数值分析实验三

肖涵薄 31360164

2019 年 6 月 3 日

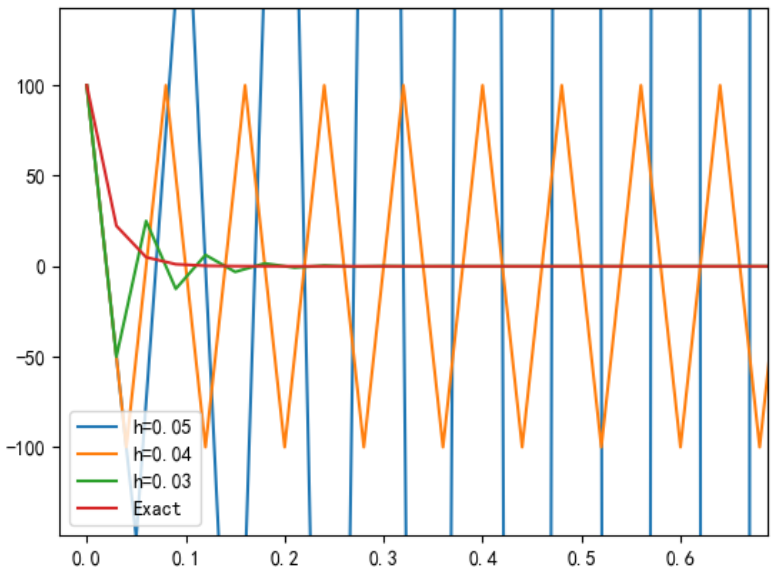
一

1

显式 Euler 公式为

$$y_{n+1} = y_n + hf(x_n, y_n)$$

其中 $f(x_n, y_n) = -50y$ 该微分方程的解析解为 $y(x) = 100e^{-50x}$.
不同步长的解与精确解的关系如下图所示



可以看到 h=0.03 时的解接近精确解, h=0.04 时在精确解上下波动, 不收敛也不继续发散, 而当 h=0.05, 出现数值不稳定, 可以看到 h 变动极大且 $|h|$ 趋于无穷大.

实际上, 利用 Euler 方法的稳定性条件

$$|1+\lambda h| \leq 1, \lambda = -50$$

可以得到方法稳定的条件为 $0 \leq h \leq 0.04$. 与实验相符.

2

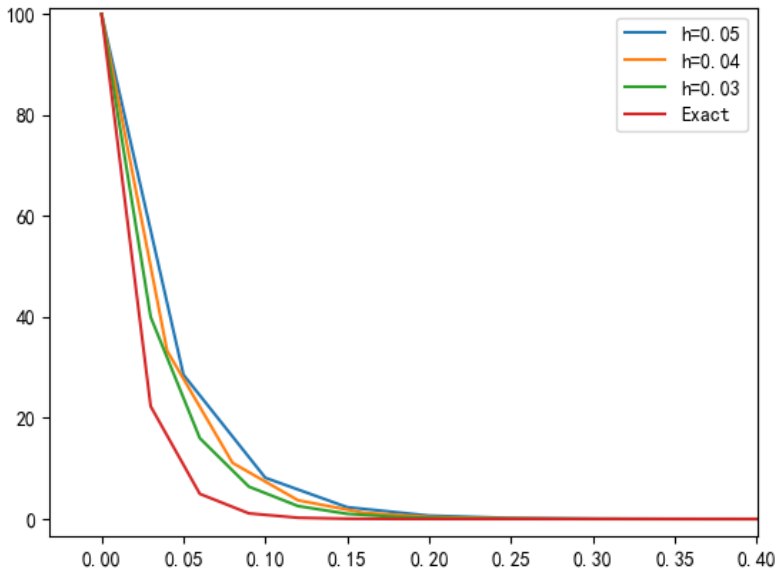
隐式 Euler 公式为

$$y_{n+1} = y_n + hf\left(x_{n+1}, y_{n+1}\right)$$

在此题中可显式地化为

$$y_{n+1} = \frac{1}{1+50h} y_n$$

不同步长的解与精确解的关系如下图所示



此时所有 h 均不会出现数值不稳定.

