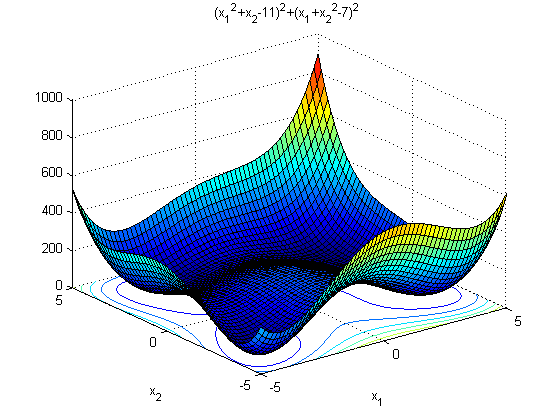
Nonlinear Optimization Project 1

Shuowen Wei

1.

(a)(b)(c)

First of all, let’s see the picture of this function, with range from -5 to 5:



Roughly we can see there are about four local minimizers, guess some start points near them and apply the ***fminunc*** function in Matlab, command and outputs are as follows:

>> format short

format compact

f = @(x1,x2) (x1.^2+x2-11).^2+(x1+x2.^2-7).^2;

ezsurfc(f,[-5,5]);

fun = @(x) f(x(1),x(2));

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

x0 = [3.5, 2];

[x, fval, exitflag, output, grad, hessian] = fminunc(fun,x0,options)

First-order

Iteration Func-count f(x) Step-size optimality

0 3 10.8125 46.5

1 9 0.151133 0.00999617 2.62

2 12 0.088031 1 1.99

3 15 0.000828784 1 0.238

4 18 8.38061e-006 1 0.0302

5 21 2.39631e-007 1 0.0057

6 24 8.36494e-012 1 2.4e-005

Local minimum found.

Optimization completed because the size of the gradient is less than

the default value of the function tolerance.

<stopping criteria details>

Computing finite-difference Hessian using user-supplied objective function.

x =

3.0000 2.0000

fval =

8.3649e-012

exitflag =

1

output =

iterations: 6

funcCount: 24

stepsize: 1

firstorderopt: 2.3971e-005

algorithm: 'medium-scale: Quasi-Newton line search'

message: [1x438 char]

grad =

1.0e-004 \*

0.2123

0.2397

hessian =

74.0264 20.0012

20.0012 34.0118

>> det(hessian)

ans =

2.1177e+003

>> x1 = [-3, 3];

[x, fval, exitflag, output, grad, hessian] = fminunc(fun,x1,options)

First-order

Iteration Func-count f(x) Step-size optimality

0 3 2 14

1 9 0.00177728 0.0137721 0.514

2 12 9.21033e-006 1 0.0341

3 15 2.55503e-008 1 0.00176

4 18 3.25833e-011 1 6e-005

5 21 1.27633e-014 1 1.09e-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>

Computing finite-difference Hessian using user-supplied objective function.

x =

-2.8051 3.1313

fval =

1.2763e-014

exitflag =

1

output =

iterations: 5

funcCount: 21

stepsize: 1

firstorderopt: 1.0904e-006

algorithm: 'medium-scale: Quasi-Newton line search'

message: [1x438 char]

grad =

1.0e-005 \*

-0.0294

0.1090

hessian =

64.9726 1.3049

1.3049 80.4697

>> det(hessian)

ans =

5.2266e+003

>> x2 = [3, -2];

[x, fval, exitflag, output, grad, hessian] = fminunc(fun,x2,options)

First-order

Iteration Func-count f(x) Step-size optimality

0 3 16 48

1 9 0.0427827 0.0122842 1.6

2 12 0.0168638 1 0.991

3 15 3.01255e-005 1 0.0412

4 18 4.40562e-007 1 0.00956

5 21 1.67933e-008 1 0.00169

6 24 3.0399e-013 1 4.26e-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>

Computing finite-difference Hessian using user-supplied objective function.

x =

3.5844 -1.8481

fval =

3.0399e-013

exitflag =

1

output =

iterations: 6

funcCount: 24

stepsize: 1

firstorderopt: 4.2579e-006

algorithm: 'medium-scale: Quasi-Newton line search'

message: [1x438 char]

grad =

1.0e-005 \*

-0.1349

-0.4258

hessian =

104.8227 6.9456

6.9456 29.3346

>> det(hessian)

ans =

3.0267e+003

>> x3 = [-4, -4];

