UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA MECÂNICA

LISTA DE OTIMIZAÇÃO CLÁSSICA

ARTHUR HENRIQUE IASBECK

UBERLÂNDIA
17 DE OUTUBRO DE 2019

EXERCÍCIO 1

No Exercício 1 foi proposto um problema baseado no posicionamento de carros interconectados por molas, que tinha como objetivo a minimização da energia potencial associada ao sistema. A função a ser minimizada neste caso é apresentada na Eq. 1

$$f(X) = \frac{1}{2}X^T K X - X^T P \tag{1}$$

sendo

$$X = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}^T \tag{2}$$

$$P = \begin{bmatrix} 1000 & 2000 & 3000 \end{bmatrix}^T \tag{3}$$

$$K = \begin{bmatrix} 8500 & -1000 & -2500 \\ -1000 & 3000 & -500 \\ -2500 & -500 & 11500 \end{bmatrix} \tag{4}$$

Efetuando a multiplicação matricial introduzida na Eq. 1 obtém-se $f(x)$, Eq. 5.

$$\begin{aligned} f(x) = 4250x_1^2 - 1000x_1x_2 - 2500x_1x_3 - 1000x_1 + 1500x_2^2 \\ -500x_2x_3 - 2000x_2 + 5750x_3^2 - 3000x_3 \end{aligned} \tag{5}$$

A partir da relação introduzida na Eq. 5 é possível determinar analiticamente $\nabla f(x)$, Eq 6.

$$\nabla f(x) = \begin{bmatrix} 8500x_1 - 1000x_2 - 2500x_3 - 1000 \\ 3000x_2 - 1000x_1 - 500x_3 - 2000 \\ 11500x_3 - 500x_2 - 2500x_1 - 3000 \end{bmatrix} \tag{6}$$

O Método das Variáveis Métricas foi empregado para determinação de x^*. Esta abordagem é baseada no Método de Newton e propõe uma aproximação para a inversa da Matriz Hessiana, que deve ser atualizada recursivamente. Assim sendo, a cada iteração x^{k+1} é determinado a partir das Eqs. 7 e 8.

$$x^{k+1} = x^k + \alpha s^k \tag{7}$$

$$s^k = -\nabla f(x^k) H^k \tag{8}$$

sendo H uma aproximação para a inversa da Matriz Hessiana. É necessário que a mesma seja atualizada ao fim de cada iteração por meio do emprego das Eqs. 9 a 14.

$$H^k = H^{k-1} + D^{k-1} \tag{9}$$

$$p = x^k - x^{k-1} \tag{10}$$

$$y = \nabla f(x^k) - \nabla f(x^{k-1}) \tag{11}$$

$$\sigma = p^T y \tag{12}$$

$$\tau = y^T H^k y \tag{13}$$

$$D^k = \frac{\sigma + \theta\tau}{\sigma^2} pp^T + \frac{\theta - 1}{\tau}(H^k y)(H^k y)^T - \frac{\theta}{\sigma}(H^k y p^T + p(H^k y)^T) \tag{14}$$

Pode-se adotar $\theta = 1$ ou $\theta = 0$. No presente trabalho ambas as possibilidades foram consideradas.

É importante ressaltar que neste caso é necessário que a ordem do problema seja reduzida, de forma a torná-lo uni-dimensional, para que seja possível determinar o valor ótimo de α, que será utilizado na determinação de x^{k+1}, Eq. 7. Essa redução se dá em todas as iterações e para operá-la adotou-se

$$g(\alpha) = f(x^{k-1} - \alpha H \nabla f(x^{k-1})) \tag{15}$$

Observe que a relação introduzida na Eq. 15 foi baseada naquela apresentada na Eq. 7 e que x^{k-1}, H e $\nabla f(x)$ são conhecidos, de forma que a única variável que resta é o próprio α. Uma vez minimizada $g(\alpha)$ é determinado α^* e já é possível empregar a Eq. 7 para determinação de x^{k+1}.

