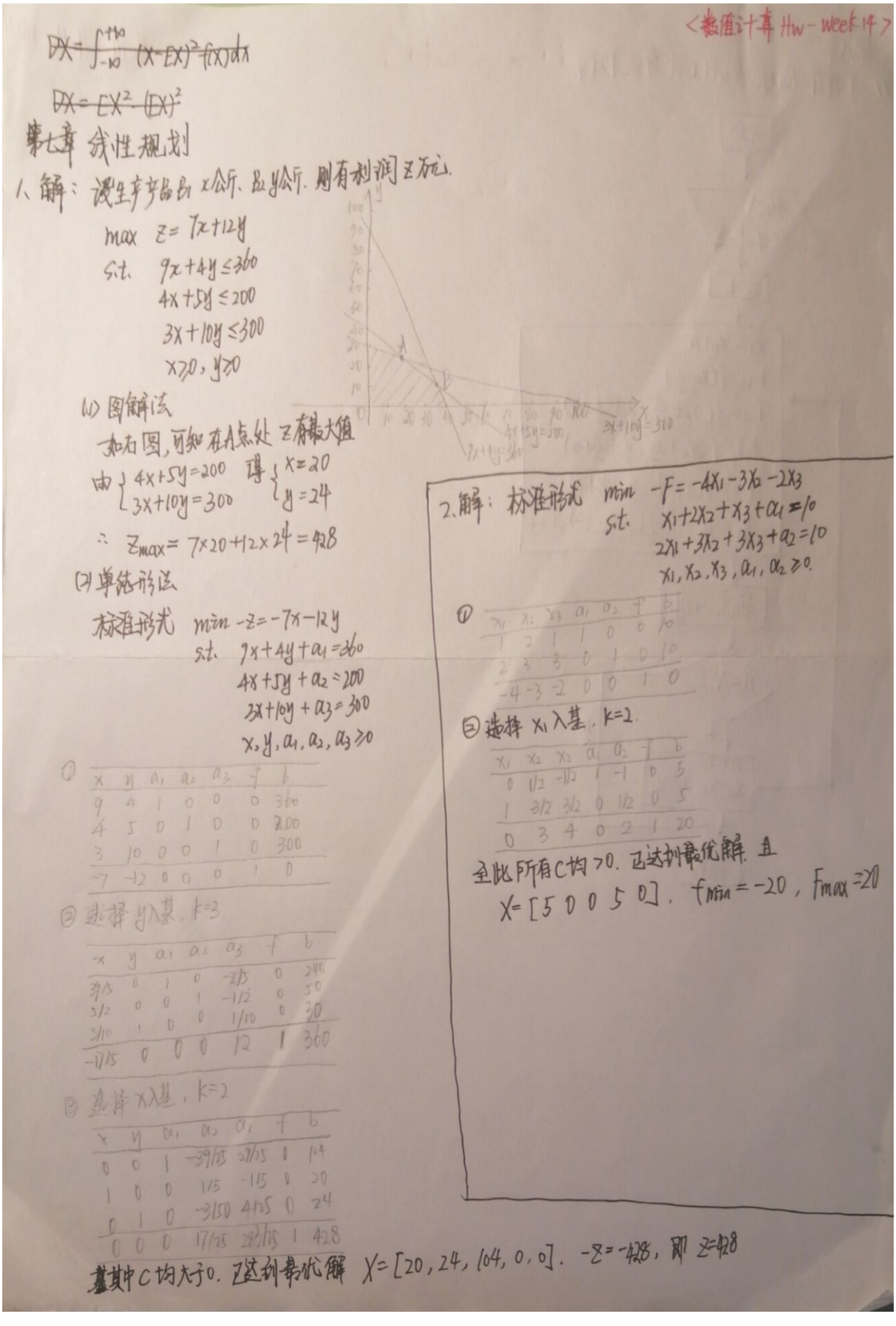
20195633 李燕琴

第七章 动态规划



〈数值计算 Hw-week14〉

~~$DX=\int_{-\infty}^{+\infty}(X-EX)^2 f(x)dx$~~

~~$DX=EX^2-(EX)^2$~~

第七章 线性规划

1、解：设生产产品B_1 x公斤、B_2 y公斤. 则有利润z万元.

