Przykład optymalizacji z więzami nieliniowymi

Niech dana jest minimalizowana

funkcja dwóch zmiennych

f(x1,x2) = exp(x1) * *(4*x1*x1+2*x2*x2+4*x1*x2+2*x2+1)

z więzami:

X1*x2-x1-x2+1.5<0

-x1*x2-10<0

Zakładamy co jest (jednym!!!) szukanym wektorem x

X(1) = x1

X(2) = x2

Etap 1:

Piszemy M-plik objfun.m dla funkcji celu

```
function f = objfun(x)
f1 = exp(x(1))*(4*x(1)^2 + 2*x(2)^2
f2=4*x(1)*x(2) + 2*x(2) + 1);
f=f1+f2
```

Etap 2:

Piszemy M-plik confun.m dla więzów.

```
function [c, ceq] = confun(x)
```

% więzy nierównościowe

c = [1.5 + x(1)*x(2) - x(1) - x(2);

-x(1)*x(2) - 10];

% więzy równościowe

ceq = [];

Etap 3:

Aktywujemy procedurę optymalizacyjną.

x0 = [-1;1]; % Przekazujemy opcjonalną

% wartość startową

options = optimset('LargeScale','off');

[x, fval] = ...

fmincon(@objfun,x0,[],[],[],[],[],[],@confun,

options)

Uzyskujemy przykładowy wynik obliczeń:

x =

-9.5474 1.0474

fval = 0.0236

Przykład z wykonywaniem obliczeń gradientów

Etap 1: piszemy M-plik obliczający wartość funkcji celu oraz gradient tej funkcji

```
function [f,G] = objfungrad(x)
f = exp(x(1))*(4*x(1)^2+2*x(2)^2+
4*x(1)*x(2)+2*x(2)+1);
% Gradient funkcji celu
t = exp(x(1))*(4*x(1)^2+2*x(2)^2+
4*x(1)*x(2)+2*x(2)+1);
G = [ t + exp(x(1)) * (8*x(1) + 4*x(2)),
exp(x(1))*(4*x(1)+4*x(2)+2)];
```

Etap 2: Piszemy M-plik obliczający wektory więzów nieliniowych oraz macierz gradientu tych więzów

```
function [c,ceq,DC,DCeq] = confungrad(x)
c(1) = 1.5 + x(1) * x(2) - x(1) - x(2); %Więzy %nieliniowe
c(2) = -x(1) * x(2)-10;
% Gradient tych więzów
DC= [x(2)-1, -x(2);
x(1)-1, -x(1)];
% Zaś przy braku więzów nieliniowych typu %równościowego:
ceq=[];
DCeq = [ ];
```

***Uwaga*: Skoro wprowadziliśmy możliwość analitycznego obliczania gradientów w funkcjach objfungrad.m oraz confungrad.m, to musimy przekazać o tym informację**

aby odpowiednie wzory zostały w ogóle przez procedurę fmincon użyte!!!:

Robi się to poleceniem:

```
options = optimset(options,'GradObj','on',
'GradConstr','on');
```

Etap 3: **Aktywujemy procedurę optymalizacyjną**:

```
x0 = [-1,1]; % Punkt startowy dla niewiadomej
options = optimset('LargeScale','off');
options = optimset(options,'GradObj','on',
'GradConstr','on');
[x,fval] = fmincon(@objfungrad,x0,[],[],[],[],lb,ub,...
@confungrad,options)
```

Otrzymujemy:

```
x =
-9.5474 1.0474
fval =
```

0.0236