OML2

Transformée de Laplace, TP1

Nicolas Boizot

April 7, 2026

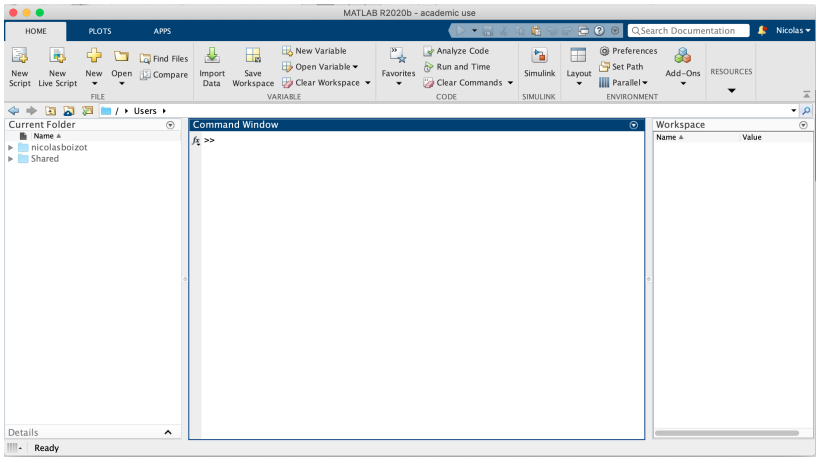
Introduction

TP1 - Startup -

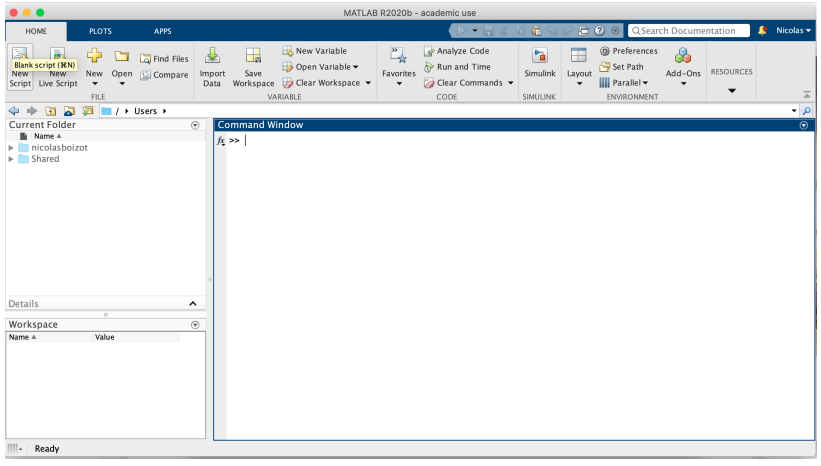
Option 1 : Installation locale.

Option 2 : Utiliser la version en ligne (*Matlab Online*, disponible avec la licence site de l'UTLN).

TP1 - Current Folder / Command Window / Workspace -



(a) Disposition en trois colonnes



(b) Disposition en deux colonnes

Exercice

Créer un répertoire de travail et le définir comme répertoire courant.

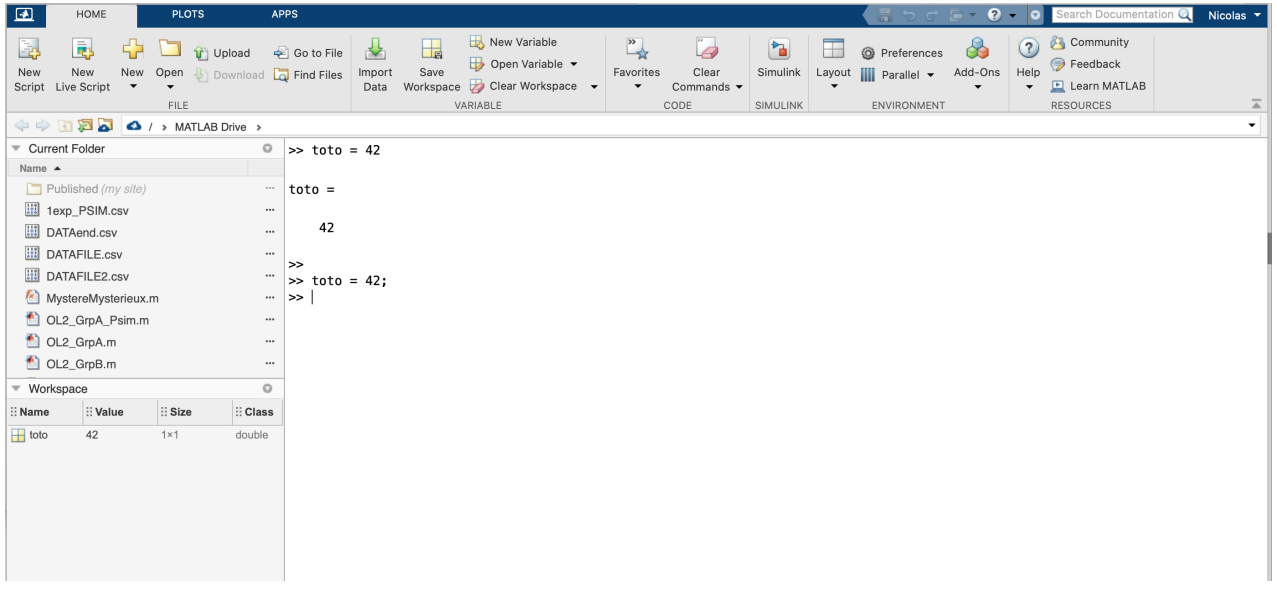
TP1 - toto = 42 ou toto = 42;?

