1.有如下两个函数：

$$f_1(x,y)=(x-y)^2+xy$$

$$f_2(x,y)=(3-y)^2+35*(x+6-(y-4)^2)^2$$

- 假设起始点$(x,y)=(2,3)$,学习率$\eta=0.05$,针对f_1运行20轮梯度下降，请画出(x,y)收敛路径，指出得到的结果。(10分) 再选择一种其他的梯度下降方法，运行20轮，请画出(x,y)收敛路径，指出得到的结果。(10分)
- 假设起始点$(x,y)=(0,2)$,学习率$\eta=0.0015$,针对f_2运行100轮梯度下降，请画出(x,y)收敛路径，指出得到的结果。(10分) 再选择一种其他的梯度下降方法，运行100轮，请画出(x,y)收敛路径，指出得到的结果。(10分)

2.考虑函数$f(x)=\frac{\beta}{4}(\frac{1}{2}x_1^2+\frac{1}{2}\sum_{i=1}^{2k}(x_i-x_{i+1})^2+\frac{1}{2}x_{2k+1}^2-x_1)$，假设$x_0=\vec{0}$

- 证明f是$\beta-smooth$函数(10分)。
- 求解f的最小值(10分)，和对应的x^*(10分)。
- 证明我们采用梯度下降法，第t轮中f的导数$\in span\{e_1,e_2,...,e_l\}$(10分)
- 证明(15分)

$$f(x_k)-f^*\geq\frac{3\beta||x_0-x^*||_2^2}{32(k+1)^2}$$

1 解：

1) 编写 python 代码绘制 $f_1(x_1, x_2)$ 函数的图像大致如下：

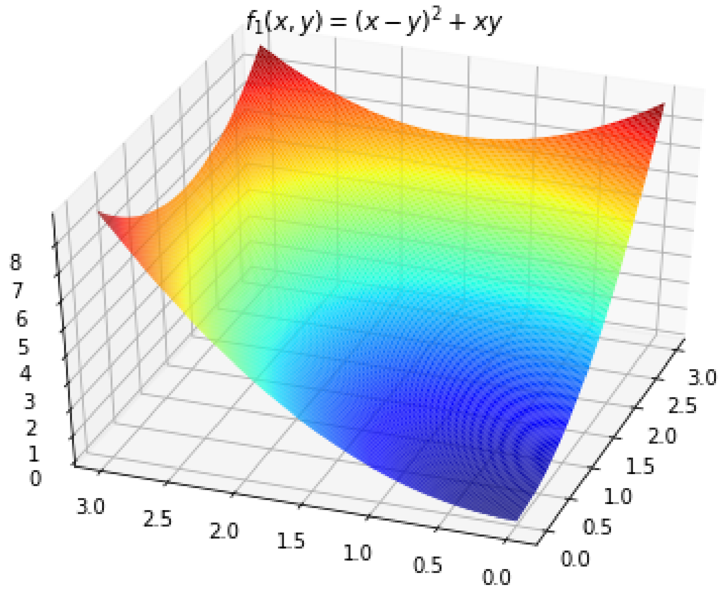

图 1: 函数 $f_1(x, y)$

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
from mpl_toolkits.mplot3d import Axes3D
from matplotlib.font_manager import FontProperties
font = FontProperties(fname=r"./fonts/simsun.ttc", size=12)
# 二维原始图像
def f(x1, x2):
return (x1-x2) ** 2 + x1 * x2
## 偏导数
def fx1(x1, x2):
return 2*x1-x2
def fx2(x1, x2):
return 2*x2-x1

X1 = np.arange(0,3,0.02)
X2 = np.arange(0,3,0.02)
X1, X2 = np.meshgrid(X1, X2) # 生成xv、yv，将X1、X2变成n*m的矩阵，方便后面绘图
Y = np.array(list(map(lambda t : f(t[0],t[1]),zip(X1.flatten(),X2.flatten()))))
Y.shape = X1.shape # 1600的Y图还原成原来的（40,40）

%matplotlib inline
#作图
fig = plt.figure(facecolor='w')
ax = Axes3D(fig)
ax.plot_surface(X1,X2,Y,rstride=1,cstride=1,cmap=plt.cm.jet)
ax.set_title(u'$ f_1(x,y) = (x-y)^2 + x y $')
ax.view_init(elev=40, azim=200)
plt.show()
```

经过梯度下降，分别对 x, y 计算偏导数，编写 python 程序计算如下：

