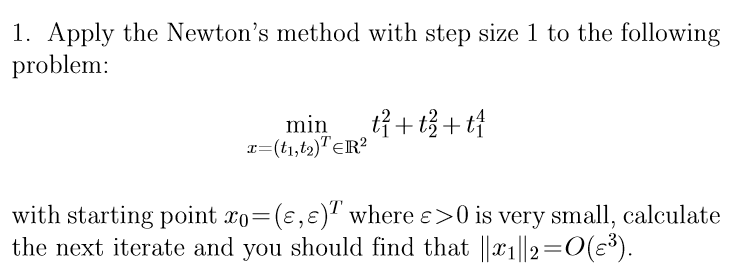
# 最优化计算方法第二次作业



**解：**梯度▽f0=(2ɛ+4ɛ3 ，2ɛ)，Hessian矩阵▽2f0=

采用Newton法，可得：

x1=x0-(▽2f0)-1▽f0=（,0）

可直接得|| x1||2=O()。

Matlab程序代码如下：

clc

clear

clc

clear

%% prepare f

ft = @(t)t(1)^2+t(2)^2+t(1)^4;

syms t1 t2;

f = ft([t1,t2]);

df = gradient(f);

ddf = hessian(f);

dft =matlabFunction(df,'vars',[t1,t2]);

dft = @(t)dft(t(1),t(2));

ddft = matlabFunction(ddf,'vars',[t1,t2]);

ddft = @(t)ddft(t(1),t(2));

%% init

epsilon = 0.1;%0.1%0.001 %此处分别取值0.1 0.01 0.001进行计算

x0 = [epsilon;epsilon];

alpha = 1;

%%

x\_current = x0;

for i = 1:1

B = ddft(x\_current);

p = -B\dft(x\_current);

x\_current = x\_current + alpha\*p

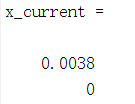
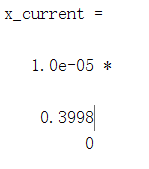
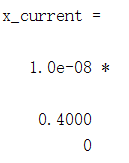
end

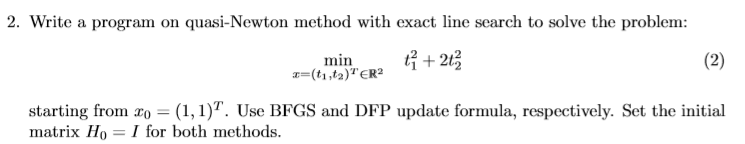
scale\_x = norm(x\_current,2)

log(scale\_x)./log(epsilon)

运行结果如下图：

epsilon = 0.1 epsilon = 0.01 epsilon = 0.001



**解：**

Matlab程序代码如下：

clc

clear

f = @(t)t(1)^2+2\*t(2)^2; %目标函数

x0 = [1,1]'; %起始值

epsilon = 0.0001; %此处可以取0.1 0.01 0.001但是为了结果精确看，此处取为0.0001

H0 = eye(2);

method = 'BFGS';%DFP 此处可以根据需要变换算法

x = quasi\_Newton(f,x0,epsilon,H0,method);

function [xk,k] = quasi\_Newton(f,x0,epsilon,H0,method)

%使用quasi\_Newton(f,x0,method)

if nargin < 3

help mfilename;

end

k = 0;

syms t1 t2;

t = [t1,t2]';

fs = f(t);

dfs = gradient(fs);

df = matlabFunction(dfs);

df = @(x) df(x(1),x(2));

df0 = df(x0);

normdf = sqrt(df0'\*df0);

H = H0;

xk = x0;

dfk = df0;

while normdf > epsilon

p = -H\*dfk;

alpha = cal\_alpha(H,dfk);

xk1 = xk + alpha\*p;

dfk1 = df(xk1);

sk = xk1 - xk;

yk = dfk1 - dfk;

eval(['H = ' method '(H,sk,yk);']);

%H = BFGS(H,sk,yk); 因为两种算法方法名及参数相同，所以可以直接使用。

k = k + 1;

xk = xk1;

dfk = dfk1;

normdf = sqrt(dfk'\*dfk);

xk

f(xk) %计算函数值

end

end

function alpha = cal\_alpha(H,dfk)

%alpha = dfk'\*H\*dfk/(dfk'\*H'\*dfk);

alpha = 1;

end

function H = BFGS(H,sk,yk)%BFGS方法代码

gammak = 1/(yk'\*sk);

skykT = sk \* yk';

skskT = sk \* sk';