[x, fval, exitflag, output, grad, hessian] = fminunc(fun,x3,options)

First-order

Iteration Func-count f(x) Step-size optimality

0 3 26 78

1 6 5.6318 0.0128205 25.9

2 9 0.312105 1 6.41

3 12 0.00466055 1 1.04

4 15 2.98344e-005 1 0.0826

5 18 2.06413e-009 1 0.000587

6 21 1.34807e-012 1 1.71e-005

Local minimum found.

Optimization completed because the size of the gradient is less than

the default value of the function tolerance.

<stopping criteria details>

Computing finite-difference Hessian using user-supplied objective function.

x =

-3.7793 -3.2832

fval =

1.3481e-012

exitflag =

1

output =

iterations: 6

funcCount: 21

stepsize: 1

firstorderopt: 1.7069e-005

algorithm: 'medium-scale: Quasi-Newton line search'

message: [1x438 char]

grad =

1.0e-004 \*

-0.0285

-0.1707

hessian =

116.3073 -28.2517

-28.2517 88.2661

>> det(hessian)

ans =

9.4678e+003

The global minimizer is at the point (3,2), which makes the value of function f equals to zero.

2.

(a).

We use Matlab commands gradient and hessian to compute the gradient and hessian matrixes for the function , commands and outputs are as follows:

clc;clear

syms x1 x2 x3 x4

f = 100\*(x1^2-x2)^2+(1-x1)^2+90\*(x3^2-x4)^2+(1-x3)^2+10.1\*((1-x2)^2+(1-x4)^2)+19.8\*(1-x2)\*(1-x4);

gradient(f)

hessian(f)

>> gradient(f)

ans =

2\*x1 - 400\*x1\*(- x1^2 + x2) - 2

- 200\*x1^2 + (1101\*x2)/5 + (99\*x4)/5 - 40

2\*x3 - 360\*x3\*(- x3^2 + x4) - 2

- 180\*x3^2 + (99\*x2)/5 + (1001\*x4)/5 – 40

>> hessian(f)

ans =

[ 1200\*x1^2 - 400\*x2 + 2, -400\*x1, 0, 0]

[ -400\*x1, 1101/5, 0, 99/5]

[ 0, 0, 1080\*x3^2 - 360\*x4 + 2, -360\*x3]

[ 0, 99/5, -360\*x3, 1001/5]

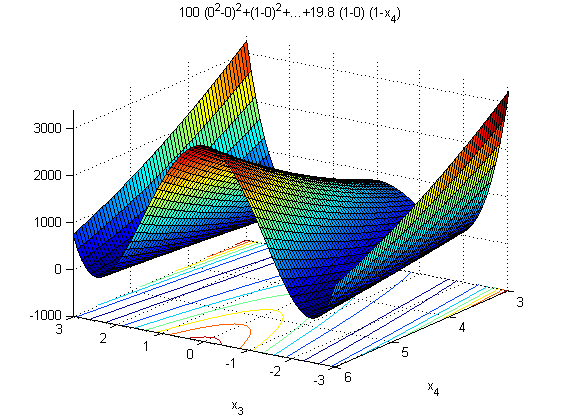
(b).

No, f is not a convex function.

I set x1 and x2 to be zero, thus get the intersecting surface of this function:

f = @(x3,x4) 100\*(0.^2-0).^2+(1-0).^2+90\*(x3.^2-x4).^2+(1-x3).^2+10.1\*((1-0).^2+(1-0).^2)+19.8\*(1-0)\*(1-x4)

ezsurfc(f,[-3,3],[3,6]);



It’s very clear to see this intersecting surface is non-convex. Thus, I pick a pair of points as follows:

f = @(x1,x2,x3,x4) 100\*(x1.^2-x2).^2+(1-x1).^2+90\*(x3.^2-x4).^2+(1-x3).^2+10.1\*((1-x2).^2+(1-x4).^2)+19.8\*(1-x2)\*(1-x4)

fun = @(x) f(x(1),x(2),x(3),x(4));

x1=[0 0 2.5 6];

x2=[0 0 -2.5 6];

x12\_mid=(x1+x2)./2;

[fun(x1)+fun(x2)]/2

fun(x12\_mid)

>> [fun(x1)+fun(x2)]/2

ans =

177.4750

>> fun(x12\_mid)

ans =

3.4056e+003

Thus, we find a pair of points x1=[0 0 2.5 6] and x2=[0 0 -2.5 6] such that hence by definition that this function f is not a convex function.

