

Exercices programmation – Matlab 2

Rappel matlab

Le help contient l'entier de la documentation

Find()

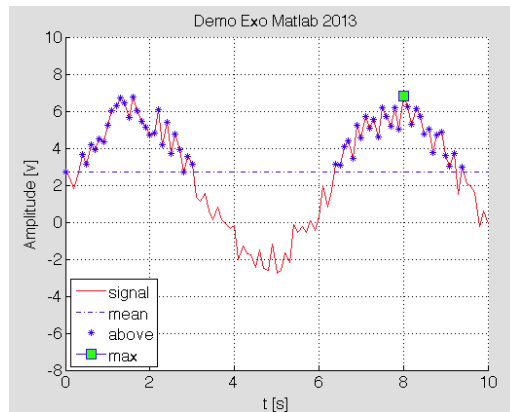
$X = [0.2, 0.6, -1.2, 3.1, 10.3, 100.5, 1000.8]$

A l'aide de `find()` cherchez les éléments de $X > 1$.

Affichez uniquement ces éléments dans un graph à l'aide de `plot()`

Affichez uniquement les éléments dans un graph avec une échelle semi log en Y à l'aide de `semilogy()`

Plot()



Créez le plot ci-dessus à l'aide des commandes suivantes:

Utilisez `doc/help` pour connaître les fonctions manquantes et leurs paramètres.

```
clear all           % efface toutes les variables du workspace
close all          % ferme toutes les fenetres
figure;           % new figure
```

```
h=0.1;
X=0:h:10;         % x vector
```

Générez le vecteur $Y = f(X)$, avec $f() = a \cdot \sin(x) + b \cdot \text{rand} + c$; a, b, c à choix

Y=...

```
plot(X,Y, 'r');    % plot Y vs X en rouge
```

```
hold on;          % stack plots
grid on;          % affiche la grille
xlabel('t [s]');  % affiche les labels et titre
ylabel('Amplitude [v]');
title('Demo Exo Matlab 2013')
```

Calculez la valeur moyenne de Y

m=...

Affichez la moyenne comme une droite (vecteur avec 2 pts) en pointillés bleus ('-.b').

mY = [m m]

Quelles sont les valeurs de mX pour mY défini ci-dessus?

mX = ...

plot(...'-.b')

Affichez un carré vert à l'emplacement (maxX, maxY) de la valeur maximum d' Y (à calculer)

maxX, maxY = ...

plot(maxX, maxY, ...
'marker', 'square', 'markersize', 12, 'markeredgecolor', 'b', 'markerfacecolor', 'g')

Sélectionnez les valeurs plus grande que la valeur moyenne et mettez-les en évidence par une étoile bleu ('b*'). Utilisez find() pour sélectionner les indexes des valeurs puis générez les X et Y correspondants.

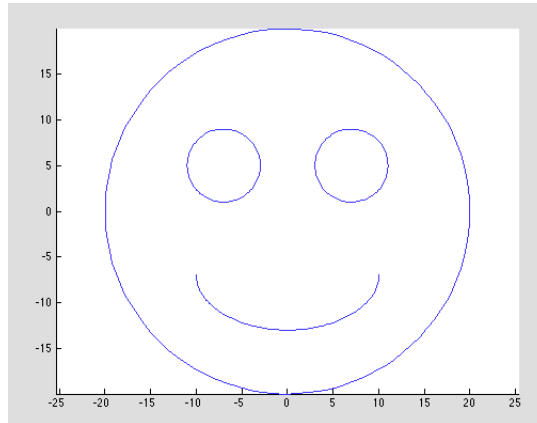
Changez les valeurs min et max de l'axe Y à l'aide de la fonction axis(). Comment faire pour garder les valeurs courantes de l'axe X ?

Créez une legend en bas à gauche ('Location', 'SouthWest') à l'aide de legend().

Créez un fichier pdf à partir de la figure courante

saveas(gca, 'demoExo.pdf')

Function smile() – optionnel



Dans Matlab, écrire une fonction qui dessine un *smile* à l'aide de 3 cercles et une ellipse. Pour cela faite un script qui contient les fonctions suivantes :

Définir une fonction `mydrawCircle()` pour afficher cercle fait d'une courbe paramétrique où $x=r*\sin(v)+offsetX$ et $y = r*\cos(v)+offsetY$ avec $v= -pi : 0.1*pi : pi$. *center* est un vecteur.

```
function mydrawCircle(radius, center)
v=...
x=... % x=f(radius,v,center)
y=... % y=f(radius,v,center)
plot(x, y);
end
```

Faire de même pour afficher une ellipse `mydrawEllipse()`. La fonction a comme paramètre de retour (`boucheHandle`) un *handle* sur le plot qui nous permettra de changer les valeurs du plot par la suite. Avec $x=r1*\sin(v)+offsetX$ et $y = r2*\cos(v)+offsetY$, avec $v= -pi : 0.1*pi : 0$

```
function [boucheHandle]=mydrawEllipse(radius1,radius2, center)
...
end
```

Faire une procédure `drawSmiley()` qui dessine le *smile*. La taille de la tête est passée en paramètre (`size`). Employer les commandes

- hold on** pour afficher plusieurs plots sur la même figure
- axis equal** pour avoir de vrais cercles (échelle x = échelle y)
- f=figure** pour créer une nouvelle fenêtre
- figure(f)** pour la mettre au 1^{er} plan

```
function [boucheHandle] = drawSmiley(size)
f=figure; % nouvelle fenetre
hold on % stocker les graphiques
axis equal; % Sinon le cercle apparait oval

... % dessin du smile

figure(f) % amene la fenetre au premier plan
end
```

Animation du Smile

Ecrire une fonction `doSmile()` qui permet de modifier la forme de la bouche, i.e. changer la valeur du rayon de la demi ellipse ($r = 0$ -> ligne horizontale, $r > 0$ ellipse)

Le paramètre de sortie *boucheHandle* retourne une référence sur le plot de la bouche du *smile*.

Pour cela accéder aux points X et Y du plot définissant la bouche, utiliser la fonction ,

get(bouchehandle, 'YData', ...) et **set(bouchehandle, 'YData', ...)** voir slide 35 du cours.

Les X de l'ellipse ne change pas, recalculer les Y en fonction du nouveau rayon, ne pas oublier d'ajouter l'offset en Y qui se calcul en prenant le 1er point en Y. Le vecteur d'angles v doit être de la même taille que **length(Yoffset)**.

```
function doSmile(boucheHandle, radius2)
Yoffset=get(boucheHandle, 'YData'); % position du 1er pt, defini l'offset en Y
v = ...
y = ... % y=f(radius2,v,Yoffset)

set(boucheHandle, 'YData', y);
end
```

Faite sourire votre *smile* en modifiant l'ouverture de la bouche via `doSmile()`.

```
function MySmiley
boucheHandle=drawSmiley(20);
for n=-1:6
    doSmile(boucheHandle,n); % anime la bouche
    pause(0.1);
end
end
```