常微分方程式の初期値問題 (1) 数値解法の紹介

桂田 祐史

1992 年 5 月 20 日

1 はじめに

今回から数回に渡って常微分方程式の初期値問題の数値シミュレーションを考えます。常微分方程式としては正規形のもののみを扱います。よく知られているように高階の方程式も一階の方程式に帰着されますから、当面一階の方程式のみを考えます。独立変数を t、未知関数を $x = x(t)$ とすれば、一階正規形の常微分方程式は

$$\frac{dx}{dt} = f(t,x) \quad (t \in [a,b]) \tag{1}$$

と表わされます。ここで f は既知関数です。方程式 (1) に初期条件と呼ばれる

$$x(a) = x_0 \tag{2}$$

の形の条件を与えて、 (1),(2) を同時に満たす関数 $x(t)$ を求めるのが常微分方程式の初期値問題です（この時 $x(t)$ を初期値問題 (1),(2) の解と呼びます）。ここで x_0 は既知の量です。

変数分離形のような方程式の場合は、簡単な演算で厳密解が求まりますが（いわゆる求積法と呼ばれるものです）、それが不可能な問題も数多くあります。そういう場合も以下に解説するような数値解法で近似解を求めることが出来ます。

基本的な考え方は次のようなものです。「問題となっている区間 $[a,b]$ を

$$a = t_0 < t_1 < t_2 < \cdots < t_N = b$$

と分割して、各‘時刻’t_n での x の値 $x_n = x(t_n)$ $(n = 1,2,\ldots,N)$ を求めることを目標とする。そのために微分方程式 (1) から $\{x_n\}_{n=0,\ldots,N}$ を解とする適当な差分方程式を作り、それを解く。」

区間 $[a,b]$ の分割の仕方ですが、以下の例では簡単のため N 等分することにします。つまり

$$h = (b-a)/N, \quad t_n = a + nh.$$

となります。

2 Euler（オイラー）法

微分 $x'(t) = \dfrac{dx}{dt}$ は差分商 $\dfrac{x(t+h)-x(t)}{h}$ の $h \to 0$ の極限です。そこで、(1) 式の微分を差分商で置き換えることによって次の方程式を得ます。

$$\frac{x_{n+1}-x_n}{h} = f(t_n, x_n) \quad (n = 0,1,\ldots,N-1)$$

変形すると

$$x_{n+1} = x_n + hf(t_n, x_n).$$

これを漸化式として使って、 x_0 から順に x_1, x_2, ..., x_N が計算出来ます。この方法を Euler 法と呼びます。

こうして得られる $N+1$ 個の点 (t_n, x_n) を順に結んで得られる折れ線関数は（f に関する適当な仮定のもとで） $N \to \infty$ の時（$h = (b-a)/N$ について言えば $h \to 0$）、真の解 $x(t)$ に収束することが証明できます（現在の数学科のカリキュラムでは 3 年次に開講されている常微分方程式の講義で学びます）。ここでは簡単な例で収束を確かめてみましょう。

例 1. 初期値問題

$$x'(t) = x \quad (t \in [0,1]), \quad x(0) = 1$$

の解は $x(t) = e^t$ であるが、 Euler 法を用いて解くと $x_N = (1+1/N)^N$ となる。確かに $N \to \infty$ とすると $x_N \to e = x(1)$ となっている。

例題 4.1 Euler 法を用いて、例 1 の初期値問題を解くプログラムを作って収束を調べよ。

`reidai4-1.f` は、指示された分割数 N を用いた Euler 法により数値解を計算するプログラムです。以下の実行例では $N = 1, 2, \ldots, 20$ に対して、折れ線関数のグラフを描いています。

```
waltz11% f77o reidai4-1.f
waltz11% reidai4-1 | mgraph -M | xplot
1,20,1                                   ← 使用する N の指定
                                           1 から 20 まで 1 刻みで
```

`reidai4-1b.f` は、 $t = 1$ での誤差 $|x_N - x(1)|$ を計算するプログラムです（上に述べたように $x(1) = e = 2.71828\ldots$ です）。誤差がどのように減少するか見てみましょう。

```
waltz11% f77o reidai4-1b.f
waltz11% reidai4-1b | mgraph | xplot
1,100,1
```