Exécuter la commande

```
toto = 42
```

Puis

```
toto = 42;
```



TP1 - Tracé d'un graphe : principe -

Objectif : tracer le graphe d'une fonction

$$f : \mathbb{R} \to \mathbb{R}$$
$$t \to f(t)$$

Principales étapes : (1) Choisir un intervalle $[a, b] \in \mathbb{R}$

Exemple : $[0, 5] \in \mathbb{R}$

TP1 - Tracé d'un graphe : principe -

Objectif : tracer le graphe d'une fonction

$$f : \mathbb{R} \to \mathbb{R}$$
$$t \to f(t)$$

Principales étapes : (2) choisir des valeurs de t dans cet intervalle.
(appelons ça la Base de Temps)

Exemple :

t	0	0,7	1,1	2	3	5

TP1 - Tracé d'un graphe : principe -

Objectif : tracer le graphe d'une fonction

$$f : \mathbb{R} \to \mathbb{R}$$
$$t \to f(t)$$

Principales étapes : (3) Calculer les valeurs de $f(t)$ correspondantes.

Exemple :

t	0	0,7	1,1	2	3	5
f(t)	0	0,49	1,21	4	9	25

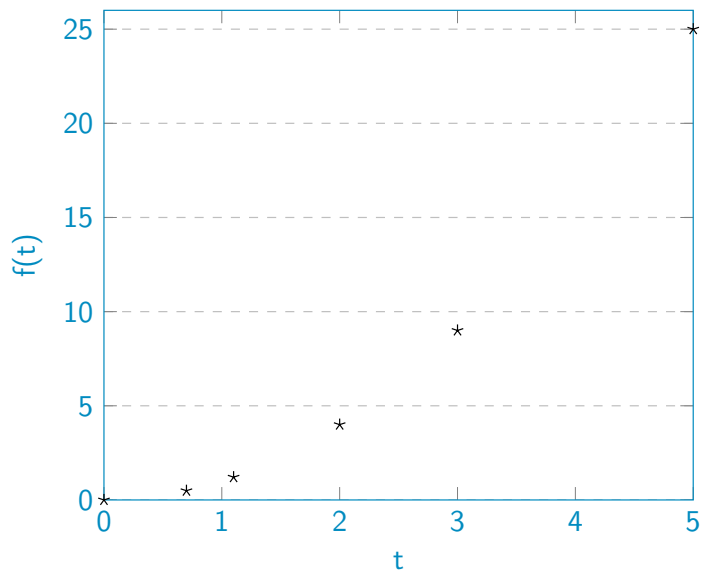
TP1 - Tracé d'un graphe : principe -

Objectif : tracer le graphe d'une fonction

$$f : \mathbb{R} \to \mathbb{R}$$
$$t \to f(t)$$

Principales étapes : (4) Marquer les points $(t, f(t))$ dans un repère du plan.

Exemple :



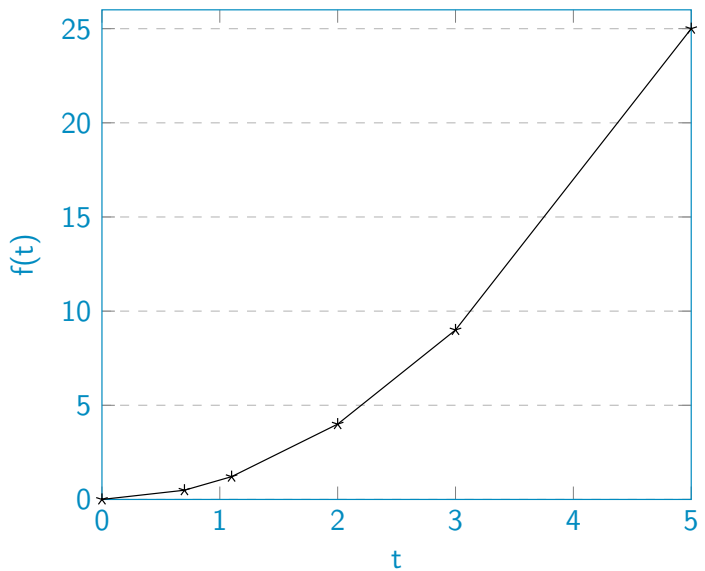
TP1 - Tracé d'un graphe : principe -

Objectif : tracer le graphe d'une fonction

$$f : \mathbb{R} \to \mathbb{R}$$
$$t \to f(t)$$

Principales étapes : (5) Les points sont ensuite reliés entre eux.

Exemple :



TP1 - Tracé d'un graphe.
un exemple sous Matlab $->$ Etape 1

L'intervalle $[a, b]$ sera représenté par une suite croissante de réels $(a < \ldots < \ldots < b)$ et stockée dans un tableau.
J'appelle ce tableau la **base de temps**.

```
debut = 0;  % c'est le a de l'intervalle [a,b]
fin =  5; % c'est le b de l'intervalle [a,b]
pas = 0.1; % c'est la difference entre deux elements ...
    successif de [a,b]
Base2Temps = debut : pas : fin;
```

Remarque : si je défini la variable *Base2Temps* sans définir le pas, alors, Matlab utilisera un pas de valeur 1. Syntaxe :

```
Base2Temps = debut : fin;
```

TP1 - Tracé d'un graphe.
un exemple sous Matlab $->$ Etape 2

Dans un premier temps, nous allons représenter le graphe de la fonction $f(t) = t^2$.
Nous devons donc créer un tableau contenant les valeurs de la fonction $f(t)$ correspondant à chacun des éléments de la **base de temps**.
Rien de plus simple.

```
y = Base2Temps.^2;
```

Remarques :

- `^` indique qu'une quantité va être évaluée *à une certaine puissance*,`^2` indique donc le passage au carré.
- **- très très important -** pour les opération `^`, `*` (multiplication) et `/` (division) nous utiliserons toujours les codes `.^`, `.*` et `./` si l'opération est réalisée entre deux tableaux de valeurs.

