Runge-Kutta 法による一階常微分方程式の解法

永井佑紀

平成 25 年 11 月 21 日

準古典理論で現れる方程式である Riccati 方程式は一階常微分方程式である。この微分方程式を数値的に解く為には、Runge-Kutta 法が使われる。8 段 7 次の Runge-Kutta 法 [1] の係数についてまとめた [1]。

1 Runge-Kutta 法

1.1 概略

Runge-Kutta 法による数値解法に関しては、一冊の本の大部分を割くほどに様々な手法が存在するため、ここでは触れない。詳細は参考文献を参考とすること [2]。Runge-Kutta 法の解説自体も多いので、ここでは概略のみを記す。

Runge-Kutta 法では、

$$\frac{dy}{dx} = f(x, y(x)) \tag{1}$$

$$y(x_0) = y_0 \tag{2}$$

という初期値問題を解く。ここで f は x と $y(x)$ の両方に依存しており、一般的には、$y(x)$ に関する非線形な項を含む。数値計算においては、x は離散的にしか取れない。この微分方程式の両辺を x で積分すると

$$\int_{x_n}^{x_{n+1}} dx \frac{dy}{dx} = \int_{x_n}^{x_{n+1}} dx f(x, y(x)) \tag{3}$$

$$y(x_{n+1}) - y(x_n) = \int_{x_n}^{x_{n+1}} dx f(x, y(x)) \tag{4}$$

となる。さらに、

$$x_{n+1} - x_n \equiv h \tag{5}$$

$$x = x_n + x'h, \ (x_n \leq x \leq x_{n+1}) \tag{6}$$

と定義すると、$dx/dx' = h$ より

$$y(x_{n+1}) = y(x_n) + h \int_0^1 dx' f(x_n + hx', y(x_n + hx')) \tag{7}$$

が得られる。この式により、ある離散点 x_n での値 $y(x_n)$ がわかっているときに、距離 h だけ離れた離散点 x_{n+1} での値 $y(x_{n+1})$ がどう書けるか、ということを書き下すことができた。しかし、計算機上で積分を実行する事は不可能なので、この積分は離散化しなければならず、

$$y(x_{n+1}) \sim y(x_n) + h \sum_{i=1}^{s} b_i f(x_n + hc_i, y(x_n + hc_i)) \tag{8}$$

と書かなければならない。この式に基づく方法を、s 段 Runge-Kutta 法と呼ぶ。

第二項はもともと積分なので、微分方程式の問題を、数値積分をどのように評価するかという問題に置き換えたことになる。積分をどのように評価するかで、様々な近似計算法が生まれる。

[1] 故林伸彦氏の超伝導渦糸系のプログラム中で使われている方法である。

1.2 1段 Runge-Kutta 法

まず、一番簡単な場合として、$s=1$ を考えてみる。手元にある情報は、x_n、$y(x_n)$ である。ここで、$f(x_n, y(x_n))$ というのは、元の式 (1) から明らかなように、点 x_n における関数 $y(x)$ の傾きである。よって、一番シンプルなのは、

$$y(x_{n+1}) = y(x_n) + hf(x_n, y(x_n)) \tag{9}$$

である。これは、積分を矩形公式を使って

$$h\int_0^1 dx' f(x_n + hx', y(x_n + hx')) \sim h\int_0^1 dx' f(x_n, y(x_n)) = hf(x_n, y(x_n)) \tag{10}$$

と置き換えた事に対応する。この方法は、オイラー法と等価である。

1.3 2段 Runge-Kutta 法

次に、もう少し積分を改善してみる。一番簡単なのは、

$$h\int_0^1 dx' f(x_n + hx', y(x_n + hx')) = h\int_0^{1/2} dx' f(x_n + hx', y(x_n + hx')) + h\int_{1/2}^1 dx' f(x_n + hx', y(x_n + hx')) \tag{11}$$

