龙格库塔方法解洛伦茨方程组

公式推导

洛伦茨方程组$\begin{cases} \frac{dx}{dt} = \sigma(y-x) \\ \frac{dy}{dt} = -xz + rx - y \\ \frac{dz}{dt} = xy - bz \end{cases}$

令$\begin{cases} f(x,y,z,t) = \sigma(y-x) \\ g(x,y,z,t) = -xz + rx - y \\ h(x,y,z,t) = xy - bz \end{cases}$

采用龙格库塔方法$\begin{cases} x_{n+1} = x_n + \frac{\Delta t}{6}(k1 + 2k2 + 2k3 + k4) \\ y_{n+1} = y_n + \frac{\Delta t}{6}(l1 + 2l2 + 2l3 + l4) \\ z_{n+1} = z_n + \frac{\Delta t}{6}(m1 + 2m2 + 2m3 + m4) \end{cases}$

$\begin{cases} k1 = f(x_n, y_n, z_n, t_n) = \sigma(y_n - x_n) \\ l1 = g(x_n, y_n, z_n, t_n) = -x_n z_n + rx_n - y_n \\ m1 = h(x_n, y_n, z_n, t_n) = x_n y_n - bz_n \end{cases}$

计算中间变量$\begin{cases} x_{n+1/2} = x_n + k1\Delta t/2 \\ y_{n+1/2} = y_n + l1\Delta t/2 \\ z_{n+1/2} = z_n + m1\Delta t/2 \\ t_{n+1/2} = t_n + \Delta t/2 \end{cases}$

$\begin{cases} k2 = f(x_{n+1/2}, y_{n+1/2}, z_{n+1/2}, t_{n+1/2}) \\ l2 = g(x_{n+1/2}, y_{n+1/2}, z_{n+1/2}, t_{n+1/2}) \\ m2 = h(x_{n+1/2}, y_{n+1/2}, z_{n+1/2}, t_{n+1/2}) \end{cases}$

更新中间变量$\begin{cases} x_{n+1/2} = x_n + k2\Delta t/2 \\ y_{n+1/2} = y_n + l2\Delta t/2 \\ z_{n+1/2} = z_n + m2\Delta t/2 \\ t_{n+1/2} = t_n + \Delta t/2 \end{cases}$

$\begin{cases} k3 = f(x_{n+1/2}, y_{n+1/2}, z_{n+1/2}, t_{n+1/2}) \\ l3 = g(x_{n+1/2}, y_{n+1/2}, z_{n+1/2}, t_{n+1/2}) \\ m3 = h(x_{n+1/2}, y_{n+1/2}, z_{n+1/2}, t_{n+1/2}) \end{cases}$

更新另一中间变量$\begin{cases} x_{n+1} = x_n + k3\Delta t \\ y_{n+1} = y_n + l3\Delta t \\ z_{n+1} = z_n + m3\Delta t \\ t_{n+1} = t_n + \Delta t \end{cases}$（注意此处$x_{n+1}$并非龙格库塔方法中的$x_{n+1}$，仅仅是一中间变量）

$\begin{cases} k4 = f(x_{n+1}, y_{n+1}, z_{n+1}, t_{n+1}) \\ l3 = g(x_{n+1}, y_{n+1}, z_{n+1}, t_{n+1}) \\ m3 = h(x_{n+1}, y_{n+1}, z_{n+1}, t_{n+1}) \end{cases}$

