Homework 1,2D Root-finding using Newton's Method
資工 3B 00957144 蔣佳純
日期:2022/10/2

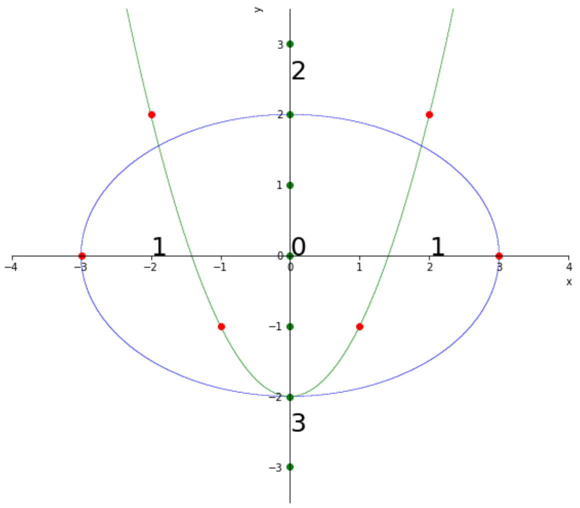
1. initial point = [1.0,1.0],compute $[x\ y]^{(i)}$

(a.) 執行結果

```
f(x,y) = 0.111111x^2+0.25y^2-1

g(x,y) = 1x^2-1y-2

fx = 0.222222x

fy = 0.5y

gx = 2x

gy = -1


start!!

initial point: [1,1]

 i       x            y           f(x,y)       g(x,y)       error         C

 0  1.0000000000 1.0000000000 -0.6388888889 -2.0000000000

 1  2.3409090909 1.6818181818 0.3160009183 1.7980371901 2.0227272727 1.2545493131

 2  1.9307877844 1.5597419822 0.0224144814 0.1681994861 0.5321975062 1.3587962308

 3  1.8861477166 1.5555604732 0.0002257863 0.0019927356 0.0488215767 1.3714806342

 4  1.8856181575 1.5555555556 0.0000000312 0.0000002804 0.0005344767 1.3716286289

 5  1.8856180832 1.5555555556 0.0000000000 0.0000000000 0.0000000744 1.3716286496

finished!!

used 5 times.
```

(b.) x=-4 to 4 and y=-3 to 3 ,grid-points$[x_i,y_i]$, which make the computation converges or divergence?

x=0 的點上會發散，其餘的點皆收斂



綠色點為發散點

(c.) 對於起始點[1,1]，以 2-norm 計算誤差｜e_n｜<= 10^{-6}，需要 5 次 iteration (d.) 在會發散的點 add small perturbations:

在 x=0 處發散，只要 x 不為 0 就會收斂

test 1:

(x,y)=(0,2) -> divergence

```
start!!
initial point: [0,2]
 i       x             y            f(x,y)        g(x,y)
 0  0.0000000000 2.0000000000 0.0000000000 -4.0000000000
divergence!!!
```

add small perturbations:

(x,y)=(0.000001,2)

```
start!!
initial point: [1e-06,2]
 i        x                   y                  f(x,y)                  g(x,y)
 0      0.00000100       2.00000000           0.00000000          -4.00000000
 1  1800000.00000050      1.60000000  359999999999.84014893 3239999999998.20117188
 2   900000.00000124      1.55605854   89999999999.85290527  809999999998.67224121
 …
25       1.88561808      1.55555556           0.00000000           0.00000000
finished!!
used 25 times.
```

test 2:

