

第二次大作业

1.问题描述

22:32 11月19日周日

第三章：无约束优化问题

第四章：等式约束优化问题

第三章：无约束优化问题

例1 两个变量的二次规划问题： $\min f(x) = \frac{1}{2}(x_1^2 + \gamma x_2^2)$, $\gamma > 0$, 最优解 $x^* = (0,0)$.

$x^k = \left(\frac{\gamma-1}{\gamma+1}\right)^k \begin{bmatrix} \gamma \\ (-1)^k \end{bmatrix}$

其Hessian矩阵 $\nabla^2 f(x)$ 是常值, 其特征值为1和 γ , 则 $\frac{M}{m} = \frac{\max\{1, \gamma\}}{\min\{1, \gamma\}} = \max\left\{\gamma, \frac{1}{\gamma}\right\}$

- $\gamma = 1$ 时, 一次迭代可得到最优解;
- 对于 γ 离1不远的情况, 如 $\frac{1}{3}$ 和3之间, 收敛速度很快
- 如果 $\gamma \gg 1$ 或 $\gamma \ll 1$ 时, 收敛速度将会非常慢

作业：编程证明以上结论

$\gamma = 10$ 时迭代过程

22:32 11月19日周日

第三章：无约束优化问题

第四章：等式约束优化问题

第三章：无约束优化问题

大作业

对于 \mathbb{R}^2 空间非二次规划问题 $\min f(x) = e^{x_1+3x_2-0.1} + e^{x_1-3x_2-0.1} + e^{-x_1-0.1}$, 分析回溯直线搜索采用不同的 α, β 值时, 误差随迭代次数改变的情况。注：初始值相同。

36 / 83

$\min f(x)$

22:32 11月19日周日

第三章：无约束优化问题

第四章：等式约束优化问题

第三章：无约束优化问题

例1：两个变量的非二次规划问题，第二次大作业补充

$$\min_x f(x) = e^{x_1+3x_2-0.1} + e^{x_1-3x_2-0.1} + e^{-x_1-0.1}$$

□ 使用Newton下降方法，令回溯直线搜索时 $\alpha = 0.1$ ， $\beta = 0.7$ 。

开始几次迭代过程

误差和迭代次数的关系

经过5次就收敛到了很高的精度

79 / 83

500

2.问题求解：

问题一：取不同的 γ 值进行计算，将每次迭代的结果保存，绘制到平面图。

部分代码如下：

```

5  def cal(r, epsilon):
6      k = 0
7      points = []
8      while True:
9          xk = ((r - 1) / (r + 1)) ** k * np.array([r, (-1) ** k])
10         points.append(xk)
11         if np.linalg.norm(xk, ord=2) <= epsilon:
12             break
13         k += 1

```

问题二：使用梯度下降算法，首先求出一个近似最优解，再在相同初始点，不同的 α, β 的值的的情况下，观察误差随迭代次数的变化。

部分代码如下：

```

6  def f(x):
7      return np.exp(x[0] + 3 * x[1] - 0.1) + \
8          np.exp(x[0] - 3 * x[1] - 0.1) + \
9          np.exp(-x[0] - 0.1)
10
11
12 def df(x):
13     return np.array([np.exp(x[0] + 3 * x[1] - 0.1) +
14                     np.exp(x[0] - 3 * x[1] - 0.1) -
15                     np.exp(-x[0] - 0.1),
16                     3 * np.exp(x[0] + 3 * x[1] - 0.1) -
17                     3 * np.exp(x[0] - 3 * x[1] - 0.1)])
18
19 def d2f(x):
20     return np.array([[np.exp(x[0]+3*x[1]-0.1)+np.exp(x[0]-3*x[1]-0.1)+np.exp(-x[0]-0.1),
21                     3*np.exp(x[0]+3*x[1]-0.1)-3*np.exp(x[0]-3*x[1]-0.1)],
22                     [3*np.exp(x[0]+3*x[1]-0.1)-3*np.exp(x[0]-3*x[1]-0.1),
23                     9*np.exp(x[0]+3*x[1]-0.1)+9*np.exp(x[0]-3*x[1]-0.1)]]

