第17卷 第4期
1996年 12月
郑州工业大学学报
Journal of Zhengzhou University of Technology
Vol.17 No.4.
Dec. 1996

对Runge－Kutta模型传统应用方法的改进及计算机验证

王文义
（郑州工业大学计自系）

摘 要：本文提出的改进型Runge－Kutta模型应用方法，不仅能够满足对计算精度的要求，而且更重要的是可以大大地提高计算速度。

关键词：Runge－Kutta模型；冗余；重复遍历。
中图分类号：TP399

在许多数学公式的理论推导中，如对微分，积分及无穷级数的推导等，都是通过极限过程来给予定义的，然而当今最现代化的计算工具——计算机对“极限”一词却是无能为力。因为计算机只能按照人们提供的模型—算法—程序的固定程式机械地去完成有限次的算术运算和逻辑运算。

真正理想化的模型是不存在的，可以说迄今凡涉及数值计算的几乎所有数学模型充其量都只是近似的、在一定范围内可行的替代品。正因为此，对应用者来说就几乎都要在精度和速度二者之间进行无奈的选择，以图找到一个折衷的方法，使得实际的需要求得以满足。因此，可以说：既保证精度、计算速度又快的算法是应用者追求的目标。使用Runge－Kutta模型求解微分方程也同样存在上述问题。

1 Runge－Kutta模型

给定微分方程 $\begin{cases}\dfrac{dy}{dx}=f(x,y)\\ y(x_0)=y_0\end{cases}$

其特解的形式为$y=f(x)$。在平面座标中该特解是一条曲线。既然计算机无法获得特解，但根据数值算法的特点，在某个给定范围内，以一定的步距（步距不一定相等），从已知点(x_0,y_0)出发，设法得到一组座标点，如$(x_1,y_1),(x_2,y_2),\cdots,(x_n,y_n)$，用这些座标点去近似的代表所求的$y=f(x)$还是可以的。

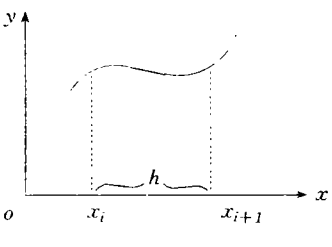

图1

如图(1)所示，若(x_i,x_{i+1})是$y=f(x)$曲线的某个区间，改进的尤拉格式可写成如下形式：

$$y_{i+1}=y_i+\frac{1}{2}(k_1+k_2)\cdot h$$

其中：$k_1=f(x_i,y_i)$

$k_2=f(x_{i+1},y_i+k_1\cdot h)$；

收稿日期：1995－12－20

这个形式可看成是把 x_i 和 x_{i+1} 两点的斜率 k_1 和 k_2 取算术平均作为平均斜率的近似值，即式中的 $\frac{1}{2}(k_1+k_2)$，而 $X_7 i+1$ 处的斜率是通过已知信息 y_i 予报的。同理，若设法在 (x_i, x_{i+1}) 区间内多予报几个点的斜率值，然后将它们加权平均作为平均斜率的近似值，就有可能构造出更为精确的计算格式。这就是 Runge－Kutta 方法的基本思路。根据这个思路，可以得到截断误差为 $O(h^5)$ 的四阶 Runge－Kutta 模型（推导与证明从略）：

$$y_{i+1}=y_i+\frac{1}{6}(k_1+2k_2+2k_3+k_4)\cdot h$$

其中：$k_1=f(x_i,y_i)$；　　$k_2=f(x_{i+\frac{1}{2}},y_i+\frac{k_1}{2}\cdot h)$；

$k_3=f(x_{i+\frac{1}{2}},y_i+\frac{k_2}{2}\cdot h)$；　　$k_4=f(x_{i+1},y_i+\frac{k_3}{2}\cdot h)$

式中的 $\frac{1}{6}(k_1+2k_2+2k_3+k_4)$ 即表示区间 (x_i,x_{i+1}) 内平均斜率的近似值。

2　Runge－Kutta 模型的传统用法及改进

对给定的微分方程和初始条件，对某区间可使用上述四阶 Runge－Kutta 模型求出 y_{i+1} 来，根据一定的精度要求，实际应用中常常把该区间划分成更小的区间，然后再对每个小区间递推的使用 Runge－Kutta 模型继而求出符合精度要求的 y_{i+1}。