(x,y)=(0,-1.999) -> divergence

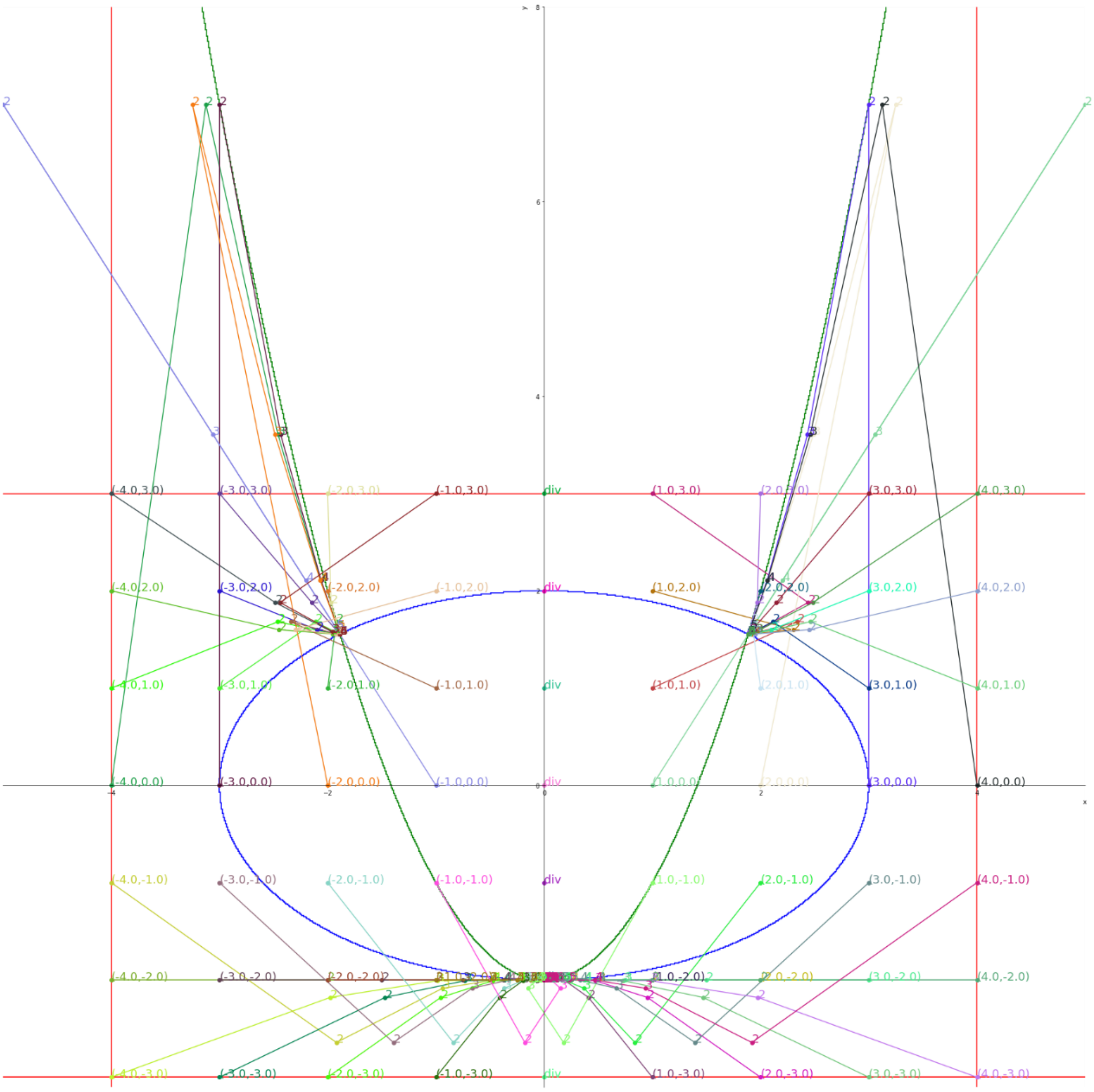
```
start!!
initial point: [0,-1.999]
 i           x                y              f(x,y)          g(x,y)        error
 0      0.00000000     -1.99900000     -0.00099975     -0.00100000
divergence!!!
```

