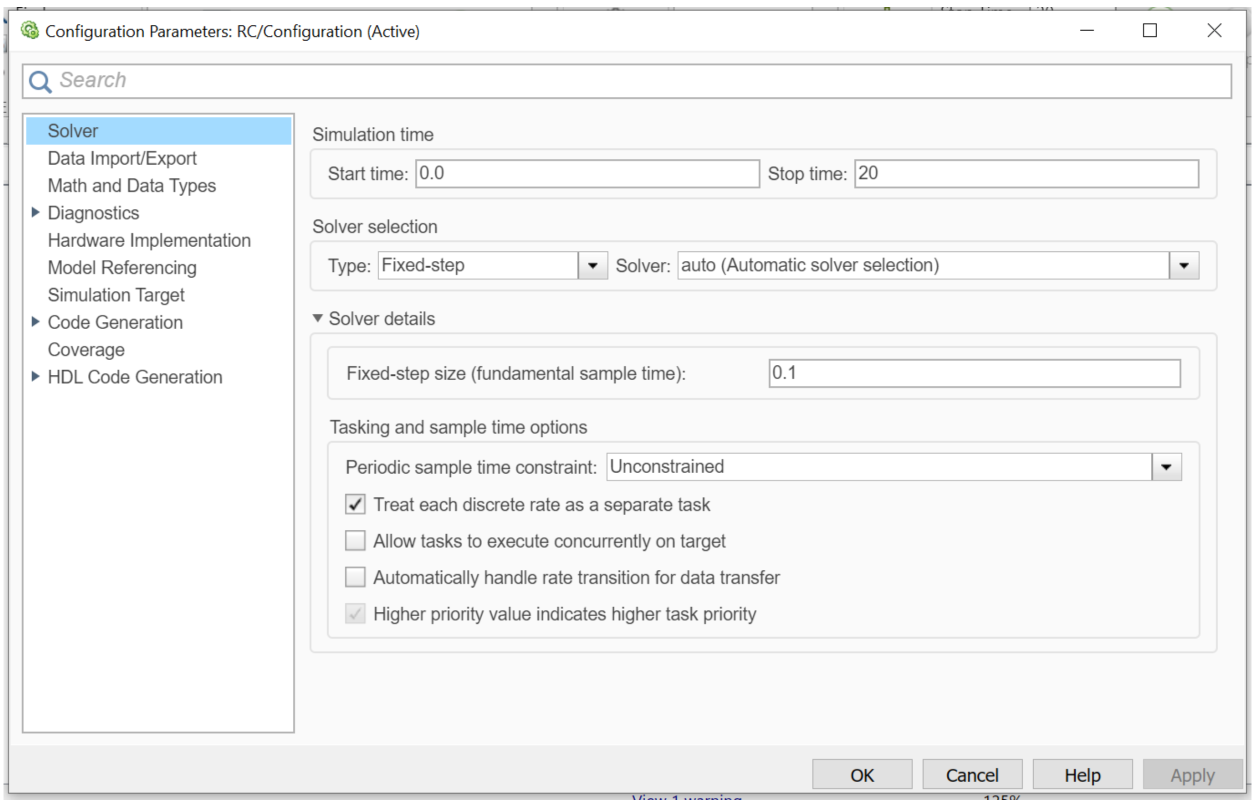


Configuration des paramètres de la simulation :

La fenêtre de configuration des paramètres de la simulation s'affiche en cliquant sur l'onglet (*Modeling*) puis sur (*Model Settings*).

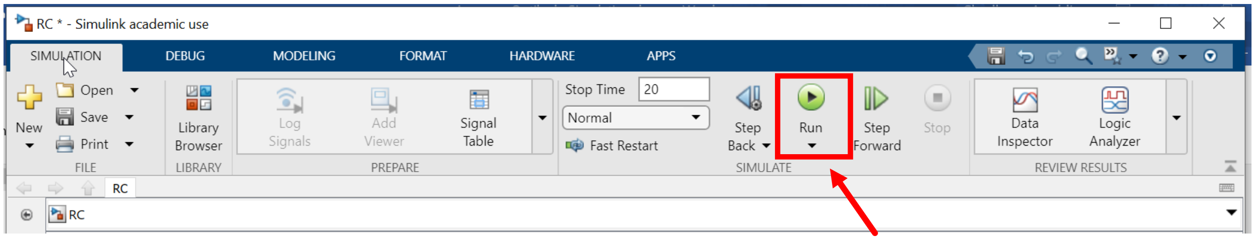


La fenêtre suivante s'affichera :

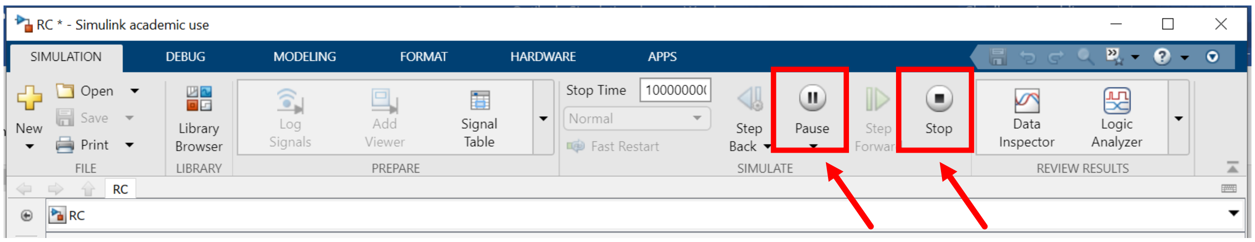


Dans cette fenêtre, les paramètres suivants peuvent être configurés : La durée désirée de la simulation, la période d'échantillonnage, le type du solveur (la technique d'intégration des équations différentielles). Le choix du solveur dépend de la nature de la simulation. Notez que les versions récentes de MATLAB permettent une sélection automatique du solveur.

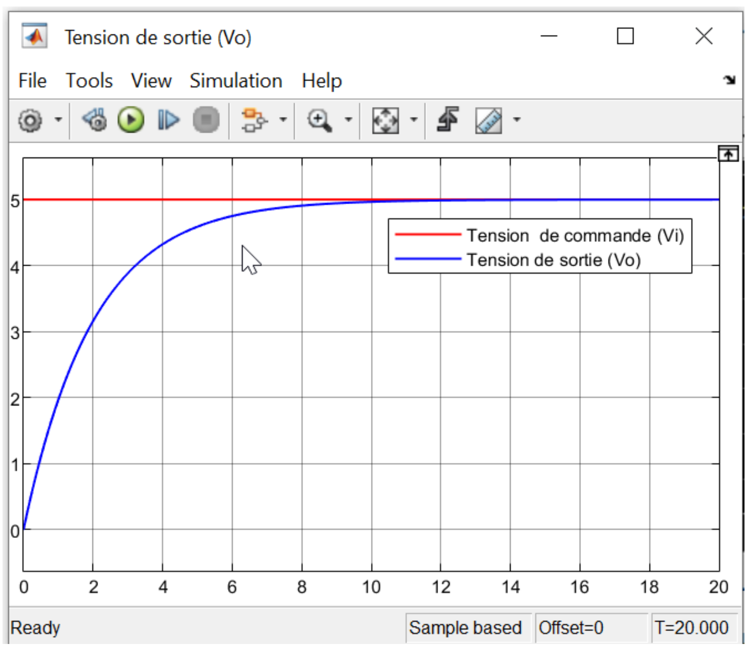
Exécution de la simulation: L'une des options pour exécuter la simulation consiste à cliquer sur l'icône (Run) :