O código no qual foi implementado o Método das Variáveis métricas é apresentado a seguir. Para minimização de $g(\alpha)$ e determinação de α^* foi empregado o Método da Seção Áurea, que foi implementado de forma genérica em `[xOpt, fOpt, k] = aureaSec(f,a,b,tol)`, sendo `xOpt` o ponto de ótimo determinado pelo processo de otimização, `f` a função a ser minimizada, `fOpt` o valor de $f(x^*)$, `k` o numero de chamadas de $f(x)$, `a` e `b` os extremos do espaço de busca, e `t` a tolerância adotada para encerramento das iterações.

```
clc; clear; close all;

addpath('..');

% Parametros pra execucao do algoritmo
n = 3;
numGrad = 0;
x0 = [0, 0, 0]';
tol = 1e-4;

% Definicao da funcao objetivo
f = @(x) 4250*x(1).^2 - 1000*x(1)*x(2) - 2500*x(1)*x(3) - 1000*x(1) + ...
1500*x(2).^2 - 500*x(2)*x(3) - 2000*x(2) + 5750*x(3).^2 - 3000*x(3);

if numGrad
    % Definicao numerica do gradiente
    h = 1e-10;
    df = @(x) [(f([x(1) + h, x(2), x(3)]) - f([x(1), x(2), x(3)]))/h
    (f([x(1), x(2) + h, x(3)]) - f([x(1), x(2), x(3)]))/h
```

```
    (f([x(1), x(2), x(3) + h]) - f([x(1), x(2), x(3)]))/h];
else
    % Definicao analitica do gradiente
    df = @(x) [8500*x(1) - 1000*x(2) - 2500*x(3) - 1000
    3000*x(2) - 1000*x(1) - 500*x(3) - 2000
    11500*x(3) - 500*x(2) - 2500*x(1) - 3000];
end

% Variaveis para controle de execucao
alfaOptValues = zeros(1,1);
k = 1;
nVal = 0;
H = eye(n);

while 1
    % Reduzir a dimensao do problema de otimizacao
    g = @(alfa) f(x0 - alfa*H*df(x0));

    % Resolver o problema de otimizacao uni-dimensional
    [alfaOpt,¬,nVal1] = aureaSec(g,-1,1,1e-4);

    % Atualizar a solucao otima
    x = x0 - alfaOpt*H*df(x0);

    % Armazenar dados de execucao
    alfaOptValues(k) = alfaOpt;
    if numGrad
        nVal = nVal + nVal1 + 6;
    else
        nVal = nVal + nVal1 + 1;
    end

    % OBS : Lembre-se que e necessaria a computacao do gradiente para ...
       atualizacao de x. No entanto, se estivermos utilizando a ...
       aproximacao numerica para o gradiente, a computacao do mesmo ...
       levara, neste caso a 6 avaliacoes da funcao objetivo.

    % Verificar a condicao de parada
    cp = norm(x - x0);
    if cp < tol
        break;
    end

    % Atualizacao de H (aproximacao para a inversa da Matriz Hessiana)
    p = x - x0;
    y = df(x) - df(x0);
    sigma = p'*y;
    tal = y'*H*y;
    theta = 1;
    D = ((sigma + theta*tal)/sigma^2)*(p*p') ...
    + ((theta - 1)/tal)*(H*y)*(H*y)' ...
    - (theta/sigma)*(H*y*p' + p*(H*y)');

    H = H + D;
```

```
    % Atualizar variaveis para a proxima iteracao
    x0 = x;
    k = k + 1;
end

xOpt = x;
fOpt = f(xOpt);

for i = 1:length(x)
    fprintf(['x',num2str(i),'* = %.4f\n'], xOpt(i));
end

fprintf('f(x*) = %.4f\n', fOpt);
fprintf('Numero de avaliacoes da funcao objetivo: %d\n', nVal);
fprintf('Numero de iteracoes: %d\n', k);
```

É importante ressaltar que a forma como o Método da Variável Métrica foi implementado possibilita que seja empregado tanto o gradiente calculado analiticamente quanto aquele obtido numericamente, bastando que seja modificada a variável `numGrad`.

Os resultados obtidos a partir da execução do algoritmo apresentado são introduzidos na Tab. 1. É interessante observar que, para o estudo de caso em questão, a forma como é computado o gradiente não influencia no valor de x^*, assim como o valor de θ também não foi relevante na solução do problema de otimização. Cabe ressaltar, no entanto, que a computação numérica do gradiente leva a um maior número de avaliações da função objetivo.

Tabela 1: Resultados obtidos a partir da implementação do Método da Variável Métrica.