TP1 - Tracé d'un graphe.
Un exemple sous Matlab $->$ Etapes 3 et 4

La commande suivante permet de tracer le graphe de $f(t) = t^2$ sur l'intervalle $[debut, fin]$.

```
plot(Base2Temps,y)
```

La méthode la plus simple pour modifier l'apparence du tracé consiste à indiquer un code de la manière suivante

```
plot(Base2Temps,y,'r') % le graphe est trace en rouge
```

```
plot(Base2Temps,y,'+') % seuls les points servant au trace ...
    sont apparents (sous la forme de croix)
```

```
plot(Base2Temps,y,'k:') % trace en noir et en pointilles
```

TP1 - Tracé d'un graphe : un exemple sous Matlab - A vous de jouer

Il est possible d'accéder à un récapitulatif des possibilités en tapant dans la console la commande

```
help plot
```

Exercice

On s'intéresse au tracé de la fonction $f(t) = \frac{sin(t)}{t}$ sur l'intervalle $[-8, 8]$.

- Tracez ce graphe en utilisant un pas de 1, 0.5, 0.1 ou encore 0.55. Qu'observez vous ?
- Que se passe t'il en 0 ?

Place aux scripts

TP1 - Scripts -

- C'est bien joli tout ça, mais tout notre travail est perdu !!
- Pour remédier à cela, nous allons maintenant utiliser l'éditeur de Matlab pour consigner notre code dans un fichier Matlab dédié (dont l'extension sera .m).
- Deux propositions simples :
  1. écrire une fonction.
     Les fonctions peuvent accepter des arguments en entrée et retourner des arguments en sortie. Les variables internes sont locales à la fonction.
     Autrement dit, les fonctions ont leur propre workspace et n'ont pas accès au workspace "public".
  2. écrire un script.
     Les scripts n'acceptent pas d'arguments en entrée et ne retournent pas d'arguments en sortie. Ils opèrent directement du workspace.

TP1 - Scripts - Trois commandes à inclure au début de vos scripts

Pensez à inclure les trois commandes suivantes au début de tous vos scripts.

```
clear variables % efface toute les variables du workspace

close all % ferme toutes les fenetres graphiques

clc % efface la console
```

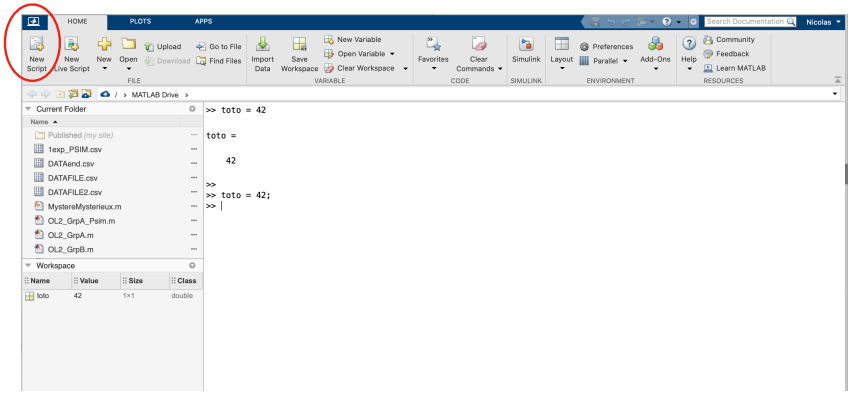
TP1 - Scripts -
A vous de jouer

Exercice

Créez un script dans lequel vous :

- définirez une base de temps;
- tracerez le graphe de la fonction $\frac{sin(t)}{t}$ sur la base de temps choisie.
  Important : choisir un pas permettant une bonne visualisation du graphe.

```
% Ceci est un commentaire
% Dans Matlab, les commentaires sont indiques a l'aide d'un %
```



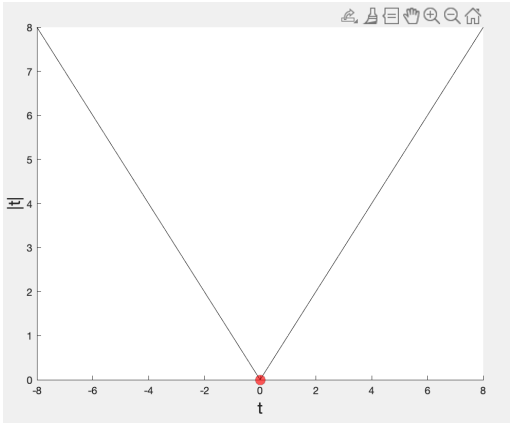
Jouer avec les tracés graphiques

TP1 - Tracé d'un graphe à l'aide de Matlab - Légendes, labels, texte et autres joyeusetés

- ajouter un label à l'un des axes :
  `xlabel('...mon texte...')` (abscisses),
  `ylabel('...mon texte...')` (ordonnées)
- ajouter un titre : `title('...mon titre...')`
- changer l'épaisseur du trait : `plot(.....,'LineWidth',2)`
  Permet d'obtenir un tracé d'épaisseur 2.
- Placer du texte au niveau du point de coordonnées $(\mathbf{x}, \mathbf{y})$:
  `text(x,y,'...du texte...')`
- tracer plusieurs courbes sur un même graphique :

```
hold on
%placer ici les commandes graphiques
hold off
```

TP1 - Tracé d'un graphe à l'aide de Matlab - Marquer un point d'intérêt



Mettre en évidence un point (x_1, y_1) du graphique.

```
scatter(x1,y1, % position du marqueur
100,'o', % taille (100) et forme du marqueur ('o')
'MarkerEdgeColor','k', % couleur du contour du marqueur
'MarkerFaceAlpha',0.2, % transparence de 20%
'MarkerFaceColor','r', % couleur interieure du marqueur
'MarkerEdgeAlpha',0.2) %  transparence de 20%
```

Note : il s'agit d'un détournement de la commande *scatter* qui sert à visualiser de nuages de points.