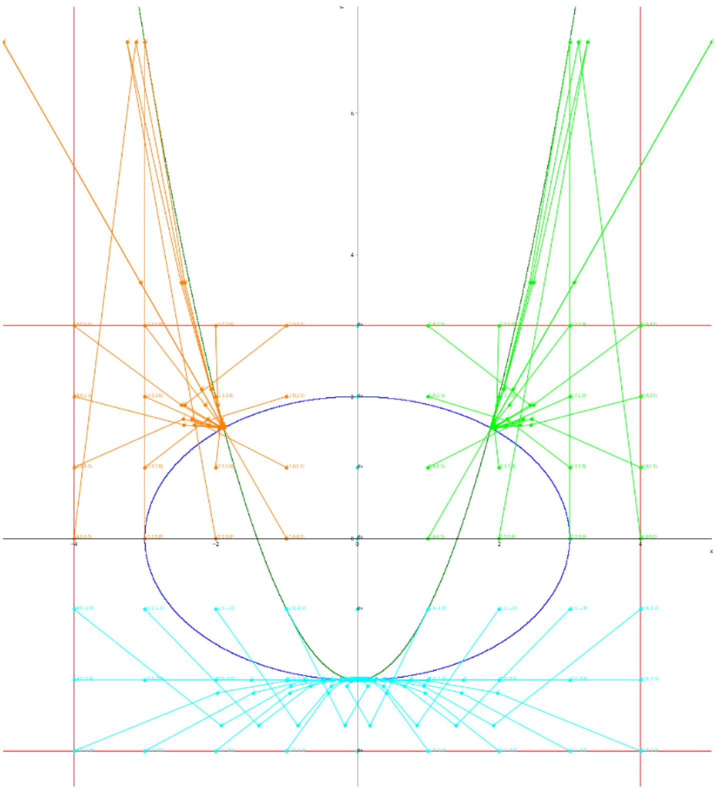
add small perturbations:

(x,y)=(0.000001,-1.999)

```
start!!
initial point: [1e-06,-1.999]
 i          x                 y                f(x,y)           g(x,y)           error
 0    0.000001000000  -1.999000000000  -0.000999750000  -0.000999999999
 1   -0.140703646062  -2.000000281408   0.002200005410   0.019797797424   0.141704927471
 2   -0.070351823031  -2.000000000000   0.000549931000   0.004949379004   0.070352104440
 …
18   -0.000001073484  -2.000000000000   0.000000000000   0.000000000001   0.000001073484
19   -0.000000536742  -2.000000000000   0.000000000000   0.000000000000   0.000000536742
finished!!
used 19 times.
```

2.





收斂區域圖(橘色為收斂到$(-\frac{4}{3}\sqrt{2}\ ,\frac{14}{9})$，綠色為收斂到$(\frac{4}{3}\sqrt{2}\ ,\frac{14}{9})$，青色為收斂到(0,-2)

3. 圖上標示 div 即為發散點

造成發散的 initial guesses 的原因：

$$\begin{bmatrix} h \\ k \end{bmatrix} = \frac{1}{f_y g_x - f_x g_y} \begin{bmatrix} g_y & -f_y \\ -g_x & f_x \end{bmatrix} \begin{bmatrix} f \\ g \end{bmatrix}$$

if $f_y g_x - f_x g_y = 0$, then divergence(h,k no solution)

$$\frac{\partial f}{\partial x} = \frac{2}{9}x \quad \frac{\partial f}{\partial y} = \frac{1}{2}y \quad \frac{\partial g}{\partial x} = 2x \quad \frac{\partial g}{\partial y} = -1$$

x=0,y=y*代入 $f_y g_x - f_x g_y = 0$　因此 x=0 上的點皆會發散

4. step 1: compute the eigenvalues of the Jacobian matrix and C

$$J = \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \\ \frac{\partial g}{\partial x} & \frac{\partial g}{\partial y} \end{bmatrix} = \begin{bmatrix} f_x & f_y \\ g_x & g_y \end{bmatrix}$$

$Jx = \lambda x$

⇨ $(J-\lambda I)x = 0$

⇨ $(J-\lambda I) = \begin{bmatrix} f_x - \lambda & f_y \\ g_x & g_y - \lambda \end{bmatrix} = 0$

由 $\det(J-\lambda I) = 0$:

⇨ $(f_x - \lambda)\cdot(g_y - \lambda) - g_x \cdot f_y = 0$

⇨ $\lambda^2 - (f_x + g_y)\lambda + f_x \cdot g_y - g_x \cdot f_y = 0$

⇨ $\lambda = \frac{(f_x+g_y)\pm\sqrt{(f_x+g_y)^2-4(f_x\cdot g_y-g_x\cdot f_y)}}{2}$