`mgraph` コマンドには、対数目盛によるグラフを描く機能（`-lx`, `-ly` ここで `l` は logarithmic の頭文字です）があります。それを用いると

```
waltz11% reidai4-1b | mgraph -lx -ly | xplot
2,1000,1
```

このグラフから、誤差 $= O(N^{-1}) \quad (N \to \infty)$ であることが読みとれます。実はこれは Euler 法の持つ一般的な性質です。

問 上の議論でグラフからどうして、誤差 $= O(N^{-1}) \quad (N \to \infty)$ が推測できるのか論ぜよ。

問題 4.2 例題 4-1 とは異なる初期値問題を適当に設定して、それを Euler 法で解いてみよ。収束の速さはどうか。

例題 4.2[1] Euler 法を用いて、以下の問題を $t = 1$ まで解け。

$$\frac{du}{dt} = v$$

$$\frac{dv}{dt} = -g_0 - \mathrm{sign}(v) C v^2 / m$$

[1] この問題は、戸川隼人著「数値計算」岩波書店から採りました。この本は学部レベルの数値計算の入門書として推薦出来ます本です。

$$u(0) = 0, \quad v(0) = v_0$$

ここで、 g_0, C, m, v_0 は正の定数である（例えば $g_0 = 9.8, C/m = 0.1$ としておきます）。 sign は符号を表わす関数である（変数が正の時 1, 0 の時 0, 負の時 -1）。（これは空気抵抗を考えた場合の、鉛直方向へのボール投げのシミュレーションです。）

上記の方程式は一見複雑そうですが $x = (u, v)$ とすれば、一階正規形常微分方程式の初期値問題 (1),(2) の形をしています。

簡単のため $\phi(u, v) = -g_0 - \mathrm{sign}(v)Cv^2/m$ とおけば、 Euler 法の式は

$$u_{n+1} = u_n + hv_n, \quad v_{n+1} = v_n + h\phi(u_n, v_n)$$

となります。この漸化式で u_n, v_n を計算するプログラムが `reidai4-2.f` です。一行ごとに t_n, u_n, v_n を三つ並べて出力します。以下の例では、初速に相当する $v_0 = 5$、区間の分割数 $N = 10$、時刻 $t = 0$ から $t = 1$ までの計算をしています。

```
waltz11% f77o reidai4-2.f
waltz11% reidai4-2
5,10,0,1
```

問題 4-3 `reidai4-2.f` を修正して（あるいは別のプログラムを書いて）、 $t - u$ 曲線（u の時間変化を表すグラフ）、 $t - v$ 曲線（v の時間変化を表すグラフ）を描け[2]。空気抵抗の強さを表す係数 C が 0 の場合と比較せよ。

3 Runge-Kutta （ルンゲ - クッタ）法

前節で解説した Euler 法は簡単で、これですべてが片付けば喜ばしいのですが、残念ながらあまり「精度が高く」ありません（率直にいって実用性は低いです）。現在まで様々な方法が開発されていますが、比較的簡単で精度が高いものに Runge-Kutta 法と呼ばれるものがあります。それは x_n から x_{n+1} を求める漸化式として次のものを用います。

$$k_1 = hf(t_n, x_n)$$

$$k_2 = hf(t_n + h/2, x_n + k_1/2)$$

$$k_3 = hf(t_n + h/2, x_n + k_2/2)$$

$$k_4 = hf(t_n + h, x_n + k_3)$$

$$x_{n+1} = x_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

これがどうやって導かれたものかは今回は解説しません。まずは使ってみて下さい。

問題 4-4

$$x'(t) = x \quad (t \in [0, 1]), \quad x(0) = 1$$

を Runge-Kutta 法を用いて $[0, 1]$ で解く時、 x_1 は x_0 のどういう式で表わされるか？またこれを Euler 法を用いた場合と比べるとどんなことが解るか？

問題 4-5 区間の分割数 N を変えながら

$$x'(t) = x \quad (t \in [0, 1]), \quad x(0) = 1$$

を Runge-Kutta 法を用いて解くプログラムを作り、 $t = 1$ での誤差 $|x_N - x(1)|$ を調べよ（例題 4-1 の真似をする）。 Euler 法と比べてどうなるか？

[2](UNIX マニア向けの注意) awk というコマンドを知っていれば、プログラムを書き換えなくても済みます。