```
x1 = 2
x2 = 3
alpha = 0.05
```

```
最终结果为:(0.87684, 0.91559, 0.80433)
迭代过程中(X1,X2)的取值，迭代次数:20
[(2, 3), (1.95, 2.8), (1.895, 2.6174999999999997), (1.836375, 2.4505), (1.7752625000000002, 2.2972687499999997), (1.7125996875000002, 2.1563049999999997), (1.6491549687500002, 2.0263044843749998), (1.58555469609375, 1.9061317843749999), (1.522305815703125, 1.7947963407421874), (1.459815051169922, 1.691431997453125), (1.398405145925586, 1.5952795502663084), (1.3383286088463429, 1.505671852535957), (1.2797793405885065, 1.4220210977246783), (1.2229024614158899, 1.3438079549816357), (1.1678026130233827, 1.2705722825542667), (1.1145509658487578, 1.201905184950009), (1.0631911285113824, 1.1374422147474461), (1.0137441263976166, 1.0768575496982706), (0.9662125912427685, 1.0198590010483244), (0.9205842821709078, 0.9661837305056303), (0.8768350404790985, 0.9155945715636127)]
```

图 2: 最终结果

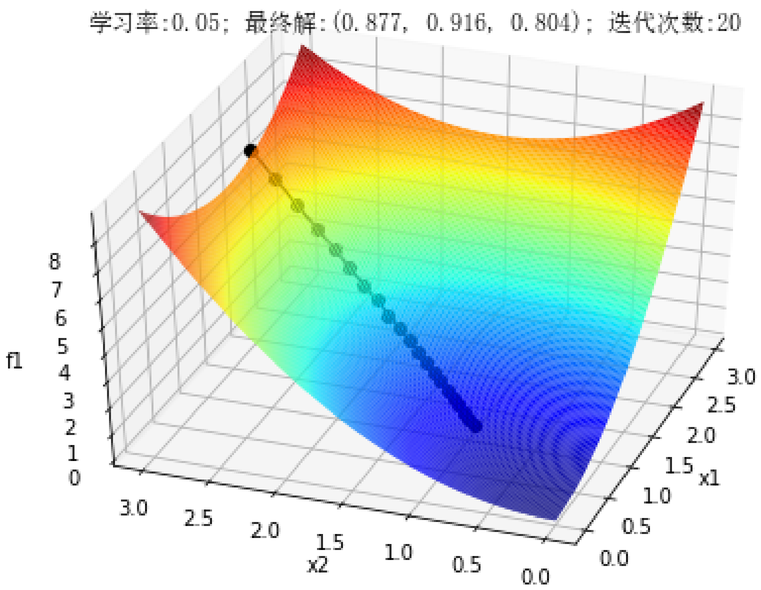

图 3: 收敛路径

```
#保存梯度下降经过的点
GD_X1 = [x1]
GD_X2 = [x2]
GD_Y = [f(x1,x2)]
# 定义y的变化量和迭代次数
y_change = f(x1,x2)
iter_num = 0

while(y_change > 1e-10 and iter_num < 20) :
        tmp_x1 = x1 - alpha * fx1(x1,x2)
        tmp_x2 = x2 - alpha * fx2(x1,x2)
        tmp_y = f(tmp_x1,tmp_x2)

        y_change = np.absolute(tmp_y - f(x1,x2))
    x1 = tmp_x1
    x2 = tmp_x2
    GD_X1.append(x1)
        GD_X2.append(x2)
    GD_Y.append(tmp_y)
    iter_num += 1
print(u"最终结果为:(%.5f, %.5f, %.5f)" % (x1, x2, f(x1,x2)))
print(u"迭代过程中(X1,X2)的取值，迭代次数:%d" % iter_num)
print(list(zip(GD_X1,GD_X2)))

# 作图
fig = plt.figure(facecolor='w')
ax = Axes3D(fig)
ax.plot_surface(X1,X2,Y,rstride=1,cstride=1,cmap=plt.cm.jet)
ax.plot(GD_X1,GD_X2,GD_Y,'ko-')
```