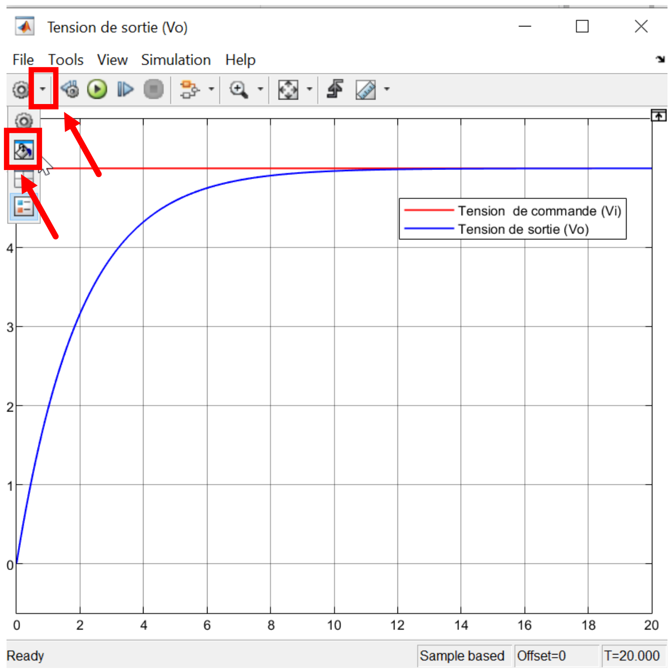
Lorsque la simulation est en cours d'exécution l'icône (*Run*) passe à (*Pause*) et le bouton (*Stop*) devient actif. Si nécessaire, il est possible d'arrêter la simulation en cours en appuyant sur le bouton (*Stop*).



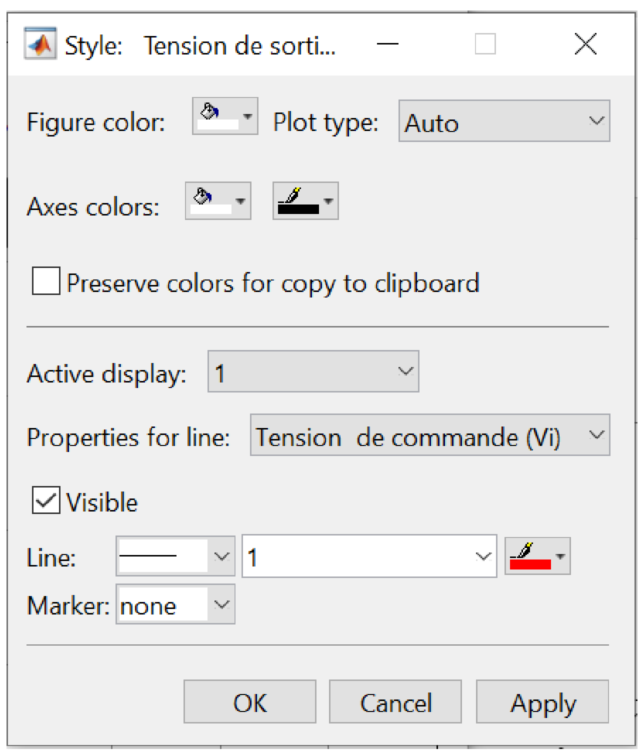
À la fin de la simulation, les signaux (Vi(t) et Vo(t)) peuvent être affichés. Pour cela, faire un double clic sur le bloc (*Scope*).



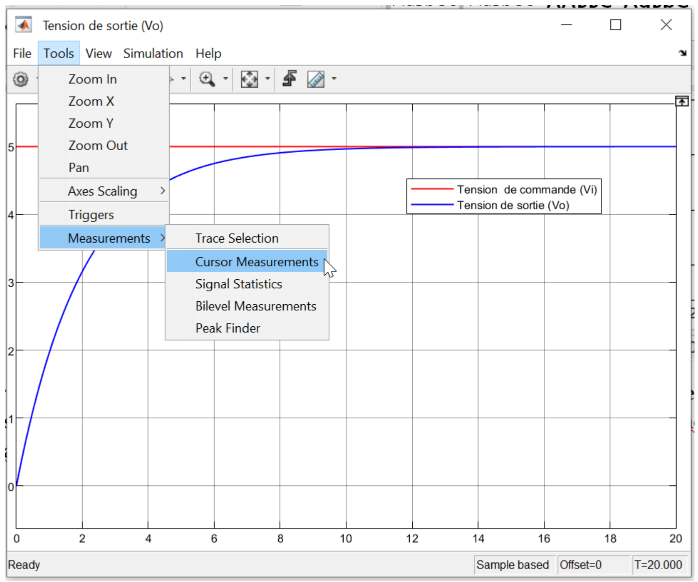
Il est possible de changer les couleurs et les dimensions de la figure ci-dessus. Pour cela, cliquer successivement sur les icônes encadrées en rouge indiquées sur la figure suivante.



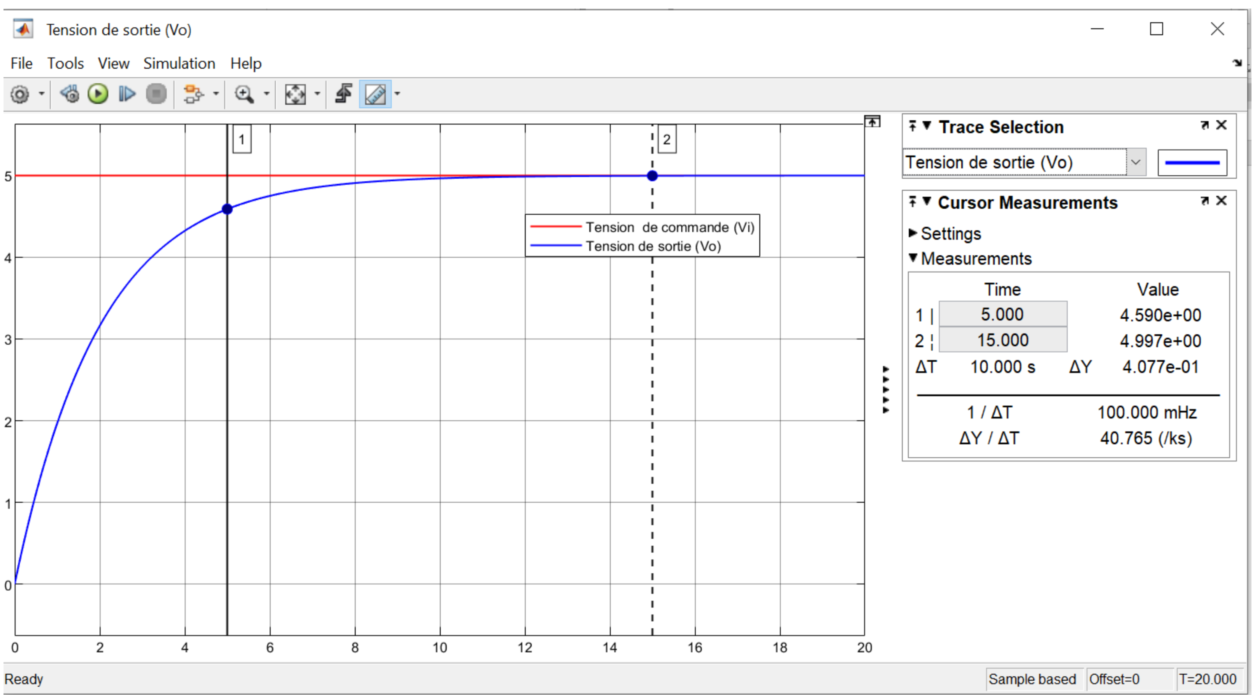
Dans la fenêtre qui s'affiche, les paramètres désirés sont configurés.



Les coordonnées d'un point du graphe peuvent être obtenues à l'aide de curseurs. Pour cela, cliquer sur le menu (*Tools*) et ensuite sur (*Measurements*) et puis sur (*Cursor Measurements*).



