数值计算方法实验报告

实验六：常微分方程数值解

数据科学与计算机学院　17大数据与人工智能

17341015　陈鸿峥

一、 实验题目

尝试用不同方法求解下面的初值问题

$$\begin{bmatrix} u' \\ v' \end{bmatrix} = \begin{bmatrix} 32 & 66 \\ -66 & -133 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} \frac{2}{3}x + \frac{2}{3} \\ -\frac{1}{3}x + \frac{1}{3} \end{bmatrix}, x \in [0, 0.5]$$

初值条件为

$$\begin{bmatrix} u(0) \\ v(0) \end{bmatrix} = \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \end{bmatrix}$$

比较各种方法的计算结果和计算时间。（该问题的精确解为$u = \frac{2}{3}x + \frac{2}{3}\mathrm{e}^{-x} - \frac{1}{3}\mathrm{e}^{-100x}, v = -\frac{1}{3}x - \frac{1}{3}\mathrm{e}^{-x} + \frac{2}{3}\mathrm{e}^{-100x}$）。

二、 实验目的

理解常微分方程的数值解法，会利用不同方法求解微分方程。

三、 实验原理与内容

本次实验主要对欧拉公式、改进欧拉公式和龙格-库塔四阶公式进行探究。记

$$A = \begin{bmatrix} 32 & 66 \\ -66 & -133 \end{bmatrix}, \mathbf{y}_0 = \begin{bmatrix} u(0) \\ v(0) \end{bmatrix} = \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \end{bmatrix}, \mathbf{c}(x) = \begin{bmatrix} \frac{2}{3}x + \frac{2}{3} \\ -\frac{1}{3}x + \frac{1}{3} \end{bmatrix}$$

则

$$f(x, y) = A\mathbf{y} + \mathbf{c}(x)$$

有欧拉公式

$$\mathbf{y}_{n+1} = \mathbf{y}_n + hf(x_n, \mathbf{y}_n)$$

改进欧拉公式

$$\begin{cases} \mathbf{y}_p = \mathbf{y}_n + hf(x_n, \mathbf{y}_n) \\ \mathbf{y}_c = \mathbf{y}_n + hf(x_{n+1}, \mathbf{y}_p) \\ \mathbf{y}_{n+1} = \frac{1}{2}(\mathbf{y}_p + \mathbf{y}_c) \end{cases}$$

标准四阶四段龙格-库塔公式

$$
\begin{cases}
k_1 = hf(x_n, \mathbf{y}_n) \\
k_2 = hf\left(x_n + \frac{1}{2}h, \mathbf{y}_n\right) \\
k_3 = hf\left(x_n + \frac{1}{2}h, \mathbf{y}_n + \frac{1}{2}k_1\right) \\
k_4 = hf\left(x_n + h, \mathbf{y}_n + k_3\right) \\
\mathbf{y}_{n+1} = \mathbf{y}_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)
\end{cases}
$$

下面为本次实验的Mathematica源代码，完整文件已在附件中`ODE.nb`。

```
u[x_] := 2/3 x + 2/3 E^-x - 1/3 E^(-100 x);
v[x_] := -1/3 x - 1/3 E^-x + 2/3 E^(-100 x);
a = 0; b = 0.5;
NList = {5, 50, 500, 5000, 50000};
x = Table[a + i*h, {i, 0, n}];
yn = {u[b], v[b]};
A = {{32, 66}, {-66, -133}};
c[x_] := {2/3 x + 2/3, -1/3 x + 1/3};
f[x_, y_] := A.y + c[x];
Do[h = (b - a)/n;
 (*Euler*)
 y = Table[Table[0, {i, 1, 2}], {j, 1, n + 1}];
 y[[1]] = {1/3, 1/3};
 time = TimeUsed[];
 For[i = 1, i <= n, i++,
   y[[i + 1]] = y[[i]] + h*f[x[[i]], y[[i]]]]
  Print["MaxIter: ", n, "\t Euler err: ", Norm[y[[n + 1]] - yn, 1],
   "\t Time: ", TimeUsed[] - time];
 (*Improved Euler*)
 y = Table[Table[0, {i, 1, 2}], {j, 1, n + 1}];
 y[[1]] = {1/3, 1/3};
 time = TimeUsed[];
 For[i = 1, i <= n, i++,
   yp = y[[i]] + h*f[x[[i]], y[[i]]];
   yc = y[[i]] + h*f[x[[i + 1]], yp];
   y[[i + 1]] = 1/2*(yp + yc)]
  Print["MaxIter: ", n, "\t Improved Euler err: ",
   Norm[y[[n + 1]] - yn, 1], "\t Time: ", TimeUsed[] - time];
 (*Runge-Kutta*)
 y = Table[Table[0, {i, 1, 2}], {j, 1, n + 1}];
 y[[1]] = {1/3, 1/3};
 time = TimeUsed[];
 For[i = 1, i <= n, i++,
```

```
  k1 = h*f[x[[i]], y[[i]]];
  k2 = h*f[x[[i]] + 1/2*h, y[[i]] + 1/2*k1];
  k3 = h*f[x[[i]] + 1/2*h, y[[i]] + 1/2*k2];
  k4 = h*f[x[[i]] + h, y[[i]] + k3];
  y[[i + 1]] = y[[i]] + 1/6*(k1 + 2 k2 + 2 k3 + k4)]
 Print["MaxIter: ", n, "\t Runge-Kutta err: ",
  Norm[y[[n + 1]] - yn, 1], "\t Time: ", TimeUsed[] - time], {n,
 NList}]
```

四、 实验结果与分析

方法	最大迭代轮数	绝对误差	运行时间(s)
欧拉	5	58261.8	0.
改进欧拉	5	$1.14311 \cdot 10^8$	0.
龙格-库塔	5	$2.0589 \cdot 10^{12}$	0.
欧拉	50	0.14357	0.
改进欧拉	50	0.142552	0.
龙格-库塔	50	0.144524	0.
欧拉	500	0.143614	0.016
改进欧拉	500	0.143514	0.
龙格-库塔	500	0.143709	0.031
欧拉	5000	0.153111	0.078
改进欧拉	5000	0.153102	0.281
龙格-库塔	5000	0.15312	0.359
欧拉	50000	0.248979	0.547
改进欧拉	50000	0.24898	1.25
龙格-库塔	50000	0.24898	2.187

从上表可以得出以下结论：

- 不同方法都随着迭代轮数的增加而精度增加
- 当迭代轮数比较少时，欧拉公式的绝对误差较小
- 随着迭代轮数的增加，改进欧拉公式和四阶龙格-库塔公式都取得了较低的绝对误差
- 欧拉公式的速度最快，龙格-库塔的速度最慢，某种程度上速度与精度呈负相关关系

五、 实验总结和心得

本次实验明白了数值求解常微分方程的原理，并且实现了多种数值方法，收获良多。