```
function optimisation()
% Exemple d'utilisation de l'optimisation numérique
% A. Desbiens, mars 2015
%
% Sur un procédé on a enregistré le vecteur d'entrée u (Nx1)
% et le vecteur de sortie y (NX1) selon une période d'échantillonnage Te.
%
% On désire trouver le modèle [b1*z^(-1) + b2*z^(-2)] / [1 + a1*z^(-1) +a2*z^(-2)]
% qui minimise le critère:
%    J = somme de i= 1 à N de [y(i) - ym(i)]^2
% où ym(i) est la sortie du modèle pour l'entrée u, tout en respectant les
% contraintes suivantes:
%   1) le gain [b1 + b2] / [1 + a1 + a2] du modèle est unitaire:
%      b1 + b2 - a1 -a2 = 1
%   2) le système est stable, i.e. les pôles du modèle ont un module
%      inférieur à l'unité:
%      | [-a1 +- sqrt(a1^2 - 4*a2) ] / 2 | < 1
%   3) 0 < b2 < 1 (juste pour ajouter ce type de contraintes!)

%Génération des données enregistrées
G=tf(1,[100 20 1]);
Te=5;
N=40;
Gd=c2d(G,Te,'zoh');
u=[0; ones(N-1,1)]+0.1*rand(N,1);
y=lsim(Gd,u)+0.1*randn(N,1);


%%%%%%%% ESTIMATION DES PARAMÈTRES %%%%%%

%X = fmincon(FUN,X0,A,B,Aeq,Beq,LB,UB,NONLCON,OPTIONS,autres_arguments_pour_FUN_et_NONLCON)
%     min F(X)  subject to:  A*X  <= B, Aeq*X  = Beq (linear constraints)
%      X                     C(X) <= 0, Ceq(X) = 0   (nonlinear constraints)
%                               LB <= X <= UB        (bounds)

% le vecteur des paramètres inconnus est p=[b1 b2 a1 a2]
p0=[0 0.5 -0.5 0]';  % valeur initiale de p respectant les contraintes

opt = optimoptions('fmincon', 'TolFun', 1e-3); % voir 'help optimoptions'
[p,fval,exitflag]=fmincon(@(p) critere(p,u,y,Te),p0,[],[],[1 1 -1 -1],[1],[-inf 0 -inf -inf],[inf 1 inf inf],@(p) contraintes(p),opt);
if exitflag~=1
    error(['EXITFLAG = ' num2str(exitflag) '. Voir le texte ci-dessus et la section sur l''EXITFLAG en tapant ''help fmincon''.']);
end


%%%%%%%% RÉSULTATS %%%%%%%%%

b1=p(1)
b2=p(2)
a1=p(3)
a2=p(4)
Gm=tf([0 b1 b2],[1 a1 a2],Te);
ym=lsim(Gm,u);
t=(0:N-1)*Te;
plot(t,y,'k-',t,ym,'r-');
title('y et ym');
Gain=sum(Gm.num{:})/sum(Gm.den{:})
Module_des_poles=abs(roots(Gm.den{:}))


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function J=critere(p,u,y,Te)
% Critère à minimiser: doit retourner un scalaire

b1=p(1);
b2=p(2);
a1=p(3);
a2=p(4);
Gm=tf([0 b1 b2],[1 a1 a2],Te);
ym=lsim(Gm,u);

J=sum((ym-y).^2);
% pour un tel critère où J = sum {fonction(p).^2}, la fonction lsqnonlin serait
% plus appropriée que fmincon


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [C,Ceq]=contraintes(p)
% Contraintes non linéaires:
% C(X) <= 0
% Ceq(X) = 0
% On pourrait passer des arguments supplémentaires à la fonction (comme u,
% y ou Te).

b1=p(1);
b2=p(2);
a1=p(3);
a2=p(4);
C=[abs((-a1+sqrt(a1^2-4*a2))/2)-1; abs((-a1-sqrt(a1^2-4*a2))/2)-1];
Ceq=[];
```