clear all; close all; clc

%-------------------------%

% USING MILP TO SOLVE NLP %

%-------------------------%

x = linspace(0,12);

y = x.^2.\*sin(x);

figure, plot(x,y,'linewidth',2), grid on

xlabel('x'), ylabel('y = f(x)')

keyboard

%--------------%

% SOLVE AS NLP %

%--------------%

fun = @(x) x^2\*sin(x);

s1 = fmincon(fun,1,[],[],[],[],0,12)

s2 = fmincon(fun,5,[],[],[],[],0,12)

s3 = fmincon(fun,9,[],[],[],[],0,12)

% Three different guess; three different answers.

hold on, plot([s1,s2,s3],[fun(s1),fun(s2),fun(s3)],'ro','markerfacecolor','r')

keyboard

%---------------%

% SOLVE AS MILP %

%---------------%

% Create the node points...

n = 100;

x = linspace(0,12,n);

y = x.^2.\*sin(x);

% Setup the minimization problem...

prob = optimproblem('ObjectiveSense','minimize');

% Define your variables...

b = optimvar('b',n-1,1, 'Type','integer', 'LowerBound',0,'UpperBound',1);

w = optimvar('w',n,1, 'Type','continuous','LowerBound',0,'UpperBound',1);

% Define your objective...

prob.Objective = sum( y\*w );

% Define your constraints...

prob.Constraints.bsum = sum(b) == 1;

prob.Constraints.wsum = sum(w) == 1;

% Create the SOS constraints...

SOS = zeros(n,n-1);

SOS([1:n+1:end]) = 1;

SOS([2:n+1:end]) = 1;

prob.Constraints.sos = SOS\*b >= w;

% Solve the problem...

sol = solve(prob)

xsol = x\*sol.w

ysol = y\*sol.w

% Plot the results...

figure, plot(x,y,'-o','linewidth',2), grid on, hold on

plot(xsol,ysol,'ro','markerfacecolor','r')

xlabel('x'), ylabel('Piecewise Linear Approximation of f(x)')