对于隐式 Euler 公式, 稳定性条件为

$$\left|\frac{1}{1-\lambda h}\right| \leq 1$$

当 $\lambda < 0$, 对一切 h 均稳定, 且 h 越小, 与精确解符合得越好. 这与实验相符.

二

方程组为

$$\begin{cases} y'(t) = -2x(t) \\ x'(t) = \frac{9}{2}y(t) \\ x(0) = 3, y(0) = 2 \end{cases}$$

定义 $r(t) = \begin{pmatrix} x \\ y \end{pmatrix}$, 则

$$r'(t) = Ar(t) = \begin{pmatrix} 0 & \frac{9}{2} \\ -2 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}, \quad r(0) = \begin{pmatrix} 3 \\ 2 \end{pmatrix}$$

梯形公式为

$$r_{n+1} = r_n + \frac{h}{2}\left[r'\left(t_n\right) + r'\left(t_{n+1}\right)\right] \implies r(t_{n+1}) = (E - \frac{h}{2}A)^{-1}(E + \frac{h}{2}A)r(t_n)$$

显式 Euler 法为

$$r(t_{n+1}) = r(t_n) + hr'(t_n) \implies r(t_{n+1}) = (E + hA)r(t_n)$$

隐式 Euler 法为

$$r_{n+1} = r_n + hAr_{n+1}$$

在此题中可显式地化为

$$r_{n+1} = (E - hA)^{-1}r_n$$

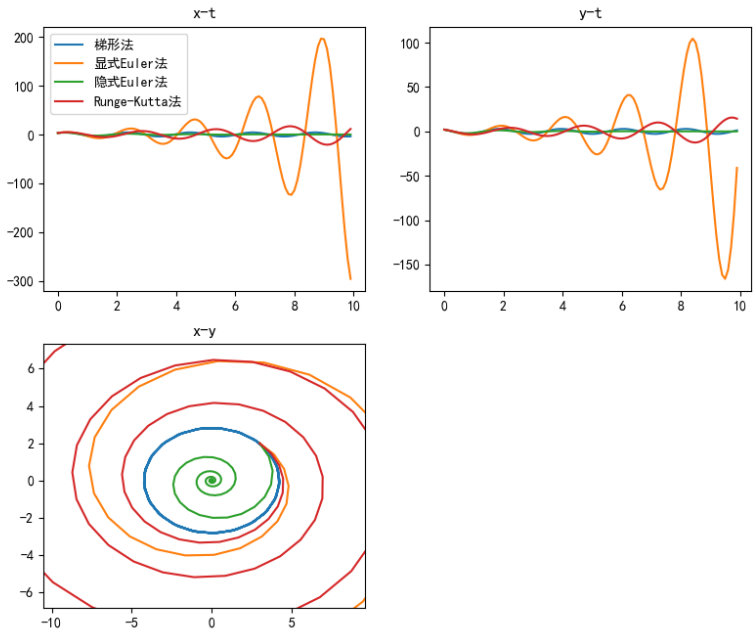
按照 Runge-Kutta 法,

$$r_{n+1} = r_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

其中

$$\begin{cases} k_1 = f\left(t_n, r_n\right) \\ k_2 = f\left(t_n + \frac{h}{2}, r_n + \frac{h}{2}k_1\right) \\ k_3 = f\left(t_n + \frac{h}{2}, r_n + \frac{h}{2}k_2\right) \\ k_4 = f\left(t_n + h, r_n + hk_3\right) \end{cases}$$

取步长为 0.1, 绘制 $t \in [0, 10]$ 的图像, 不同方法的解如下图所示:



三个图分别为 $x-t, y-t, x-y$ 的关系图. 可以非常清楚地看到仅有梯形法保持了相平面轨迹不变. 显式 Euler 法和 Runge-Kutta 法轨迹逐渐远离原点, 隐式 Euler 法轨迹逐渐收缩到原点.