とわけて、まず、第一項を矩形公式で評価：

$$k_1 = \int_0^{1/2} dx' f(x_n + hx', y(x_n + hx')) = \int_0^{1/2} dx' f(x_n, y(x_n)) = \frac{1}{2} f(x_n, y(x_n)) \tag{12}$$

し、$x_n + h/2$ の点での値を

$$y(x_n + h/2) = y(x_n) + hk_1 \tag{13}$$

求め、第二項はこの値を使ってまた矩形公式で評価：

$$\int_{1/2}^1 dx' f(x_n + hx', y(x_n + hx')) = \int_{1/2}^1 dx' f(x_n + h/2, y(x_n) + hk_1)) = \frac{1}{2} f(x_n + h/2, y(x_n) + hk_1) \tag{14}$$

する。その結果、

$$y(x_{n+1}) = y(x_n) + \frac{h}{2}(f(x_n, y(x_n)) + f(x_n + h/2, y(x_n) + hk_1)) \tag{15}$$

が得られる。これは積分を台形公式で近似したことと等しい。つまり、台形公式は

$$h\int_0^1 dx' f(x_n + hx', y(x_n + hx')) = \frac{h}{2}(f(x_n, y(x_n)) + f(x_n + h, y(x_n + h))) \tag{16}$$

であるが、$f(x_n + h, y(x_n + h))$ は $y(x_n + h)$ がそもそも求めたいものなので求まらないため、一つ手前の $f(x_n + h/2, y(x_n + h/2))$ を使っているのである。

1.4 古典的 Runge-Kutta 法

古典的 Runge-Kutta 法とは、4段4次 Runge-Kutta 法のことであり、よく使われる。文献も多いのでここでは詳細については説明しない。

この方法では、積分は

$$h\int_0^1 dx' f(x_n + hx', y(x_n + hx')) = \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{17}$$

であり、それぞれの k_i は $i = 1$ から順番に

$$k_1 = f(x_n, y_n) \tag{18}$$
$$k_2 = f(x_n + h/2, y(x_n) + hk_1/2) \tag{19}$$
$$k_3 = f(x_n + h/2, y(x_n) + hk_2/2) \tag{20}$$
$$k_4 = f(x_n + h, y(x_n) + hk_3) \tag{21}$$

のように逐次的に求める。

1.5 一般の場合と誤差評価

s 段の Runge-Kutta 法は

$$y(x_{n+1}) = y(x_n) + h\sum_{i=1}^{s} b_i k_i \tag{22}$$

で与えられ、ここで k_i は逐次的に

$$k_1 = f(x_n, y(x_n)) \tag{23}$$
$$k_2 = f(x_n + c_2 h, y(x_n) + a_{21} hk_1) \tag{24}$$
$$k_3 = f(x_n + c_3 h, y(x_n) + a_{31} hk_1 + a_{32} hk_2) \tag{25}$$
$$\vdots \tag{26}$$
$$k_s = f\left(x_n + c_s h, y(x_n) + h\sum_{i}^{s-1} a_{s,i} k_i\right) \tag{27}$$

で定義される。これらの係数の値は様々なものが提案されており、どのような Runge-Kutta 法であるかを見る為に tableau：

$$\begin{array}{c|ccccc} 0 & & & & & \\ c_2 & a_{21} & & & & \\ c_3 & a_{31} & a_{32} & & & \\ \vdots & \vdots & & \ddots & & \\ c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & \\ \hline & b_1 & b_2 & \cdots & b_{s-1} & b_s \end{array} \tag{28}$$

というものが使われている。ここで、$\boldsymbol{b}^T \equiv (b_1, \cdots, b_s)$ と定義しておく。

ここで、誤差の評価を考える。ある点 x_n においての真の値を $y_{\text{true}}(x_n)$ とする。このとき、p 次の Runge-Kutta の結果 $y(x_n)$ と q 次の Runge-Kutta の結果 $\hat{y}(x_n)$ は

$$y(x_n) = y_{\text{true}}(x_n) + O(h^{p+1}) \tag{29}$$
$$\hat{y}(x_n) = y_{\text{true}}(x_n) + O(h^{q+1}) \tag{30}$$

