微分方程式の課題

学籍番号 2120029, 氏名 政野玄空

2023 年 8 月 16 日

1　レポート課題 1

まず $\frac{dy}{dt} = f(t, y)$, 初期値が $y_0 = 1.0$, $t_0 = 0.0$, $dt = 0.1$ のオイラー法のコードを提示する.

ソースコード 1: kadai5-1

```
1  #include <stdio.h>
2  #include <math.h>
3
4  int main(void) {
5      double y = 1.0;
6      double t = 0.0;
7
8      double dt = 0.1;
9      while (t < 10.0) {
10         t += dt;
11         y += dt * y;
12         printf("%f,%f \n",t,y);
13     }
14     return 0;
15 }
```

今回は 39 行目の dt と 42 行目の t を変化させて実行し,Δy 表を作成する.Δy は $\left|\frac{\text{数値解}-\text{厳密解}}{\text{数値解}}\right|$ で計算する.

表 1: 表 1

手法	計算終了時間				
	2	4	6	8	10
オイラー法：dt = 0.1	8.953187e-02	1.710478e-01	2.452654e-01	3.128382e-01	3.743611e-01
オイラー法：dt = 0.025	2.429106e-02	4.799214e-02	7.111745e-02	9.368103e-02	1.156965e-01

dt の値が小さくなればなるほど y の相対誤差が小さくなることが確認できた.

2　レポート課題 2-1

まず $\frac{dy}{dt} = f(t, y)$, 初期値が $y_0 = 1.0$, $t_0 = 0.0$, $dt = 0.2$ の 4 次のルンゲクッタ法のコードを提示する. オイラー法については同様のコードの k1 の値を y に足す,2 次のルンゲクッタ法については同様のコードの k2 の値を y に足すことで計算できるので省略する.

ソースコード 2: 4 次のルンゲクッタ法

```
#include <stdio.h>
#include <math.h>

double f(double y) {
   return y;
}
// 4nd Runge-Kutta Method
void rungeKutta(double t0, double y0, double dt, double t_end) {
    double t = t0;
    double y = y0;

    while (t < t_end) {
        double k1 = dt*f(y);
        double k2 = dt*f(y + (k1 / 2.0));
        double k3 = dt*f(y + (k2 / 2.0));
        double k4 = dt*f(y + k3);
        double exacty = exp(t);
        printf("Runge-Kutta Method\n");
        printf("t = %f, y = %f\n", t, y);
        printf("Exact Solution\n");
        printf("t = %f, y = %f\n", t, exacty);
        printf("relative error\n");
        printf("t = %f, deltay = %e\n", t, fabs((y-exacty)/exacty));
        y += (k1+2*k2+2*k3+k4)/6;
        t += dt;
    }
}

int main() {
    double t0 = 0.0;
    double y0 = 1.0;
    double dt = 0.2;
    double t_end = 10.0;

    printf("Runge-Kutta Method\n");
    rungeKutta(t0, y0, dt, t_end);

    return 0;
}
```

それぞれ dt の値を 0.2,0.1,0.05,0.025,0.0125 に変更してレポート課題 2-1 の表を完成させる.

表 2: 表 2

手法	タイムステップ Δt				
	0.2	0.1	0.05	0.025	0.0125
オイラー法	5.868407e-01	3.743612e-01	2.139677e-01	1.154246e-01	6.010208e-02
2 次ルンゲクッタ法	5.583757e-02	1.534751e-02	3.985508e-03	1.019255e-03	2.579542e-04
4 次ルンゲクッタ法	1.129033e-04	7.667773e-06	4.970922e-07	3.180150e-08	2.013389e-09

3　レポート課題 2-2

エクセルで同様の表を作成し, グラフを作成した.

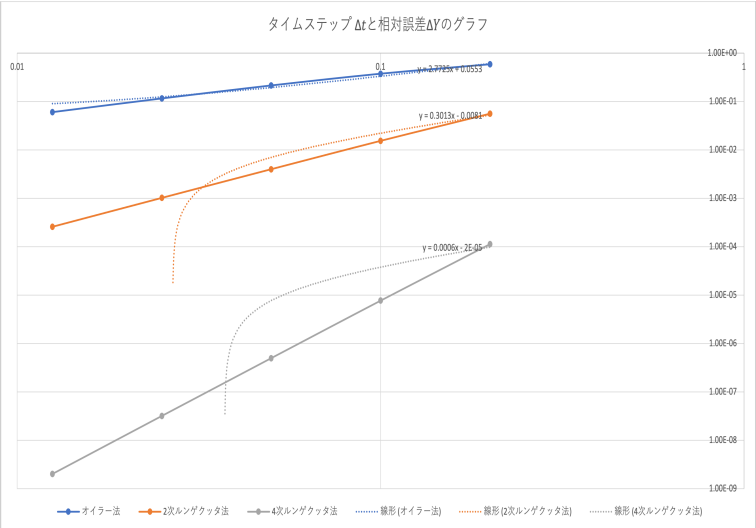

図 1: タイムステップ Δt と相対誤差 ΔY のグラフ

傾きはオイラー法が 2.7723,2 次のルンゲクッタ法が 0.3013,4 次のルンゲクッタ法が 0.0006 となった.

4　レポート課題 2-3

各方法のタイムステップを小さくする利点と欠点を考察していく.

4.1　オイラー法

利点:精度が向上し近似的な解が真の解により近づく. 欠点:t の範囲が小さくても高い精度を得るためには多くのステップが必要になり計算量が大きくなる.

4.2　2 次のルンゲクッタ法

利点:タイムステップを短くすることで, 数値解の精度が向上し, オイラー法よりも高い精度の結果を得ることができるうえにオイラー法に比べて収束性が向上し, 安定して解を得ることができる. 欠点:タイムステップを短くすると, 計算量が増加する. しかし,4 次ランゲクッタ法に比べては計算量は少ない.

4.3　4 次のルンゲクッタ法

利点:高い数値解の精度を持つ. 近似的な解を高次の精度で求めることができる. タイムステップを短くしても, 高次の精度を保ちながら計算を行える. 欠点:計算量が多くなる. タイムステップを非常に小さくする場合, 高い精度を保つことができるが, 計算コストが増加する.

それぞれタイムステップを小さくする場合は精度が向上するが, 計算量が増加し, 特に 4 次のルンゲクッタ法は顕著である. またタイムステップを小さくしすぎると数値誤差が累積していき問題になる場合もある. 問題によって計算時間とリソースを考慮し適切な方法を選ぶのが重要だと考えられる.