原因解释

在此题中, 微分方程的解为

$$\frac{x^2}{9} + \frac{y^2}{4} = 2$$

做变量代换 $r' = \begin{pmatrix} 1/3 & 0 \\ 0 & 1/2 \end{pmatrix} r = Qr$. 则方程曲线变为圆. 相平面稳定要求半径不变, 即 $r_n'^2 = 2$.
设 $r_{n+1} = Br_n$. 变量代换后矩阵 B 也要做变换, $B' = QBQ^{-1}$. $r_n'^2 = 2$ 要求 $\|B'\| = 1$.
对于梯形公式, 代入 $r(t_{n+1}) = [(E - \frac{h}{2}A)^{-1}(E + \frac{h}{2}A)]r(t_n) = Br(t_n)$. 当 $h = 0.1$,

$$B' = \begin{pmatrix} 0.9560 & 0.2934 \\ -0.2934 & 0.9560 \end{pmatrix}, \quad \|B'\|_2 = 1$$

因此相平面稳定.

对于显式 Euler 法, $B = E + hA$, 当 $h = 0.1$,

$$B' = \begin{pmatrix} 1 & 0.3 \\ -0.3 & 1 \end{pmatrix}, \quad \|B'\|_2 = 1.044 > 1$$

因此相平面不稳定且距离原点越来越远.

对于隐式 Euler 法, $B = (E - hA)^{-1}$, 当 $h = 0.1$,

$$B' = \begin{pmatrix} 0.9174 & 0.2752 \\ -0.2752 & 0.9174 \end{pmatrix}, \quad \|B'\|_2 = 0.9578 < 1$$

因此相平面不稳定. 且距离原点越来越近.

对于 Runge-Kutta 法, 较难用 $r_{n+1} = Br_n$ 形式表达, 但可以验证其并不满足 $\|B'\|_2 = 1$:

在本实验中 $|r_0'| = \sqrt{2}$, 而计算可得 $r_1 = [3.7156625, 1.46655833]^T$,

$$r_1' = Qr_1 = \begin{pmatrix} 1.2386 \\ 0.7333 \end{pmatrix}, \quad |r_1'| = 1.4393 \neq |r_0'|$$

因此不具有相平面稳定性.

三

待解的方程为

$$\begin{cases} \frac{\mathrm{d}^2\theta(t)}{\mathrm{d}t^2} + \sin\theta = 0, \quad 0 < t \leq 10 \\ \theta(0) = \frac{\pi}{3} \\ \left.\frac{\mathrm{d}\theta(t)}{\mathrm{d}t}\right|_{t=0} = -\frac{1}{2} \end{cases}$$

设 $x = \theta, y = \theta'$, 则 $x' = y, y' = -\sin x$.

隐式 Euler 公式为

$$x_{n+1} = x_n + hy_{n+1}$$

$$y_{n+1} = y_n - h\sin x_{n+1}$$

将 x_n, y_n 看做常数, 可写出迭代格式

$$x_{n+1} = x_n + hy_n - h^2\sin x_{n+1}$$

$$y_{n+1} = y_n - h\sin(x_n + hy_{n+1})$$

当 $x_{n+1}^{(k)} - x_{n+1}^{(k-1)} < \varepsilon$, $y_{n+1}^{(k)} - y_{n+1}^{(k-1)} < \varepsilon$, 可解出 $x_{n+1} = x_{n+1}^{(k)}, y_{n+1} = y_{n+1}^{(k)}$.

梯形公式为

$$r_{n+1} = r_n + \frac{h}{2}\left[r'\left(t_n\right) + r'\left(t_{n+1}\right)\right]$$

