UDC 517.91 : 518.5 160

可変ピッチ法による微分方程式の数値計算

富田 昇
石田博章
松本勝也

Numerical Solution of Differential Equations by Using Variable Pitch Method

Noboru Tomita
Hiroaki Ishida
Katsuya Matsumoto

We devised variable pitch method in order to economise the time required to solve numerically the differential equations by the digital computer without making the sacrifice of accuracy.

The differential equations are solved numerically often by reducing difference equations. The various kinds of these methods developed so far were constant pitch methods. Euler's method and Runge-Kutta method are also constant pitch methods, and the orders of the error produced at every one pitch using by these methods change variously.

On the variable pitch method, the magnitude of the pitch is determined at every one pitch so that the error may not over the range previously determined. In place of the true error which may not be known, we use the difference between two resultant values which are calculated by different way each other.

1. まえがき

技術計算用の電子計算機が当社に導入されたのを契機として，反応装置や熱交換器の解析などで，微分方程式を数値計算で解く機会は，今後飛躍的に増加することが予想される。特に動特性を問題にする場合には，往々にして偏微分方程式を数値計算する必要を生ずる。しかしながら偏微分方程式を電子計算機（ここでは特にデジタル型を前提とする）で解くことは，なかなかやっかいであって，短時間でやろうとすれば精度が悪く，精度よく計算しようとすれば時間がかかりすぎるということが多い。このような欠点を緩和し，短時間に精度よく計算することを狙いとして1ピッチごとに誤差を評価し，これが所定の範囲内になるようにピッチを変えて計算する方法（これを仮りに可変ピッチ法と名ずける）を考案したので，その概要を報告する。

2. 可変ピッチ法の基礎概念

微分方程式を電子計算機で数値計算するには，これを差分に直して解くのが普通であって，オイラー法，ルンゲ・クッタ法，ミルン法等種々のものがあるが，これらはいずれもピッチを固定した方式である。オイラー法やルンゲ・クッタ法で気のつくことは1ピッチで生ずる計算誤差が必ずしも一定でなく，解曲線の性質によっては著しく変化する。解の全体としての精度は1ピッチで生ずる計算誤差の最大値によって左右されることが多いから，この誤差はなるべく均一化されることが望ましい。可変ピッチ法は上記の目的を達成すべく，誤差がある値より大きくなるとピッチを小さくして誤差の増大をさけ，逆に誤差が別のある値より小さくなるとピッチを大きくして計算時間を短縮する。

3. 基礎方式の可変ピッチ法

固定ピッチの方式で最も簡単なオイラ法は1次近似であって，誤差評価の仕方がないため，これをそのまま可変ピッチ法に適用することはできない。そこで，可変ピッチ法の最も簡単な方法とし

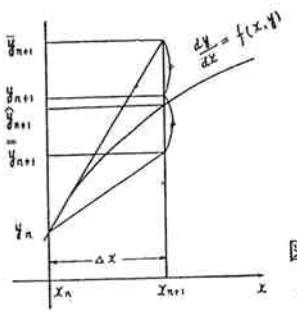

図 1

て，2次近似の方法を採用する。簡単のために1変数で考えることとし，与式を次のようにとる。

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

Δx をピッチとするとき，オイラ法と同様にして，先ず

$$\bar{y}_{n+1} = y_n + \Delta x \cdot f(x_n, y_n) \tag{2}$$

を計算し，次いで1ピッチ先での勾配を利用して

$$\bar{\bar{y}}_{n+1} = y_n + \Delta x \cdot f(x_{n+1}, y_{n+1}) \tag{3}$$

を計算する。もし区間 (x_n, x_{n+1}) の間で解曲線に変曲点が存在しないならば真の値は $\bar{y}_{n+1}$ と $\bar{\bar{y}}_{n+1}$ との間にあるはずである。（実際には(3)式の右辺第2項内にある y_{n+1} の値は，これから求めようとする値であるから代りに $\bar{y}_{n+1}$ を用いる）そこで求める値として

$$y_{n+1} = (\bar{y}_{n+1} + \bar{\bar{y}}_{n+1})/2 \tag{4}$$

を用い，誤差評価としては次の式を用いる。

$$\delta y_{n+1} = |\bar{y}_{n+1} - \bar{\bar{y}}_{n+1}| \tag{5}$$

今仮りに与えられた微分方程式が

$$\frac{dy}{dx} = ax + b \tag{6}$$

とすると，解析解は

$$y = \frac{1}{2}ax^2 + bx + c \tag{7}$$

であるから

$$y_{n+1} = y_n + \Delta x\left(ax_n + b + \frac{1}{2}\Delta x \cdot a\right) \tag{8}$$

