# Traitement du signal – MATLAB

**MATLAB** (« *matrix laboratory* ») est un langage de script émulé par un environnement de développement du même nom ; il est utilisé à des fins de calcul numérique. Développé par la société The MathWorks, MATLAB permet de manipuler des matrices, d'afficher des courbes et des données, de mettre en œuvre des algorithmes, de créer des interfaces utilisateurs, et peut s’interfacer avec d’autres langages comme le [C](https://fr.wikipedia.org/wiki/C_(langage)), C++, Java, et Fortran. Les utilisateurs de MATLAB (environ 4 millions en 2019) sont de milieux très différents comme l’ingénierie, les sciences et l’économie dans un contexte aussi bien industriel que pour la recherche. Matlab peut s’utiliser seul ou bien avec des *toolboxes* (« boîte à outils »).

Dans cette étude nous utiliserons MATLAB pour programmer des courbes sinusoïdales et nous utiliserons un de ces modules (Simulink) pour effectuer des opérations de traitement analogique.

## Le régime sinusoïdal :

* **Ouvrir** MATLAB.
* **Taper** les lignes de code suivantes :

x=0 :0.001 :10 // crée un tableau x allant de 0 à 10 par pas de 0.001

y=sin(x) // crée un tableau y fonction de x

plot(x,y) // affichage de la courbe x en abscisse, y en ordonnée

Vous venez de créer une sinusoïde simple dont vous pouvez retrouver l’équation grâce au cours.

Question 1 : **Donner** l’équation de la sinusoïde tracée.

* **Créer** un signal sinusoïdal y1(x) de fréquence 100 Hz, d’amplitude 0.5 avec une phase à 0 rad sur un temps allant de 0 à 0.04 secondes avec un pas de 50 μs et **afficher** ce signal.
* De la même façon et sur le même graphique, **créer** un signal sinusoïdal y2(x) de fréquence 200 Hz, d’amplitude 0,5 avec une phase à 0 rad.

**REMARQUE :** Pour mettre deux courbes sur le même graphique écrire « plot(x,y1,x,y2) », y1(x) et y2(x) étant les fonctions à tracer.

* **Créer** maintenant la fonction y3(x) : addition de y1 et y2. (Une seule ligne de commande suffit).
* **Tracer** la sur le même graphique que les deux autres.

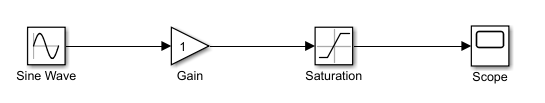
Question 2 : **Copier-coller** la capture d’écran du graphique obtenu.

* **Refaire** les étapes précédentes pour **créer** y4, soustraction de y1 et y2.
* **Tracer** sur le même graphique y1, y2 et y4.

Question 3 : **Copier-coller** la capture d’écran du graphique obtenu.

## Les opérations

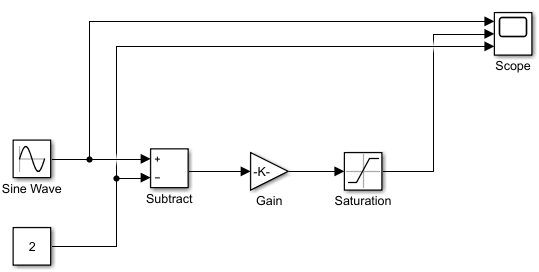
* **Ouvrir** Simulink dans MATLAB.
* **Choisir** « blanck model ».
* En allant chercher dans la bibliothèque **créer** le schéma ci-dessous :

*En faisant un double clic gauche dans la fenêtre Simulink vous pouvez rechercher le nom des blocs pour y accéder directement.*

* En double-cliquant sur les blocs **paramétrer** la sinusoïde de départ à une amplitude de 1 et une fréquence de 105 Hz. Paramétrer aussi la saturation entre -3 et 3.
* En allant dans les paramètres de Simulink **régler** le temps de simulation allant de 0 à 1 seconde **régler** le en pas fixe à une valeur de 0.001.
* **Cliquer** ensuite sur run pour lancer la simulation. **Observer** le résultat en double-cliquant sur le scope.
* **Régler** maintenant le gain sur 6. **Relancer** la simulation et regarder de nouveau le scope.

Question 4 : **Expliquer** ce que fait ce montage.

* **Réaliser** le schéma ci-dessous :

****

**REMARQUE :** Pour dédoubler une connexion, clic droit sur le point d’intersection à créer.

* **Régler** l’amplitude du signal d’entrée à 3 avec une fréquence de 105 Hz. **Régler** le gain à 10000, les limites de saturation à -5/5 ainsi que la constante à 2. Le temps d’exécution reste le même. **Lancer** la simulation et **observer** le scope.

Question 5 : **Expliquer** ce que fait ce montage.

* **Modifier** la constante à 0. **Relancer** la simulation. **Comparer** vos deux résultats.

Question 6 : **Expliquer** le rôle de cette constante.

## Echantillonneur

* Toujours sur Simulink, **faire** un montage permettant d’obtenir le même graphique que celui de la question 2 avec y1, y2 et y3. **Simuler** votre montage pour **vérifier** son bon fonctionnement.
* **Dupliquer** le signal y3 pour ajouter un bloc « quantizer » sur l’un avec en paramètre un « quantization interval » de 0.3. **Simuler** le montage.

Question 7 : **Expliquer** le rôle du bloc « quantizer ».

Question 8 : **Copier-coller** la capture d’écran du scope obtenu et **donner** la description de chaque courbe de différente couleur.

Question 9 : **Expliquer** le rôle du paramètre « quantization interval ».