```

```

26 def minf(x, alpha, beta, epsilon):
27     while True:
28         d = -df(x)
29         if np.linalg.norm(df(x), ord=2) <= epsilon:
30             break
31         t = 1
32         while f(x + t * d) > f(x) + alpha * t * np.transpose(df(x)).dot(d):
33             t = beta * t
34         x = x + t * d
35     return f(x)
36
37 def cal(x, alpha, beta, num, epsilon=1e-5):
38     errors = []
39     while True:
40         errors.append(f(x) - num)
41         d = -df(x)
42         if np.linalg.norm(df(x), ord=2) <= epsilon:
43             break
44         t = 1
45         while f(x + t * d) > f(x) + alpha * t * np.transpose(df(x)).dot(d):
46             t = beta * t
47         x = x + t * d
48     return errors

```

问题三：使用牛顿下降法，首先求出一个近似最优解，再在相同初始点，不同的 α, β 的值的情况下，观察误差随迭代次数的变化。

部分代码如下：

```

26 def minf(x, alpha, beta, epsilon):
27     while True:
28         d = -df(x)
29         if np.linalg.norm(df(x), ord=2) <= epsilon:
30             break
31         t = 1
32         while f(x + t * d) > f(x) + alpha * t * np.transpose(df(x)).dot(d):
33             t = beta * t
34         x = x + t * d
35     return f(x)
36
37 def newton(x, alpha, beta, num, epsilon):
38     errors = []
39     while True:
40         d = df(x)
41         d2 = d2f(x)
42         errors.append(f(x) - num)
43         d = -np.linalg.inv(d2).dot(d)
44         lambda2 = d.transpose().dot(d2).dot(d)
45         if 0.5 * lambda2 <= epsilon:
46             break
47         t = 1
48         while f(x + t * d) > f(x) - alpha * t * lambda2:
49             t = beta * t
50         x = x + t * d
51     return errors

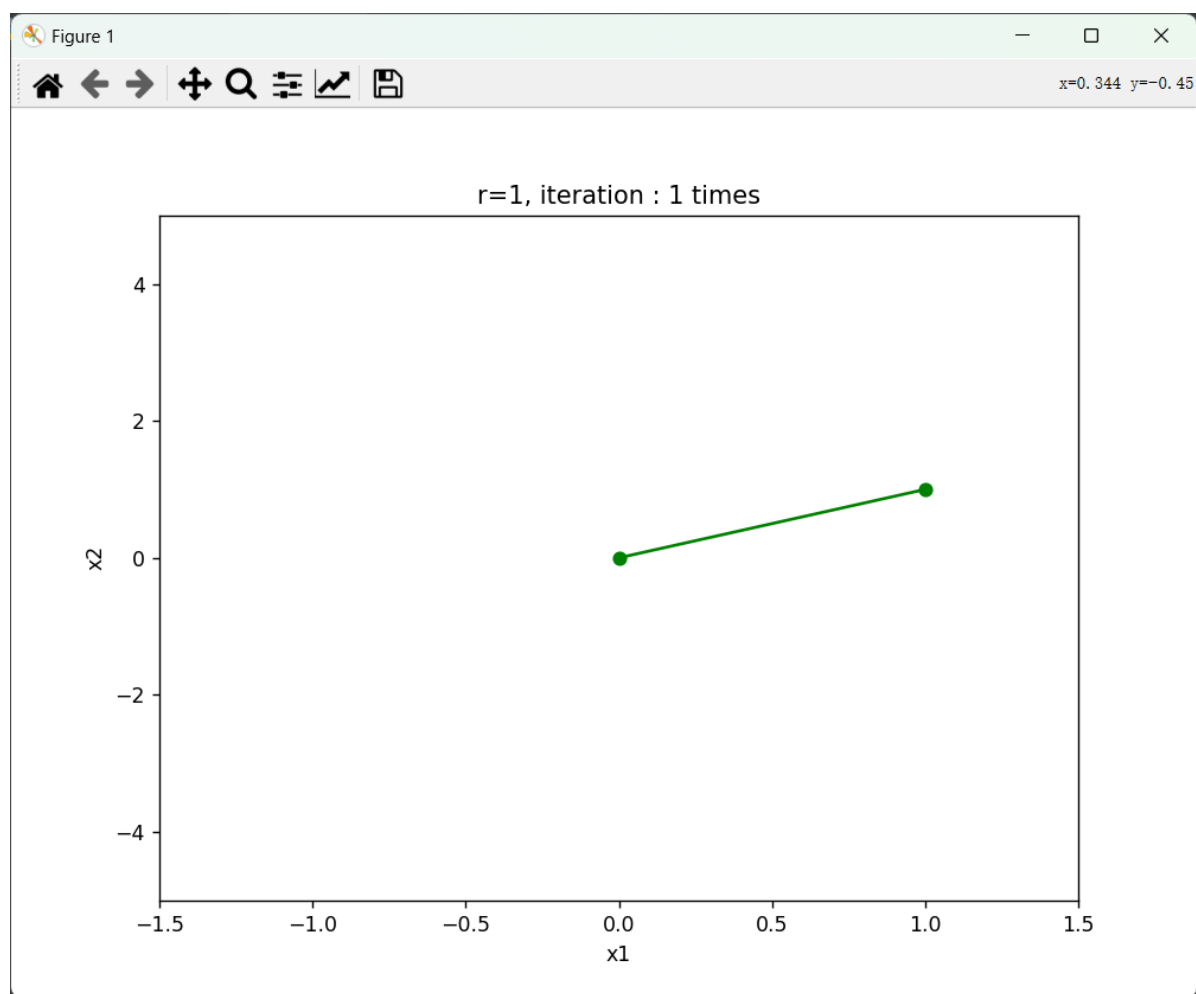
```