となって(4)式の結果と一致し，Δx^3 のオーダの誤差を有することが分る。

4. ルンゲ・クッタ方式の可変ピッチ法

前項の方式では精度よく計算するためには，Δx をかなり小さくとる必要がある。1ピッチでの精度を上げるために，ルンゲ・クッタ方式の可変ピッチ法が考えられ，この場合には Δx^5 のオーダの誤差を有する。計算方式はルンゲ・クッタ法と全く同じである。すなわち

$$\left.\begin{aligned}
\Delta y_n^1 &= \Delta x \cdot f(x_n, y_n) \\
\Delta y_n^2 &= \Delta x \cdot f(x_n + \Delta x/2, y_n + \Delta y_n^1/2) \\
\Delta y_n^3 &= \Delta x \cdot f(x_n + \Delta x/2, y_n + \Delta y_n^2/2) \\
\Delta y_n^4 &= \Delta x \cdot f(x_n + \Delta x, y_n + \Delta y_n^3) \\
y_{n+1} &= y_n + \frac{1}{6}(\Delta y_n^1 + 2\Delta y_n^2 + 2\Delta y_n^3 + \Delta y_n^4)
\end{aligned}\right\} \tag{9}$$

誤差評価の方法としては種々のものが考えられるが，現在のところ，最も簡単な手法として

$$\delta y_{n+1} = |\Delta y_n^1 - \Delta y_n^4| \tag{10}$$

$$\delta y_{n+1} = |\Delta y_n^2 - \Delta y_n^3| \tag{11}$$

の2通りが考えられる。どちらがよいかは今後の問題である。

5. 誤差評価の考え方

3節で述べたように，図1において，区間 (x_n, x_{n+1}) で変曲点がないならば，x_n および x_{n+1} における勾配はそれぞれこの区間における最大および最小であるから真の解 $\hat{y}_{n+1}$ は必ず $\bar{y}_{n+1}$ と $\bar{\bar{y}}_{n+1}$ との間にある。y_{n+1} は $\bar{y}_{n+1}$ と $\bar{\bar{y}}_{n+1}$ との中点であるから，計算誤差 $y_{n+1} - \hat{y}_{n+1}$ は常に δy_{n+1} の半分より小さい。実際には，この区間で曲率の変化があまり大きくないものとすれば，誤差は δy_{n+1} よりは，はるかに小さいはずである。今全長を l，所要精度を ε，Δx の出発値を Δx_s とし，適当な係数を a とするとき，

$$\delta y_{n+1} \leqslant a \cdot \Delta x_s \cdot \varepsilon / l \tag{12}$$

となるように Δx を定める。

ルンゲ・クッタ方式の場合で誤差評価として(10)式を採用した場合には，上と全く同様のことがいえる。(11)式を採用した場合には上のような保証はないが，実用上はさして支障がないようである。ただどの方式を採用するかによって a の値も大きく変る。a の値をいくらに採るかは重要な問題であって，これの選択をあやまると，時間がかかりすぎたり，精度が悪くなったりするが，今のところ a の最適値については，まだはっきりした結論は得られていない。

