システム数理IV 第2回 1999/4/26 改訂版

牧野淳一郎

1999 年 4 月 27 日

1 数値解法の基本 ：Euler method

前章では、駆け足で常微分方程式の解の存在を示した。今回は、数値解法のもっとも単純なものである Euler 法を題材に、常微分方程式の数値解とはどういうものかについて考えていこう。

というわけで、 Euler 法とは何かということから始めよう。

2 Euler 法—一変数の場合

まず、一変数の場合を考えよう。微分方程式の初期値問題

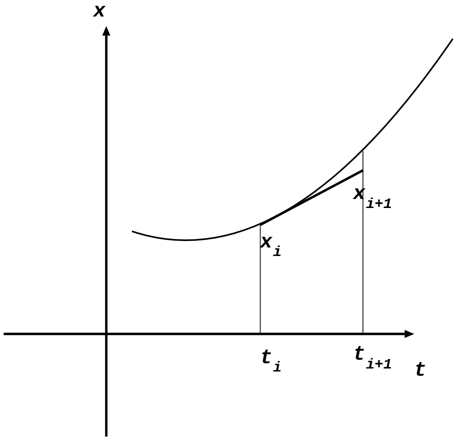
$$\dot{x} = f(x,t), \quad x(t_0) = x_0 \tag{1}$$

に対して、（前進）オイラー法とは以下のような方法である。

時刻 t_i での数値解が x_i であったとすると、時刻 $t_{i+1} = t_i + \Delta t$ での数値解を

$$x_{i+1} = x_i + \Delta t f(x_i, t_i) \tag{2}$$

という形に書く。



上の図に様子を示す。ここでは、真の解（がわかっているものとして）を曲線で示し、そこから出発したとした。

やっていることはなんということはなくて、真の解に対する時刻 t_i での接線を数値解ということにしようということである。

ある時刻 t_s での初期条件が与えられて、 t_e での解を求めようという時に、例えばその区間を n 等分して

$$t_0 = t_s, \quad t_k = t_0 + \Delta t k, \quad \Delta t = \frac{t_e - t_s}{n} \tag{3}$$

というふうに t_i を決めてやる。あとは x_1 から順に $x_2, x_3, ...x_n$ と計算していけば、 $t_e = t_n$ での数値解 x_n が求まるわけである。

直観的には、 n を大きくしていけば、正しい答えに近付いていきそうな気がするであろう。しかし、そうなるということをどうやって証明すればいいであろうか ?

2.1　例　:線形方程式の場合

とりあえず、答がわかっている場合にオイラー法がどう振舞うかということをみてみることにしよう。初期値問題

$$\dot{x} = kx, \quad x(0) = 1 \tag{4}$$

の $t = 1$ での数値解を考える。いうまでもないが、厳密解が $x = e^{kt}$で与えられるわけである。

$$x_{i+1} = x_i + \Delta t k x_i = x_i(1 + \Delta t k) \tag{5}$$

なので、

$$x_e = (1 + k/n)^n \tag{6}$$

である。指数関数は定義により

$$e^t = \lim_{n\to\infty} (1 + t/n)^n \tag{7}$$

と書けるので、問題は、この　数値解」がどれくらい正しいかということであるが、

$$\lim_{n\to\infty} x_e = e^k \tag{8}$$

となり、$n \to \infty$ の極限で数値解が厳密解に一致するということが示せたことになる。上の証明は任意の係数、初期条件と積分区間に一般化できる。

なお、多変数の場合にもまったく同様に厳密解に収束することが証明できるが、これは計算練習ということにしておく。各自証明を試みること。

2.2　計算機の精度

さて、前節では、とりあえずもっとも簡単な一変数線形の方程式で極限で厳密解が得られるということを証明できたわけだが、じつは実際に計算機で計算してみるとこんなにうまくはいかない。

うまくいかないというのはどういう意味かというと、実際に n をどんどん大きくしていっても、答が厳密解に近付かなくなるどころかかえって遠ざかっていくということが起きる。これは実際にプログラムをつくって確かめてみればすぐにわかる。

なぜそういうことが起きるか、また、そうならないようにして意味がある数値解を得るためにはどうすればいいかというのが、さまざまな数値解法が研究される大きな目標の一つである。ちなみに、もうひとつの目標は、なるべく短い計算時間で正確な答を得るということである。

計算機で、例えば C でプログラムを書くと、`int` と宣言する、普通整数とよぶデータ型と、`float` または `double` と宣言する、普通「実数」とよぶデータ型を必要に応じて使い分ける。微分方程式の解を求めようという場合なら、普通従属変数には `double` を使うであろう。