1 | 其中 $f(x)$ $df(x)$ $d2f(x)$ 的实现与第二题相同

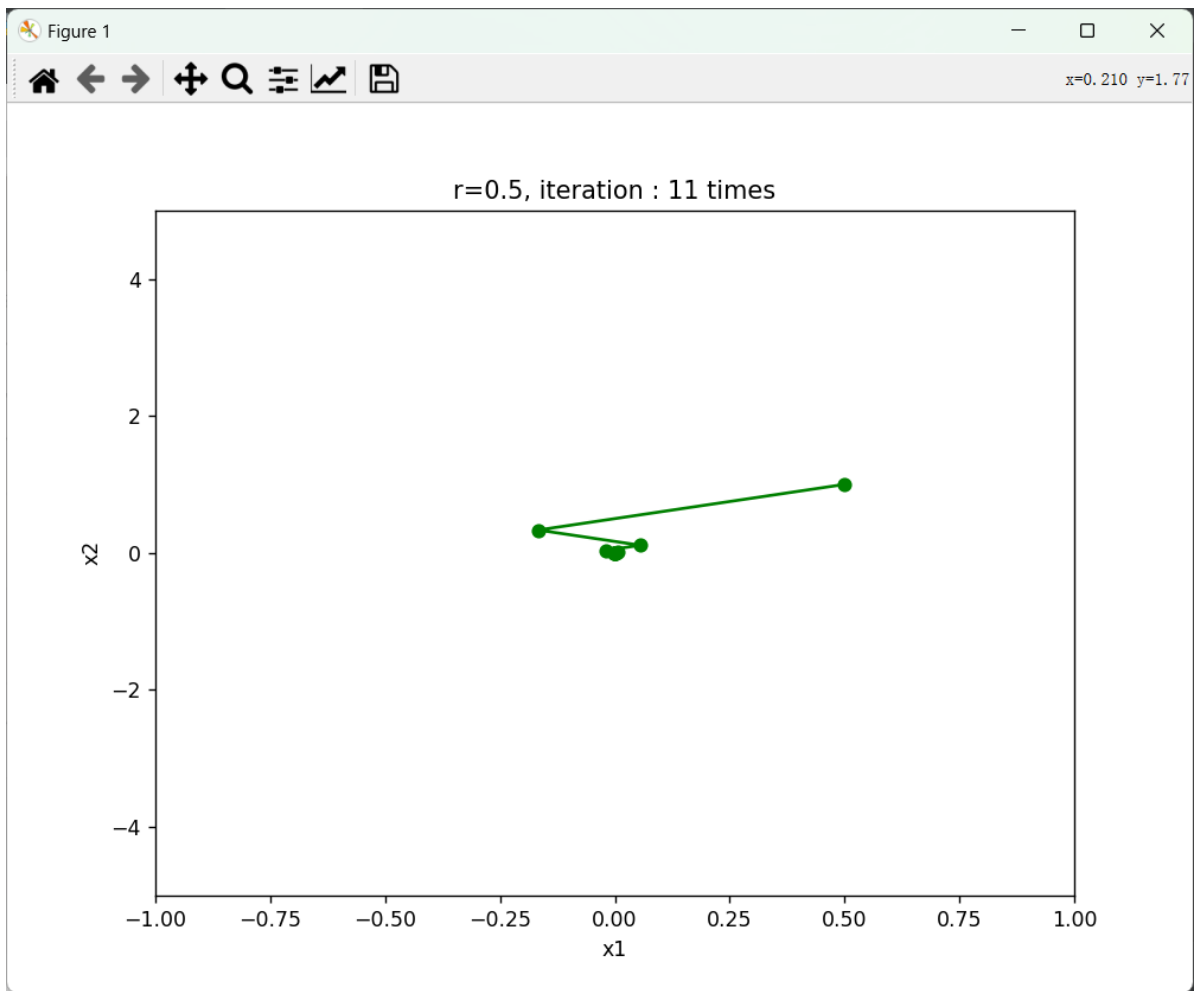
3.结果分析

题目一：

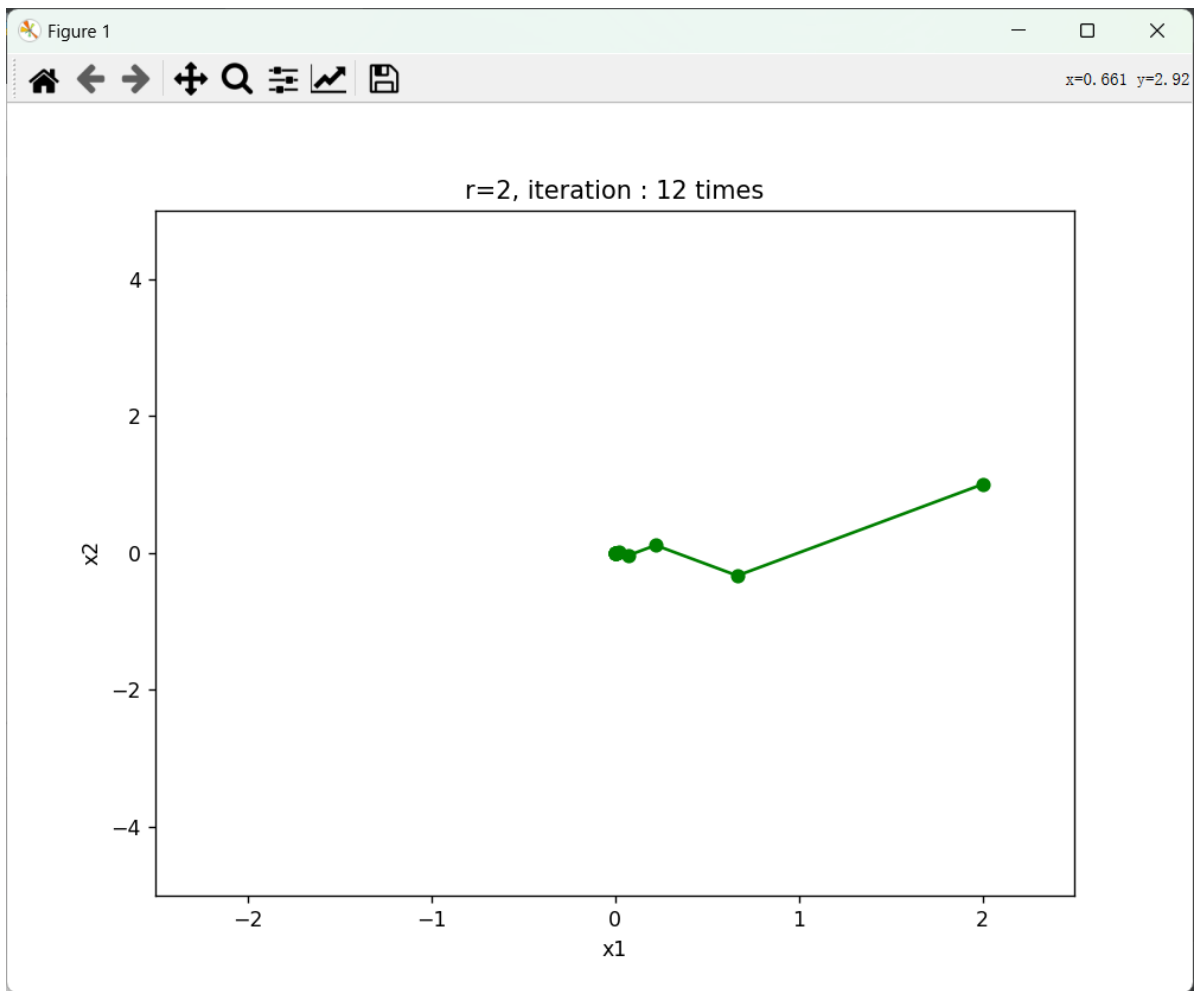
