بهینه سازی با fminunc.m با دو آپشن مقیاس متوسط و بزرگ:

(medium and Large scale)

صورت مسئله به صورت زیر می باشد:

$$f(x) = x_1^2 + 3x_2^2 + 5x_3^2 \rightarrow min$$
$$x \in \mathbb{R}^n.$$

برای استفاده از مقیاس بزرگ از ام فایل زیر می توان بهره برد:

```
function [f,g]=fun1(x)
%Objective function for example (a)
%Defines an unconstrained optimization problem to be solved
with fminunc
f=x(1)^2+3*x(2)^2+5*x(3)^2;
if nargout > 1
g(1)=2*x(1);
g(2)=6*x(2);
g(3)=10*x(3);
end
```

هم چنین برای استفاده از fminunc داریم:

```
function [xopt,fopt,exitflag]=unConstEx1
options=optimset('fminunc');
options.LargeScale='off'; options.HessUpdate='bfgs';
%assuming the function is defined in the
%in the m file fun1.m we call fminunc
%with a starting point x0
x0=[1,1,1];
[xopt,fopt,exitflag]=fminunc(@fun1,x0,options);
```

اگر قصد استفاده از مقیاس بزرگ را داشته باشیم با روشن و خاموش کردن آن می توان از آن به علاوه کد زیر بهره برد.

```
function [xopt,fopt,exitflag]=unConstEx1
options=optimset('fminunc');
options.LargeScale='on';
options.Gradobj='on';
%assuming the function is defined as in fun1.m
%we call fminunc with a starting point x0
x0=[1,1,1];
[xopt,fopt,exitflag]=fminunc(@fun1,x0,options);
```

حال برای فهم بهتر مسئله بهینه سازی با این دو آپشن، این دو را با هم مقایسه می کنیم:

```
function TestFminunc
% Set up shared variables with OUTFUN
history.x = []; history.fval = [];
%searchdir = [];
% call optimization
disp('****************************************')
disp('Iteration steps of the Matlab quasi-Newton algorithm')
i=1;x0= [1,1,1]; options
=optimset('outputfcn',@outfun,'display','iter','largescale','o
ff'); xopt = fminunc(@fun1,x0,options);
function stop = outfun(x,optimValues,state)
    stop = false;
switch state case 'iter'
% Concatenate current point and objective function
% value with history. x must be a row vector.
history.fval = [history.fval; optimValues.fval];
history.x = [history.x; x];
case 'done'
m=length(history.x(:,1));
n=length(history.fval);
if i==1
A=history.x;
subplot(2,2,1);
plot([1:m],A(:,1),'b',[1:m],A(:,2),'r',[1:m],A(:,3),'g','LineW
idth',1.5);
title('Quasi-Newton iterates');
subplot(2,2,2);
plot([1:n],history.fval,'*-r','LineWidth',2);
title('Quasi-Newton - Objective Function Values');
end
if i==2
A=history.x;
subplot(2,2,3);
plot([1:m],A(:,1),'b',[1:m],A(:,2),'r',[1:m],A(:,3),'g','LineW
idth',1.5);
title('Trust-region iterates');
subplot(2,2,4);
plot([1:n],history.fval,'*-r','LineWidth',2);
title('Trust-region Objective Function Values');
end
end
end disp('****************************************')
disp('Iteration steps of the Matlab trust-region algorithm')
i=2; history.x = [];
history.fval = []; options
=optimset('outputfcn',@outfun,'display','iter','largescale','o
n','Gradobj','on'); xopt = fminunc(@fun1,x0,options);
end
```


که با اجرای برنامه مقایسه ای بالا خواهیم داشت:

Iteration steps of the Matlab quasi-Newton algorithm

Iteration	Func-count	f(x)	Step-size	First-order optimality
0	4	9		10
1	8	1.12	0.1	2.4
2	12	0.378915	1	1.18
3	16	0.147472	1	0.722
4	20	0.0478284	1	0.506
5	24	0.00377754	1	0.202
6	28	0.000132586	1	0.0323
7	32	8.67955e-007	1	0.00172
8	36	2.13552e-009	1	8.09e-005
9	40	9.21575e-013	1	3.24e-006

Local minimum found.

Optimization completed because the size of the gradient is less than

the default value of the function tolerance.

<stopping criteria details>

****

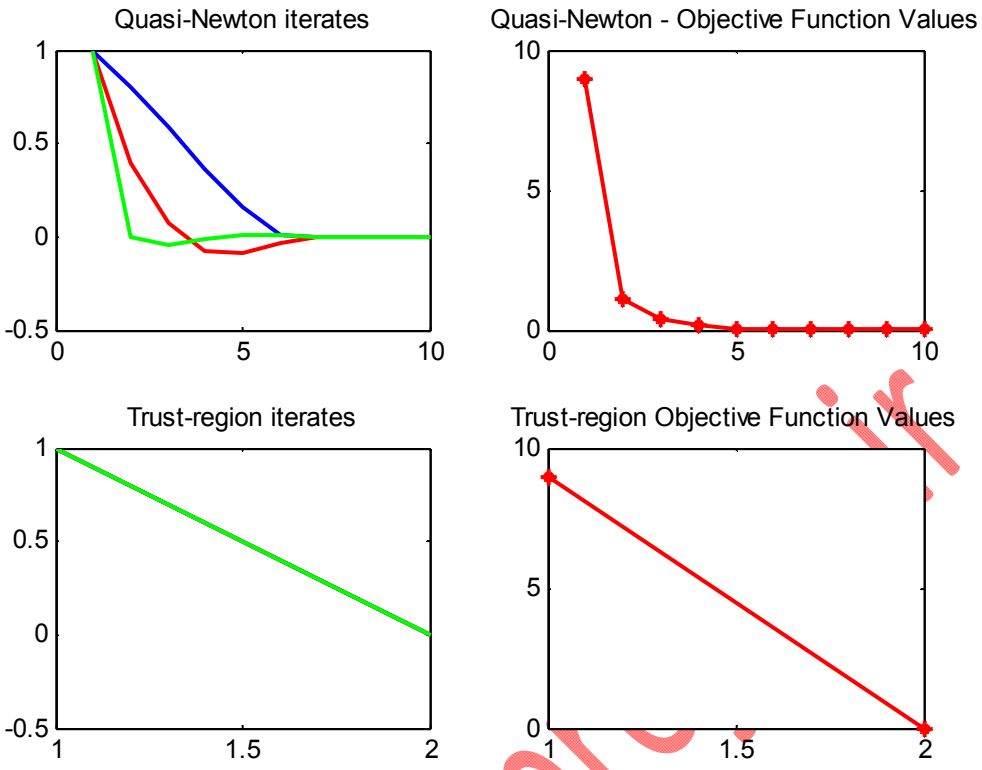
Iteration steps of the Matlab trust-region algorithm

Iteration	f(x)	step	Norm of optimality	First-order CG-iterations
0			9	10
1	1.52842e-030	1.73205	4e-015	1

Local minimum found.

Optimization completed because the size of the gradient is less than

the default value of the function tolerance.



که با توجه به شکل مشخص است که آپشن مقیاس بزرگ fminunc بعد از یک مرحله یا یک تکرار همگرا می شود و با توجه به محاسبات انجام شده در متلب آپشن مقیاس بزرگ و متوسط مقایسه می شوند.