```
    ax.set_xlabel('x1')
    ax.set_ylabel('x2')
    ax.set_zlabel('f1')

    ax.set_title(u'学习率:%.2f; 最终解:(%.3f, %.3f, %.3f); 迭代次数:%d' % (alpha, x1, x2, f(x1,x2),
                                                        iter_num),fontproperties=font)
    ax.view_init(elev=40, azim=200)
    plt.show()
```

选用动量梯度下降法，结果和代码如下：

```
最终结果为:(1.26915, 1.14441, 1.46799)
迭代过程中(X1,X2)的取值，迭代次数:20
[(2, 3), (1.995, 2.98), (1.9854500000000002, 2.9421749999999998), (1.9717113750000002, 2.888638), (1.9540726887500002, 2.821426876875),
(1.932764278621875З, 2.7424929607375), (1.9079715315240315, 2.6536913279994847), (1.8798468004607294, 2.5567728029128958), (1.8485199385
137145, 2.4533776363081405), (1.8141074515578048, 2.345030809693348), (1.7767202928303747, 2.23313889490089), (1.7364703415218883, 2.118
988384102821), (1.693475623849545б, 2.0037453922511403), (1.6478643486671976, 1.888456623781364), (1.5997778406353191, 1.774051487664087
6), (1.5493724624385958, 1.6613452394850745), (1.4968206236345842, 1.551043026041305), (1.4423109776048346, 1.4437447067996723), (1.3860
4790993601, 1.3399503273022302), (1.3282504215712188, 1.2400661220311902), (1.2691505084373504, 1.1444109281747983)]
```

图 4: 最终结果

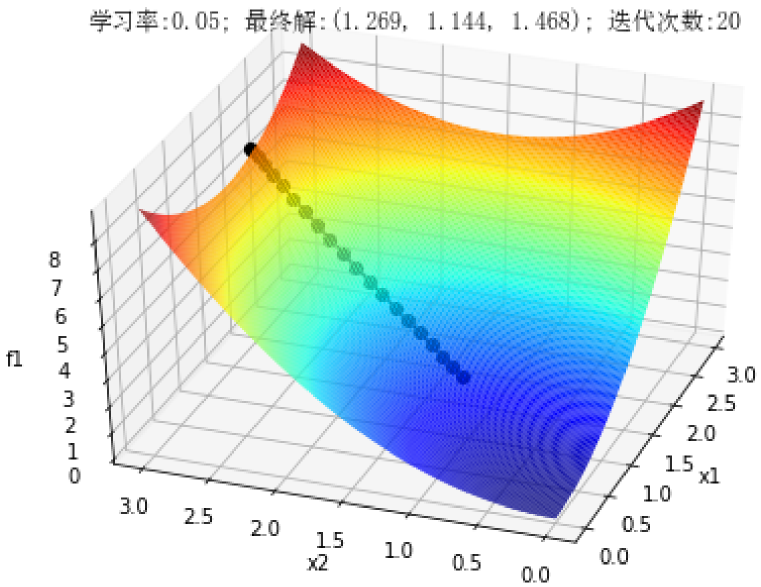

图 5: 收敛路径

```
    x1 = 2
    x2 = 3
    alpha = 0.05
    #保存梯度下降经过的点
    GD_X1 = [x1]
    GD_X2 = [x2]
    GD_Y = [f(x1,x2)]
    # 定义y的变化量和迭代次数
    y_change = f(x1,x2)
    iter_num = 0

    # 动量梯度下降
    beta = 0.9
    v_dx1 = v_dx2 = 0
    while(y_change > 1e-10 and iter_num < 20) :
        v_dx1 = beta*v_dx1+(1-beta)*fx1(x1, x2)
        v_dx2 = beta*v_dx2+(1-beta)*fx2(x1, x2)
```

```
        tmp_x1 = x1 - alpha*v_dx1
        tmp_x2 = x2 - alpha*v_dx2
        tmp_y = f(tmp_x1,tmp_x2)

        y_change = np.absolute(tmp_y - f(x1,x2))
        x1 = tmp_x1
        x2 = tmp_x2
        GD_X1.append(x1)
        GD_X2.append(x2)
        GD_Y.append(tmp_y)
        iter_num += 1
    print(u"最终结果为:(%.5f, %.5f, %.5f)" % (x1, x2, f(x1,x2)))
    print(u"迭代过程中(X1,X2)的取值, 迭代次数:%d" % iter_num)
    print(list(zip(GD_X1,GD_X2)))