TP1 - Tracé d'un graphe à l'aide de Matlab - Marquer un point d'intérêt (fonction)

```
function PointInteret(x1,y1,color)
% 'b'  (blue); 'g' (green); 'r' (red); 'c' (cyan);
%  'm' (magenta);
% 'y' (yellow); 'k' (black); 'w' (white)

scatter(x1,y1, 100,'o', ...
    'MarkerEdgeColor','k','MarkerFaceAlpha', ...
   0.2,'MarkerFaceColor',color,'MarkerEdgeAlpha',0.2);
```

- Copier le code ci-dessus dans un script vide.
- Sauvegarder dans le répertoire courant/de travail (*PointInteret.m*)

$\Rightarrow$ La commande PointInteret(x1,y1,'r') fait apparaître un point d'intérêt rouge en $(x1, y1)$.

TP1 - Tracé d'un graphe à l'aide de Matlab - Légendes, labels, texte et autres joyeusetés

Tracé de plusieurs graphes **non-superposés** dans une même fenêtre graphique : `subplot(li,col,z)`

li - nombre de lignes;
col - nombre de colonnes;
z - fenêtre dans laquelle le tracé est effectué.

Un exemple à une ligne et deux colonnes.

```
% 1 ligne, 2 colonnes, ...
    graphe 1
subplot(1,2,1)
plot(0,0)
% 1lignes, 2 colonnes, ...
    graphe 2
subplot(1,2,2)
plot(0,0)
```

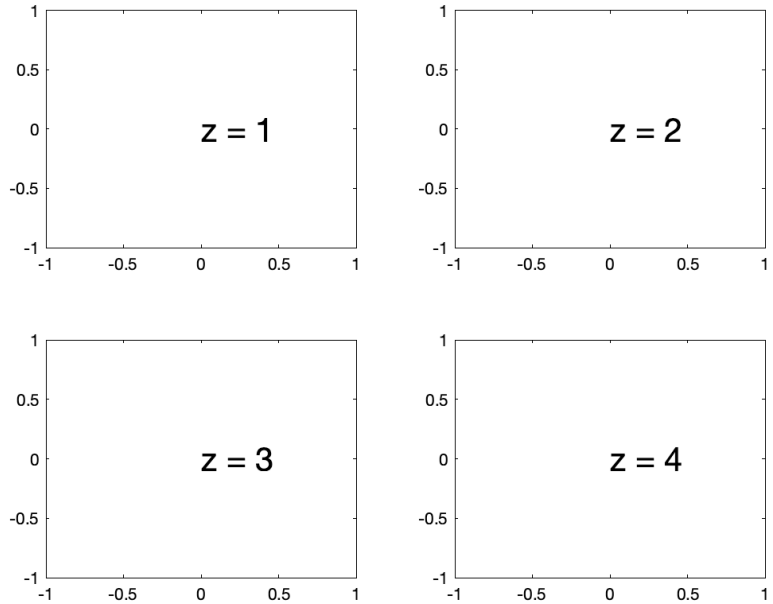
Un exemple à deux lignes et une colonne.

```
% 2 lignes, 1 colonnes, ...
    graphe 1
subplot(2,1,1)
plot(0,0)
% 2 lignes, 1 colonnes, ...
    graphe 2
subplot(2,1,2)
plot(0,0)
```

TP1 - Tracé d'un graphe à l'aide de Matlab - Légendes, labels, texte et autres joyeusetés

```
% 2 lignes, 2 colonnes, fenetre 1
subplot(2,2,1)
plot(0,0)
text(0,0,'z = 1','Fontsize',20)
% 2 lignes, 2 colonnes, fenetre 2
subplot(2,2,2)
plot(0,0)
text(0,0,'z = 2','Fontsize',20)
% 2 lignes, 2 colonnes, fenetre 3
subplot(2,2,3)
plot(0,0)
text(0,0,'z = 3','Fontsize',20)
% 2 lignes, 2 colonnes, fenetre 4
subplot(2,2,4)
plot(0,0)
text(0,0,'z = 4','Fontsize',20)
```

Graphique correspondant au code ci-contre



TP1 - Tracé d'un graphe à l'aide de Matlab - Tracer un ligne Horizontale, ou verticale.

Horizontale :

```
plot(0,0) % cree une fenetre graphique
yline(0)  % yline(valeur)
```

Verticale :

```
plot(0,0) % cree une fenetre graphique
xline(0) % xline(valeur)
```

Segment :

```
plot(0,0) % cree une fenetre graphique
line([-1, 1], [-1,1]) % line([x1, x2],[y1, y2]) : segment de ...
    droite entre (x1,y1) et (x2,y2)
```

N'oubliez pas de combiner ces commandes avec des "*hold on*...*hold off*"

Ces commandes nous seront utiles au second TP.