計算機を使う時に「実数」と呼ぶものは、数学的な意味での実数とはもちろん違うものである。なにが違うかというと、基本的には「有限の桁数」しかないということである。

つまり、数学的な実数というのは、もちろん、連続であるわけで、ということは、たとえば 1.000.....0001 と 0 が何個続いても、そういう実数というものはあるわけである。

ところが、計算機では、ある数を有限の情報量（ビット数）で表現する必要がある。今のふつうの計算機では、例えば C の float なら 32 ビット、double なら 64 ビットを使う。例えば float の場合、実際の表現は（符号）$\times(2^{\text{指数}})\times$（仮数）という形になっていて、符号に 1 ビット（0 なら正、1 なら負）、指数に 8 ビット、仮数に残りの 23 ビットを使う。

仮数は 1 と 2 の間を表現すればいいので、0 を 1 ということにして、オール 1（16 進数で `7fffff`）を $2-2^{-23}$に割り当てる。指数については、正負両方にとるために、7f のときに $2^0=1$ ということにする。

なお、あと、0 というものを表現する必要があるが、これは「全ビット 0 」が 0 ということにする。

ちょっと話が細かくなったが、問題は要するに有限の桁数（ビット数）で表現されているということである。このために、四則演算、あるいは関数計算をしたときに得られる答というものが、厳密に正しいわけではなくなっている。つまり、無限小数になるものをどこかで打ち切るので、その分誤差がでるわけである。

なお、打ち切り方には、切捨て、切り上げ、四捨五入などいろいろあり得るが、最近の計算機は大抵四捨五入になっている。といっても 2 進数なので 0 捨 1 入 (最近接丸め) ということになる。さらに細かいことをいうと、捨てるところがちょうど 1 の時にどうするべきかという問題があるが、これはさらにその上が 0 になるほうにもっていく（最近接偶数丸め）ということをする計算機もある。

double であれば仮数が 52 ビット、指数が 11 ビットになる。

2.3 丸め誤差のある場合の収束性

実は丸め誤差がある場合の計算精度の議論というのはかなり難しい。しかし、このあたりがちゃんとわかってないと、計算して出てきた答が正しいかどうかということが理解できないので、少し詳しい解説を試みる。

前にやった、線形方程式の例をそのまま使うことにしよう。式ではわからないので、操作手順を表してみる。

```
x = 1;
dt = 1.0/n;
for(i=0;i<n;i++){
   x = x + x*k*dt
}
```

例えば C で書けばこんな感じだろう。丸め誤差の結果、答がどう変わるかということを考えてみる。

これにはいくつかの方針がありえる。一つは、「悲観的」解析である。（区間解析という専門用語が

ある）つまり、演算毎に、誤差があったら答が最大これだけずれるというのを見積もって、答の両側に幅をつけていく。もう一つは、「楽観的」解析（確率的解析ともいう）である。丸めが最近接丸めであるときには、「計算結果は真の値の回りのある幅で一様分布する」と仮定してよい場合が多い。この時には、計算結果は確率的に真の値の回りをランダム・ウォークすることになる。

これらの 2 つの方法は原理的には美しいが、実際にはどちらも役に立たないことが多い。というのは、現実にはどちらも恐ろしく手間がかかるうえに、悲観的解析はあまりに悲観的であり、楽観的解析は逆に過度に楽観的であるからである。上の例で、どうなるか考えてみる。

このためには、まず、丸め誤差のモデル化が必要である。いま、簡単のために、任意の演算に対して丸め誤差が以下のように表現できるとする：演算の「真の」結果が p であるとき、計算機で表現された結果は区間 $[(1-\epsilon)p, (1+\epsilon)p]$ の間に一様分布する。

まず、区間解析を試みよう。面倒なので $k=1$ とし、k との乗算では誤差は生じないとする。この時、$dt = [(1-\epsilon)/n, (1+\epsilon)/n]$ となり、`for` の最初の反復の後では

$$
\begin{aligned}
x^{(1)} &= [(1+(1-\epsilon)/n)(1-\epsilon), (1+(1+\epsilon)/n)(1+\epsilon)] \\
&\simeq [1+1/n-\epsilon(1+2/n), 1+1/n+\epsilon(1+2/n)] \qquad (9)
\end{aligned}
$$

で、次の反復ではどうなるかとかいって順に計算していけばいいが、これはあまりに大変であるので以下の近似を行なう：

- 誤差は `x + x*dt` の、加算のところだけで出る。

これは、通常の場合にはそれほど悪い近似ではない。このときは、結局、数値解が

$$
\left(1+\frac{1}{n}-\epsilon\right)^n < x < \left(1+\frac{1}{n}+\epsilon\right)^n \qquad (10)
$$