(c).

>> x0 = [-3,-1,-3,-1];

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

[x, fval, exitflag, output, grad, hessian] = fminunc(fun,x0,options)

First-order

Iteration Func-count f(x) Step-size optimality

0 5 19192 1.2e+004

1 10 4966.71 8.32778e-005 3.98e+003

2 15 1918.61 1 1.82e+003

3 20 656.028 1 734

4 25 250.951 1 293

5 30 107.869 1 143

6 35 59.3993 1 83.8

7 40 42.8945 1 49.5

8 45 37.406 1 27.9

9 50 35.6153 1 14.1

10 55 35.0042 1 **4.92**

11 60 34.6551 1 **13.2**

12 85 3.45603 3.06633 **32.2**

13 100 3.44566 0.0271379 **29.4**

14 110 1.91732 10 **17.9**

15 115 1.818 1  **18**

16 120 1.5351 1 **17.5**

17 125 1.05806 1 **17.1**

18 130 0.412725 1 **18.4**

19 135 0.0474582 1 7.79

First-order

Iteration Func-count f(x) Step-size optimality

20 140 0.00513989 1 1.61

21 145 0.00112419 1 0.0994

22 150 0.00103843 1 0.0489

23 155 0.00103743 1 0.0296

24 160 0.00103718 1 0.0262

25 165 0.0010361 1 0.0292

26 170 0.0010337 1 0.0542

27 175 0.00102703 1 0.0971

28 180 0.00101022 1 0.163

29 185 0.000967317 1 0.265

30 190 0.000864441 1 0.405

31 195 0.00064466 1 0.545

32 200 0.000306013 1 0.543

33 205 7.88021e-005 1 0.299

34 210 1.52932e-005 1 0.0987

35 215 1.84896e-007 1 0.00339

Local minimum found.

Optimization completed because the size of the gradient is less than

the default value of the function tolerance.

<stopping criteria details>

Computing finite-difference Hessian using user-supplied objective function.

x =

1.0002 1.0004 0.9998 0.9996

fval =

1.8490e-007

exitflag =

1

output =

iterations: 35

funcCount: 215

stepsize: 1

firstorderopt: 0.0034

algorithm: 'medium-scale: Quasi-Newton line search'

message: [1x438 char]

grad =

-0.0001

0.0004

-0.0034

0.0013

hessian =

802.6453 -400.1126 0 -0.0000

-400.1126 220.2000 0.0000 19.8000

0 0.0000 721.9346 -359.9404

-0.0000 19.8000 -359.9404 200.2000

>> det(hessian)

ans =

2.2186e+007

After a long iteration, this process finally reaches the local minimizer (its Hessian matrix’s determinant is positive). As you can easily find that the values from the “First-order optimality” are not always decreasing. The reason for this is that the function f is not convex, then when the iteration points get close to the local minimizer, they meet the concave part of the function, which makes the “First-order optimality” not decreasing, or say a little vibrating for some points, and also leads to slower convergence speed.

Other two start points to test convergence are as follows:

>> test\_points1 = [0,0,0,0];

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

[x, fval, exitflag, output, grad, hessian] = fminunc(fun,test\_points1,options)