    # 作图
    fig = plt.figure(facecolor='w')
    ax = Axes3D(fig)
    ax.plot_surface(X1,X2,Y,rstride=1,cstride=1,cmap=plt.cm.jet)
    ax.plot(GD_X1,GD_X2,GD_Y,'ko-')
    ax.set_xlabel('x1')
    ax.set_ylabel('x2')
    ax.set_zlabel('f1')

    ax.set_title(u'学习率:%.2f; 最终解:(%.3f, %.3f, %.3f); 迭代次数:%d' % (alpha, x1, x2, f(x1,x2),
                                                      iter_num),fontproperties=font)
    ax.view_init(elev=40, azim=200)
    plt.show()
```

2) 对于函数 $f_2(x,y)$，修改代码相应部分：

```
    # 二维原始图像
    def f(x1, x2):
      return (3-x2)**2+35*(x1+6-(x2-4)**2)**2
    ## 偏导数
    def fx1(x1, x2):
      return 70*(x1+6-(x2-4)**2)
    def fx2(x1, x2):
      return 2*(x2-3)+70*(x1+6-(x2-4)**2)*(-2*(x2-4))
```

得到的结果如下：

```
最终结果为:(-1.57164, 1.92520, 1.68966)
迭代过程中(X1,X2)的取值，迭代次数:100
[(0, 2), (-0.21, 1.163), (0.0271, 2.5141), (-0.3739, 1.3238), (-0.2126, 2.192), (-0.4771, 1.2382), (-0.2561, 2.4641), (-0.6115, 1.374),
(-0.4532, 2.2102), (-0.6993, 1.3318), (-0.5083, 2.3558), (-0.8011, 1.395), (-0.6344, 2.2681), (-0.8829, 1.4097), (-0.7157, 2.2806), (-0.
9601, 1.4422), (-0.8024, 2.2539), (-1.028, 1.4682), (-0.877, 2.2373), (-1.0887, 1.4934), (-0.9446, 2.22), (-1.1428, 1.517), (-1.0054, 2.
2035), (-1.191, 1.5392), (-1.0601, 2.1877), (-1.2339, 1.5601), (-1.1093, 2.1727), (-1.2722, 1.5798), (-1.1536, 2.1582), (-1.3063, 1.598
3), (-1.1935, 2.1445), (-1.3366, 1.6157), (-1.2294, 2.1313), (-1.3636, 1.6322), (-1.2617, 2.1187), (-1.3876, 1.6477), (-1.2909, 2.1068),
