情報工学実験１
C++/Octaveによる微分方程式の数値解法

035743A：比嘉雅樹

実験日： 2004/07/02
提出日： 2004/07/09
共同実験者： F グループ
035739B： 任家林
035741D： 浜川ありさ

1　実験の目的

微分方程式の解をコンピュータを用いて数値的に解く方法について学習する。今回は、オイラー法のアルゴリズムをＣ＋＋を用いてプログラミングする事を目的とする。

2　野球ボールの軌道計算プログラムのソース

```
#include "NumMeth.h"

int main() {

  //* ボールの初期位置及び初期速度を設定する.
  double y1, speed, theta;
  double r1[2+1], v1[2+1], r[2+1], v[2+1], accel[2+1];
    cout << "高さの初期値（メートル）: "; cin >> y1;
  r1[1] = 0; r1[2] = y1;    // 初期位置ベクトル
  cout << "初期速度（m/s）: "; cin >> speed;
  cout << "初期角度（度）: "; cin >> theta;
  const double pi = 3.141592654;
  v1[1] = speed*cos(theta*pi/180);   // 初期速度（x）
  v1[2] = speed*sin(theta*pi/180);   // 初期速度（y）
  r[1] = r1[1]; r[2] = r1[2];   // 初期位置および初期速度を設定
  v[1] = v1[1]; v[2] = v1[2];

  //* 物理パラメータを設定（質量，Cd 値など）
  double Cd = 0.35;        // 空気抵抗（無次元）
  double area = 4.3e-3;    // 投射物の横断面積（m^2）
  double grav = 9.81;      // 重力加速度（m/s^2）
  double mass = 0.145;     // 投射物の質量（kg）
  double airFlag, rho;
  cout << "空気抵抗（あり:1，なし:0）: "; cin >> airFlag;
  if( airFlag == 0 )
    rho = 0;   // 空気抵抗なし
  else
    rho = 1.2;   // 空気の密度（kg/m^3）
  double air_const = -0.5*Cd*rho*area/mass;   // 空気抵抗定数
```

```
  //* ボールが地面に着くまで，あるいは最大の刻み数になるまでループ
  double tau;
  cout << "時間刻みτ（秒）: "; cin >> tau;
  int iStep, maxStep = 1000;   // 最大の刻み数
  double *xplot, *yplot, *xNoAir, *yNoAir;
  xplot = new double [maxStep + 1];
  yplot = new double [maxStep + 1];
  xNoAir = new double [maxStep + 1];
  yNoAir = new double [maxStep + 1];


for( iStep=1; iStep<=maxStep; iStep++ ) {

    //* プロット用に位置（計算値および理論値）を記録する
    xplot[iStep] = r[1];   // プロット用に軌道を記録
    yplot[iStep] = r[2];
    double t = ( iStep-1 )*tau;   // 現在時刻
    xNoAir[iStep] = r1[1] + v1[1]*t; // 位置（x）
yNoAir[iStep] = r1[2] + v1[2]*t  - 0.5*grav*t*t;          // 位
置（y）

    //* ボールの加速度を計算する
    double normV = sqrt( v[1]*v[1] + v[2]*v[2] );
    accel[1] = air_const*normV*v[1];   // 空気抵抗
    accel[2] = air_const*normV*v[2];   // 空気抵抗
    accel[2] -= grav;                  // 重力

    //* オイラー法を用いて，新しい位置および速度を計算する

        /*
        ここにオイラー法のアルゴリズムを書く
        */
r[1] = r[1] + tau * v[1];
r[2] = r[2] + tau * v[2];
v[1] = v[1] + tau * accel[1];
v[2] = v[2] + tau * accel[2];

    //* ボールが地面に着いたら（y < 0）ループを抜ける
```

```
        /*
        if 文を使って書く
        */
if(r[2] < 0 ) break;
  }

  //* 最大到達距離と滞空時間を表示する
cout << "到達距離は" << r[1] << "メートル" << endl;
cout << "滞空時間は" << tau * (iStep - 1) << "秒" << endl;
            // 滞空時間の表示をここに書く

  //* プロットする変数を出力する
  //   xplot, yplot xNoAir, yNoAir
  ofstream xplotOut("xplot.txt"), yplotOut("yplot.txt"),
    xNoAirOut("xNoAir.txt"), yNoAirOut("yNoAir.txt");

  int i;
  for( i=1; i<=iStep+1; i++ ) {
    xplotOut << xplot[i] << endl;
    yplotOut << yplot[i] << endl;
  }
  for( i=1; i<=iStep+1; i++ ) {
    xNoAirOut << xNoAir[i] << endl;
    yNoAirOut << yNoAir[i] << endl;
  }

  delete [] xplot, yplot, xNoAir, yNoAir;  // メモリを開放

}
```