当 $\gamma=1$ 时，只用了一次迭代就收敛



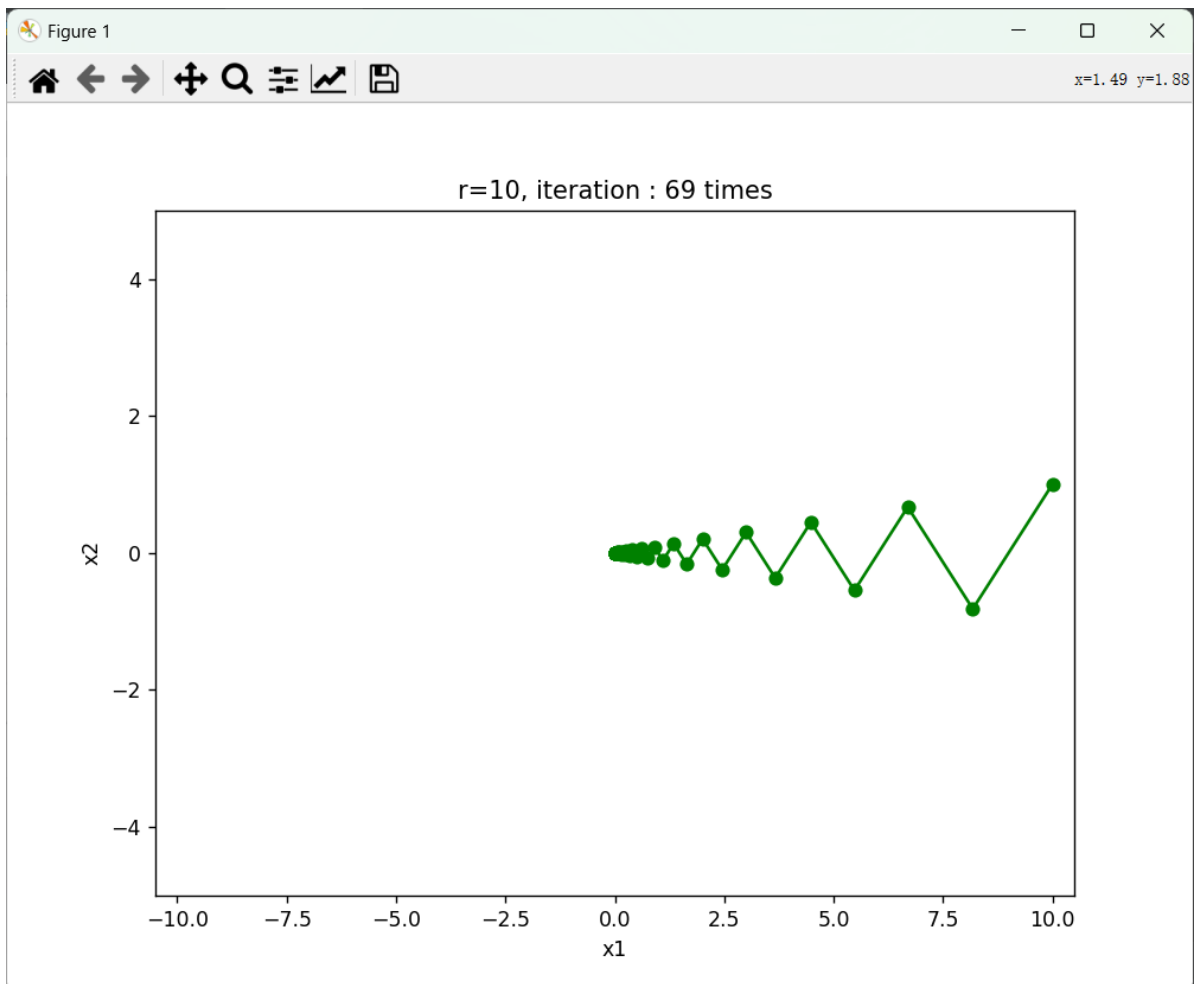
当 $\gamma=0.5$ 时, 进行了11次迭代