(-1.409, 1.6623), (-1.3173, 2.0954), (-1.428, 1.6761), (-1.341, 2.0845), (-1.4449, 1.6891), (-1.3625, 2.0741), (-1.46, 1.7014), (-1.381
9, 2.0642), (-1.4733, 1.713), (-1.3995, 2.0548), (-1.4852, 1.724), (-1.4153, 2.0459), (-1.4958, 1.7344), (-1.4298, 2.0374), (-1.5052, 1.
7442), (-1.4428, 2.0293), (-1.5135, 1.7535), (-1.4547, 2.0216), (-1.521, 1.7623), (-1.4655, 2.0143), (-1.5276, 1.7706), (-1.4753, 2.007
3), (-1.5335, 1.7785), (-1.4843, 2.0007), (-1.5387, 1.786), (-1.4925, 1.9945), (-1.5435, 1.7931), (-1.5, 1.9886), (-1.5477, 1.7998), (-
1.5069, 1.983), (-1.5514, 1.8061), (-1.5132, 1.9777), (-1.5549, 1.8121), (-1.519, 1.9727), (-1.5579, 1.8178), (-1.5243, 1.9679), (-1.560
7, 1.8232), (-1.5293, 1.9634), (-1.5632, 1.8284), (-1.5339, 1.9592), (-1.5656, 1.8332), (-1.5382, 1.9553), (-1.5677, 1.8378), (-1.5422,
1.9515), (-1.5697, 1.8422), (-1.5459, 1.948), (-1.5715, 1.8463), (-1.5494, 1.9447), (-1.5732, 1.8502), (-1.5527, 1.9417), (-1.5748, 1.85
39), (-1.5559, 1.9388), (-1.5764, 1.8573), (-1.5588, 1.9361), (-1.5779, 1.8606), (-1.5616, 1.9336), (-1.5793, 1.8638), (-1.5643, 1.931
2), (-1.5807, 1.8667), (-1.5668, 1.9291), (-1.582, 1.8695), (-1.5693, 1.9271), (-1.5833, 1.8721), (-1.5716, 1.9252)]
```

图 6: 方法一结果

方法二的结果如下：

2

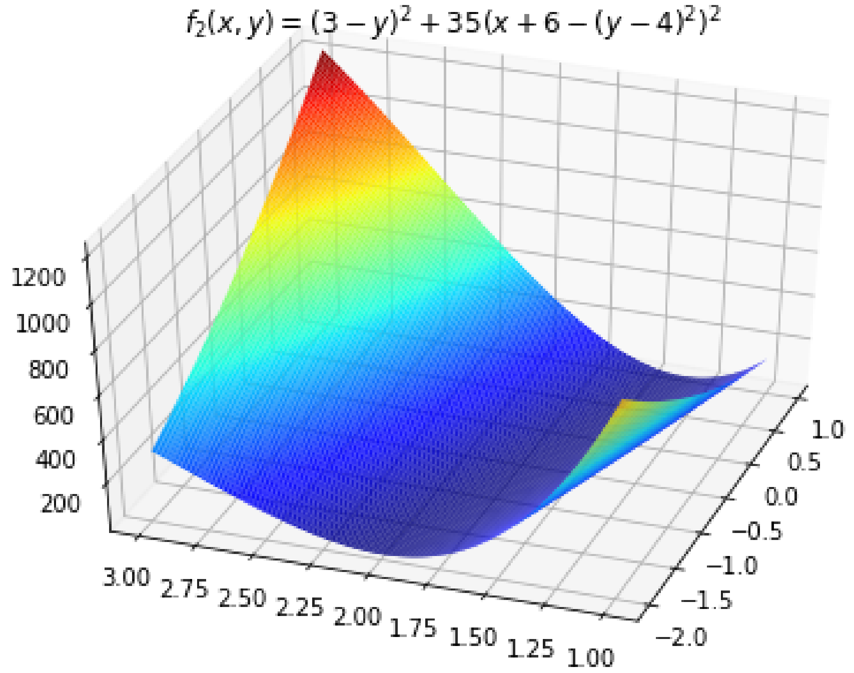

图 7: 函数 $f_2(x, y)$

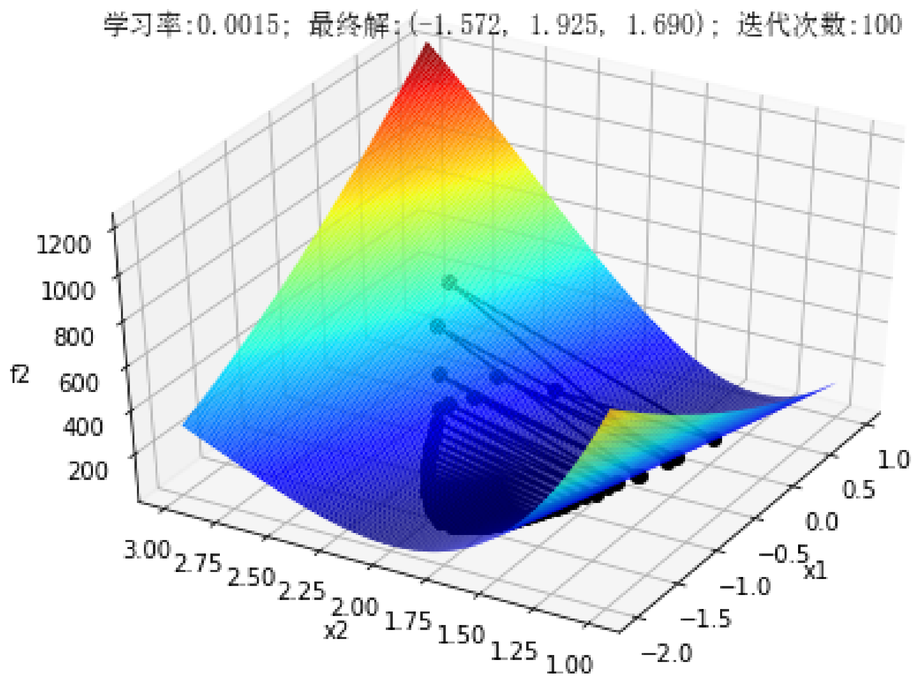

图 8: 方法一收敛路径