6. プログラムの実例

可変ピッチ法のプログラムを作成し，FACOM 231に掛けてみたので，その概要をのべる。プログラムは下記の通りである。

Problem NO. 6-605 F

```
begin comment variable pitch method. ;
integer n ; Readinteger(n) ;
begin integer I, J, i, j, m, p, pp ;
real dx, l, a, d6, fc, dxll, dxul ;
array Y, YS, F[0 : n], FC[1 : 3], DY[0 : n, 1 : 4],
  EY, DDY, EYDX[1 : n] ;
procedure FF(Y) ; array Y ; F[1]:=a-a*Y[1] ;
procedure TL ; begin CRLF(2) ; Space(21) ;
  Printstring('X') ; Space(19) ;
Printstring('EXP (CAL)') ; Space(16) ; Printstring
  ('EXP (ANA)') ; Space(16) ;
Printstring('EXP (DIF)') ; CRLF(2) end ;
procedure PR ; begin real ex ; Space(16) ; Printreal
  (YS[0], 7) ; Printfix(YS[1], 10, 9) ;
ex := 1.0-exp(-a*YS[0]) ; Printfix(ex, 13, 9) ;
  Printfix(YS[1]-ex, 13, 9) ; CRLF end ;
```

```
procedure DT ; begin CRLF(2) ; Space(10) ;
  Printstring('A =') ; Printfix(a, 4, 2) ; Space(7) ;
Printstring('L =') ; Printfix( l, 3, 4) ; Space(6) ;
  Printstring('DX =') ; Printfix(dx, 3, 6) ;
Space(4) ; Printstring('XS =') ; Printfix(YS[0],
  3, 5) ; Space(3) ; Printstring('YS =') ;
Printfix(YS[1], 3, 5) ; CRLF(2) ; Space(10) ;
  Printstring('PP =') ; Printinteger(pp) ;
ASW(1, BA) ; Space(10) ; Printstring('FC = ') ;
  Printreal(fc, 4) ; Space(4) ;
Printstring('EY =') ; Printreal(EY[1], 4) ;
  Space(4) ; Printstring('DXUL= ') ;
Printreal(dxul, 4) ; Printstring(' DXLL= ') ;
  Printreal(dxll, 4) ; BA : end ;
procedure ST ; begin LFEED ; CRLF(3) ; Space(20)
  ; ASW(1, BC) ; Printstring ('VARIABLE') ; go
  to BD ; BC : Printstring('CONSTANT') ; BD :
  Printstring(' PITCH METHOD. (' ) ;
ASW(2,BE) ; Printstring('RUNGE-KUTTA)') ; go
  to BF ; BE : Printstring ('FUNDAMENTAL)') ;
BF : Space(10) ; Printstring('(EXPONENTIAL
  CURVE)') ; Space(21) ; Printstring('PAGE') ;
Printinteger(p) end ;
comment end of declaration ;
Readreal(a) ; Readreal(l) ; Readreal(dx) ; Readarray
  (YS) ; Readinteger(pp) ;
ASW (1,BB) ; Readreal(fc) ; Readarray(EY) ;
  Readreal (dxul) ; Readreal(dxll) ; BB :
j:=m:=0 ; FC[1]:=FC[2]:=0.5 ; F[0]:=FC[3]:=1.0 ;
  p:=1 ; d6:=1.0/6.0 ;
ST ; DT ; TL ; dxul:=0.5*dxul ; dxll:=dxll+dxll ;
ASW(1,BG) ; for J:=1 step 1 until n do EYDX[J]
  :=fc*EY[J]*dx/l ; BG : PR ;
if YS[0] >=1 then go to BH ; j:=j+1 ; m:=0 ;
if j >=50 then begin p:=p+1 ; LFEED ; CRLF(3)
  ; Space(108) ; Printstring('PAGE') ;
Printinteger(p) ; CRLF(4) ; TL ; j:=0 end ;
BI : FF(YS) ;
ASW(2,BK) ; for I :=1, 2, 3 do begin for J := 0
  step 1 until n do
begin DY[J,I] :=dx*F [J] ; Y[J] := FC[I]*DY[J,I]
  +YS[J] end ; FF(Y) end ;
for J := 0 step 1 until n do DY[J,4] := dx*F[J] ;
ASW(1,BM) ; if i = 1 then begin i := 0 ; go to
  BL end ;
for J := 1 step 1 until n do begin ASW(3,BN) ;
  DDY[J] := abs(DY[J,1]-DY[J,4]) ;
go to BP ; BN : DDY[J] := abs (DY[J,2]-DY[J,3]) ;
  BP : end ; BM : go to BQ ; BK :
for J := 0 step 1 until n do begin DY[J,1] :=
  dx*F[J] ; Y[J] := DY[J,1]+YS[J] end ; FF(Y) ;
for J := 0 step 1 until n do DY[J,2] := dx*F[J] ;
ASW(1,BQ) ; if i = 1 then begin i := 0 ; go to BL
  end ;
for J := 1 step 1 until n do DDY[J] := abs(DY
  [J,1]-DY[J,2]) ; BQ : ASW(1,BL) ;
for J := 1 step 1 until n do if DDY[J] >=4.0*
  EYDX[J] then go to BR ; BL :
YS[0] := YS[0]+dx ; m := m+1 ; ASW(2,BS) ;
  for J := 1 step 1 until n do
YS[J] := YS[J]+(DY[J,1]+DY[J,4]+2.0*(DY
  [J,2]+DY[J,3]))*d6 ; go to BT ; BS :
for J := 1 step 1 until n do YS[J] := YS[J]+(DY
  [J,1]+DY[J,2])*0.5 ; BT : ASW(1,BU) ;
for J := 1 step 1 until n do if DDY[J] >=EYDX
  [J] then go to BR ;
for J := 1 step 1 until n do if 2.0*DDY[J] >=
  EYDX[J] then go to BU ;
if dx <= dxul then dx := dx+dx ; go to BU ; BR
  : if dx>= dxll then dx := 0.5*dx else i := 1 ;
BU : if m < pp then go to BI ; m := 0 ; go to BG
  ; BH : Space(100) ; Printstring ('END') ;
CRLF end end
```