$C = \frac{abs(\lambda_{max})}{abs(\lambda_{min})}$

step 2: show some compute result

```
start!!
initial point: [4.0000000000,2.0000000000]
 i        x              y             f(x,y)          g(x,y)           error          C
 0   4.000000000000   2.000000000000   1.777777777778 12.000000000000                    1.0379687136
 1   2.450000000000   1.600000000000   0.306944444444   2.402500000000   1.950000000000 1.2400962230
 2   1.950734196117   1.556097560976   0.028178116363   0.249266342926   0.543168242907 1.3539106486
 3   1.886704902366   1.555555638153   0.000455601249   0.004099750458   0.064571216574 1.3713261979
 4   1.885618396190   1.555555555556   0.000000131166   0.000001180496   0.001086588773 1.3716285624
 5   1.885618083164   1.555555555556   0.000000000000   0.000000000000   0.000000313026 1.3716286496
finished!!
used 5 times.
```

```
start!!
initial point: [0.0000000000,1.0000000000]
 i        x              y             f(x,y)          g(x,y)         error         C
 0   0.000000000000   1.000000000000  -0.750000000000 -3.000000000000               inf
divergence!!!
```

```
start!!

initial point: [-2.0000000000,3.0000000000]

 i         x               y              f(x,y)          g(x,y)         error            C

 0   -2.000000000000   3.000000000000   1.694444444444 -1.000000000000                   nan

 1   -1.969827586207   1.879310344828   0.314087478531   0.000910374554   1.150862068966 nan

 2   -1.893748111485   1.580493823280   0.022965949107   0.005788086474   0.374895996270 nan

 3   -1.885681077777   1.555728050053   0.000160566662   0.000065077033   0.032832806935 nan

 4   -1.885618086435   1.555555563923   0.000000007879   0.000000003968   0.000235477472 nan

 5   -1.885618083164   1.555555555556   0.000000000000   0.000000000000   0.000000011639 nan

finished!!
```