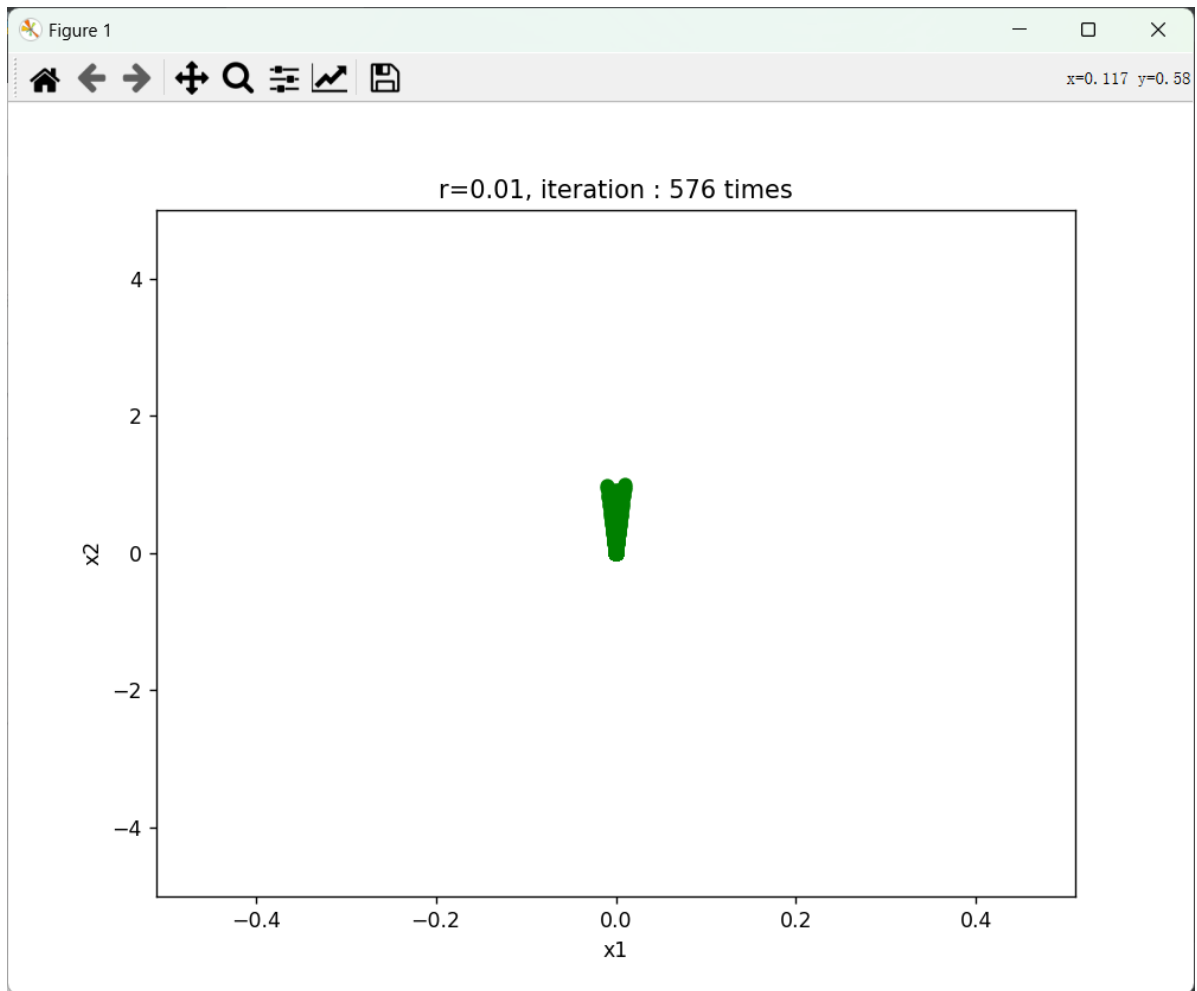
当 $\gamma=2$ 时，进行了12次迭代



当 $\gamma=10$ 时, 进行了69次迭代



当 $\gamma=0.01$ 时，进行了576次迭代