$$\max\ z=7x+12y$$

$$\text{s.t.}\quad 9x+4y\le 360$$
$$4x+5y\le 200$$
$$3x+10y\le 300$$
$$x\ge 0,\ y\ge 0$$

y
100
90
80
70
60
50
40
30
20
10
0
10
20
30
40
50
60
70
80
90
100
x
A
B
4x+5y=200
3x+10y=300
9x+4y=360

(1) 图解法

如右图，可知在A点处 z有最大值

由$\begin{cases}4x+5y=200\\3x+10y=300\end{cases}$ 得$\begin{cases}x=20\\y=24\end{cases}$

$\therefore z_{max}=7\times 20+12\times 24=428$

(2) 单纯形法

标准形式 $\min\ -z=-7x-12y$

$$\text{s.t.}\quad 9x+4y+a_1=360$$
$$4x+5y+a_2=200$$
$$3x+10y+a_3=300$$
$$x,y,a_1,a_2,a_3\ge 0$$

①

x	y	a_1	a_2	a_3	f	b
9	4	1	0	0	0	360
4	5	0	1	0	0	200
3	10	0	0	1	0	300
-7	-12	0	0	0	1	0

② 选择y入基，$k=3$.

x	y	a_1	a_2	a_3	f	b
37/15	0	1	0	-2/15	0	240
5/2	0	0	1	-1/2	0	50
3/10	1	0	0	1/10	0	30
-17/5	0	0	0	12	1	360

③ 选择x入基，$k=2$

x	y	a_1	a_2	a_3	f	b
0	0	1	-39/25	28/25	1	104
1	0	0	1/5	-1/5	0	20
0	1	0	-3/50	4/25	0	24
0	0	0	17/25	283/75	1	428

~~基~~其中c均大于0. 已达到最优解 $X=[20,24,104,0,0]$. $-z=-428$，即 $z=428$

2、解：标准形式 $\min\ -F=-4x_1-3x_2-2x_3$

$$\text{s.t.}\quad x_1+2x_2+x_3+a_1=10$$
$$2x_1+3x_2+3x_3+a_2=10$$
$$x_1,x_2,x_3,a_1,a_2\ge 0.$$

①

x_1	x_2	x_3	a_1	a_2	f	b
1	2	1	1	0	0	10
2	3	3	0	1	0	10
-4	-3	-2	0	0	1	0

② 选择x_1入基，$k=2$.