と書く事ができる。もし、$q > p$ であれば、その差は

$$\hat{y}(x_n) - y(x_n) = y_{\text{true}}(x_n) - y(x_n) + O(h^{p+2}) \tag{31}$$

となり、真の値からのずれ $y_{\mathrm{true}}(x_n)-y_n(x_n)$ を見積もる事ができる。従って、同じ段で二種類の次数の Runge-Kutta を作る事ができれば、誤差を推定できる。そこで、$\boldsymbol{b}^T$ の下に $\hat{\boldsymbol{b}}^T$ とその差 $\boldsymbol{d}^T = \hat{\boldsymbol{b}}^T - \boldsymbol{b}^T$ を書き、

$$\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & & \ddots & & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s \\
 & \hat{b}_1 & \hat{b}_2 & \cdots & \hat{b}_{s-1} & \hat{b}_s \\
\hline
 & d_1 & d_2 & \cdots & d_{s-1} & d_s
\end{array} \tag{32}$$

という tableau を用いて、誤差の評価を行う。

2 Verner の方法による 8 段 7 次の Runge-Kutta 法

ここでは、Verner の方法による 8 段 7 次の Runge-Kutta 法の tableau を示す。このときの tableau は

$$\begin{array}{c|cccccccc}
0 & & & & & & & & \\
\frac{1}{18} & \frac{1}{18} & & & & & & & \\
\frac{1}{6} & -\frac{1}{12} & \frac{1}{4} & & & & & & \\
\frac{2}{9} & -\frac{2}{81} & \frac{4}{27} & \frac{8}{81} & & & & & \\
\frac{2}{3} & \frac{40}{33} & -\frac{4}{11} & -\frac{56}{11} & \frac{54}{11} & & & & \\
1 & -\frac{369}{73} & \frac{72}{73} & \frac{5380}{219} & -\frac{12285}{584} & \frac{2695}{1752} & & & \\
\frac{8}{9} & -\frac{8716}{891} & \frac{656}{297} & \frac{39520}{891} & -\frac{416}{11} & \frac{52}{27} & 0 & & \\
1 & \frac{3015}{256} & -\frac{9}{4} & -\frac{4219}{78} & \frac{5985}{128} & -\frac{539}{384} & 0 & \frac{693}{3328} & \\
\hline
 & \frac{3}{80} & 0 & \frac{4}{25} & \frac{243}{1120} & \frac{77}{160} & \frac{73}{700} & 0 & 0 \\
 & \frac{57}{640} & 0 & -\frac{16}{65} & \frac{1377}{2240} & \frac{121}{320} & 0 & \frac{891}{8320} & \frac{2}{32} \\
\hline
 & \frac{33}{640} & 0 & -\frac{132}{325} & \frac{891}{2240} & -\frac{33}{320} & -\frac{73}{700} & \frac{891}{8320} & \frac{2}{32}
\end{array} \tag{33}$$

である。ここで、$\boldsymbol{b}^T$ は 5 段 5 次、$\hat{\boldsymbol{b}}^T$ は 8 段 7 次の Runge-Kutta 法を与える係数である。

このテーブルを使っての誤差評価を行い、その誤差評価からステップ幅を決めることを考える。真の値 $y_{\mathrm{true}}(x_{n+1})$ からのずれは

$$\hat{y}(x_{n+1}) - y_{\mathrm{true}}(x_{n+1}) = \hat{C}(x')h^8 \tag{34}$$

$$y(x_{n+1}) - y_{\mathrm{true}}(x_{n+1}) = C(x'')h^6 \tag{35}$$

である。よって h が十分に小さいときは、

$$||\hat{y}(x_{n+1}) - y(x_{n+1})|| \sim C(x'')h^6,\ \ x_n \leq x'' \leq x_{n+1} \tag{36}$$