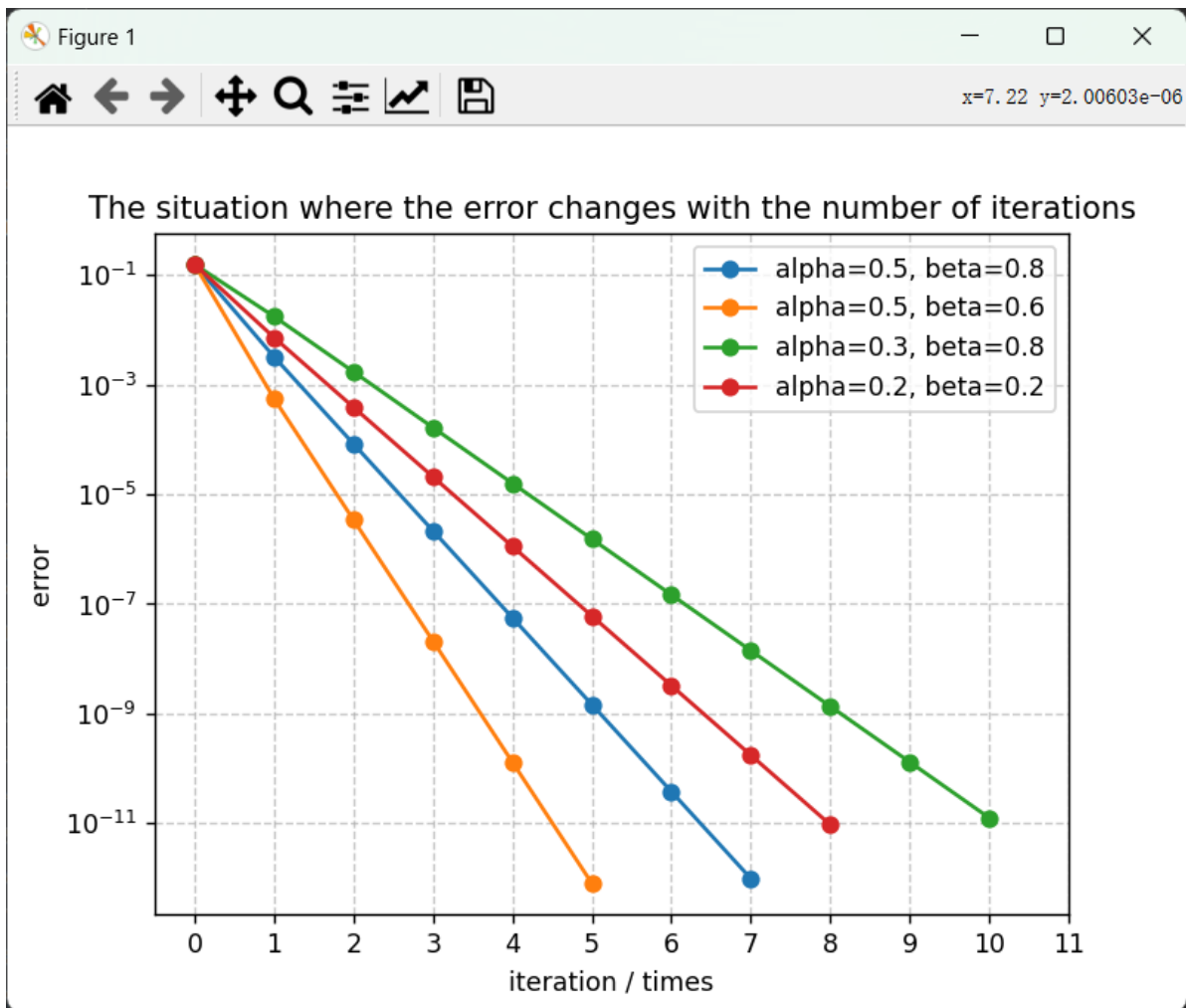


结果符合规律：

当 $\gamma = 1$ 时，一次迭代可得到最优解；当 γ 离1不远时，10次迭代得到了最优解；

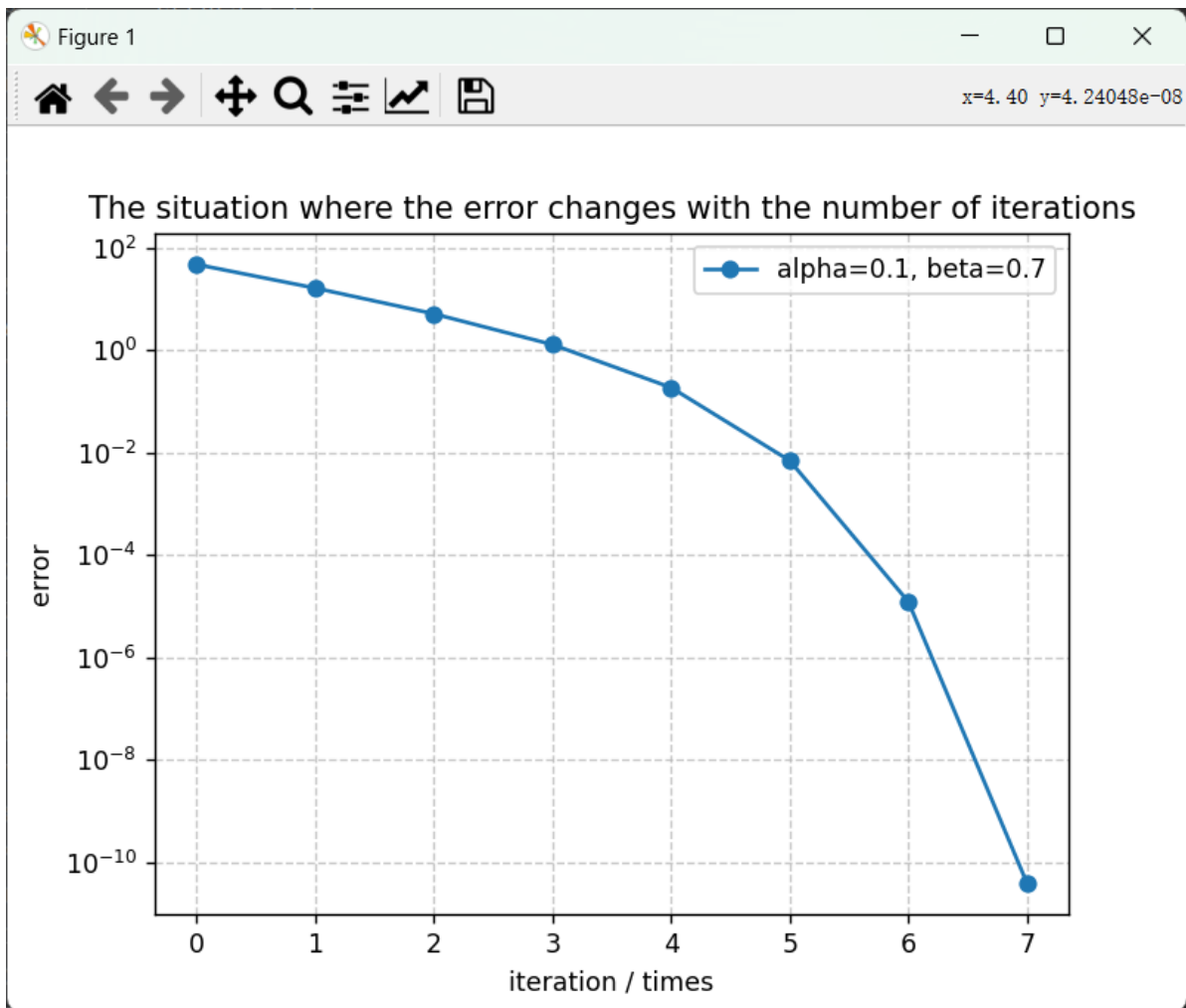
当 $\gamma \gg 1$ 或 $\gamma \ll 1$ 时，经过了非常多的迭代才能收敛。

问题二：



可以看出 α, β 的值需要在适中的范围内，过大或过小都会导致迭代次增加。当 β 过小时误差的改变会非常慢，过大时回溯搜索需要迭代很多次。 α 对于迭代速度的影响没有 β 大，但仍然会影响收敛速度。（可以在代码内修改 α, β 的值观察得出，因考虑美观仅绘制部分）

问题三：



可在代码中修改 γ 和 α 的值，观察不同参数下牛顿下降方法的误差收敛情况