Construction de signaux classiques

TP1 - Echelon de Heaviside

$$H(t) = \begin{cases} 0 \text{ si } t < 0 \\ 1 \text{ si } t \geq 0 \end{cases} \quad \begin{array}{c} \xrightarrow{TL} \\ \xleftarrow[TL^{-1}]{} \end{array} \quad \frac{1}{s}$$

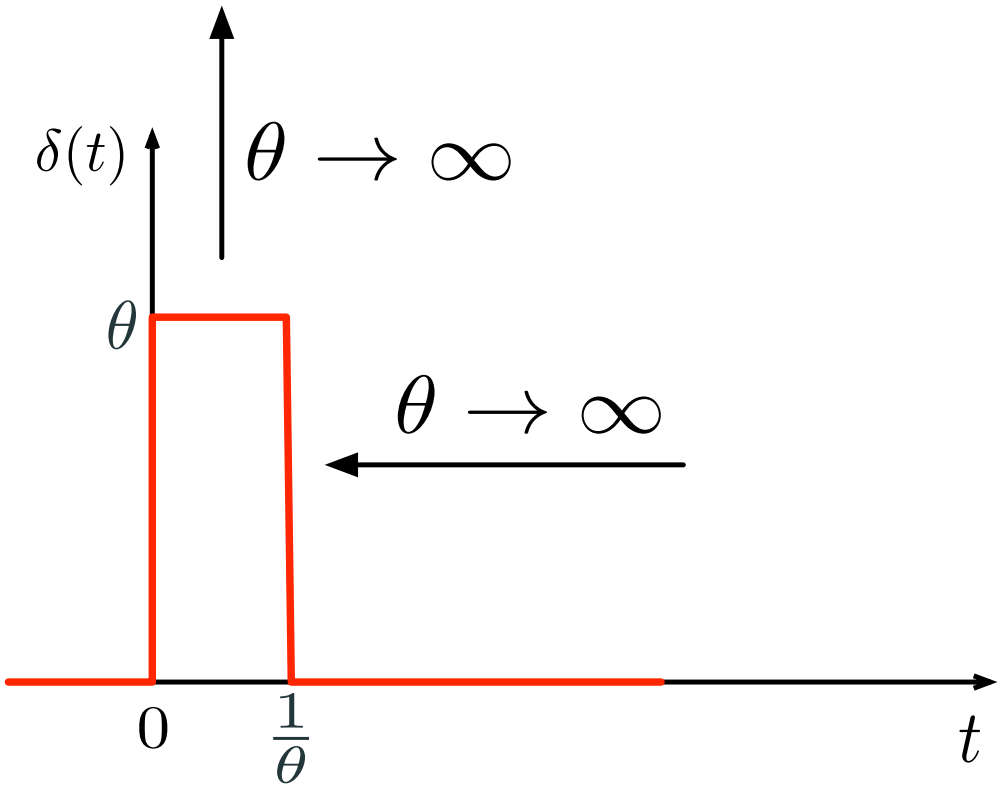
TP1 - Echelon de Heaviside (code)

Heaviside défini par nos soins

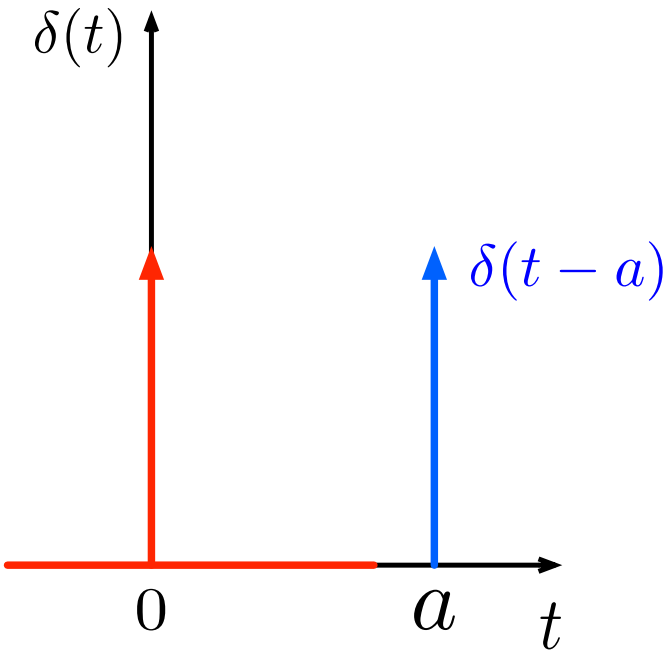
(nous n'utilisons pas la fonction toute faite)

```
unitstep = Base2Temps>=0;
```

TP 1 - Impulsion de Dirac



"Dérivée" de l'échelon $\Rightarrow$ Impulsion de Dirac

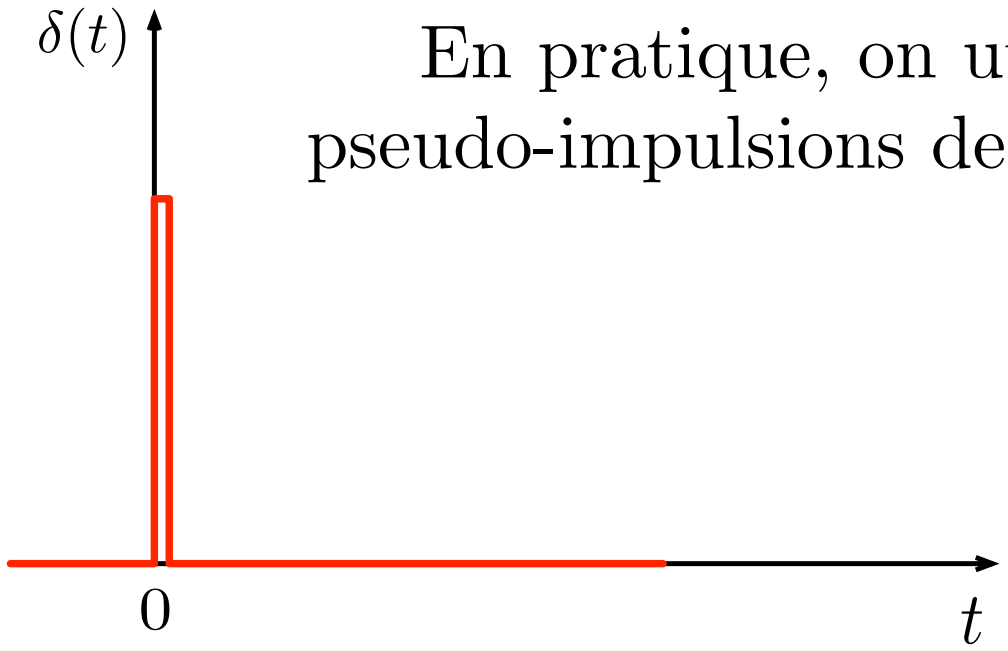


Propriétés :