可以解得

$$x_{n+1} = x_n + \frac{h}{2}\left(y_n + y_{n+1}\right)$$

$$y_{n+1} = y_n + \frac{h}{2}\left(-\sin x_n - \sin x_{n+1}\right)$$

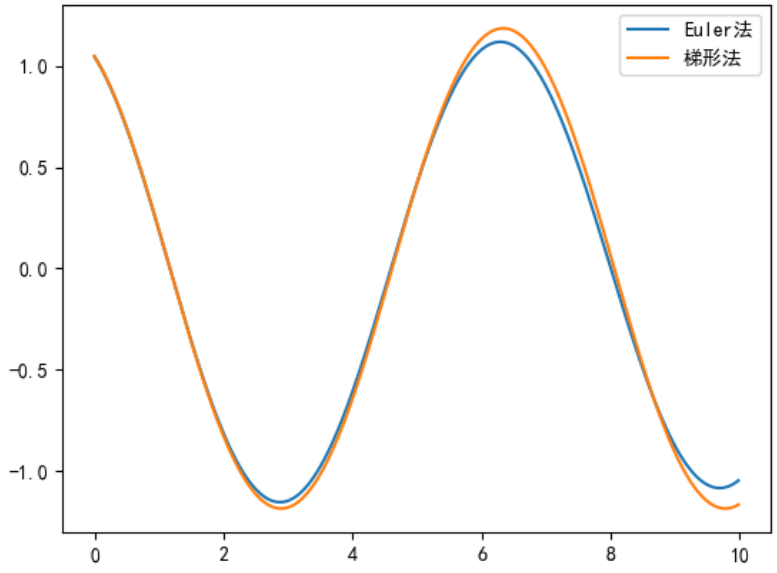
将 x_n, y_n 看做常数, 可写出迭代格式

$$x_{n+1} = x_n + \frac{h}{2}\left(2y_n + \frac{h}{2}\left(-\sin x_n - \sin x_{n+1}\right)\right)$$

$$y_{n+1} = y_n + \frac{h}{2}\left(-\sin x_n - \sin\left(x_n + \frac{h}{2}\left(y_n + y_{n+1}\right)\right)\right)$$

当 $x_{n+1}^{(k)} - x_{n+1}^{(k-1)} < \varepsilon$, $y_{n+1}^{(k)} - y_{n+1}^{(k-1)} < \varepsilon$, 可解出 $x_{n+1} = x_{n+1}^{(k)}, y_{n+1} = y_{n+1}^{(k)}$.

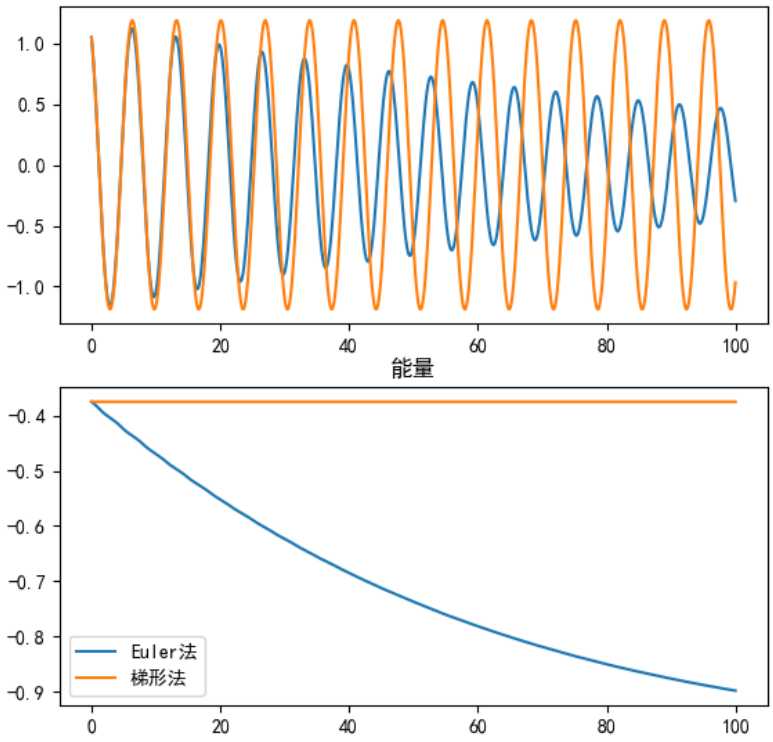
绘制 $t \in [0, 10]$ 的图像, $t - \theta$ 图如下图所示:



本题中单摆的能量为

$$\varepsilon = \frac{1}{2}\left(\frac{\mathrm{d}\theta}{\mathrm{d}t}\right)^2 + \cos\theta = \frac{1}{2}y^2 + \cos x$$

两种方法计算下能量如下图所示



可以看到梯形公式满足能量守恒, 但隐式 Euler 法能量逐渐降低. 这与上一题的计算相符.

