常微分方程

常微分方程的求解

1.符号解

通过sympy库的dsolve()函数求解。

例：求解二阶线性微分方程：

$$y'' - 2y' + y = e^x, \quad y(0) = 1,\ y'(0) = -1$$

解法：

```
import sympy as sp

sp.var('x')
y = sp.Function('y')
eq = y(x).diff(x, 2) - 2 * y(x).diff(x) + y(x) - sp.exp(x)
con = {y(0): 1, y(x).diff(x).subs(x, 0): -1}
s = sp.dsolve(eq, ics=con)
# s = sp.simplify(s) 可选用进行化简
print(s)
```

例：求

$$\begin{cases} \dfrac{\mathrm{d}x}{\mathrm{d}t} = Ax, \\ x(0) = [1,1,1]^T \end{cases}$$

的解，其中

$$x(t) = [x_1(t),\ x_2(t),\ x_3(t)]^T,\ A = \begin{bmatrix} 3 & -1 & 1 \\ 2 & 0 & -1 \\ 1 & -1 & 2 \end{bmatrix}$$

解法：

```
import sympy as sp
sp.var('x1:4', cls=sp.Function)
x = sp.Matrix([x1(t), x2(t), x3(t)])
A = sp.Matrix([[3, -1, 1], [2, 0, 1], [1, -1, 2]])
```

```
eq = x.diff(t) - A @ x
s = sp.dsolve(eq, ics={x1(0): 1, x2(0): 1, x3(0): 1})
print(s)
```

2.数值解

Python只能求一阶常微分方程（组）的数值解，高阶常微分方程需要化为一阶方程组来求解，
使用scipy.intedrate模块的odeint函数求解
调用方式为

$$sol = odeint(func,\ y0,\ t)$$

其中，func 为定义微分方程的函数，y0 为初始条件序列，t 为自变量取值序列，返回值sol为对应于序列 t 中元素的数值解，若微分方程组中有 n 个函数，则返回值 sol 为 n 列的矩阵，第 i 列对应第 i 个函数的数组解.

例：求解二阶线性微分方程在[-1, 1]的数值解：

$$y'' - 2y' + y = e^x, \quad y(0) = 1,\ y'(0) = -1$$

解法：
方程转化为

$$\begin{cases} y_1' = y_2, & y_1(0) = 1, \\ y_2' = 2y_2 - y_1 + e^x, & y_2(0) = -1. \end{cases}$$

```
from scipy.integrate import odeint
import numpy as np
import sympy as sp
import pylab as plt

df = lambda f, x: [f[1], 2 * f[1] - f[0] + np.exp(x)] # 描述一个二阶常微分方程
x1 = np.linspace(0, 1, 51) # 生成包含51个等间距点的数组，范围 0~1
s1 = odeint(df, [1, -1], x1) # 数值积分，[1,-1]表示初始条件，即y(0)=1.y'(0)=-1
x2 = np.linspace(0, -1, 51)
s2 = odeint(df, [1, -1], x2)

x = sp.var('x')
y = sp.Function('y')
eq = y(x).diff(x, 2) - 2 * y(x).diff(x) + y(x) - sp.exp(x)
con = {y(0): 1, y(x).diff(x).subs(x, 0): -1}
s = sp.dsolve(eq, ics=con)
```

```
sx = sp.lambdify(x, s.args[1], 'numpy') # 将符号解 s 转化为数值计算函数 sx
x3 = np.linspace(-1, 1, 101)

# 绘图
plt.rc('font', family='SimHei')
plt.rc('axes', unicode_minus=False)
plt.rc('font', size=16)
plt.plot(x2, s2[:, 0], 'P-', x1, s1[:, 0], '^-')
plt.plot(x3, sx(x3), 'k-')
plt.legend(['[-1, 0]上的数值解', '[0, 1]上的数值解', '符号解'])
plt.show()
```

常微分方程建模实例

1.Malthus模型

设 $x(t)$ 为 t 时刻的人口数，人口增长率 $r = const$，则有

$$\begin{cases} \frac{dx}{dt} = rx, \\ x(0) = x_0. \end{cases}$$

其解为

$$x(t) = x_0 e^{rt}$$

2.Logistic模型

设 $x(t)$ 为 t 时刻的人口数，人口增长率 $r(x) = r - sx$ 为 x 的线性函数，且 $x = x_m$ 时，$r(x_m) = 0$，
故可得

$$r(x) = r(1 - \frac{x}{x_m})$$

其中 $r = const$ 为人口较少时的增长率，则有

$$\begin{cases} \frac{\mathrm{d}x}{\mathrm{d}t} = r(1 - \frac{x}{x_m}), \\ x(t_0) = x_0. \end{cases}$$

其解为

$$x(t) = \frac{x_m}{1 + (\frac{x_m}{x_0} - 1)e^{-r(t - t_0)}}$$

3.种群竞争模型

设 t 时刻两个种群数量分别为 $x_1(t)$ 和 $x_2(t)$，每个种群的增长都受到 Logistic 模型的限制。
设两个种群的自然增长率分别为 r_1 和 r_2 ，生存极限分别为 K_1 和 K_2 .
设第二个种群每个个体消耗的资源量为第一个种群的 α_1 倍，第一个种群每个个体消耗的资源量为第二个种群的 α_2 倍，则有

$$\begin{cases} \frac{dx_1}{dt} = r_1(1 - \frac{x_1+\alpha_1 x_2}{K_1})x_2, \\ \frac{dx_2}{dt} = r_2(1 - \frac{x_2+\alpha_2 x_1}{K_2})x_1, \\ x_1(0) = x_1^0, \quad x_2(0) = x_2^0. \end{cases}$$

4.弱肉强食模型

假设第二个种群被第一个种群捕食，
单位时间内第一个种群与第二个种群的有效接触数为 $b_{12}x_2(t)$ ，其中 $b_{12} = const$ ，
设两个种群之间的接触系数为 b_{21}， 则有

$$\begin{cases} \frac{dx_1}{dt} = r_1(1 - \frac{x_1}{K_1})x_1 + b_{12}x_1x_2, \\ \frac{dx_2}{dt} = r_2(1 - \frac{x_2}{K_2})x_2 - b_{21}x_1x_2, \\ x_1(0) = x_1^0, \quad x_2(0) = x_2^0. \end{cases}$$