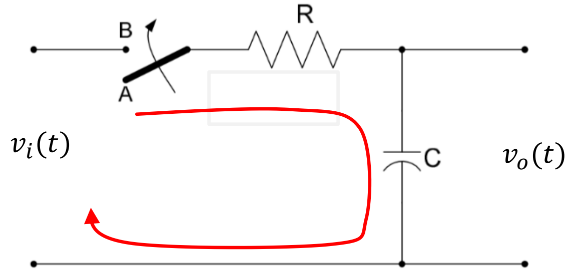
La figure ci-dessous apparait :



Remarque : Les curseurs peuvent être attachés à différents signaux. Dans la figure ci-dessus les curseurs sont attachés au signal de sortie Vo(t) en deux endroits du graphe. Les paramètres ΔT et ΔY indiquent les différences des abscisses et des ordonnées des positions des curseurs.

2-4 Exemple 2 : Cas discret.

Système 1er ordre : Circuit RC



$R = 1K\Omega$, $C = 2mF$, l'entrée $v_i(t)$ est une fonction échelon $u(t)$ de 5V. Supposer une condition initiale $v_o(0) = 0$.

Modélisation par fonction de transfert :

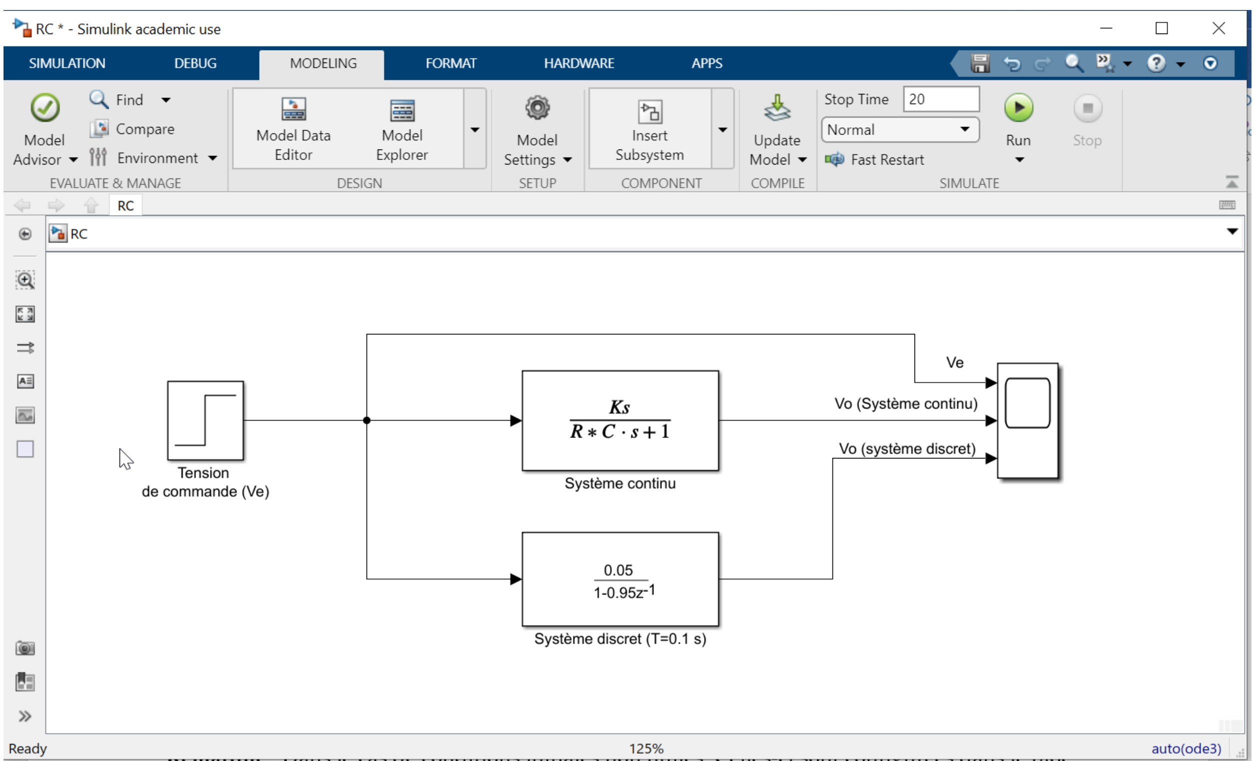
$$\frac{dv_o(t)}{dt} = \frac{1}{RC}(v_i(t) - v_o(t)) \rightarrow sV_o(s) = \frac{1}{RC}(V_i(s) - V_o(s)) \rightarrow P(s) = \frac{V_o(s)}{V_i(s)} = \frac{1}{RCs + 1}$$

Conversion vers la transformée en z pour $T = 0.1\ s$:

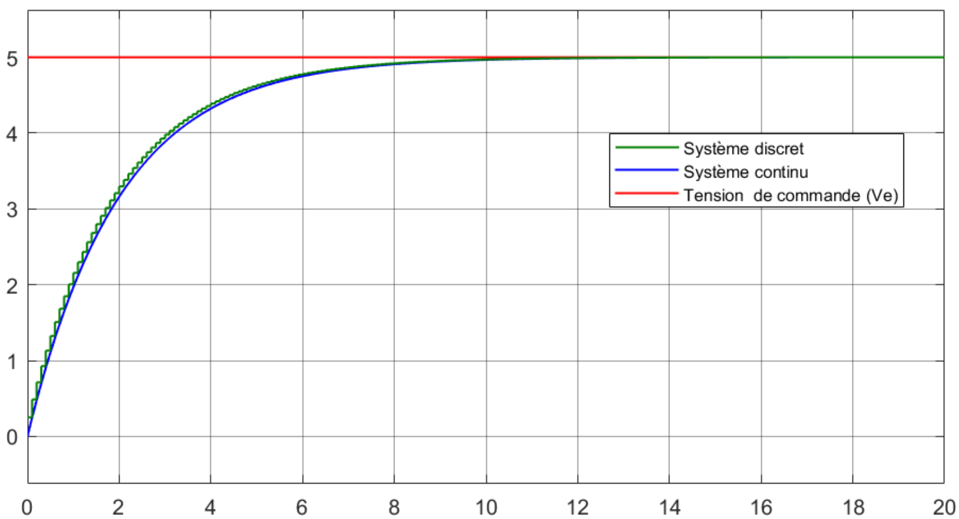
$$\frac{1}{s+a} \qquad \rightarrow \qquad \frac{1}{(1 - e^{-aT})z^{-1}}$$

D'où :

$$\frac{K_s}{\tau s + 1} = \frac{\frac{K_s}{\tau}}{s + \frac{1}{\tau}} = \frac{0.5}{s + 0.5} \qquad \rightarrow \qquad \frac{T\frac{K_s}{\tau}}{\left(1 - e^{-\frac{1}{\tau}T}\right)z^{-1}} = \frac{0.05}{(1 - 0.95)z^{-1}}$$



Dans la fenêtre des paramètres de configuration, la période d'échantillonnage est égale à $T = 0.1\ s$ et le solveur automatique est choisi. À la fin de la simulation, les signaux (Vi(t) et Vo(t) peuvent être affichés en faisant un double clic sur le bloc (*Scope*).