このプログラムは ASW 1 を on にするか off にするかによって固定ピッチか可変ピッチかいずれかを選択できるようになっており，また ASW 2 を on にするか off にするかによって基礎方式かルンゲ・クッタ方式かを選択できるようになっている。従ってこれらの組合せによって4通りが可能である。更にルンゲ・クッタ方式の可変ピッチ法の場合には ASW 3 をon にするか off にするかにより，誤差評価に(11)式を用いるか(10)式を用いるかが選択できる。またこのプログラムでは微分方程式

$$\frac{dy}{dx}=k(1-y) \tag{13}$$

を初期条件 (y=0, x=0) で解き，これと厳密解とを比較するようになっているが，procedure の部分を適宜変更することにより，任意の常微分方程式を取扱うことができる。

このプログラムで(13)式の k の値を 100 とし，全長を 1.0 で1ピッチごとにプリントし，誤差の最大値が±0.0001程度となるように計算させたところ次のような結果

が得られた。

表 1

方式	基礎方式		ルンゲ・クッタ方式		
ピッチ	可変	固定	可変		固定
誤差評価	—	—	(10)式	(11)式	—
所要時間	3分41秒	45分30秒	3分22秒	2分43秒	8分53秒
基本ピッチ	0.004	0.0005	0.004	0.004	0.004
最大誤差 ($\times 10^{-4}$)	1.368	1.591	0.880	1.098	1.077
同上の x	0.0535	0.01	0.083	0.081	0.012
最大誤差変化 ($\times 10^{-4}$)	0.523	0.205	0.742	0.844	0.799
同上の x	0.1035	0.0005	0.083	0.081	0.004
印刷行数	144	2001	97	78	251
係数	2000	—	10000	4000	—
最大ピッチ	0.016	—	0.016	0.016	—
最小ピッチ	0.00025	—	0.0005	0.001	—
上限ピッチ	0.02	—	0.02	0.02	—
下限ピッチ	10^{-6}	—	10^{-6}	10^{-6}	—

表1より分ることを列挙すれば,

① 可変ピッチの方が所要時間が短い。

② ルンゲ・クッタ方式の方が所要時間が短い。

③ ルンゲ・クッタの可変ピッチ法では誤差評価に(11)式を用いた方が所要時間が短い。

④ 固定ピッチの場合，1ピッチで生ずる誤差は初めは大きく，次第に小さくなるのに反し，可変ピッチでは，初めは小さく，次第に大きくなり，ある値をすぎるとまた次第に小さくなる。

⑤ 可変ピッチの場合，いずれも最大ピッチは上限ピッチでおさえられている。

また表1以外の計算例から分ったことを列挙すると,

⑥ 上限ピッチを0.04にすると，すなわち最大ピッチを0.032にすると計算値が振動を起し，1に収束しなくなる。このことはどの可変ピッチ法においても同様である。

⑦ kを1とし，xに関する諸定数を100倍すれば全く同じ結果が得られる。

⑧ (12)式での Δx_s の代りに Δx を用いるとこの場合，ピッチの変化がはげしすぎてうまくゆかない。

⑨ $\delta y_{n+1}=|\Delta y_n{}^1-\Delta y_n{}^4\pm 2(\Delta y_n{}^2-\Delta y_n{}^3)|$ を(10)式の代りに誤差評価として用いても結果は大して変りない。

7. 今後に残された問題

前節の例でわかるように，区間内で曲率が大きく変るような解曲線をもつ微分方程式については可変ピッチ法が有効である。また，この例では係数aの値は2000～10000となったが，これは式や精度が変ればそれに伴って変ることが予想される。この関係をつかむことが残された問題の第1である。

次に，すでによく用いられている方法で1ピッチごとに誤差評価をするものとしてミルンの方法があるが，これと可変ピッチ法との長短の比較が第2の問題として残る。

ところで，まえがきで指摘したように偏微分方程式の数値計算は非常に時間がかかる。FACOM 231では，数時間はあたりまえで場合によっては数日はかかるものとみなければならない。従って可変ピッチ法が偏微分方程式に適用できるかどうかが今後に残された第3の，そして最も大きな問題である。