First-order

Iteration Func-count f(x) Step-size optimality

0 5 42 40

1 15 35.0038 0.00436333 2.59

2 25 34.4995 10 15.1

3 35 33.5783 0.412766 45.3

4 45 32.8345 0.316599 67

5 50 28.5319 1 84.7

6 55 15.1628 1 47.4

7 60 7.78401 1 28.8

8 70 0.553626 0.650304 14.1

9 75 0.0840885 1 7.51

10 80 0.0306737 1 3.79

11 85 0.0213766 1 0.884

12 90 0.0206039 1 0.442

13 95 0.0203525 1 0.166

14 100 0.0202273 1 0.179

15 105 0.0192572 1 0.573

16 110 0.0175041 1 1.13

17 115 0.0134975 1 1.87

18 120 0.00827703 1 2.24

19 125 0.00326252 1 1.71

First-order

Iteration Func-count f(x) Step-size optimality

20 130 0.000262356 1 0.427

21 135 7.30241e-006 1 0.0417

22 140 9.22275e-007 1 0.0343

23 145 5.37392e-008 1 0.00458

24 150 3.70751e-009 1 0.000332

25 155 1.20506e-011 1 0.000104

26 160 3.72233e-013 1 8.18e-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>

Computing finite-difference Hessian using user-supplied objective function.

x =

1.0000 1.0000 1.0000 1.0000

fval =

3.7223e-013

exitflag =

1

output =

iterations: 26

funcCount: 160

stepsize: 1

firstorderopt: 8.1770e-006

algorithm: 'medium-scale: Quasi-Newton line search'

message: [1x438 char]

grad =

1.0e-005 \*

0.8177

-0.3032

-0.0210

0.1229

hessian =

802.2927 -400.0243 0.0000 0.0000

-400.0243 220.2000 0.0000 19.8000

0.0000 0.0000 722.2638 -360.0220

0.0000 19.8000 -360.0220 200.2000

>> test\_points2 = [2,2,2,2];

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

[x, fval, exitflag, output, grad, hessian] = fminunc(fun,test\_points2,options)

First-order

Iteration Func-count f(x) Step-size optimality

0 5 802 1.6e+003

1 10 296.998 0.00062422 490

2 15 102.592 1 292

3 20 66.5201 1 270

4 25 41.2572 1 48.7

5 30 39.3213 1 37.3

6 35 34.15 1 64.5

7 40 26.1825 1 110

8 45 14.8655 1 127

9 50 7.00882 1 93.7

10 55 1.34885 1 38.4

11 60 0.0972515 1 7.75

12 65 0.00382467 1 0.904

13 70 0.000176836 1 0.174

14 75 0.000147449 1 0.0817

15 80 0.00014173 1 0.032

16 85 0.00014062 1 0.00876

17 90 0.000140482 1 0.0104

18 95 0.000140144 1 0.0128

19 100 0.000139349 1 0.0162

First-order

Iteration Func-count f(x) Step-size optimality

20 105 0.000137218 1 0.0286

21 110 0.000131911 1 0.0518

22 115 0.000119146 1 0.0847

23 120 9.25297e-005 1 0.12

24 125 5.11719e-005 1 0.131

25 130 1.4495e-005 1 0.085

26 135 1.41931e-006 1 0.0218

27 140 4.9803e-008 1 0.00145

Local minimum found.

Optimization completed because the size of the gradient is less than

the default value of the function tolerance.

<stopping criteria details>

Computing finite-difference Hessian using user-supplied objective function.

x =

1.0001 1.0002 0.9999 0.9998

fval =

4.9803e-008

exitflag =

1

output =

iterations: 27

funcCount: 140

stepsize: 1

firstorderopt: 0.0014

algorithm: 'medium-scale: Quasi-Newton line search'

message: [1x438 char]

grad =

0.0009

0.0002

-0.0014

0.0010

hessian =

802.4781 -400.0705 -0.0000 0.0000

-400.0705 220.2000 -0.0000 19.8000

-0.0000 -0.0000 722.1117 -359.9843

0.0000 19.8000 -359.9843 200.2000

>> det(hessian)

ans =

2.2195e+007

As we can see from test points [0 0 0 0] and [2 2 2 2], both of them act better the point [-3 -1 -3 -1], that’s because their convergence path didn’t meet the concave part of the function, which avoid wasting a lot of time there.

3.

>> H=[2 0; 0 -4]

H =

2 0

>> det(H)

ans =

-8

>> eig(H)

ans =

-4

2

Since the eigenvalues of the Hessian Matrix contains both positive values and negative values, thus this only one critical one point is neither a local minimizer nor a maximizer.