Conversion vers la transformée en z pour $T = 0.2\ s$:

$$\frac{1}{s+a} \qquad \rightarrow \qquad \frac{1}{(1-e^{-aT})z^{-1}}$$

D'où :

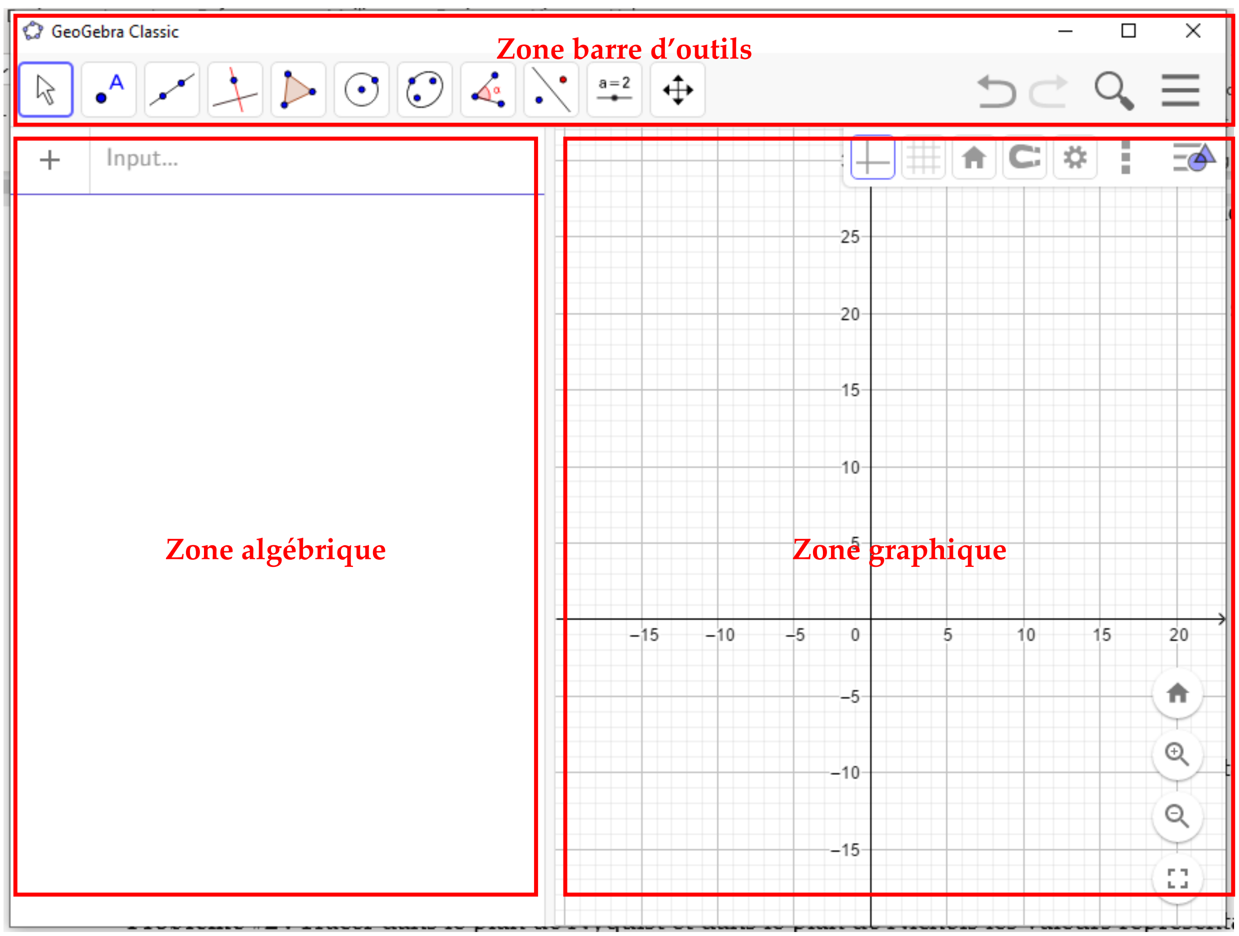
$$\frac{K_s}{\tau s+1} = \frac{\frac{K_s}{\tau}}{s+\frac{1}{\tau}} = \frac{0.5}{s+0.5} \qquad \rightarrow \qquad \frac{T\,\frac{K_s}{\tau}}{\left(1-e^{-\frac{1}{\tau}T}\right)z^{-1}} = \frac{0.05}{(1-0.95)z^{-1}}$$

3- GEOGEBRA

GeoGebra est un logiciel dynamique mathématique réunissant géométrie, algèbre et calcul. Il a été développé par M.Hohenwarter de l'Université de Salzburg en 2001. . D'une part, GeoGebra est un système de représentation géométrique interactif. Il permet d'élaborer des constructions 2D et 3D incluant des points, des cercles, des segments, des droites, des coniques ainsi que des courbes représentatives de fonctions avec la possibilité d'introduire des modifications de façon interactive. Nous présenterons dans ce qui suit les traitements utiles en commande.

L'interface de Geogebra est très intuitive et facile à gérer et comporte par défaut 3 cadrans :

- Zone barre d'outils : située en haut, elle se compose de tous les éléments de la géométrie sur plan : point, segment, droite, polygone, cercle, ellipse etc.
- Zone algébrique : c'est le cadran à gauche utilisé pour saisir les équations à représenter.
- Zone graphique : c'est le cadran à droite ou le graphe est visualisé.



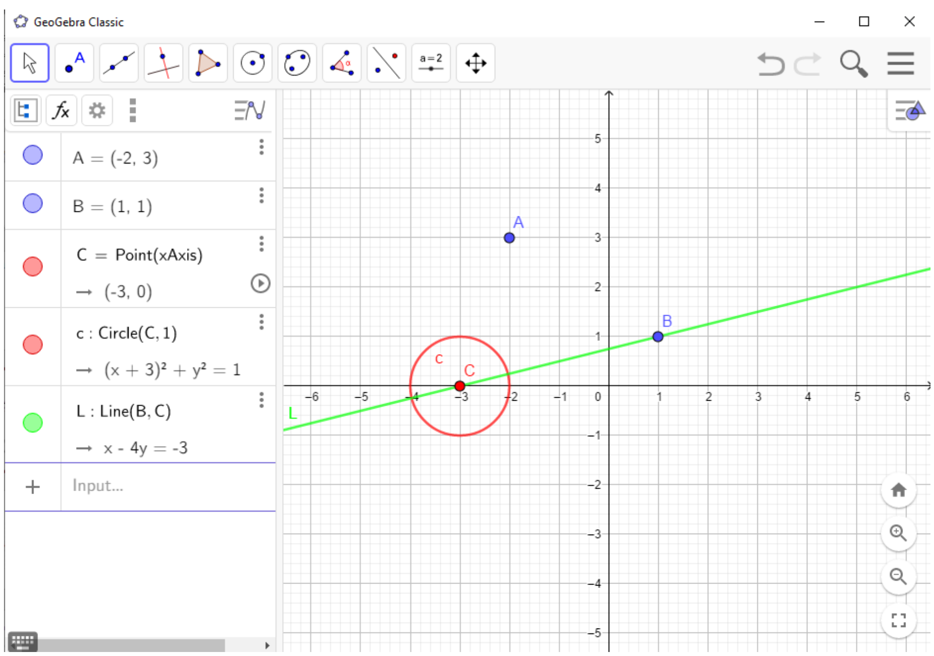
3-1 Exemple #1 : Présentation de points, lignes et cercles

Un point est représenté en faisant appel à la fonction **Point**, dont l'icône se trouve dans la barre d'outils. À l'aide de la souris, le point est placé dans la zone graphique. Par défaut, le point sera noté A. La fonction **Point** peut être aussi saisie dans la zone algébrique. Dans ce cas, elle aura besoins de deux paramètres qui sont les coordonnées du point à représenter.

Une droite est représentée en faisant appel à la fonction **Line**, dont l'icône se trouve dans la barre d'outils. À l'aide de la souris, les deux points passants par la droite sont sélectionnés. La fonction **Line** peut être aussi saisie dans la zone algébrique. Dans ce cas, elle aura besoin de deux paramètres qui sont les deux points passant par la droite.

Un cercle est représenté en faisant appel à la fonction **Circle**, dont l'icône se trouve dans la barre d'outils. À l'aide de la souris, le centre du cercle est sélectionné et puis la sourie sera déplacé pour fixer le rayon du cercle. La fonction **Circle** peut être aussi saisie dans la zone algébrique. Dans ce cas, elle aura besoin de deux paramètres qui sont le centre et le rayon.

Dans cet exemple trois points A, B et C sont respectivement représentés à (-2,3), (1,1) et (-3,0). Il en va de même de la droite L, passant par les points B et C et du cercle dénoté c, définit par son centre C et son rayon r=1.