の範囲にくるということになる。

今、$n >> 1, n\epsilon << 1$ として展開すれば、

$$
e(1-n\epsilon) < x < e(1+n\epsilon) \qquad (11)
$$

ということになり、n に比例する誤差が入るということになる。

さて、確率的解析の場合にも同様に上の単純化をすると、結果は誤差の平均値は 0 で分散が $n\epsilon^2$ の程度ということになる。

実際に計算してみると、区間評価では $n > \sqrt{1/\epsilon}$ で誤差を大きめに、逆に確率評価では $n > \epsilon^{-2/3}$ で小さめに見積もっている。

この、非常に単純な場合には、どちらについてもそのようにずれる理由がわかり、精密な解析を行なえば誤差のもう少し正しい上限を与えることは不可能ではない。が、解くべき方程式や計算法がすこし複雑になると、解析が非常に困難になることは注意するべきであろう。

なお、一々丸め誤差のことを考えていると話が進まないので、以下、基本的な話は丸めのことを無視して、必要に応じて丸めの影響を考えることにする。なお、ふつう「打ち切り誤差」という時には、この意味の丸めを無視したもののことである。

3　オイラー法の収束性と打ち切り誤差

線形方程式の場合に、ステップサイズ 0 の極限で真の解に収束するということを前に示したわけだが、計算機が実際に計算するときにはもちろん有限のステップサイズである。また、上に述べたように丸めの影響があって、どうせ収束はしない。そうすると、どの程度のステップサイズの時に計算精度がどうなるかということを知る必要がある。

もちろん、プログラムを書いて、真の解との誤差を調べればいいが、もともと数値解を計算するのは真の解がわからないからなので、そうはいかないわけである。

以下、一般の微分方程式に対してオイラー法がどのように収束するかということを考えてみる。先週やった存在定理と同じように、常微分方程式の初期値問題

$$dx/dt = f(x,t), \quad x(t_s) = x_0 \tag{12}$$

の $t_s \le t \le t_e$ での解を考える。今、関数 f が領域 $t_s \le t \le t_e, |x - x_0| \le b$ で連続であり、最大値、最小値をもつとしよう。絶対値が M でおさえられ、$t_e - t_s < b/M$ という関係がなりたつとする。さらに、上の範囲の任意の t, x と x' についてリプシッツ条件

$$|f(x,t) - f(x',t)| \le L|x - x'| \tag{13}$$

を満たすものとする。さらに、n, h を、$nh = t_e - t_s$ を満たすようにとる（n は整数）。また、f は 2 回以上微分可能であるとする。

このとき、オイラー法が一様に一次収束する、すなわち、ある定数 C が存在し、

$$|x_i - x(t_i)| \le Ch \tag{14}$$

となることを示すことができる。

こんなのは当たり前と思う人もいるかもしれないが、なかなかそうでもない。証明の方針としては、例によって具体的に x_{i+1}を構成しておく：

$$x_{i+1} = x_i + hf(x_i, t_i) \tag{15}$$

今、真の解との差が欲しいので、こちらも書いておくと

$$x(t_{i+1}) = x(t_i) + \int_{t_i}^{t_{i+1}} f(x(t),t)dt = x(t_i) + hf(x(t_i), t_i) + O(h^2) \tag{16}$$

ここで、何故最後の $O(h^2)$ が出てくるかというと、これは f が x, t のそれぞれで 2 回以上微分可能としたからである。平均値の定理を適当に使えばこの形に変形できる。差をとれば

$$x_{i+1} - x(t_{i+1}) = x_i - x(t_i) + h[f(x_i, t_i) - f(x(t_i), t_i)] - O(h^2) \tag{17}$$

絶対値をとれば

$$|x_{i+1} - x(t_{i+1})| \le |x_i - x(t_i)| + |h[f(x_i, t_i) - f(x(t_i), t_i)]| + Ah^2 \tag{18}$$

ここで A は定数である。今、

$$e_i = x_i - x(t_i) \tag{19}$$

とおいて、さらにリプシッツ条件から変形して

$$|e_{i+1}| \le |e_i|(1 + hL) + Ah^2 \tag{20}$$

ここで、明らかに $e_0 = 0$ なので、容易に（！）わかるように

$$|e_i| \leq Ah^2 \sum_{k=0}^{i-1} (1+hL)^k = \frac{Ah}{L}[(1+hL)^i - 1] \tag{21}$$

ここで、$nh = t_e - t_s$ であったことを思い出すと

$$(1+hL)^i - 1 \leq e^{L(t_e - t_s)} - 1 \tag{22}$$

結局、

$$|y_i - y(t_i)| \leq \frac{Ah}{L}[e^{(t_e - t_s)L} - 1] \tag{23}$$

となる。

というわけで、一応、誤差の上限が存在して、それが h に比例するということが示された。なお、上限の形をみるとなかなか嫌な格好をしていることがわかる。つまり、時刻依存性が e^{tL} の形をしているので、積分を先に進めていくと、かならず誤差が指数関数的に増大していくことになる。これは、言い換えれば、長時間積分した場合の精度については理論的にはなかなか難しい問題があるということである。この問題については、より詳しくはまたあとで扱うことにしよう。