E = eye(2);

H = (E-gammak\*skykT)\*H\*(E-gammak\*skykT) + gammak\*skskT;

end

function H = DFP(H,sk,yk)%DFP方法代码

Hyk = H\*yk;

ykTsk = yk'\*sk;

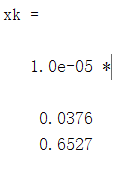
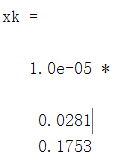
skskT = sk \* sk';

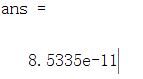
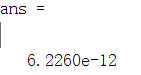
H = H - (Hyk\*yk'\*H)/(yk'\*Hyk) + skskT/ykTsk;

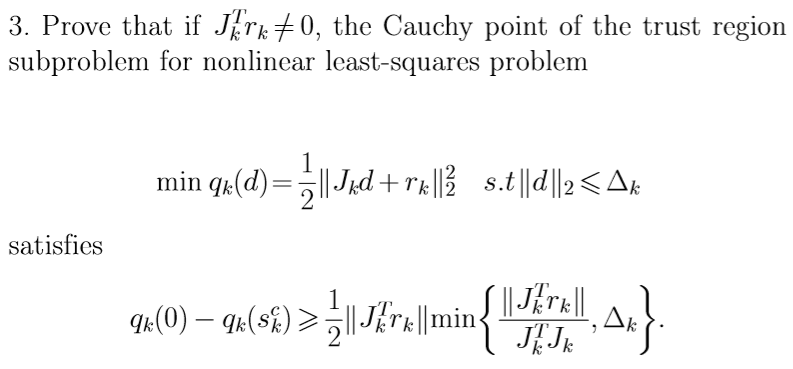
End

结果如下图所示：

BFGS（8次迭代）方法： DFP（5次迭代）方法：



**解：**

Cauchy点：

s.t.,||d||≤,

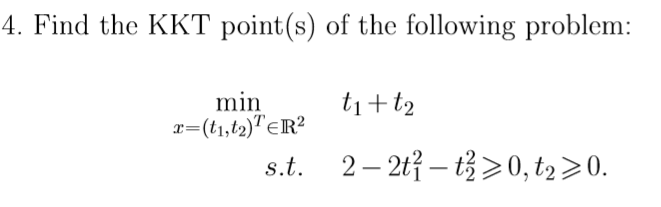
且我们有=²+  2/2()T(),

其中，0≤≤。

如果≠0，则有：

1. ,=
2. ,

结合①与②式，可得,得证。



**解：**

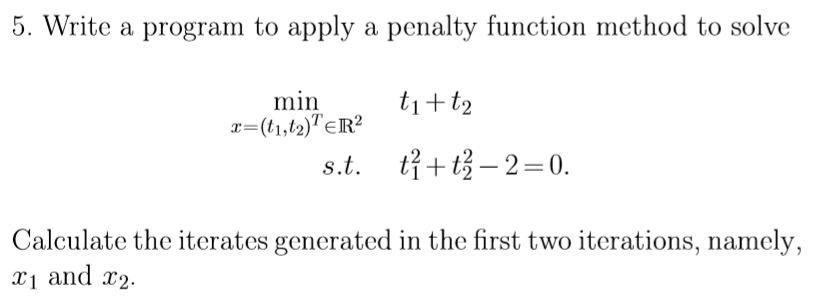
L(x,λ)=t1+t2-λ1(2-2t12-t22)- λ2t2

求解▽L=0可得到：

求解可得：,或,

然后结合KKT互补条件，可得：

,



**解：**

Matlab程序代码如下：

clc

clear

f = @(t) t(1)+t(2);%目标函数

c = @(t) t(1)^2 + t(2)^2 - 2;%目标函数约束函数

Ce = [c];%此为等式约束

alpha = 2;

sigma0 = 10;

eps = 0.01;

x0 = [-10,10]';%初始点选在了可行域外

[xk,k] = penalty(f,Ce,x0,sigma0,alpha,eps);

function [xk,k] = penalty(f,Ce,x0,sigma0,alpha,eps)

P = @(x) sum(Ce(x).^alpha);

sigma = sigma0;

xk = x0;

k = 0;

c = 2;

while abs(sigma\*P(xk))>= eps

F = @(x) f(x)+ sigma\*P(x);

warning off£»

xk = fminunc(F,xk);

sigma = c\*sigma;

k = k+1;

xk

end

end

运行结果如下图：

1次迭代,罚因子=1 2次迭代,罚因子=2