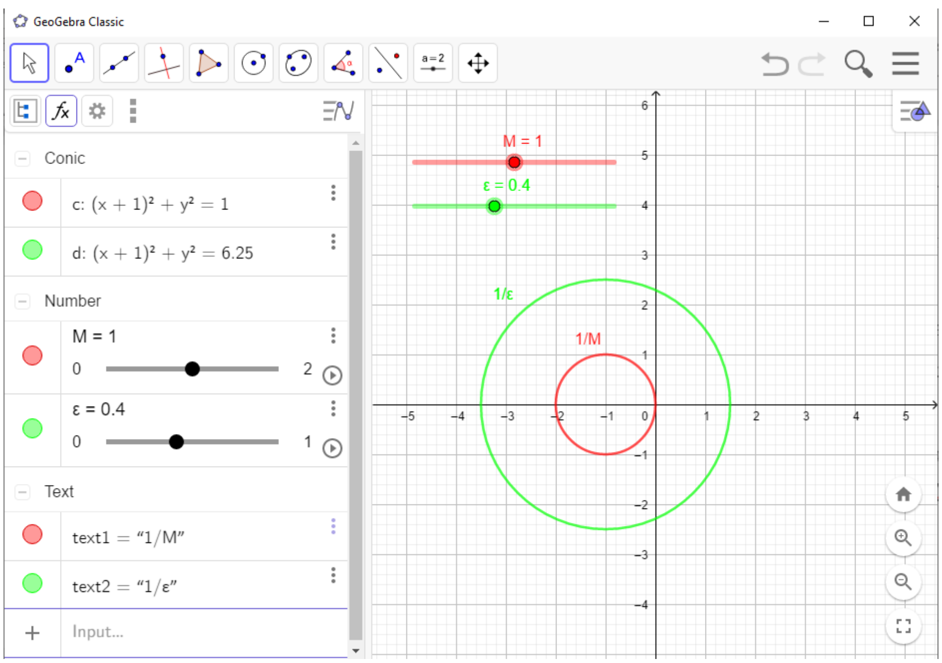


3-2 Exemple #2 : Présentation des cercles de rayons variables 1/M et 1/ε centrés au point (-1,0)

On définit M et ε comme des grandeurs qui varient entre un maximum et un minimum. Pour cela, nous allons introduire un curseur en faisant appel à la fonction **Slider,** dont l'icône se trouve dans la barre d'outils. Le curseur permet de définir et de fixer les limites de variation.

Pour tracer ces cercles, il faut faire appel à la fonction **Circle,** dont l'icône se trouve dans la barre d'outils**.** Cette icône permet de pointer le centre (-1,0) directement sur la feuille de travail. Une fenêtre s'ouvrira pour saisir le rayon (1/M ou 1/ε). Si M et ε n'ont pas été créés auparavant, GeoGebra va attribuer automatiquement un curseur pour chaque élément. Les cercles peuvent être identifiés en faisant appel à la fonction **text**. Ces grandeurs peuvent être modifiées facilement au moyen du curseur. Le changement de la position de chaque curseur entraîne un changement direct des rayons des cercles.

Dans cet exemple, M varie entre 1 et 3 avec un pas 0.1 et ε varie entre 0 et 1 avec un pas 0.1. Les cercles sont identifiés par 1/M et 1/ε. Nous aurons recours par la suite à ces familles de cercles qui définiront des cercles de sensibilité constante.



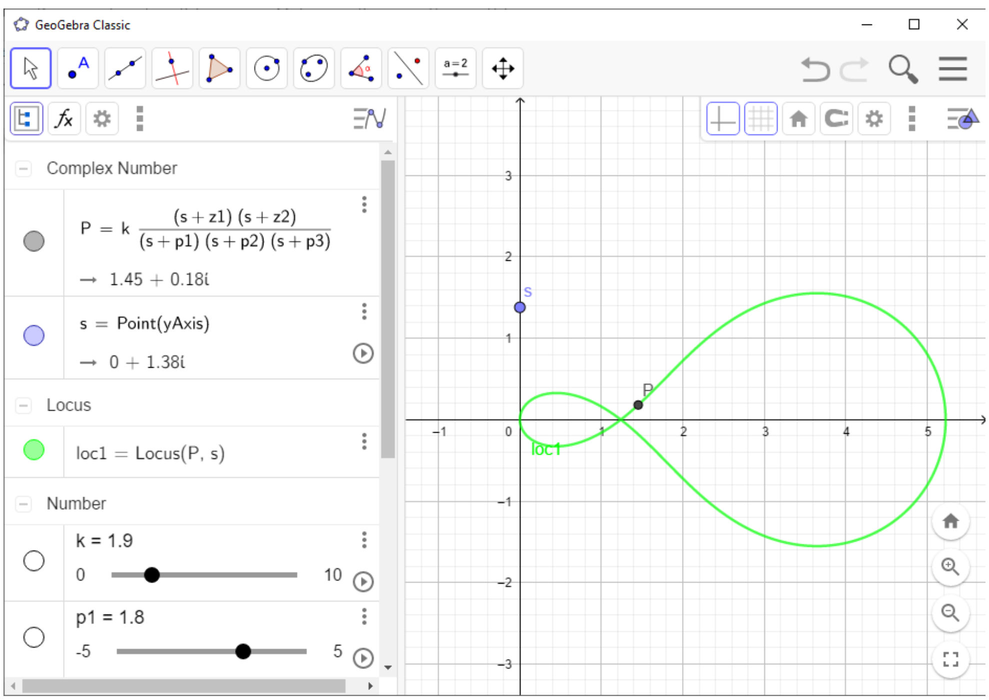
3-3 Exemple #3 : Présentation de Nyquist sur GeoGebra

Différents paramètres doivent être définis pour tracer le Nyquist d'un système.

Un point s est défini par la valeur complexe s=jω. La fonction **Point,** dont l'icône se trouve dans la barre d'outils est utilisée pour le représenter sur le graphique**.** Avec la souris, le point s est placé sur l'axe des imaginaires. La fonction **Point** peut être saisie dans la zone algébrique. Dans ce cas, le premier paramètre (valeur réelle) est mis à zéro. Par défaut, la fonction **Point** représente le point s en coordonnées cartésiennes. Il doit être converti en nombre complexe en faisant appel à la fonctionnalité **setting** (appuyer sur les trois points verticaux en haut et à droite dans la zone algébrique). Un point peut être représenté par ses coordonnées cartésiennes $a + jb$ ou ses coordonnées polaires. Ainsi, le nombre complexe $z = 2 + j1$ peut être représenté dans GeoGebra et son équivalent en module et argument peut être obtenu en faisant appel à la fonction **Polar Coordinates** qui se trouve à droite dans le menu GeoGebra sous l'onglet **Algebra**. Il est possible de lire: le module 1.11 et la phase 26°. Inversement, il est possible d'introduire des coordonnées polaires et de retrouver les coordonnées cartésiennes en faisant appel à la fonction **Cartesian Coordinates** qui se trouve à droite dans le menu GeoGebra sous l'onglet **Algebra**

La fonction de transfert du la boucle ouverte $L(s) = \frac{k(S+z_1)(S+z_2)}{(S+p_1)(s+p_2)(s+p_3)}$ est saisie directement dans la zone algébrique. Si les curseurs ne sont pas déjà attribués à $k, z_1, z_2, p_1, p_2, p_3$, ils seront attribués automatiquement par GeoGebra qui demandera une confirmation.

Faire appel à la fonction **locus**, dont l'icône se trouve dans la barre d'outils**.** La fonction **locus** aura besoin de deux paramètres qui sont L et s. Le Nyquist paraitra alors dans la zone graphique.