Computação do gradiente	θ	x_1^*	x_2^*	x_3^*	$f(x^*)$	n_{val}	k
Numérica	0	0,3241	0,836	0,3677	-1549,5888	140	5
	1						
Analítica	0					115	
	1						

A implementação do Método da Seção Áurea pode ser avaliado abaixo.

```
function [xOpt, fOpt, k] = aureaSec(f,a,b,tol)
    tal = 0.618;

    if nargin < 4
    tol = 1e-8;
    end

    alfa = a + (1 - tal)*(b - a);
    beta = a + tal*(b - a);
    fAlfa = f(alfa);
    fBeta = f(beta);
```

```
    k = 1;

    while abs(a-b) > tol
        if fBeta < fAlfa
            a = alfa;
            alfa = beta;
            fAlfa = fBeta;
            beta = a + tal*(b - a);
            fBeta = f(beta);
        elseif fAlfa ≤ fBeta
            b = beta;
            beta = alfa;
            fBeta = fAlfa;
            alfa = a + (1 - tal)*(b - a);
            fAlfa = f(alfa);
        end
        k = k + 1;
    end

    xOpt = (alfa+beta)/2;
    fOpt = f(xOpt);

end
```

Todos os códigos introduzidos nos presente trabalho podem ser acessados no link `https://github.com/ArthurIasbeck/OTMC3`.

EXERCÍCIO 2

No Exercício 2 foi proposta a solução de Problemas de Programação Linear a partir do emprego do Método Gráfico. As funções a serem minimizadas ou maximizadas, denotadas por $f_1(x)$, $f_2(x)$ e $f_3(x)$ são introduzidas nas Eqs. 16, 17 e 18.

$$min\ f_1(x) = 2x_1 \tag{16}$$
$$max\ f_2(x) = -4x_2 \tag{17}$$
$$max\ f_3(x) = 3x_1 + 3 \tag{18}$$

As restrições às quais estão sujeitas $f_1(x)$, $f_2(x)$ e $f_3(x)$ são apresentadas nas Eqs. 19 a 23.

$$-x_1 + 2x_2 \leq 0 \tag{19}$$
$$2x_1 - 3x_2 \leq 3 \tag{20}$$
$$x_1 + 3x_2 \leq 6 \tag{21}$$
$$x_1 \geq 0 \tag{22}$$
$$x_2 \geq 0 \tag{23}$$

A implementação do Método Gráfico para minimização ou maximização de $f_1(x)$, $f_2(x)$ e $f_3(x)$ pode ser verificada nas Figs. 1, 2 e 3.

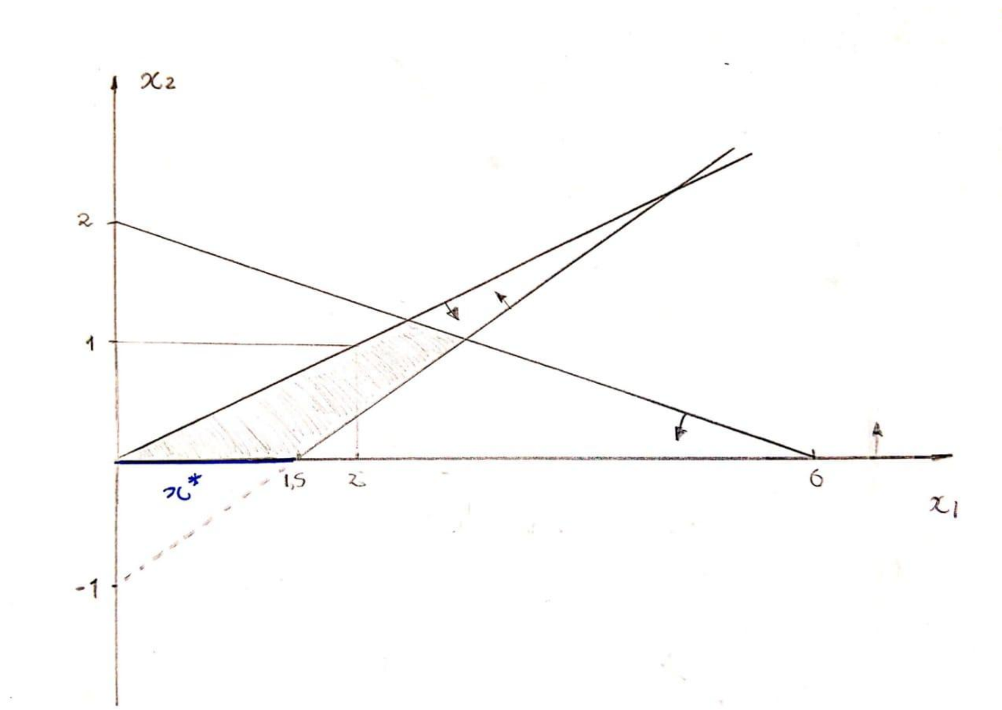

Figura 1: Resultado da implementação do método gráfico para minimização de $f_1(x)$.