2.1　传统的应用方法

虽然四阶 Runge－Kutta 方法的模型是一定的，但使用该模型的方法却不尽相同。我们的目的是用上述模型求出描述微分方程特解 $y=f(x)$ 在给定范围内的一组离散的座标点（函数值）。Runge－Kutta 模型的传统应用方法是：在座标点给出后，如 $x_0,x_1,x_2,\cdots,x_i,x_n$，在具体计算某一区间，如 (x_i,x_j) 的 y_j 时，都普遍使用了变步长的递推方法，而在计算除 x_1 之外的其它各后继点的函数值时，却都不例外的又把计算的控制回退到了初始点 x_0 处，这样做的结果势必造成了计算步骤的冗余问题，同时也一定会影响到计算的整体速度。这种应用方法的源程序可列出如下。

2.2　改进应用方法的可行性

```
PROGRAM MAIN
write(*,*)'dy/dx=***,xo=***,
y0=***
x0=0.0
y0=1.0
write(*,200)x0,y0
do 100 i=1,10
b1=0.1*i
a1=rgkt(x0,y0,b1,1000,1.e-5)
100   write(*,200)b1,a1
200   format(1x,'x=',f4.2,'y=',f11.7)
end
function rgkt(xa,ya,xb,m,d)
real k0,k1,k2,k3
call init(x0,xa,y0,ya,x1,xb)
h=x1-x0
      call xs(k0,k1,k2,k3,x0,y0,h)
rgkt=y0+(k0+2.*k1+2.*k2+
k3)/6.0
n=2
10  call init(x0,xa,y0,ya,x1,xb)
h=(x1-x0)/n
do 20 i=1,n
call xs(k0,k1,k2,k3,x0,y0,h)
y1=y0+(k0+2.*k1+2.*k2+k3)/
6.0
```

```
    x0=x0+h
20  y0=y1
    if(abs((rgkt-y1)/rgkt).le.d) then
    else
    rgkt=y1
    if (n.gt.m) then
    write(*,*)'n=',n
    write(*,*)'The partitons n>1000! '
    return
    else
    n=2*n
    goto 10
    endif
    endif
    write(*,*)'n=',n
```

```
    end
    subroutine init(x0,xa,y0,ya,x1,xb)
    x0=xa
    y0=ya
    x1=xb
    end
    subroutine xs(k0,k1,k2,k3,x0,y0,h)
    real k0,k1,k2,k3
    f(x,y)=*********
    k0=h*f(x0,y0)
    k1=h*f(x0+h/2.,y0+k0/2.)
    k2=h*f(x0+h/2.,y0+k1/2.)
    k3=h*f(x0+h,y0+k2)
    end
```

传统应用方法可通过图(2)加以描述。

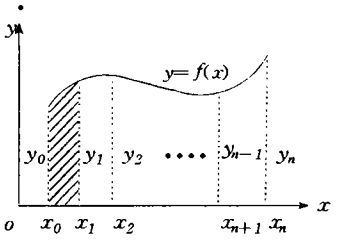

图2

图中所示阴影部分,传统应用方法除了在计算 y_1 时要遍历一次外,在计算其它任何后继点的函数值时,由于都要以初始点(x_0,y_0)作为基准,故也都要重新对阴影进行遍历,也就是说采用该方法时,后继点函数值的计算没有有效的利用其前面一点已算出的函数值,故当最终 y_n 被计算出来以后,上述阴影部分被重复遍历的次数达 n−1 次,加之在计算中对该阴影右边部分各区间的重复计算遍历次数 n−2,n−3,……,1,总的重复计算次数为 $\sum_{i=1}^{n-1}(n-i)$,这就使得计算效率大为降低。

为了提高速度,我们可以设法剔除上述的计算冗余部分,即在计算 x_j 点处的函数值 y_j 时,可直接把前面计算出的 y_{j-1} 作为初始值考虑而不必再舍近求远的回退到 y_0 处进行计算,因为从道理上讲,对 y_{j-1} 的计算追溯起来也是从 y_0 出发计算出来的,所以说从 y_0 出发计算出的 y_j 和从 y_{j-1} 计算出的 y_j 间一定存在着密切的关系。基于上述认识,我们把前面的源程序加以改进,变成下面的程序(子程序从略)。问题是经过如此改进,速度是否真的会提高,精度能否得以保证?

```
PROGRAM MAIN
    write(*.*)'dy/dx=***,x0=***,y0=***'
      x0=0.0
      y0=1.0
      write(*,200)x0,y0
      do 100 i=1,10
      b1=0.1*i
```

```
      a1=rgkt(x0,y0,b1,1000,1.0 e-6)
      write( * ,200) b1,a1
      x0=b1
100   y0=a1
200   format(1×,'×=',f4.2,' y=',f11.7)
      end
```