$$\int_{-\infty}^{+\infty} \delta(t)dt = 1$$

$$\int_{-\infty}^{+\infty} \delta(t)f(t)dt = f(0)$$

$$\int_{-\infty}^{+\infty} \delta(t-a)f(t)dt = f(a)$$



En pratique, on utilise des pseudo-impulsions de Dirac.

$$\delta(t) \quad \underset{TL^{-1}}{\overset{TL}{\rightleftarrows}} \quad 1$$

TP1 - Impulsion de Dirac (code)

Dirac défini par nos soins
(nous n'utilisons pas la fonction tout faite)

```
impulse = (Base2Temps==0);
```

Attention : cela n'a réellement de sens que si 0 est dans la Base de temps (faites des tests pour voir).

TP1 - Signal sinusoïdal – commande *sin( )*

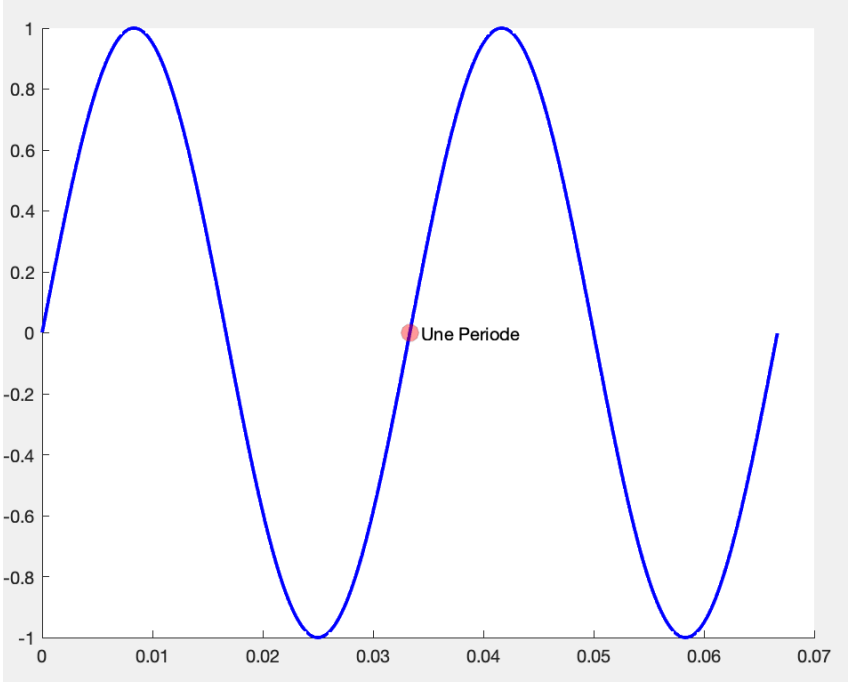
La fonction $\sin(\omega t)$, où $\omega > 0$ est une constante.

Fonction sinusoïdale périodique basée sur la sinus.

De période égale à $P = \frac{2\pi}{\omega}$ *(période)*

De fréquence égale à $f = \frac{\omega}{2\pi}$ *(fréquence)*.

ω s'appelle la **pulsation**.



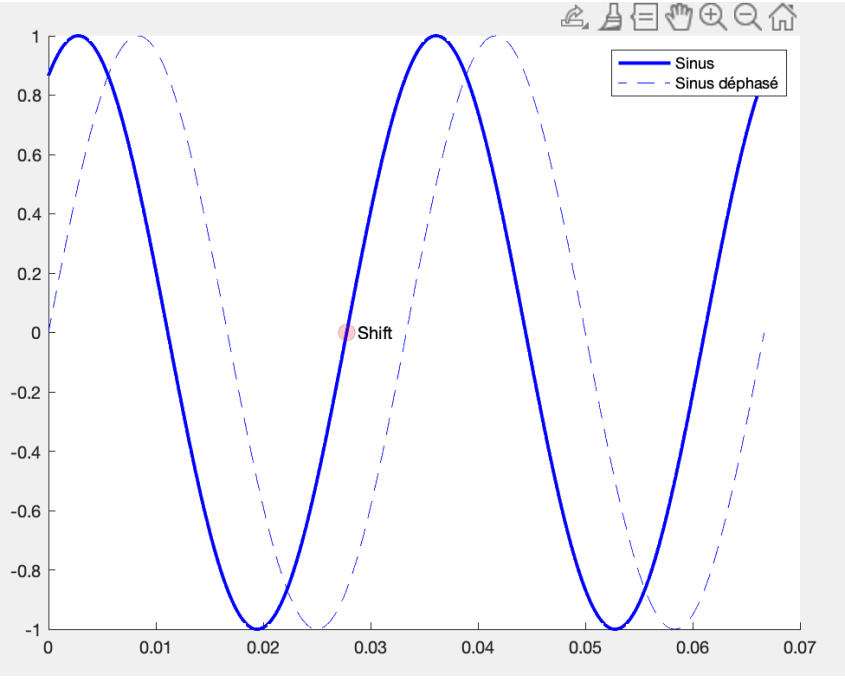
TP1 - Signal sinusoïdal – commande *sin( )*

```
debut = 0;
Freq = 30;
P = 1/30;
fin = 2*P; % on affiche deux periodes

pas = %... a completer...
Base2Temps = %... a completer...
signal = sin(2*pi*Freq*Base2Temps); % sin(2*pi*w*t)

x1 = P; % Abscisse d'une periode
y1 = sin(2*pi*Freq*P); % ordonnee correspondante

hold on % permet de superposer les traces sur une meme figure
plot(Base2Temps,signal,'b','LineWidth',2);
PointInteret(x1,y1,'r') % Point d'interet rouge
hold off % relache la commande hold on
text(P+0.001,sin(2*pi*Freq*P),'Une Periode')
```