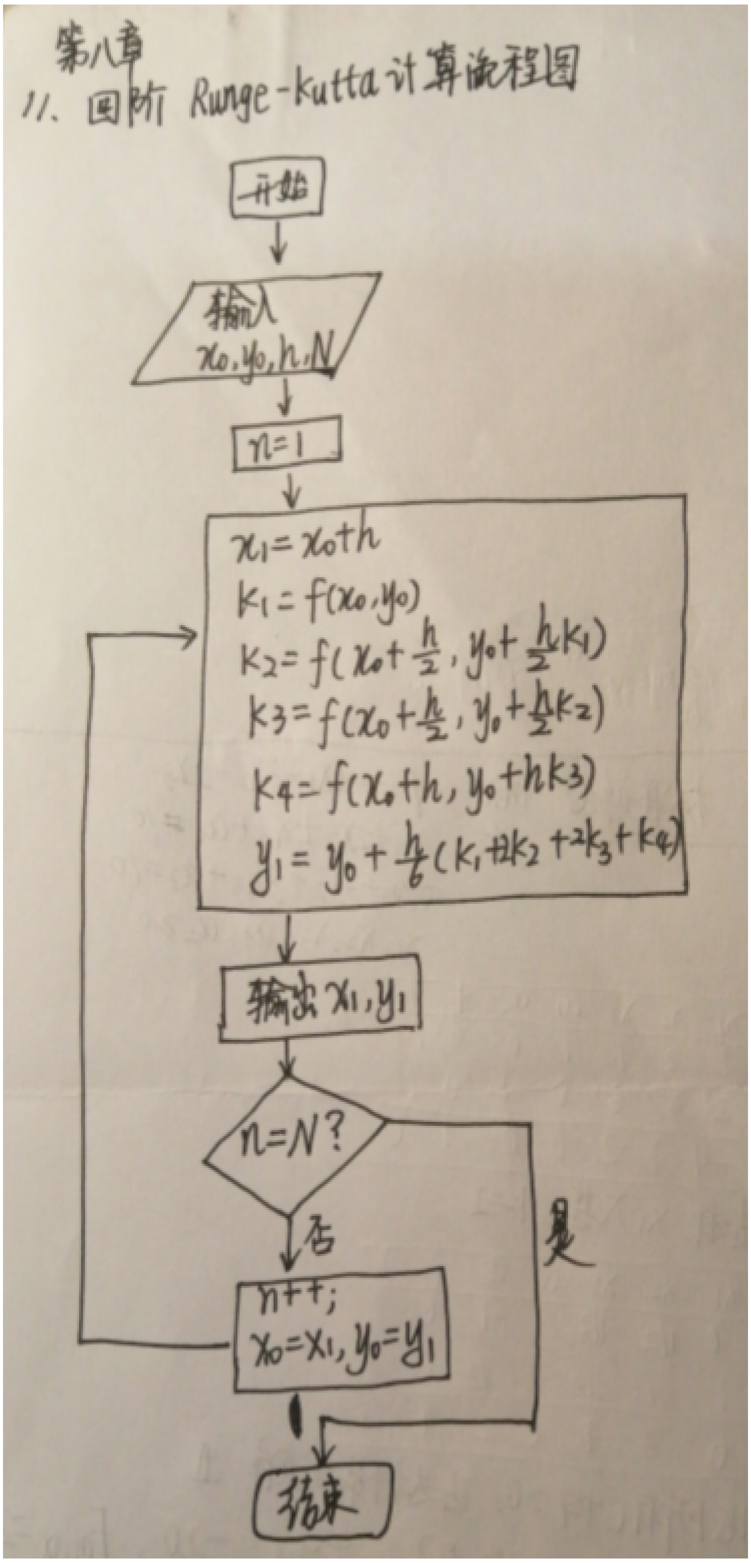
x_1	x_2	x_3	a_1	a_2	f	b
0	1/2	-1/2	1	-1	0	5
1	3/2	3/2	0	1/2	0	5
0	3	4	0	2	1	20