```
最终结果为:(-0.34241, 1.62329, 1.89806)
迭代过程中(X1,X2)的取值，迭代次数:100
[(0, 2), (-0.021, 1.9163), (-0.0571, 1.7697), (-0.0997, 1.5927), (-0.1392, 1.4285), (-0.1669, 1.3218), (-0.1777, 1.3017), (-0.1721, 1.36
67), (-0.1555, 1.4869), (-0.1355, 1.6204), (-0.1197, 1.7308), (-0.1132, 1.7958), (-0.1181, 1.807), (-0.1337, 1.7681), (-0.1571, 1.692),
(-0.1836, 1.5988), (-0.208, 1.5128), (-0.2258, 1.4565), (-0.2345, 1.4434), (-0.2342, 1.4734), (-0.2275, 1.5337), (-0.2182, 1.6044), (-0.
2103, 1.6664), (-0.2068, 1.7057), (-0.2092, 1.7159), (-0.2173, 1.698), (-0.2298, 1.6589), (-0.244, 1.6099), (-0.2573, 1.564), (-0.2672,
1.5329), (-0.2724, 1.5237), (-0.2729, 1.5369), (-0.2697, 1.5668), (-0.2649, 1.6038), (-0.2604, 1.6379), (-0.2581, 1.661), (-0.2589, 1.66
89), (-0.2628, 1.6614), (-0.2692, 1.6419), (-0.2767, 1.6163), (-0.2839, 1.5916), (-0.2894, 1.5741), (-0.2926, 1.5676), (-0.2933, 1.572
9), (-0.292, 1.5875), (-0.2896, 1.6067), (-0.2873, 1.6253), (-0.286, 1.6388), (-0.2862, 1.6445), (-0.2882, 1.6418), (-0.2916, 1.6323),
(-0.2957, 1.6191), (-0.2998, 1.6059), (-0.3031, 1.596), (-0.3052, 1.5916), (-0.3061, 1.5935), (-0.3058, 1.6005), (-0.3048, 1.6104), (-0.
3038, 1.6205), (-0.3033, 1.6283), (-0.3036, 1.6322), (-0.3048, 1.6316), (-0.3067, 1.6272), (-0.3092, 1.6205), (-0.3117, 1.6135), (-0.313
8, 1.6079), (-0.3154, 1.6051), (-0.3163, 1.6056), (-0.3166, 1.6089), (-0.3165, 1.614), (-0.3163, 1.6195), (-0.3163, 1.624), (-0.3167, 1.
6266), (-0.3176, 1.6268), (-0.3189, 1.6249), (-0.3205, 1.6216), (-0.3222, 1.6179), (-0.3238, 1.6148), (-0.325, 1.6131), (-0.3259, 1.613
1), (-0.3265, 1.6147), (-0.3269, 1.6173), (-0.3271, 1.6203), (-0.3275, 1.6229), (-0.328, 1.6246), (-0.3288, 1.625), (-0.3299, 1.6243),
(-0.3311, 1.6228), (-0.3324, 1.6209), (-0.3336, 1.6193), (-0.3347, 1.6183), (-0.3356, 1.6182), (-0.3363, 1.6189), (-0.3369, 1.6203), (-