3　報告事項

3.1　実際にオイラー法を適用しようとすると、幾つかの問題のために精度が悪く使われることはほとんどない。オイラー法の問題点を説明せよ。

$$\frac{dx}{dt} = f(t,x) \tag{1}$$
$$x(t_0) = x_0 \tag{2}$$

オイラー法とは、上記の２つの式を利用して次々に線形近似で解 (x) の値を決定していくという方法である。
この方法では線形近似をして次の x の値を計算しているので、勾配が大きいと誤差の累積が大きくなっていく。また、式も対称でない為、逆から計算しても元に戻らないという欠点を持っている。その為、精度が悪くあまり使われない。

3.2　微分方程式 $\frac{d}{dt}x = -x, x(0) = 1$ の解を導出して確かめよ。

$$\begin{aligned} \frac{d}{dt}x &= -x \\ dt &= -\frac{dx}{x} \end{aligned}$$

ここで、両辺を積分すると

$$\begin{aligned} t &= -log(x) + c \\ -loge^t &= log(x) + c \\ e^{-t} &= x + c \end{aligned}$$

x(0) = 1 より

$$\begin{aligned} x(0) &= e^0 + c \\ 1 &= 1 + c \\ c &= 0 \end{aligned}$$

よって、$x(t) = e^{-t}$ となる。

3.3 ボールの軌道のグラフを gnuplot で出力せよ。

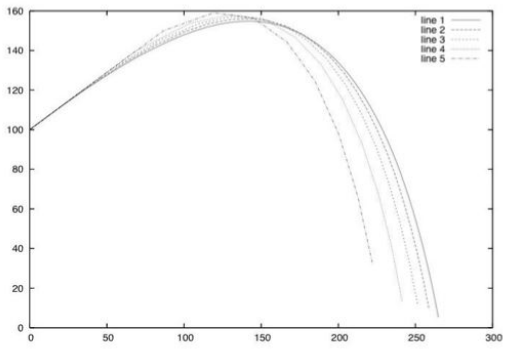

図 1: ボールの軌道のグラフ

上の図での、
高さの初期値（メートル）：100
初期速度（m/s）：100
初期角度（度）：30
空気抵抗ありで、時刻刻み τ は次の通りである。
line1 → 0.2
line2 → 0.4
line3 → 0.6
line4 → 0.8
line5 → 1.0

3.4 時間刻み τ の設定によって、計算結果が大きく異なることが確認できる。その理由を考察せよ。

3.1 でも説明したように、時間刻み τ が大きいとその分誤差が大きくなる。これにより演算結果が異なり、グラフも異なってくると考えられる。

3.5 オイラー法よりも高精度な数値計算アルゴリズムについて調べよ。

- 修正オイラー法
  今回の実験で学んだオイラー法とは、図 2 の左図のように積分値を長方形の面積で近似するという粗い近似を行っています。これよりもう一段細かい近似をするのが修正オイラー法です。修正オイラー法は、図 2

の右図のように台形の面積で近似するという方法をとっています。

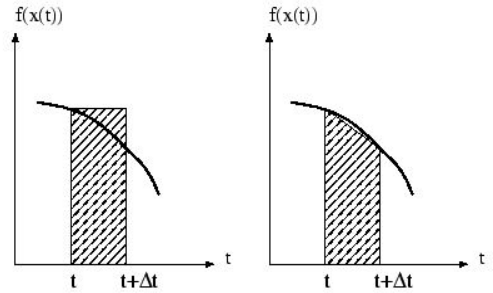

図 2: オイラー法と修正オイラー法の近似方法の違い

式は台形の面積の式

$$\frac{\Delta}{2}[f(x(t)) + f(x(t+\Delta))]$$

で表せます。しかし、$x(t)$ まで計算した時点では、$x(t+\Delta)$ は計算したい量であって、その値は未知であり、既知の量は $x(t)$ だけです。そこで、$x(t+\Delta)$ として予測値

$$x^p(t+\Delta) = x(t) + \Delta f(x(t))$$

をまず計算し、それを用いて

$$x(t+\Delta) = x(t) + \frac{\Delta}{2}[f(x(t)) + f(x^p(t+\Delta))]$$

と与えることで求められます。

参考文献

[1] http://yosirin9.hp.infoseek.co.jp/contents/no3/paper_no3.htm

[2] http://cosmos.js.yamanashi.ac.jp/~toyoki/js_prog01/prac_diff_eqns/node11.html