used 5 times.

```
start!!
initial point: [0.0001,-2]
 i          x               y                 error           C
 0    0.000100000000  -2.000000000000
 1    0.000050000000  -2.000000000000   0.000050000000 11247.7499124808
 2    0.000025000000  -2.000000000000   0.000025000000 22497.7499562368
 3    0.000012500000  -2.000000000000   0.000012500000 44997.7499781438
 4    0.000006250000  -2.000000000000   0.000006250000 89997.7499893133
 5    0.000003125000  -2.000000000000   0.000003125000 179997.7499939504
 6    0.000001562500  -2.000000000000   0.000001562500 359997.7500014548
 7    0.000000781250  -2.000000000000   0.000000781250 719997.7499899489
finished!!
used 7 times.
```

```
start!!
initial point: [1e-05,1e-05]
 i          x                      y                  error                   C
 0        0.000010000000        0.000010000000
 1   449984.250724967686        6.999685014399  449991.250389982073    3078.887720146635
 2   224992.125368714653        3.607542664922  224995.517498602509    2903.033564357254
 3   112496.062693480519        2.105282989548  112497.564934909504    2388.639429468489
 4    56248.031362831818        1.620474650308   56248.516138987943    1509.157772313577
 5    28124.015713032117        1.556699117939   28124.079425332071     782.469359955207
 6    14062.007919728156        1.555555926549   14062.008936495351     392.358740954954
 7     7031.004086288255        1.555555555442    7031.003833811007     197.050674242068
 8     3515.502295992477        1.555555555442    3515.501790295778      99.392940490420
 9     1757.751653692901        1.555555555442    1757.750642299576      50.556975174618
10      878.876838239484        1.555555555595     878.874815453569      26.125608907706
11      439.440441903482        1.555555555576     439.436396336021      13.885943707513
12      219.724266500599        1.555555555557     219.716175402902       7.726588651776
13      109.870224199042        1.555555555557     109.854042301557       4.589165823409
14       54.951292805356        1.555555555555      54.918931393687       2.947376334719
15       27.507998285355        1.555555555555      27.443294520001       2.045738673261
16       13.818626810577        1.555555555556      13.689371474778       1.514097007081
17        7.037964233055        1.555555555556       6.780662577522       1.172662162298
18        3.771580413266        1.555555555556       3.266383819789       1.064497212182
19        2.357151700485        1.555555555556       1.414428712782       1.261243759385
20        1.932781774881        1.555555555556       0.424369925603       1.358748091635
21        1.886193526767        1.555555555556       0.046588248114       1.371468480446
22        1.885618170943        1.555555555556       0.000575355824       1.371628625130
23        1.885618083164        1.555555555556       0.000000087779       1.371628649568
finished!!
used 23 times.
```

```
start!!
initial point: [1.88888,1.55556]
 i          x               y                 error           C
 0    1.888880000000   1.555555000000
 1    1.885620899675   1.555555555556   0.003259655881 1.3716278654
 2    1.885618083166   1.555555555556   0.000002816509 1.3716286496
 3    1.885618083164   1.555555555556   0.000000000002 1.3716286496
finished!!
used 3 times.
```

```
initial point: [4,-3]
 i          x                 y                    error                  C
 0       4.000000000000     -3.000000000000
 1       1.977500000000     -2.180000000000      2.842500000000                    nan
 2       0.986657790681     -2.008274687855      1.162567521464                    nan
 3       0.493319181690     -2.000019168098      0.501594128748                    nan
 4       0.246659590740     -2.000000000103      0.246678758945                    nan
 5       0.123329795370     -2.000000000000      0.123329795473      1.739484520127
 6       0.061664897685     -2.000000000000      0.061664897685      6.724895001043
 7       0.030832448843     -2.000000000000      0.030832448843     15.931855717122
 8       0.015416224421     -2.000000000000      0.015416224421     34.208729098265
 9       0.007708112211     -2.000000000000      0.007708112211     70.711138420126
10       0.003854056105     -2.000000000000      0.003854056105    143.693280488377
11       0.001927028053     -2.000000000000      0.001927028053    289.646866446449
12       0.000963514026     -2.000000000000      0.000963514026    581.548838013927
13       0.000481757013     -2.000000000000      0.000481757013   1165.350216862374
14       0.000240878507     -2.000000000000      0.000240878507   2332.951701232492
15       0.000120439253     -2.000000000000      0.000120439253   4668.154035493494
16       0.000060219627     -2.000000000000      0.000060219627   9338.558387320487
17       0.000030109813     -2.000000000000      0.000030109813  18679.366932765555
18       0.000015054907     -2.000000000000      0.000015054907  37360.983944588836
19       0.000007527453     -2.000000000000      0.000007527453  74724.217928885133
20       0.000003763727     -2.000000000000      0.000003763727 149450.685877818090
21       0.000001881863     -2.000000000000      0.000001881863 298903.621761281218
22       0.000000940932     -2.000000000000      0.000000940932 597809.493537744274
finished!!
used 22 times.
```