Comme les valeurs de k, z_1, z_2, p_1, p_2 et p_3 peuvent varier à l'aide des curseurs, nous pouvons donc afficher à chaque fois le Nyquist dédié à la nouvelle fonction L(s). Suite à la translation de s sur l'axe des imaginaires (-90° à 90°), réalisée par la fonction **move**, dont l'icône se trouve dans la barre d'outils, le point L sur Nyquist changera de position selon le sens d'évolution de ce dernier.

3-4 Exemple #4 : Transmittance T et Sensibilité S

Visualisons les variations du vecteur de la transmittance de la boucle fermée T = L/(1+L) et ainsi que celui de la sensibilité S = 1 (1+L) quand le vecteur L représentant la fonction de transfert en boucle ouverte varie en fonction de ω.

Étapes de construction :

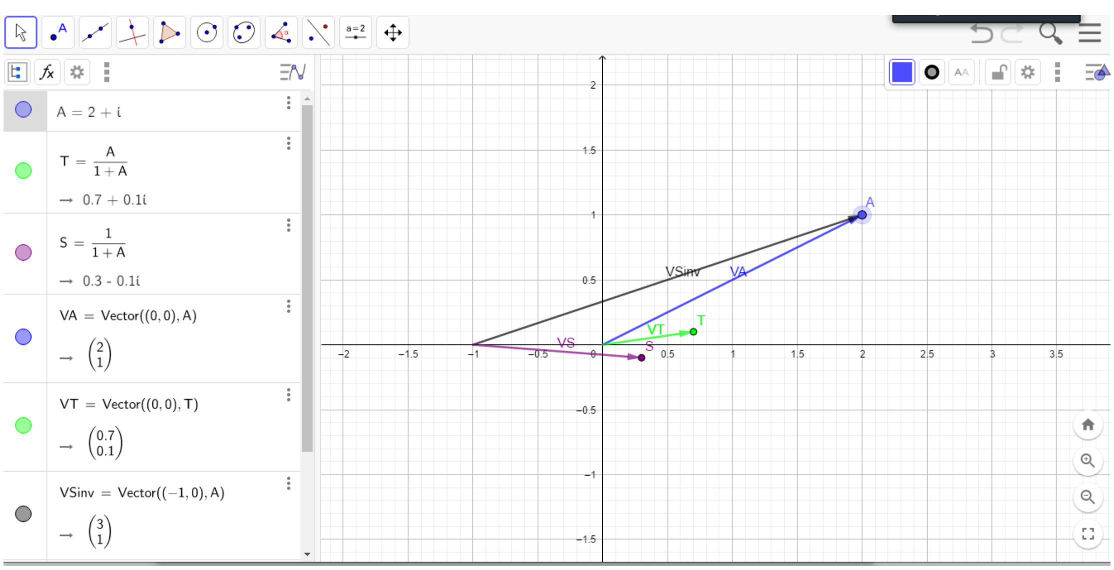
- Construction d'un point A quelconque.
- Construction d'un cercle C du centre (0,0) et passant par A.
- Construction du nombre complexe B sur le cercle.
- Construction du vecteur G = AB.
- Construction des nombres complexes $s = (1+z)^{-1}$ et $t = z(1+z)^{-1}$.
- Correspondance des vecteurs T (Transmittance) et S (Sensibilité) aux nombres t et s.

La grandeur complexe z est représentée par le point A, dont la position dans le plan complexe détermine sa grandeur (sa norme). Dans ce qui suit, les grandeurs z, t et s représenteront le gain en boucle ouverte, le gain en boucle fermée et la sensibilité. Ces grandeurs peuvent être représentées par des vecteurs dont les angles respectifs sont α_P, α_T et α_S.

Dans ce qui suit, nous illustrons la grandeur $z = 2 + j1$ (en bleu), ainsi que les valeurs correspondantes de $s = (1+z)^{-1}$ (en mauve) et $t = z(1+z)^{-1}$ (en vert). Nous représentons également le point critique C de coordonnées (-1,0), les vecteurs $\overrightarrow{OA}$, $\overrightarrow{CS}$ et $\overrightarrow{OT}$. Notez que $\overrightarrow{CS} + \overrightarrow{OT} = 1$.

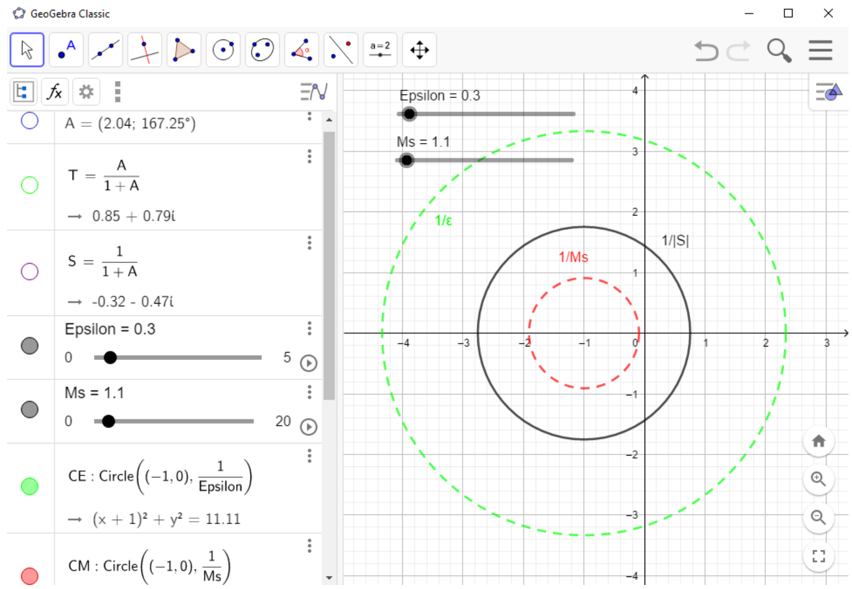
Déplaçons le point A dans le plan complexe au moyen de la souris et observons la variation de ces vecteurs.

Notons que l'utilisation de curseurs permet de visualiser l'effet des grandeurs variables de façon interactive. Pour cela, il faut faire appel à la fonction **Slider,** dont l'icône se trouve dans la barre d'outils. Les exemples suivants intégreront de telles variables contrôlées par curseur.



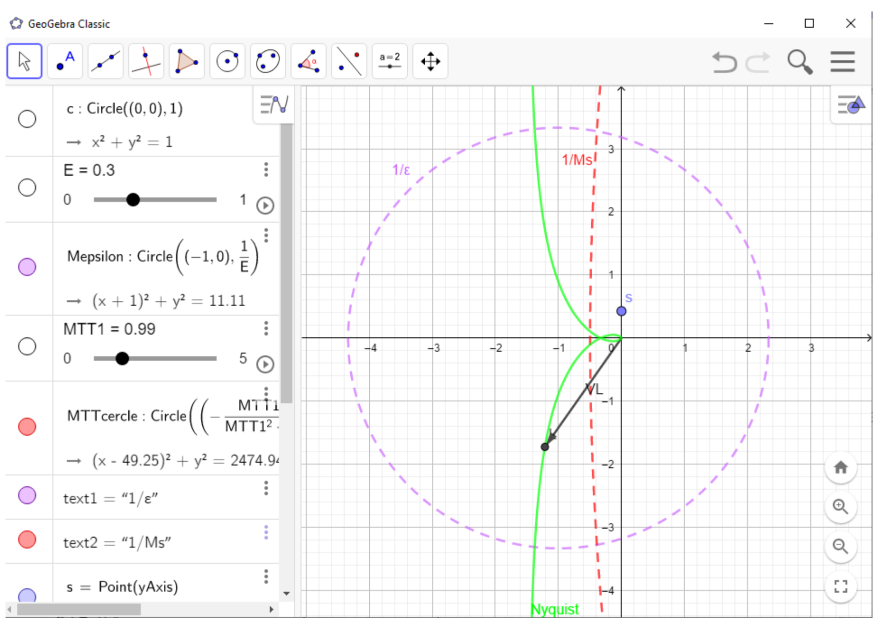
3-5 Exemple #5 : Sensibilité $|S(\omega)|$ constante

Déterminons dans le plan complexe les points représentant une sensibilité $|S(\omega)|$ constante. Il s'agit de cercle centré au point critique (-1,0) et de rayon $\frac{1}{|S(\omega)|}$ (en noir). De façon générale, la sensibilité en basse fréquence $|\omega| < \omega_1$ sera délimitée par l'extérieur du cercle de sensibilité de rayon $\frac{1}{\varepsilon}$, $\varepsilon < 1$ (en vert) et la sensibilité en haute $|\omega| > \omega_1$ sera délimitée par l'extérieur du cercle de sensibilité de rayon $\frac{1}{M_s}$ (en bleu). De la sorte, notre conception respectera les conditions : $|S(\omega)| \leq \varepsilon$ pour $|\omega| < \omega_1$et $|S(\omega)| \leq M_s$ pour $|\omega| > \omega_1$.