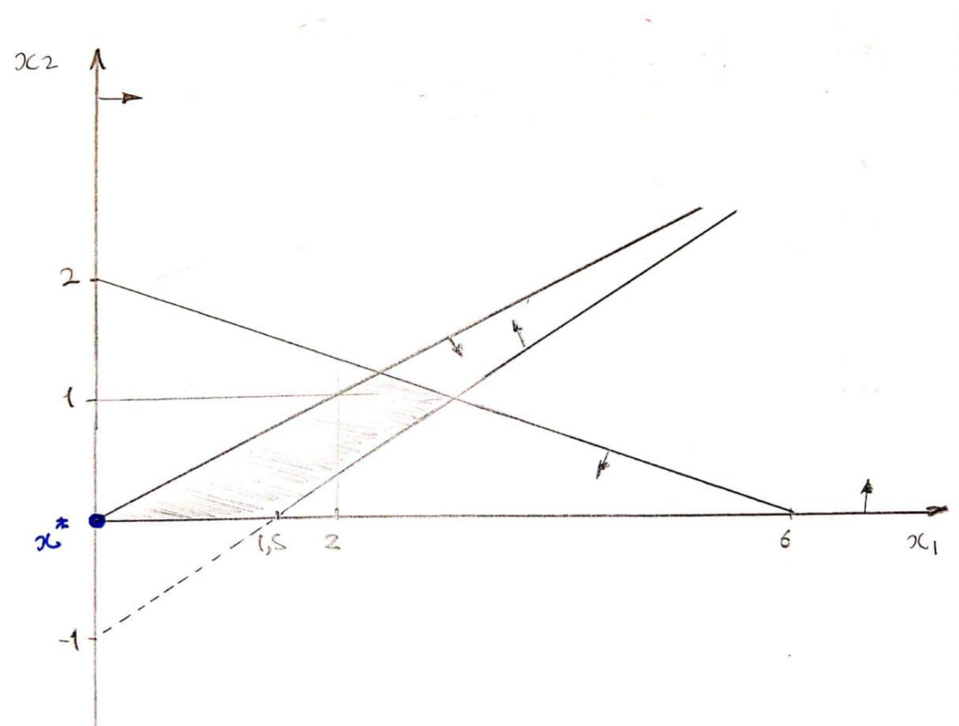

Figura 2: Resultado da implementação do método gráfico para minimização de $f_2(x)$.

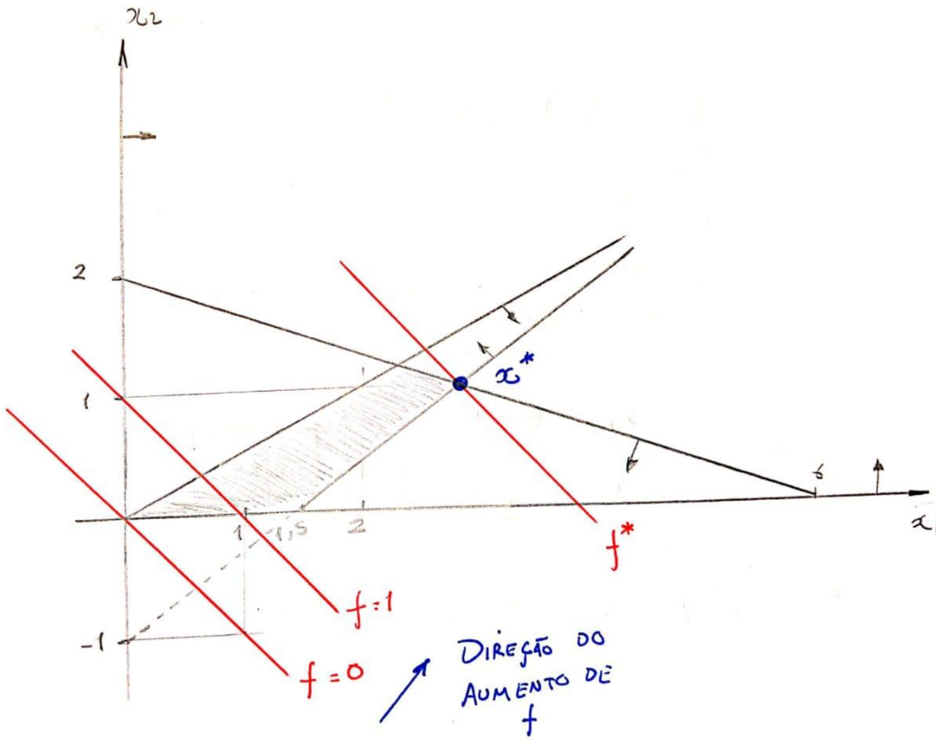

Figura 3: Resultado da implementação do método gráfico para minimização de $f_3(x)$.