step 3: analysis

計算發現：C 只和計算時的(x,y)有關

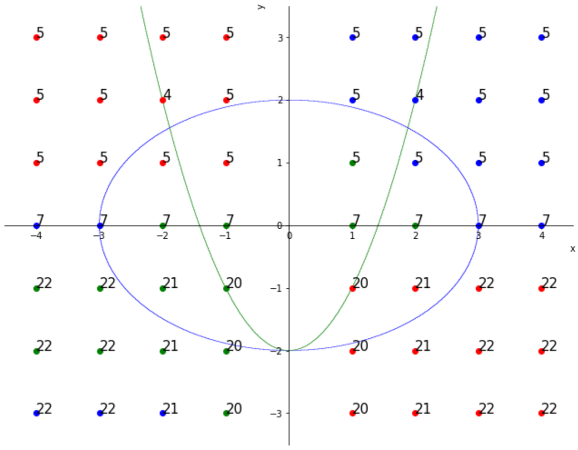
觀察發現：

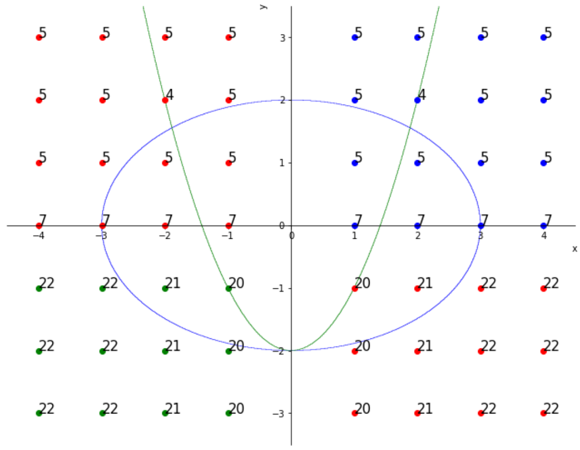
會 divergence 的 C 一定是 inf

會 convergence 的 C 有時為 nan 有時是數值

x 靠近 0 時的 C 會很大

x 為 0 時的 C 為 inf(這種 initial guess 會導致 divergence)





紅色點: 第 0 次 iteration C 為 nan

藍色點: 第 0 次 iteration C <= 1.9

綠色點: 第 0 次 iteration C > 1.9

紅色點: 第 1 次 iteration C 為 nan

藍色點: 第 1 次 iteration C <= 1.9

綠色點: 第 1 次 iteration C > 1.9

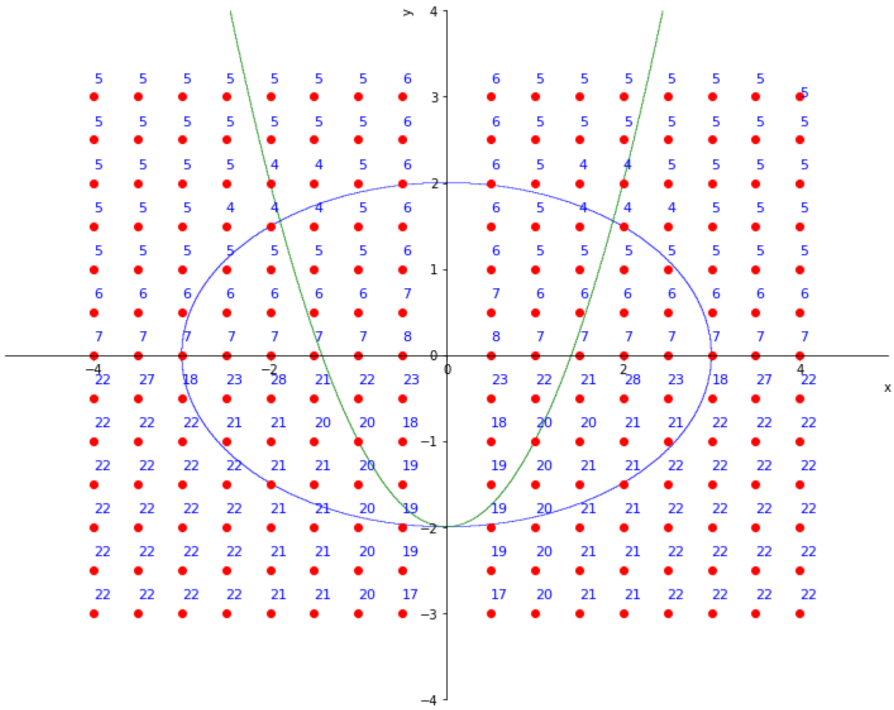
- divergence , x=0 時：

$$\lambda_1 = \frac{(f_x+g_y)+\sqrt{(f_x+g_y)^2-4(f_x\cdot g_y-g_x\cdot f_y)}}{2} = \frac{-1+\sqrt{1}}{2} = 0 \quad \lambda_2 = \frac{(f_x+g_y)-\sqrt{(f_x+g_y)^2-4(f_x\cdot g_y-g_x\cdot f_y)}}{2} = \frac{-1-\sqrt{1}}{2} = -2$$

⇨ $C = \frac{2}{0} = \infty$ (inf)

- 產生 nan 的原因：根號內數字< 0

5. 0.5 一間隔的格子點 iteration 次數圖：



6. 梯度方向和收斂速度的推論

start!!