3-6 Exemple #6 : Diagramme de Nyquist avec $M_s = 1.01$ et $\varepsilon = 0.3$

Visualiser le diagramme de Nyquist (en vert), les cercles M_s (en rouge) et ε (en mauve), et le vecteur VL représentant le gain de boucle ouverte $L(s) = \frac{2}{s(s+1)(s+2)}$. Dans cet exemple, $M_s = 1.01$ et $\varepsilon = 0.3$. Le vecteur VL représentant le gain en boucle ouverte peut être visualisé en déplaçant le point s sur l'axe imaginaire (Le graphe représente le vecteur VL pour $s = j0.43$).



3-7 Exemple #7 : Gain de boucle fermée $|T(s)|$ constant

Soit X et Y les parties réelles et imaginaires du gain de boucle ouverte $L(\omega) = X + jY$.

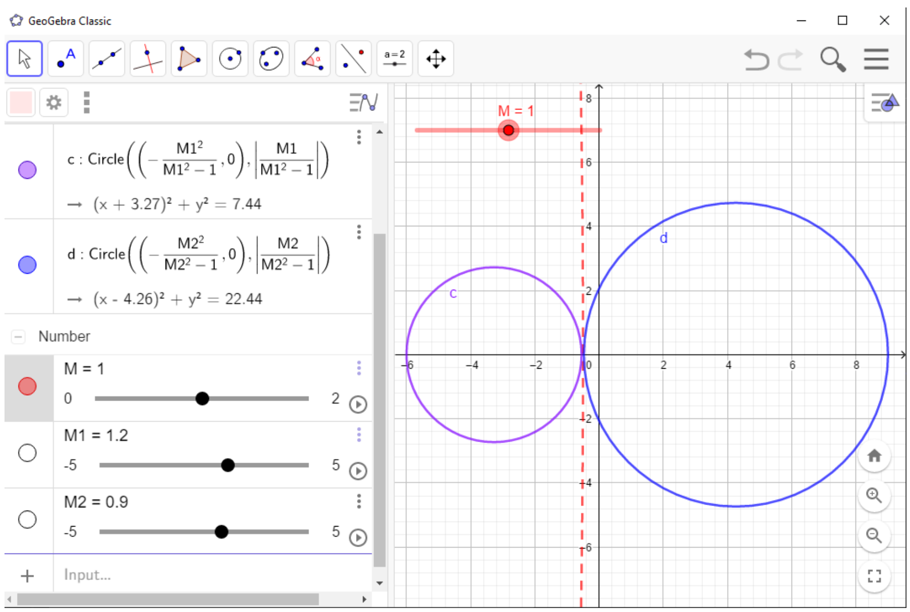
Trouvons les lieux vérifiant $|T(s)| = \left|\frac{L(s)}{1+L(s)}\right| = M$

$M = \frac{|X+jY|}{|1+X+jY|}$ ou $M^2 = \frac{X^2+Y^2}{(1+X)^2+Y^2}$, d'où :

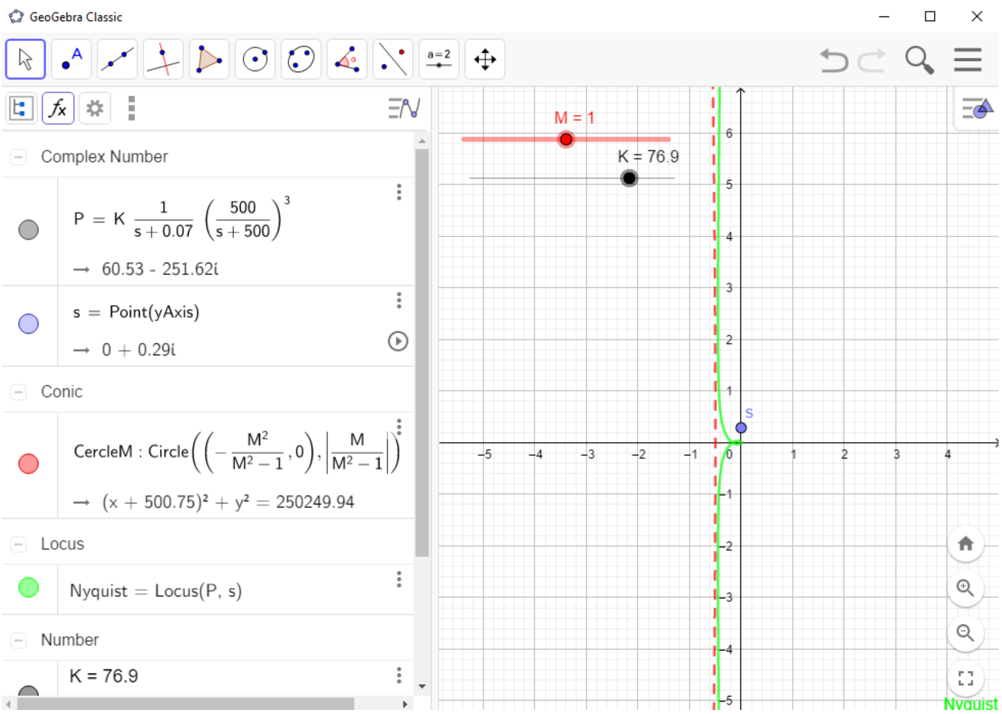
$$X^2 + \frac{2M^2}{M^2-1}X + \frac{M^4}{(M^2-1)^2} + Y^2 = \frac{M^2}{(M^2-1)^2}$$

$$\left(X^2 + \frac{M^2}{M^2-1}\right)^2 + Y^2 = \frac{M^2}{(M^2-1)^2}$$

Il s'agit de l'équation d'un cercle de centre $\left(-\frac{M^2}{M^2-1}, 0\right)$ et de rayon $\left|\frac{M}{M^2-1}\right|$.



Noter que le gain de boucle idéal M=1 se traduit par un diagramme de Nyquist qui longe la ligne droite dont la valeur réelle est -0.5. Un diagramme de Nyquist qui longerait le gain de boucle idéal M=1 signifierait une transmission parfaite entre l'entrée et la sortie de la boucle de rétroaction. Notez les modifications du diagramme de Nyquist en faisant varier le gain K dans l'exemple suivant.



3-8 Exemple #8 : Modules et phases des gains de boucle fermée M constants dans le plan de Nichols.