É possível observar que a partir da minimização de $f_1(x)$ são obtidas infinitas soluções para x^*, uma vez que o valor de $f_1(x)$ depende somente de x_1, que deve ser igual a zero para que $f_1(x)$ seja minimizado. Já a maximização de $f_2(x)$ conduz a $x^* = \begin{bmatrix} 0 & 0 \end{bmatrix}$. A maximização de $f_3(x)$ por sua vez, indica que x^* se encontra na interseção das retas construídas a partir das restrições introduzidas nas Eqs. 20 e 21, o que implica que a determinação de x^* se dá pela solução do sistema

$$\begin{cases} 2x_1^* - 3x_2^* = 3 \\ x_1^* + 3x_2^* = 6 \end{cases} \tag{24}$$

de forma que $x_1^* = 3$ e $x_2^* = 1$.

1. EXERCÍCIO 3

No Exercício 3 foi proposta a solução do Problema de Programação Linear descrito em Carpio, R. C; Silva, R. J.; Jorge, A. B. *"Otimização da Mistura de Combustíveis Secundários Alternativos Visando Atender as Restrições Operacionais e Ambientais em Fornos de Cimenteiras"*. Foi proposta neste caso a função objetivo introduzida na Eq. 25.

$$\begin{aligned} min\ f(x) = 0{,}93x_1 + 0{,}54x_2 + 1{,}54x_3 + 0{,}77x_4 + 35x_5 + 40x_6 - 50x_7 \\ +0{,}031((5{,}76ms - 5{,}82)e^{-0{,}2ms+0{,}98}) \end{aligned} \tag{25}$$

sendo

$$ms = \frac{5x_1 + 61{,}62x_2 + 93x_3 + 7{,}6x_4 + 9{,}32x_5 + 22x_7}{1{,}86x_1 + 25{,}6x_2 + 4{,}07x_3 + 84{,}1x_4 + 12{,}29x_5 + 10{,}54x_7} \tag{26}$$

Além disso, o estudo de caso proposto apresenta 19 restrições de ordem operacional, representadas como restrições de igualdade e desigualdade, todas elas lineares.

Para verificação da influência da parcela não linear de $f(x)$ na determinação da solução do problema, foram adotadas duas abordagens. Na primeira delas, esta parcela não linear foi desconsideradas e x^* foi determinado a partir do emprego de uma rotina do Matlab® chamada `linprog`. Já na segunda abordagem, a parcela não linear de $f(x)$ foi levada em consideração e a obtenção de x^* se deu a partir da implementação de outra rotina do Matlab® denominada `fmincon`. Cabe ressaltar que ambas as funções recebem como entrada restrições de desigualdade do tipo $Ax \leq b$, o que implica que restrições da forma $Ax \geq b$ devem ser transformadas multiplicando-se ambos os lados por -1.

Os resultados obtidos em cada uma das abordagens são apresentados na Tab. 2. O valor de $f(x^*)$ apresentado por Carpio *et al.* foi o mesmo obtido no emprego da primeira abordagem. Além disso, como era esperado, a consideração da parcela não linear acabou por acarretar um aumento no valor da função objetivo, apesar de terem sido observadas mudanças pouco significativas no valor de x^*.

Tabela 2: Resultados obtidos a partir da minimização de $f(x)$, ora levando em conta sua parcela não linear e ora desconsiderando-a.