Code(Python)

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import fsolve
plt.rcParams['font.sans-serif']=['SimHei'] #用来正常显示中文标签
plt.rcParams['axes.unicode_minus']=False #用来正常显示负号
############ 第一题 1 ##################
def f(y): return -50*y
for h in [0.05,0.04,0.03]:
    x,y = np.arange(0,1,h),np.arange(0,1,h)
    y[0]=100
    for i in range(0,x.size-1): y[i+1] = y[i] +h*f(y[i])
    plt.plot(x,y)
plt.plot(x,100*np.exp(-50*x))
plt.legend(labels = ['h=0.05', 'h=0.04', 'h=0.03', 'Exact'], loc = 'best')
plt.show()
############ 第一题 2 ##################
for h in [0.05,0.04,0.03]:
    x,y = np.arange(0,1,h),np.arange(0,1,h)
    y[0]=100
    for i in range(0,x.size-1): y[i+1] = y[i]/(1+50*h)
    plt.plot(x,y)
plt.plot(x,100*np.exp(-50*x))
plt.legend(labels = ['h=0.05', 'h=0.04', 'h=0.03', 'Exact'], loc = 'best')
plt.show()
############ 第二题  ##################
h=0.1
A = np.matrix([[0,9/2],[-2,0]])
def main(B):
    r = list(np.arange(0,10,h))
    r[0] = np.matrix([[3],[2]])
    for i in range(0,len(r)-1): r[i+1] = B*r[i]
```

```
    r = np.array(r).T
    plt.subplot(221)
    plt.plot(np.arange(0,10,h),r[0][0])
    plt.subplot(222)
    plt.plot(np.arange(0,10,h),r[0][1])
    plt.subplot(223)
    plt.plot(r[0][0],r[0][1])
main((np.eye(2)-A*h/2)**(-1)*(np.eye(2)+A*h/2)) #梯形法
main(np.eye(2)+h*A) #显式Euler法
main((np.eye(2)-h*A)**(-1)) #隐式Euler法
def RK4(rn): #Runge-Kutta法
    k1 = A*rn
    k2 = A*(rn+h*k1/2)
    k3 = A*(rn+h*k2/2)
    k4 = rn + h*k3
    return rn+h*(k1+2*k2+2*k3+k4)/6
r = list(np.arange(0,10,h))
r[0] = np.matrix([[3],[2]])
for i in range(0,len(r)-1): r[i+1] = RK4(r[i])
r = np.array(r).T
plt.subplot(221)
plt.title('x-t')
plt.plot(np.arange(0,10,h),r[0][0])
plt.legend(labels = ['梯形法','显式Euler法','隐式Euler法','Runge-Kutta法'], loc = 'best')
plt.subplot(222)
plt.title('y-t')
plt.plot(np.arange(0,10,h),r[0][1])
plt.subplot(223)
plt.title('x-y')
plt.plot(r[0][0],r[0][1])
plt.show()
############ 第三题  ##################
h,ep,t = 0.02,1.e-10,100
def euler(xn,yn):# Euler 法
```

```
    global h,ep
    x1,y1,x2,y2 = xn,yn,-1,-1
    while abs(x2-x1)>ep or abs(y2-y1)>ep:
        x1,y1 = x2,y2
        x2,y2 = xn + h*yn-h*h*np.sin(x1),yn -h*np.sin(xn+h*y1)
    return (x2,y2)
def ech(xn,yn):# 梯形法
    global h,ep
    x1,y1,x2,y2 = xn,yn,-1,-1
    while abs(x2-x1)>ep or abs(y2-y1)>ep:
        x1,y1 = x2,y2
        x2,y2 = xn + 0.5*h*(2*yn-0.5*h*(np.sin(xn)+np.sin(x1))),yn -
            0.5*h*(np.sin(xn)+np.sin(xn+0.5*h*(yn+y1)))
    return (x2,y2)
for func in [euler, ech]:
    r = list(np.arange(0,t,h))
    r[0] = [np.pi/3,-0.5]
    for i in range(0,len(r)-1): r[i+1] = func(r[i][0], r[i][1])
    r = np.array(r).T
    plt.subplot(211)
    plt.plot(np.arange(0,t,h),r[0])
    plt.subplot(212)
    plt.title('能量')
    plt.plot(np.arange(0,t,h),0.5*r[1]*r[1]-np.cos(r[0]))
plt.legend(labels = ['Euler法','梯形法'], loc = 'best')
plt.show()
```