0.3374, 1.622), (-0.338, 1.6235), (-0.3386, 1.6246), (-0.3394, 1.625), (-0.3403, 1.6248), (-0.3414, 1.6242), (-0.3424, 1.6233)]
```

图 9: 方法二结果

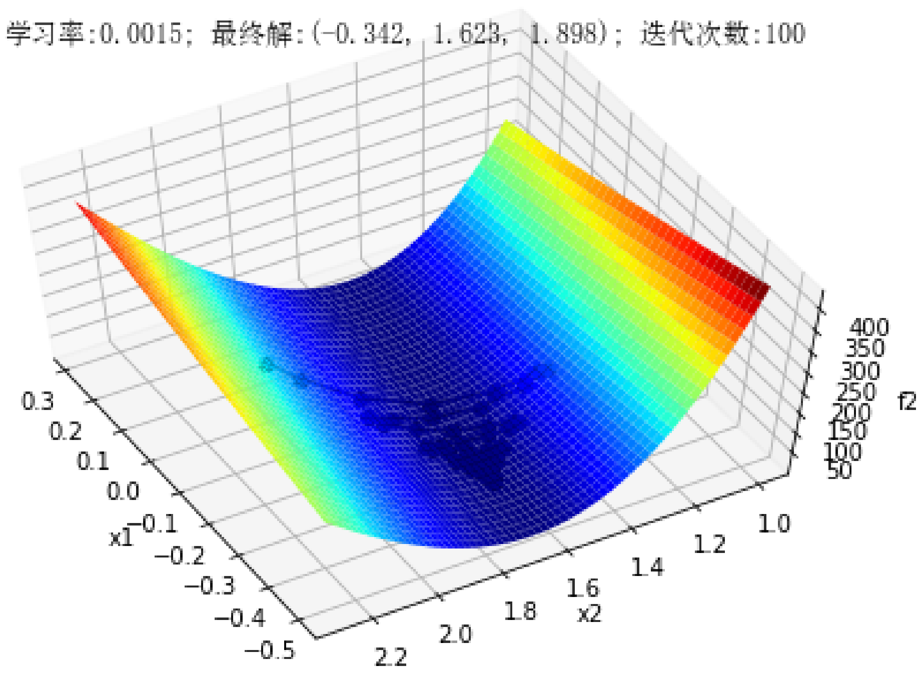

图 10: 方法二收敛路径

1) 证明：

$$
\begin{aligned}
f(x) &= \frac{\beta}{4}(\frac{1}{2}x_1^2 + \frac{1}{2}\sum_{i=1}^{2k}(x_i - x_{i+1})^2 + \frac{1}{2}x_{2k+1}^2 - x_1) \\
&= \frac{\beta}{4}(x_1^2 - x_1x_2 + x_2^2 - x_2x_3 + ... - x_{2k}x_{2k+1} + x_{2k+1}^2 - x_1)
\end{aligned}
$$

$$
= \frac{\beta}{4}([x_1\ x_2\ \cdots\ x_{2k+1}]\begin{pmatrix} 1 & -\frac{1}{2} & 0 & 0 & \cdots & 0 & 0 \\ -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & \cdots & 0 & 0 \\ 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & -\frac{1}{2} \\ 0 & 0 & 0 & 0 & \cdots & -\frac{1}{2} & 1 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{2k+1} \end{pmatrix} - x_1)
$$

$$\bigtriangledown f(x)=\frac{\beta}{4}\begin{pmatrix}2&-1&0&0&\cdots&0&0\\-1&2&-1&0&\cdots&0&0\\0&-1&2&-1&\cdots&0&0\\\vdots&\vdots&\vdots&\vdots&\ddots&\vdots&\vdots\\0&0&0&0&\cdots&2&-1\\0&0&0&0&\cdots&-1&2\end{pmatrix}\begin{pmatrix}x_1\\x_2\\\vdots\\x_{2k+1}\end{pmatrix}-\begin{pmatrix}1\\0\\\vdots\\0\end{pmatrix})$$

令

$$A=\begin{pmatrix}2&-1&0&0&\cdots&0&0\\-1&2&-1&0&\cdots&0&0\\0&-1&2&-1&\cdots&0&0\\\vdots&\vdots&\vdots&\vdots&\ddots&\vdots&\vdots\\0&0&0&0&\cdots&2&-1\\0&0&0&0&\cdots&-1&2\end{pmatrix}$$

所以 $||\bigtriangledown f(x)-\bigtriangledown f(y)||_2=\frac{\beta}{4}||A(x-y)||_2$，又因为 $||A(x-y)||_2^2=(x-y)^TA^TA(x-y)$，而 $(x-y)^T16I(x-y)-(x-y)^TA^TA(x-y)=(x-y)^T(16I-A^TA)(x-y)$，经计算可得：$16I-A^TA$ 的顺序主子式均大于等于 0，所以 $(x-y)^T(16I-A_TA)(x-y)$ 是正定矩阵，即

$$(x-y)^T(16I-A^TA)(x-y)\geq 0$$

所以 $||A(x-y)||_2\leq||16(x-y)||_2=4||x-y||_2$，所以 $||\bigtriangledown f(x)-\bigtriangledown f(y)||_2=\frac{\beta}{4}||A(x-y)||_2\leq\beta||x-y||_2$. 所以该函数是 $\beta-smooth$.