	Abordagem	
	1	2
x_1	1,21928	1,21938
x_2	0,22635	0,21749
x_3	0	0
x_4	0	0,00842
x_5	0	0
x_6	0,07841	0,07841
x_7	0,02804	0,02804
f	2,991	3,376

Os códigos empregados na implementação das abordagens 1 e 2, respectivamente, são apresentados a seguir.

```
clc; clear; close all; format long;

% As restricoes sao representadas matricialmente, logo, cada linha da
% matriz A representa uma retricao de desigualdade, enquanto cada linha da
% matriz Aeq representa uma restricao de igualdade.

% Restricoes de desigualdade
A = [-50.6 -1.23 -1.13 -0.71 -1.03 0 -0.93
     50.6 1.23 1.13 0.71 1.03 0 0.93
     -5.04  -61.62  -93  -7.6 -9.32 0 -1.93
     5.04 61.62 93 7.6 9.32 0 1.93
```

```
     -1.19 -16.59 -2.87 -1.13 -5.08 0 -0.09
     1.19 16.59 2.87 1.13 5.08 0 0.09
     -0.67 -9.01 -1.2 -82.97 -7.21 0 -0.13
     0.67 9.01 1.2 82.97 7.21 0 0.13
     0.78 0 0.1 0 0.44 0 0.12
     -0.762 -2.74 -83.64 185.83 18.96 0 -1.422
     0.018 -7.5 82.011 -219.47 -23.88 0 1.335
     -0.319 -4.877 -1.31 106.73 4.29 0 0.074
     -0.619 -7.737 -0.37 -222.88 -14.387 0 -0.25
     -38.24 155.67 173.6 164.34 37.86 0 2.93
     35.48 -190.65 -212.43 -201 -46.51 0 -3.78
     0 0 0 0 0.046 0.07 0.0123
     0 0 0 0 25392 34436 0];

b = [-62 67 -9 25 -2 9 -1 5 6.5 0 0 0 0 0 0 0.05 2700];

% Restricoes de igualdade
Aeq = [0 0 0 0 25392 34436 32100
0 0 0 0 0 0 32100];
beq = [3600 900];

% Restricoes laterais
lb = [0 0 0 0 0 0 0];

f = [0.93 0.54 1.54 0.77 35 40 -50];

x = linprog(f,A,b,Aeq,beq,lb);

C = f*x;

display(x);
display(C);
```

```
clc; clear; close all; format long;

fun = @fObj;
x0 = rand(1,7);
% Restricoes de desigualdade
% Restricoes de desigualdade
A = [-50.6 -1.23 -1.13 -0.71 -1.03 0 -0.93
     50.6 1.23 1.13 0.71 1.03 0 0.93
     -5.04  -61.62  -93  -7.6 -9.32 0 -1.93
     5.04 61.62 93 7.6 9.32 0 1.93
     -1.19 -16.59 -2.87 -1.13 -5.08 0 -0.09
     1.19 16.59 2.87 1.13 5.08 0 0.09
     -0.67 -9.01 -1.2 -82.97 -7.21 0 -0.13
     0.67 9.01 1.2 82.97 7.21 0 0.13
     0.78 0 0.1 0 0.44 0 0.12
     -0.762 -2.74 -83.64 185.83 18.96 0 -1.422
     0.018 -7.5 82.011 -219.47 -23.88 0 1.335
     -0.319 -4.877 -1.31 106.73 4.29 0 0.074
     -0.619 -7.737 -0.37 -222.88 -14.387 0 -0.25
```

```
     -38.24 155.67 173.6 164.34 37.86 0 2.93
     35.48 -190.65 -212.43 -201 -46.51 0 -3.78
     0 0 0 0 0.046 0.07 0.0123
     0 0 0 0 25392 34436 0];

b = [-62 67 -9 25 -2 9 -1 5 6.5 0 0 0 0 0 0 0.05 2700];

% Restricoes de igualdade
Aeq = [0 0 0 0 25392 34436 32100
0 0 0 0 0 0 32100];
beq = [3600 900];

% Restricoes laterais
lb = [0 0 0 0 0 0 0];

x = fmincon(fun,x0,A,b,Aeq,beq,lb);

C = fun(x);

display(x);
display(C);

function F = fObj(x)
    ms = (5*x(1) + 61.62*x(2) + 93*x(3) + 7.6*x(4) + 9.32*x(5) + 22*x(7))/ ...
    (1.86*x(1) + 25.6*x(2) + 4.07*x(3) + 84.1*x(4) + 12.29*x(5) + ...
    10.54*x(7));
    F = 0.93*x(1) + 0.54*x(2) + 1.54*x(3) + 0.77*x(4) + 35*x(5) + ...
    40*x(6) - 50*x(7) + 0.031*((5.76*ms - 5.82)*exp(-0.2*ms + 0.98));
end
```