なお、このような、ある時間範囲で積分したあとの誤差のことを「大域誤差」という。これに対し、１ステップだけ積分したあとの誤差を局所誤差、あるいは局所離散化誤差という。

4　後退オイラー法

オイラー法では、

$$x_{i+1} = x_i + hf(x_i, t_i) \tag{24}$$

という風に、t_i のところの量だけを使って x_{i+1} を計算した。これの変形として、

$$x_{i+1} = x_i + hf(x_{i+1}, t_{i+1}) \tag{25}$$

というものを考えることができよう。これでは、t_{i+1} のところの量が使われる。

この場合、左辺値が右辺に入っているので、全体として x_{i+1} についての代数方程式になっている。この方程式を解くことによって新しい時刻での値がわかるという仕掛けになる。

このようなやり方のことを一般に陰的公式 implicit scheme と呼ぶ。それに対して、方程式を解かなくても新しい時刻での値が直接計算できるものを、陽的公式 explicit scheme と呼ぶ。

この方法は、精度（収束性）についてはオイラー法と変わらないが、一々方程式をとかないといけないので計算はずっと大変になる。が、それ以上にメリットがある場合もあり、そういった問題には後退オイラー法、あるいはそれに由来するようないろいろな方法が使われる。

5　もう少し賢い方法

オイラー法は、すでに述べたように 1 次収束しかしない。つまり、計算精度をあげていこうとすると、それに比例して計算量が増えてしまう。また、ステップ数が大きくなるので丸め誤差の影響が出てくる。それでは、もう少し賢い方法はないのであろうか？

もちろん、賢い方法はいろいろあるわけである。この講義では、主に以下の3つのクラスの公式を扱う。ほかのものもないわけではないが、実用になっているのはほぼこの3つに限られるからである。

1. ルンゲ・クッタ法
2. 線形多段階法
3. 補外法

とりあえず、次回（ともう一回くらい？）でルンゲ・クッタ法を扱う。

6 ルンゲ・クッタ法

ルンゲ・クッタ法というのは、ある一般的な形に書ける公式のクラスであるが、ちょっとわかりにくいのでまず例をあげておこう。

6.1 二次のルンゲ・クッタ法

以下のような計算法を考える

$$\begin{aligned} k_1 &= x_i + \frac{h}{2} f(x_i, t_i) \\ x_{i+1} &= x_i + h f(k_1, t_i + h/2) \end{aligned} \tag{26}$$

これは二次の公式になっている。

6.2 公式の次数について

ここで、局所離散化誤差という概念を導入する。これは以下のように定義される。(t_i, x_i) から出発した数値解が x_{i+1} であったとし、また厳密解が $x(t)$ であるとする。これは、あくまでも x_i からの解—局所的な解—であって、初期条件からの解ではないことに注意して欲しい。この時に、数値解の局所離散化誤差が m 次であるとは、

$$x(t_{i+1}) = x_{i+1} + O(h^m) \tag{27}$$

という関係を満たすこと、言い換えれば、ある定数 A が存在して、

$$|x(t_{i+1}) - x_{i+1}| \le A h^m \tag{28}$$

なる関係を満たすことである。A は方程式に依存してよいが、h には依存しない必要がある。

なお、ある公式が m 次の局所離散化誤差を持つ時、大域誤差は $m-1$ 次になる。これは、先にやったオイラー法の一次収束の証明とまったく同様に行なうことができる。従って、誤差について議論するには、とりあえず局所離散化誤差がわかればいいということになる。

オイラー法の局所離散化誤差は、

$$x_{i+1} = x_i + h f(x_i, t_i) \tag{29}$$

と、t_i での x の真の解のテイラー展開

$$x(t_i+h) = x_i + h\frac{dx}{dt} + \frac{h^2}{2}\frac{d^2x}{dt^2} + O(h^3) \tag{30}$$

を比べてみればわかるように、$O(h^2)$ になっている。

上に書いた方法はどうなっているかというのを考えると、少し計算すればわかるように、$O(h^3)$ になっている。従って、局所で 3 次、大域誤差で 2 次になるわけである。

これは、例えば 1 次では必要な精度を実現するのに 100 万ステップ必要であったものが、大雑把にいってその平方根の 1000 ステップで済むということになる。

7　次週予告

来週は、ルンゲ・クッタ公式についてもう少し一般的な議論を行なう。