2) 解：令 $\bigtriangledown f(x)=0$ 可得，$Ax=\begin{pmatrix}1\\0\\\vdots\\0\end{pmatrix}$，可以解得 $x=\begin{pmatrix}\frac{2k+1}{2k+2}\\\frac{2k}{2k+2}\\\vdots\\\frac{1}{2k+2}\end{pmatrix}$，此时 $f(x^*)=-\frac{\beta(2k+1)}{8(2k+2)}$

3) 证明：对于第 1 轮，由于 $x^{(0)}=\vec{0}$，$\bigtriangledown f(x^{(0)})=-\frac{\beta}{4}\begin{pmatrix}1\\0\\\vdots\\0\end{pmatrix}$，所以第 1 轮中 f 的导数 $\in span\{e_1\}$.

假设第 t 轮中 f 的导数 $\in span\{e_1,e_2,\cdots,e_t\}$ 成立,那么 $x^{(t+1)}=x^{(t)}-\eta\bigtriangledown f(x^{(t)})\in span\{e_1,e_2,\cdots,e_t\}$，则第 $t+1$ 轮中 f 的导数为

$$\bigtriangledown f(x^{(t+1)})=\frac{\beta}{4}Ax^{(t+1)}-\begin{pmatrix}1\\0\\\vdots\\0\end{pmatrix}$$

对于矩阵 A 中的第 $t+1$ 行，由于 $a_{t+1,t+1}=2$，$a_{t+1,t}=-1$，所以 $\bigtriangledown f(x^{(t+1)})$ 的第 $t+1$ 行的值与 $x^{(t+1)}$ 的第 t 行的值有关，所以 $\bigtriangledown f(x^{(t+1)})\in span\{e_1,e_2,\cdots,e_t,e_{t+1}\}$，由数学归纳法，原命题成立.

4) 证明：

$$f(x_k)-f^*\geq\frac{3\beta||x_0-x^*||_2^2}{32(k+1)^2}$$

由 (3) 的证明可得第 t 轮中，f 的导数 $\in span\{e_1,e_2,\cdots,e_t\}$，所以 $x^{(t)}\in span\{e_1,e_2,\cdots,e_t\}$，所

以 $x_i = 0 \quad i \in \{t+1, t+2, \cdots, 2k+1\}$，所以 $x^{(t)} A_{2k+1} x^{(t)} = x^{(t)} A_t x^{(t)}$. 令

$$f_t(x) = \frac{\beta}{8} x_T A_t x - \frac{\beta}{4} \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

$$f(x_k) - f(x^*) = f_k(x_k) - f^* \geq f_k^* - f^*$$

由第二问可得 $f_k^* = -\frac{\beta}{8} \frac{k}{k+1}$，所以

$$f_k^* - f^* = \frac{\beta}{8} (\frac{2k+1}{2k+2} - \frac{k}{k+1}) = \frac{\beta}{16} \frac{1}{k+1}$$

$$||x_0 - x^*||_2^2 = ||x^*||_2^2 = \sum_{i=1}^{2k+1} (\frac{i}{t+1})^2 \leq \frac{2(k+1)}{3}$$

而

$$f_k^* - f^* = \frac{\beta}{16(k+1)} = \frac{3 \times \beta \times 2(k+1)}{32(k+1)^2 \times 3} \geq \frac{3\beta ||x_0 - x^*||_2^2}{32(k+1)^2}$$

所以原命题得证.