至此所有c均>0. 已达到最优解. 且

$X=[5\ 0\ 0\ 5\ 0]$. $f_{min}=-20$，$F_{max}=20$

第八章 常微分房方程数值解法

11



12

代码

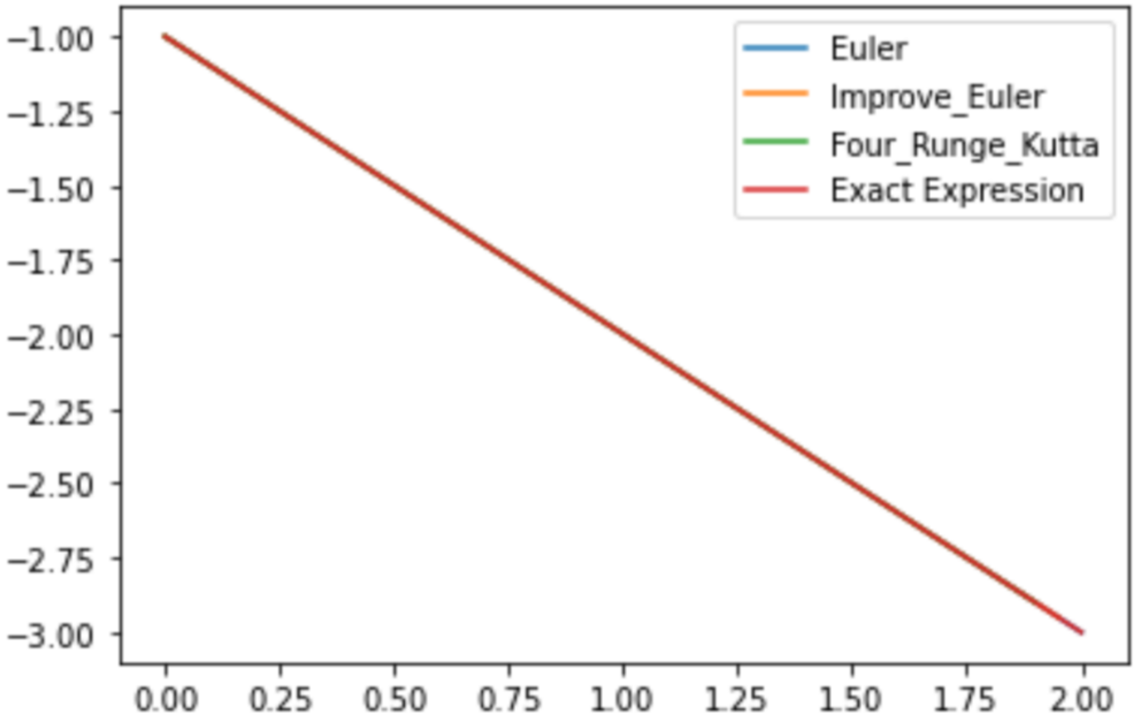
```
#-*-coding:utf-8-*-
import numpy as np
import matplotlib.pyplot as plt

# def fun(x):
#     ''' 第一问 '''
```

```python
#     return -1-x
# def diff(x,y):
#     ''' 第一问 '''
#     if 0<=x<=2:
#         return x+y
#     else:
#         print("ERROR: x的数值不在[0,2]内")
#         exit(-1)

def fun(x):
    ''' 第二问 '''
    return np.exp(-2*x)+x**2
def diff(x,y):
    ''' 第二问 '''
    if 0<=x<=1:
        return -2*y+2*(x**2)+2*x
    else:
        print("ERROR: x的数值不在[0,2]内")
        exit(-1)

def Euler(x0,y0,h,x_ub):
    n = 1
    x,y = [],[]
    x.append(x0), y.append(y0)
    while True:
        x1 = x0 + h
        if x1 > x_ub:
            return x,y
        y1 = y0 + h * diff(x0,y0)
        x.append(x1), y.append(y1)
        x0,y0 = x1,y1

def Improve_Euler(x0,y0,h,x_ub):
    n = 1
    x,y = [],[]
    x.append(x0), y.append(y0)
    while True:
        x1 = x0 + h
        if x1 > x_ub:
            return x,y
        yp = y0 + h * diff(x0,y0)
        yc = y0 + h * diff(x1,yp)
        y1 = (yp+yc)/2
        x.append(x1), y.append(y1)
        x0,y0 = x1,y1

def Four_Runge_Kutta(x0,y0,h,x_ub):
    n = 1
    x,y = [],[]
    x.append(x0), y.append(y0)
    while True:
        x1 = x0 + h
        if x1 > x_ub:
            return x,y
        k1 = diff(x0,y0)
        k2 = diff(x0+h/2,y0+h*k1/2)
        k3 = diff(x0+h/2,y0+h*k2/2)
        k4 = diff(x0+h,y0+h*k3)
        y1 = y0 + h*(k1+2*k2+2*k3+k4)/6
        x.append(x1), y.append(y1)
        x0,y0 = x1,y1

if __name__ == '__main__':
    x0,y0 = 0,-1
#     x_ub = 2 # 第一问
    x_ub = 1 # 第二问
    x1,y1 = Euler(x0,y0,h=0.025,x_ub=x_ub)
    plt.plot(x1,y1,'-')
    x2,y2 = Improve_Euler(x0,y0,h=0.05,x_ub=x_ub)
    plt.plot(x2,y2,'-')
    x3,y3 = Four_Runge_Kutta(x0,y0,h=0.1,x_ub=x_ub)
    plt.plot(x3,y3,'-')
    x4 = np.linspace(0,x_ub,50)
    y4 = fun(x4)
    plt.plot(x4,y4,'-')
    plt.legend(["Euler","Improve_Euler","Four_Runge_Kutta","Exact Expression"])
```