initial point: [4.000000000000,-1.000000000000]

i	x	y	fx	fy	gx	gy
0	4.000000000000	-1.000000000000	0.888888888889	-0.500000000000	8.000000000000	-1.000000000000
1	1.919642857143	-2.642857142857	0.426587301587	-1.321428571429	3.839285714286	-1.000000000000
…						
21	0.000001784319	-2.000000000000	0.000000396515	-1.000000000000	0.000003568638	-1.000000000000
22	0.000000892160	-2.000000000000	**0.000000198258**	**-1.000000000000**	**0.000001784319**	**-1.000000000000**

finished!!

used 22 times.

start!!

initial point: [4.000000000000,0.000000000000]

i	x	y	fx	fy	gx	gy
0	4.000000000000	0.000000000000	0.888888888889	0.000000000000	8.000000000000	-1.000000000000
1	3.125000000000	7.000000000000	0.694444444444	3.500000000000	6.250000000000	-1.000000000000
…						
6	1.885618232904	1.555555923398	0.419026273979	0.777777961699	3.771236465807	-1.000000000000
7	1.885618083164	1.555555555556	**0.419026240703**	**0.777777777778**	**3.771236166328**	**-1.000000000000**

finished!!

used 7 times.

推論：若找到的是(0,-2)的 root，f 梯度和 g 梯度方向比較接近平行，因此收斂比較不容易(需要更多次 iteration)

7. 收斂速度分析

start!!

initial point: [4,-3]

i	x	y	f(x,y)	g(x,y)	error
0	4.000000000000	-3.000000000000	3.027777777778	17.000000000000	
1	1.977500000000	-2.180000000000	0.622600694444	4.090506250000	**2.842500000000**
2	0.986657790681	-2.008274687855	0.116457760571	0.981768283767	**1.162567521464**
3	0.493319181690	-2.000019168098	0.027059592082	0.243382983122	**0.501594128748**
4	0.246659590740	-2.000000000103	0.006760106070	0.060840953807	**0.246678758945**
…					
20	0.000003763727	-2.000000000000	0.000000000002	0.000000000014	**0.000003763727**
21	0.000001881863	-2.000000000000	0.000000000000	0.000000000004	**0.000001881863**
22	0.000000940932	-2.000000000000	0.000000000000	0.000000000001	**0.000000940932**

finished!!

used 22 times.

```
start!!
initial point: [1,0]
 i        x                 y                 f(x,y)            g(x,y)            error
 0   1.000000000000   0.000000000000   -0.888888888889   -1.000000000000
 1   5.000000000000   7.000000000000   14.027777777778   16.000000000000   11.000000000000
 2   3.060769230769   3.607692307692    3.294784089415    3.760615976331    5.331538461538
 3   2.201023439531   2.105341355810    0.646393798495    0.739162825553    2.362096743121
 4   1.922967082695   1.620487063553    0.067361514244    0.077315337577    0.762910649093
 5   1.886278246012   1.556699550640    0.001166775116    0.001346070737    0.100476349596
 6   1.885618296191   1.555555923398    0.000000375363    0.000000435534    0.001803577063
 7   1.885618083164   1.555555555556    0.000000000000    0.000000000000    0.000000580869
finished!!
used 7 times.
```

由觀察可看出：

收斂到(0, -2)的 initial guess 是線性收斂

收斂到$(\pm\frac{4}{3}\sqrt{2}\ ,\frac{14}{9})$ 的 initial guess 則為二次收斂

8. 不同評估誤差方式不會影響結果(iteration 次數)

```
ld error_1(ld e0, ld e1) {// 1 norm
    return abs(e0) + abs(e1);
}
ld error_2(ld e0,ld e1) {// 2 norm
    return sqrt(e0 * e0 + e1 * e1);
}
ld error_inf(ld e0, ld e1) {// inf norm
    return max(abs(e0), abs(e1));
}
```

9. 想法和心得

收斂速度和點在橢圓或拋物線的哪一側(內部或外部)沒有太大關係，在靠近三個 root 的初始點收斂速度稍微快了一點但不明顯。觀察收斂 path 時有時候會有點突然離函數很遠，猜測是因為該點在兩函數梯度方向比較平行。

或許是因為這兩個函數比較溫和，此圖沒有產生在計算過程中離 root 越來越遠而發散的點，且那些會發散的點的 condition number 都是無限大。

在會發散的初始點只要加上一點點小變動，就會很快的收斂。