3　计算机验证

笔者对上述改进方法在计算机上进行了大量的计算验证，因篇幅所限，仅选出四个已知初始条件和特解的微分方程，把传统的方法、改进的方法和由已知特解得出的函数值 $y_{(传)}$、$y_{(改)}$、$y_{(精)}$，以及速度（由改进方法和传统方法对区间细分个数 g/c 模拟）列成下表。为统一期间，假定四例各函数点的间距（步长）都定为 0.1 且只计算十个点的函数值，并把各个微分方程和初始条件作为表头列出。

	1:dy/dx=y−2x/y,x0=0,y0=1				2:dy/dx=e^(2x−y),x0=0,y0=0			
x	y(改)	y(传)	y(精)	g/c	y(改)	y(传)	y(精)	g/c
0.0	1.000000	1.000000	1.000000		.0000000	.0000000	.0000000	
0.1	1.095446	1.095446	1.095445	2/2	.1049917	.1049919	.0149916	4/2
0.2	1.183217	1.183217	1.183216	2/4	.2198681	.2198686	.2198680	4/4
0.3	1.264912	1.264917	1.264911	2/4	.3443411	.3443410	.3443407	2/8
0.4	1.341642	1.341642	1.341641	2/8	.4779541	.4779545	.4779535	2/8
0.5	1.414215	1.414219	1.414214	2/8	.6201154	.6201175	.6201145	2/8
0.6	1.483242	1.483253	1.483240	2/8	.7701364	.7701427	.7701353	2/8
0.7	1.549196	1.549195	1.549193	2/16	.9272715	.9272711	.927202	2/16
0.8	1.612455	1.612455	1.612452	2/16	1.090755	1.090755	1.090754	2/16
0.9	1.673324	1.673328	1.673320	2/16	1.259832	1.259833	1.259830	2/16
1.0	1.732056	1.732064	1.732051	2/16	1.433782	1.433785	1.433781	2/16

	3:dy/dx=(1−xy)/(1−x^2),x0=0,y0=1				4:dy/dx=sinxcosx−ycosx,x0=0,y0=1			
x	y(改)	y(传)	y(精)	g/c	y(改)	y(传)	y(精)	g/c
0.0	1.000000	1.000000	1.000000		1.000000	1.000000	1.000000	
0.1	1.094987	1.0949870	1.0949870	2/2	.9098098	.9098098	.9098098	2/2
0.2	1.179796	1.1797940	1.1797960	2/2	.8383114	.8383147	.8383112	2/2
0.3	1.253939	1.2539380	1.2539390	2/4	.7838092	.7838103	.7838089	2/4
0.4	1.316514	1.3165060	1.3165150	2/4	.7443202	.7443248	.7443200	2/4
0.5	1.366024	1.3660220	1.3660250	2/8	.7177038	.7177042	.7177035	2/8
0.6	1.399999	1.3999990	1.4000000	4/16	.7017695	.7017709	.7017693	2/8
0.7	1.414141	1.4141380	1.4141430	4/16	.6943643	.6943671	.6943640	2/8
0.8	1.399999	1.3999970	1.4000000	8/32	.6934382	.6934434	.6934379	2/8
0.9	1.335889	1.3358870	1.3358900	16/64	.6970942	.6970944	.6970938	2/16

1.0	1.000000	.7036232	.7036238	.7036229	2/16

4 结束语

由表中四例的计算结果可以看出，使用unge－Kutta模型的改进应用方法，不仅可使计算结果比传统应用方法更接近精确解，而且更重要的是，它还大大的提高了计算速度。根据笔者在计算机上所作的大量算例的统计结果表明，采用改进方法比采用传统方法的模拟速度要平均提高4倍以上，而且若精度再适当增大，这个相对倍数还会进一步提高（当然，精度在人们实际的计算中，并非都需要那么高）。正是由于改进应用方法所具有的上述特点，使得它具有十分显著的实际意义。

参考文献

1 中科院沈阳计算所等. 电子计算机常用算法. 科学出版社. 1986
2 徐士良. FORTRAN 常用算法程序集. 清华大学出版社. 1993
3 谭浩强等. FORTRAN 77 结构化程序设计. 清华大学出版社. 1994
4 华中工学院. 算法语言. 计算方法. 人民教育出版社. 1981

The improvement of the traditional application method of Runge－Kutta model and its verification by computer

Wang Wenyi
(Zhengzhou University of Technology)

Abstract: This paper presents an improvement's application method for Runge－Kutta model. It is able to meet the requirements of the computing precission. It is more important that this method can raise the computing speed considerably.

Keywords: Runge－Kutta model. Redundancy. Repeat traversal.