TP1 - signal sinusoïdal – commande *sin( )*

Fonction $\sin(\omega t + \phi)$, où $\omega > 0$ est une constante.

Fonction sinusoïdale des diapositives précédentes, décalée sur la droite d'un temps égal à $\frac{2*\pi-\phi}{\omega}$.



TP1 - signal sinusoïdal – commande *sin( )*

```
debut = 0;
Freq = 30;
P = 1/30;
fin = 2*P; % on represente deux periodes

Omega = 2*pi*Freq;
phi = pi/3;
pas = %...a completer...;
Base2Temps = %...a completer...;

signal = sin(2*pi*Freq*Base2Temps+phi); % sin(2*pi*f*t+phi)

x1 = (2*pi-phi)/Omega; % Abscisse du decalage / dephasage
y1 = 0; % sin(0)

hold on % permet de superposer les traces sur une meme figure
plot(Base2Temps,signal,'b','LineWidth',2);
PointInteret(x1,y1,'r') % Point d'interet rouge
hold off % relache la commande hold on
text((2*pi-phi)/Omega+0.001,0,'Shift')
```

TP1 - Signal carré — commande *square*

Signal carré de **période 2*pi**, **d'amplitude 1**

```
square(T)  %(Fonctionne donc comme le sinus.)
```

Signal carré avec rapport cyclique

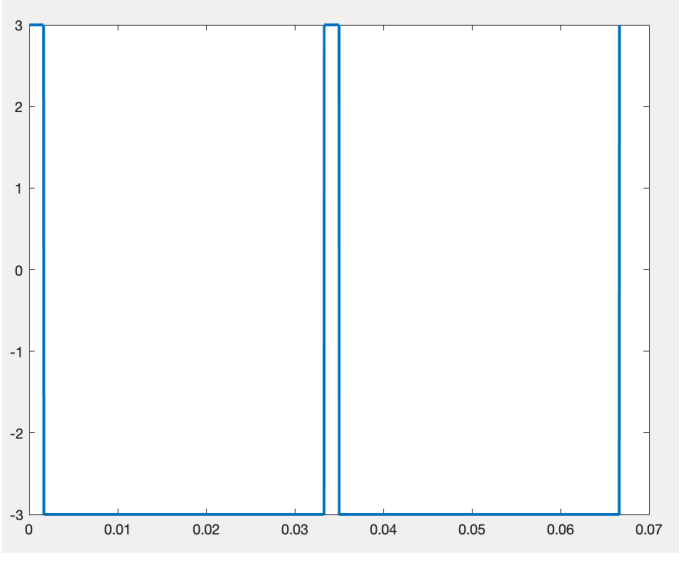
```
square(T,DUTY) %signal d'onde carree de periode 2*pi
% Avec un rapport cyclique de DUTY %
%
% DUTY indique le % de la periode pendant lequel le signal ...
    reste etat HAUT
%
% square(T) et square(T,50) font la meme chose
```

TP1 - Signal carré — commande *square*

Exercice

Tracer deux périodes d'un signal carré

- de fréquence 30 Hz;
- d'amplitude 3;
- de rapport cyclique 5%.



TP1 - Signal PWM et signal triangulaire

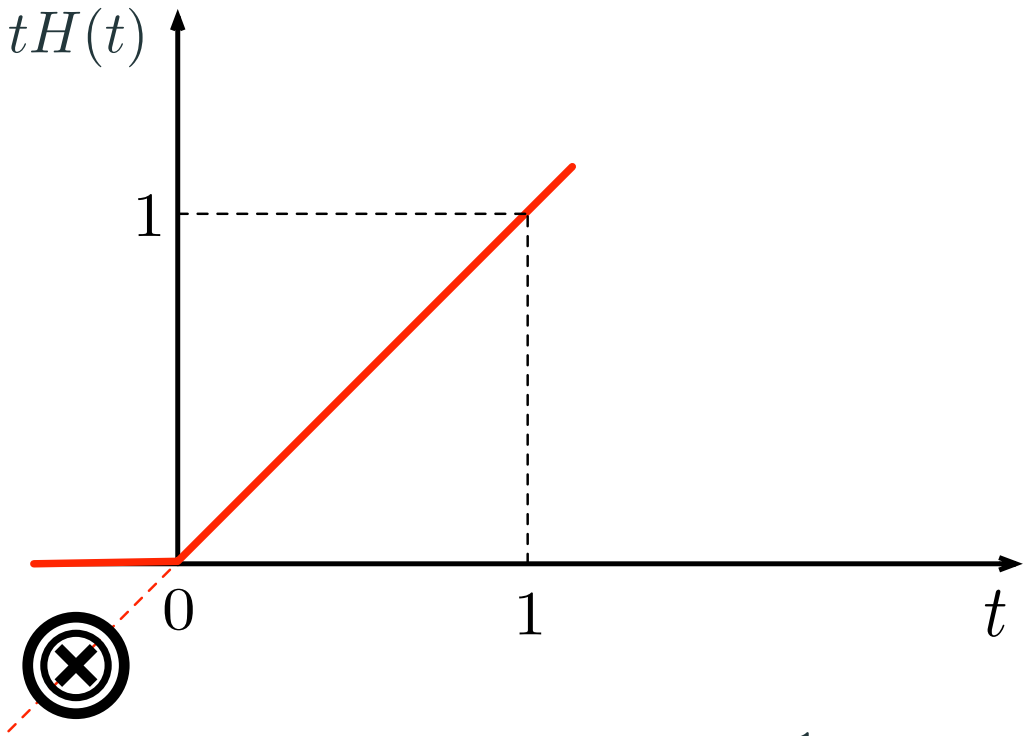
Signal carré d'amplitude allant de 0 à 1 (au lieu de -1, 1) et de fréquence donnnée.

```
square(T,DUTY) %signal d'onde carree de periode 2*pi
% Avec un rapport cyclique de DUTY %
%
% square(2*pi*FREQ*T,DUTY) %signal d'onde carree de ...
    frequence <FREQ>
%
% on ajoute 1 : signal entre 0 et 2
%
% on divise par 2.
```