结果

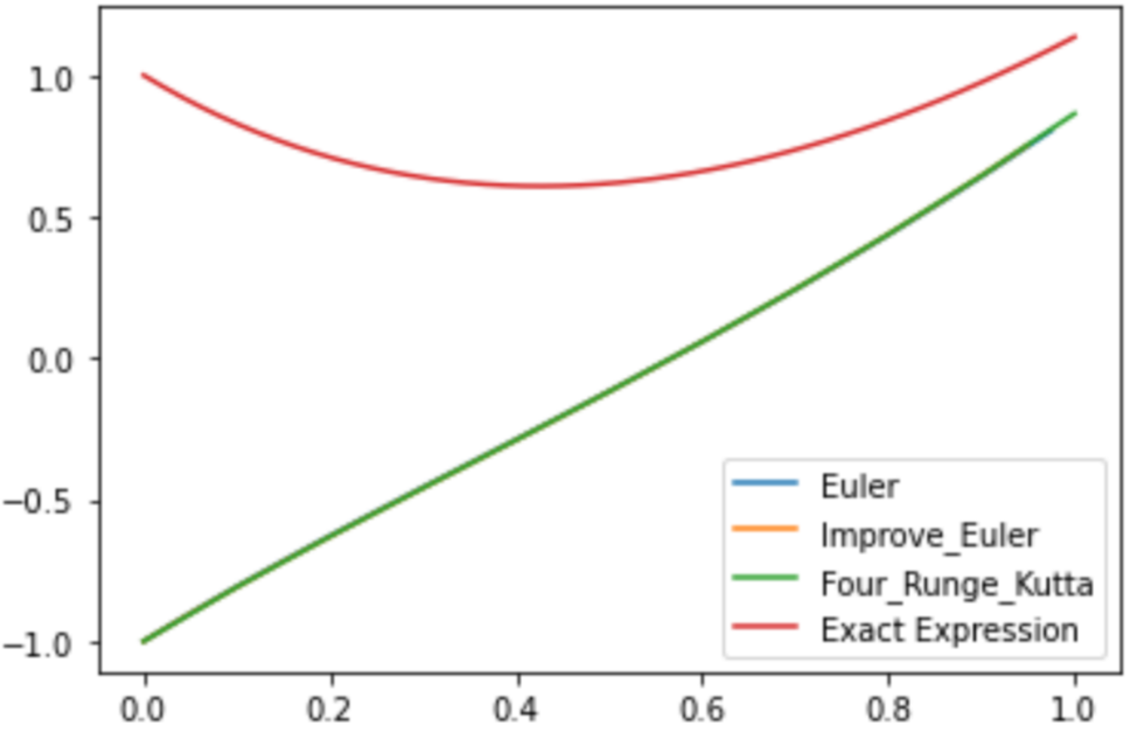
(1)

结果在$x \in [0,2]$范围内吻合度较高。



(2)

结果在$x \in [0,1]$范围内吻合度存在较大偏差。


1.0
0.5
0.0
−0.5
−1.0
0.0
0.2
0.4
0.6
0.8
1.0
Euler
Improve_Euler
Four_Runge_Kutta
Exact Expression