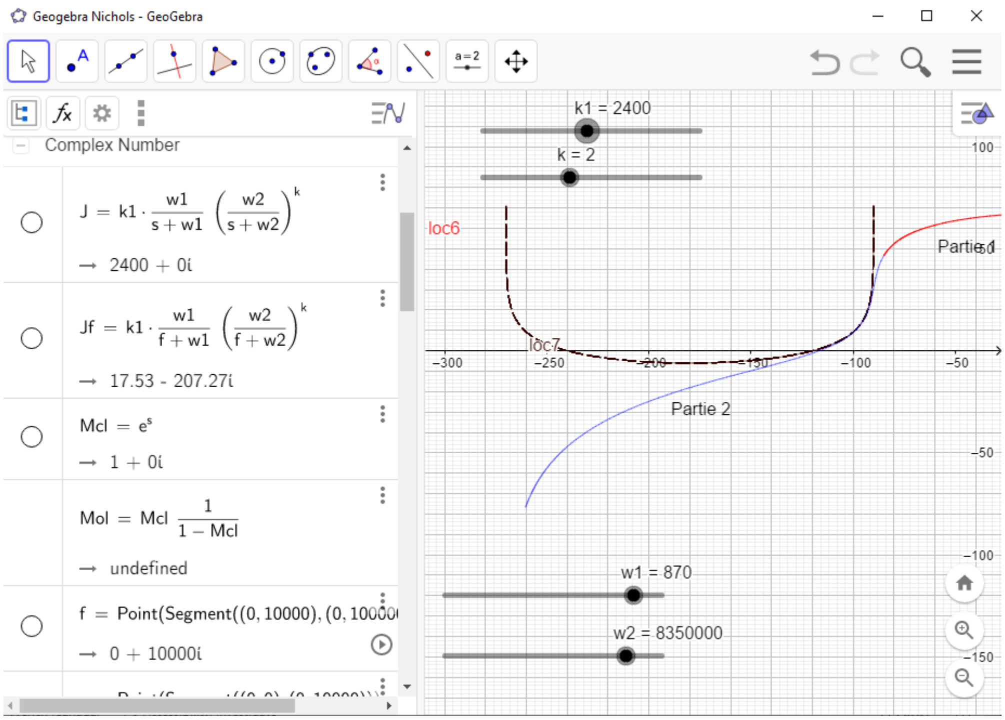
En remplaçant la boucle ouverte L par 1/L, il s'ensuit que la boucle fermée devient :

$$\frac{1/L}{1+1/L} = \frac{1}{1+L} = S$$

Il suffit donc de représenter 1/L plutôt que L. De la sorte, les cercles M sont lus comme des cercles de sensibilité S.

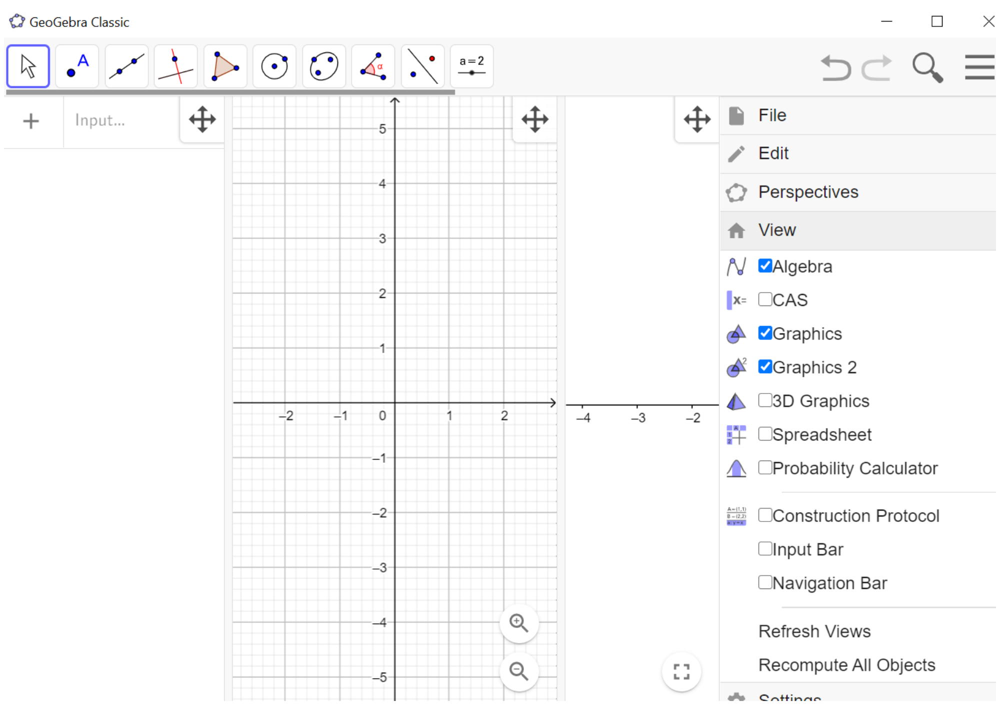
La conception sur le diagramme de Nichols représentant L(s) doit faire en sorte que le diagramme soit tangent au cercle $M_t = 0dB$ pour assurer une transmission en boucle fermée unitaire. Alternativement, la conception sur le diagramme de Nichols représentant $L^{-1}(s)$ doit faire en sorte que le diagramme soit tangent au cercle $M_s = 0dB$ pour assurer une sensibilité minimale.

Il y'a moyen d'ajouter un circuit de phase pour agrandir le domaine de fréquence pour lequel ces conditions de tangence sont valides.

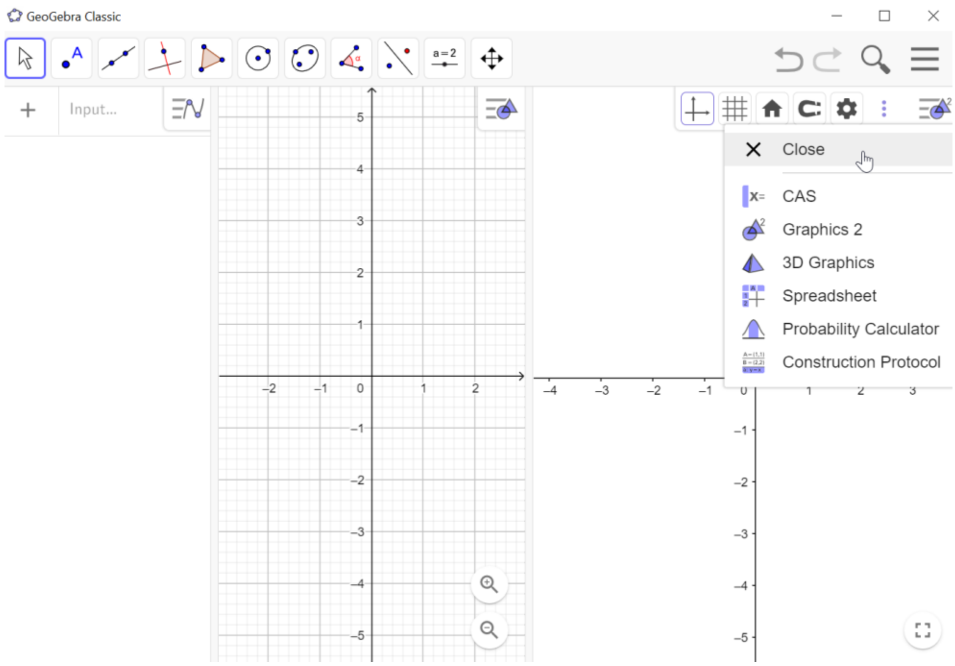


3-9 Exemple #9 : L'ajout de courbes U, S et T

GeoGebra donne la possibilité de représenter un deuxième graphique associé au premier. Le deuxième graphique peut avoir différents repères. Dans le menu **View**, ouvrir une seconde fenêtre graphique en sélectionnant **Graphics 2**.



La fenêtre **Graphics 2** peut être fermée en cliquant sur **Close** dans le menu à droite.



Dans l'exemple qui suit, Le diagramme de Nyquist ainsi que les cercles de sensibilité M et ε sont visualisés dans la fenêtre graphique principale (à gauche) et les courbes, transmittance $|T(\omega)|$, sensibilité $|S(\omega)|$ et signal de commande $|U(\omega)|$ sont ajoutées dans la fenêtre **Graphics 2** (à droite). Les limites max et min peuvent être ajoutées aussi dans la fenêtre **Graphics 2** comme ε, M_s et U_{max}.