Construction d'(autres) signaux classiques

TP1 - Rampe

Remarque : pour rendre un signal causal, on le multiplie par un échelon (comme dans la table des transformées) *de sorte à le rendre causal* (c-à-d. nul pour les temps négatifs).



$$tH(t) \quad \underset{\mathcal{TL}^{-1}}{\overset{\mathcal{TL}}{\rightleftarrows}} \quad \frac{1}{s^2}$$

TP1 - Rampe et Parabole unitaire

Rampe

```
ramp = Base2Temps.*unitstep;  % Attention au .*
plot(Base2Temps,ramp,'r','LineWidth',2);
```

Parabole unitaire

```
quad = Base2Temps.^2.*unitstep; % Attention au .*
plot(Base2Temps,quad,'r','LineWidth',2);
```

TP1 - Signal PWM et signal triangulaire

Signal carré d'amplitude allant de 0 à 1 (au lieu de -1, 1) et de fréquence donnnée.

```
square(T,DUTY) %signal d'onde carree de periode 2*pi
% Avec un rapport cyclique de DUTY %
%
% square(2*pi*FREQ*T,DUTY) %signal d'onde carree de ...
    frequence <FREQ>
%
% on ajoute 1 : signal entre 0 et 2
%
% on divise par 2.
```

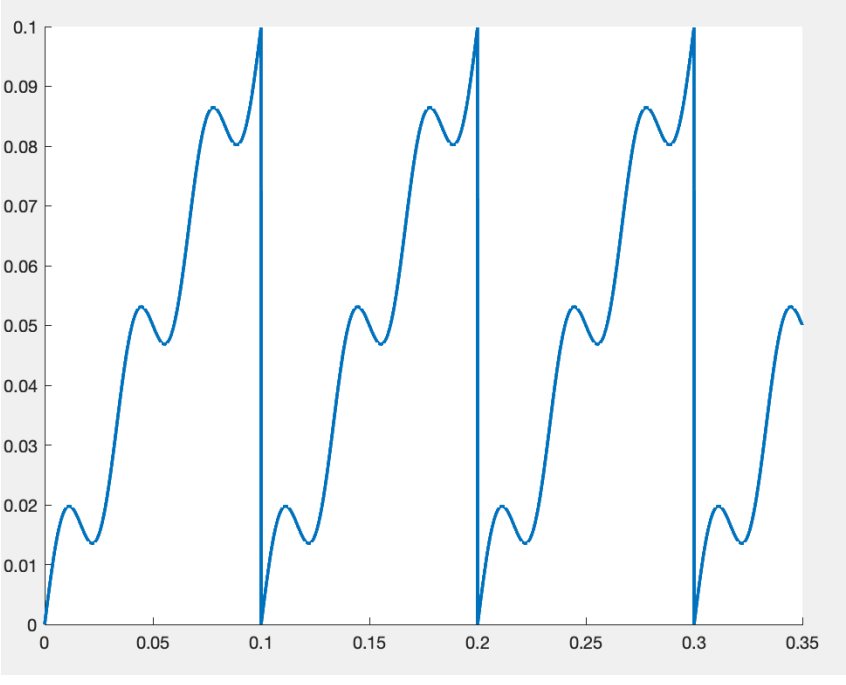
Signal triangulaire

```
signal = sawtooth(t) % Fonctionne comme sin : signal ...
    triangle entre -1 et 1 de periode 2 pi
```

TP1 - Signal périodique

Le signal est toujours tracé relativement à sa *Base de Temps*, cependant il est **crée** en utilisant une *Base de temps modifiée* (à l'aide le a commande modulo = reste d'un division euclidienne).

Exemple d'une rampe portant un sinus de fréquence $30Hz$, répétée selon une période 0.1.



TP1 - Signal périodique

```
debut = 0;
Freq = 30; % frequence du sinus
Omega = 2*pi*Freq;
P = 0.1; % une periode dure 0.1 unite de temps

fin = 3.5*P; % on represente trois periodes et demi

pas = 1/(Freq*1000);
Base2Temps = debut:pas:fin;

Base2Temps_modif = mod(Base2Temps,P); % mod : commande modulo

signal1 = Base2Temps_modif;
signal2 = 0.01*sin(2*pi*Freq*Base2Temps_modif);
signal = signal1 + signal2;

plot(Base2Temps,signal,'LineWidth',2);
```