代码实现

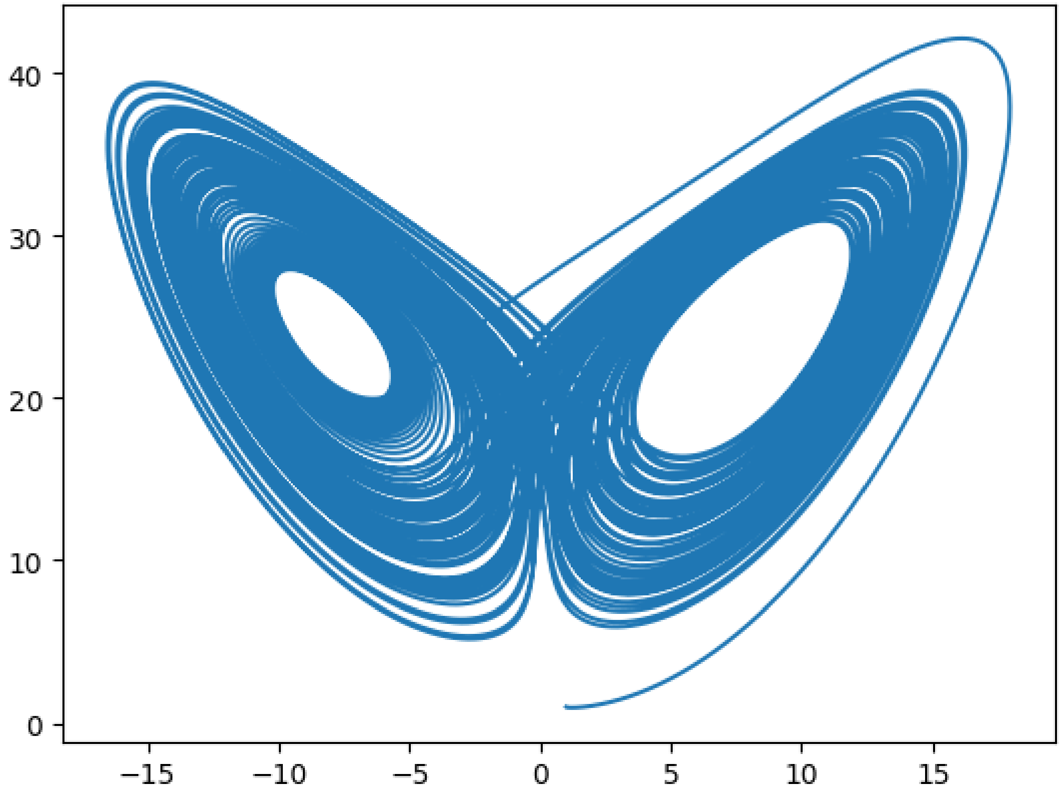
```
import numpy as np
import matplotlib.pyplot as plt
from numba import jit
from mpl_toolkits.mplot3d import Axes3D
sigma=10                                ###参数取值
b=8/3
r=25
@jit
def f(x1,y1,z1,t1):                     ###定义第一个方程
    return sigma*(y1-x1)
@jit
def g(x2,y2,z2,t2):                     ###定义第二个方程
    return -x2*z2+r*x2-y2
@jit
def h(x3,y3,z3,t3):                     ###定义第三个方程
    return x3*y3-b*z3
x_list=np.array([1])                    ###定义初值
y_list=np.array([1])
z_list=np.array([1])
t_list=np.array([0])
delta_t=0.001                           ###定义步长
for i in range(100000):
    t_n=t_list[-1]                      ###将上一循环结果存入变量
    x_n=x_list[-1]
    y_n=y_list[-1]
    z_n=z_list[-1]
    k1=f(x_n,y_n,z_n,t_n)               ###第一次计算斜率
    l1=g(x_n,y_n,z_n,t_n)
    m1=h(x_n,y_n,z_n,t_n)
    x_nplusahalf=x_n+k1*delta_t/2       ###第一次计算中间变量
    y_nplusahalf=y_n+l1*delta_t/2
    z_nplusahalf=z_n+m1*delta_t/2
    t_nplusahalf=t_n+delta_t/2
    k2=f(x_nplusahalf,y_nplusahalf,z_nplusahalf,t_nplusahalf)   ###第二次计算斜率
    l2=g(x_nplusahalf,y_nplusahalf,z_nplusahalf,t_nplusahalf)
    m2=h(x_nplusahalf,y_nplusahalf,z_nplusahalf,t_nplusahalf)
    x_nplusahalf=x_n+k2*delta_t/2       ###第二次计算中间变量
    y_nplusahalf=y_n+l2*delta_t/2
    z_nplusahalf=z_n+m2*delta_t/2
    t_nplusahalf=t_n+delta_t/2
    k3=f(x_nplusahalf,y_nplusahalf,z_nplusahalf,t_nplusahalf)   ###第三次计算斜率
    l3=g(x_nplusahalf,y_nplusahalf,z_nplusahalf,t_nplusahalf)
    m3=h(x_nplusahalf,y_nplusahalf,z_nplusahalf,t_nplusahalf)
    x_nplusone=x_n+k3*delta_t           ###第三次计算中间变量
    y_nplusone=y_n+l3*delta_t
    z_nplusone=z_n+m3*delta_t
    t_nplusone=t_n+delta_t
    k4=f(x_nplusone,y_nplusone,z_nplusone,t_nplusone)           ###第四次计算斜率
    l4=g(x_nplusone,y_nplusone,z_nplusone,t_nplusone)
    m4=h(x_nplusone,y_nplusone,z_nplusone,t_nplusone)
    new_x=x_n+(k1+2*k2+2*k3+k4)*delta_t/6    ###计算新一轮结果
    new_y=y_n+(l1+2*l2+2*l3+l4)*delta_t/6

    new_z=z_n+(m1+2*m2+2*m3+m4)*delta_t/6
```

```
    new_t=t_n+delta_t
    x_list=np.append(x_list,new_x)          ###将新一轮结果存入列表
    y_list=np.append(y_list,new_y)
    z_list=np.append(z_list,new_z)
    t_list=np.append(t_list,new_t)
fig=plt.figure()                            ###画图
ax=Axes3D(fig)
ax.plot3D(x_list,y_list,z_list,linewidth=0.5)
plt.show()
```

画图

1.当$\sigma=10$; b=8/3; r=25时，画出z-x平面中的轨迹



2.当$\sigma=10$; b=8/3时，改变r值，计算z-x平面中的轨迹

r从20变到30的动图：见压缩包中gif文件或点击如下链接
动图