である。これは、誤差が大きい方にそろう為である。よって、

$$C(x'') \sim \frac{||\hat{y}(x_{n+1}) - y(x_{n+1})||}{h^6} \tag{37}$$

である。一方、次のステップにおいて、ステップ幅を h' とする。このとき、

$$C(x''') \sim \frac{||\hat{y}(x_{n+2}) - y(x_{n+2})||}{h'^6},\ \ x_{n+1} \leq x''' \leq x_{n+2} \tag{38}$$

である。h が十分小さいとき、両者は等しい：

$$C(x''') \sim C(x'') \tag{39}$$

とする。さて、次のステップでの絶対許容誤差 ϵ を

$$||\hat{y}(x_{n+2}) - y(x_{n+2})|| = C(x''')h'^6 \leq \epsilon \tag{40}$$

としたい。上の式の関係を使うと、

$$C(x'') \sim C(x''') \leq \frac{\epsilon}{h'^6} \tag{41}$$

$$\frac{||\hat{y}(x_{n+1}) - y(x_{n+1})||}{h^6} \sim \frac{||\hat{y}(x_{n+2}) - y(x_{n+2})||}{h'^6} \leq \frac{\epsilon}{h'^6} \tag{42}$$

$$\frac{||\hat{y}(x_{n+1}) - y(x_{n+1})||}{h^6} \leq \frac{\epsilon}{h'^6} \tag{43}$$

なので、

$$h' \leq h\left[\frac{\epsilon}{||\hat{y}(x_{n+1}) - y(x_{n+1})||}\right]^{1/6} \tag{44}$$

となる。よって、新しい h' として、

$$h' = ah\left[\frac{\epsilon}{||\hat{y}(x_{n+1}) - y(x_{n+1})||}\right]^{1/6} \tag{45}$$

を採用する。ここで、a は通常 $a = 0.9$ とおく。なお、$||\hat{y}(x_{n+1}) - y(x_{n+1})||$ は

$$\hat{y}(x_n + 1) = y(x_n) + h\sum_{i=1}^{s} \hat{b}_i k_i \tag{46}$$

$$\hat{y}(x_n + 1) - y(x_n + 1) = h\sum_{i=1}^{s} g_i k_i \tag{47}$$

より

$$||\hat{y}(x_{n+1}) - y(x_{n+1})|| = ||h\sum_{i=1}^{s} g_i k_i|| \tag{48}$$

である。

実用上は、Numerical Recipe 等の可変刻み Runge-Kutta のプログラムに、tableau として上のものを用いれば、簡単に、8 段 7 次の可変刻み Runge-Kutta プログラムを作ることができる。なお、4 変数の一階微分方程式である spin-triplet の Riccati 方程式に対してこの方法を適用した場合、渦糸系の場合には 10 倍近く高速になった。

3　Riccati 方程式の解法

準古典理論で出てくる Riccati 方程式は

$$v_F\frac{da}{ds} = -2\omega_n a - \Delta^* a^2 + \Delta \tag{49}$$

のような形をしている。よって、

$$f(s,a) = (-2\omega_n a - \Delta^* a^2 + \Delta)/v_F \tag{50}$$

とおけば Runge-Kutta 法が使える。このとき、一般には、最初に解くときに初期値 $a(s_0)$ が必要であるが、このタイプの Riccati 方程式の場合、s_0 が求めたい点 s よりも十分に（コヒーレンス長よりもはるかに）離れていれば、初期値をどうとっても数値的には同じ結果が得られることが示されている [3]。準古典理論の Riccati 方程式に関しては、$a(s_0) = 0$ を初期値として解けばよい。

参考文献

[1] J. H. Verner, “Explicit Runge-Kutta Methods with Estimates of the Local Truncation Error”, SIAM J. Numer. Anal. 15-4 (1978). 772-790

[2] J. C. Butcher, “Numerical Methods for Ordinary Differential Equations”, Wiley, (2003).

[3] Y. Nagai, K. Tanaka, and N. Hayashi, “Quasiclassical numerical method for mesoscopic superconductors: bound states in a circular d-wave island with a single vortex”, Phys. Rev. B **86** (2012) 094526. (e-print: arXiv:1202.2661)