Chebyshev technique in the numerical solution of Delay Differential Equations

Madeha shaltagh yousif
University of Technology
Applied Science Department

Abstract

We investigate the stability of a numerical solution of the delay differential equation by using explicit Runge-kutta method, when the delay has been approximated by using Chebyshev polynomial instead of using Lagrange and Hirmate interpellation polynomials.

1. Introduction

The model problem Delay Differential Equation which we considered, has the forms

$$y'(t) = f(t, y(t), y(t - d(t, y(t)))). \quad \ldots (1)$$

where $d(t, y(t)) \geq 0$ is referred to as the "delay" and the term $t - d(t, y(t))$ is referred to as the "log". In general, the delay is a function of both t and the solution $y(t)$.

In our research, we use various types of Runge-kutta methods to find a numerical solution for equation (1).Previous work on numerical methods for delay differential equations has been done by paul [2,3]

2. Basic Explicit Runge-kutta methods

The s-stages Runge-kutta formula for computing the numerical solution eq.(1) at t_n+h_n is defined by

$$y_{n+1} = y_n + h_n \sum_{i=1}^{s} b_i k_i$$

$$k_i = f(t_n + c_i h_n, y_i) \qquad (2)$$

$$y_i = y_n + h_n \sum_{i=1}^{i-1} a_{ij} k_j , \qquad i=1,2,\ldots,s$$

where a_{ij} ,b_i and c_i are the coefficients of the Runge-Kutta formula and y_n is the associated approximation of $y(t_n)$.

The Runge-Kutta formula is usually represented by the following Table:

C_1	a_{11}	a_{12}	…	a_{1s}
C_2	a_{21}	a_{22}	…	a_{2s}
.	.	.	…	.
.	.	.		.
.	.	.		.
C_s	a_{s1}	a_{s2}	…	a_{ss}
	b_1	b_2	…	b_s

where

$$c_i = \sum_{j=1}^{s} a_{ij} \quad , \sum_{i=1}^{s} b_i = 1$$

Definition:

A Runge-kutta process is said to be explicit if $a_{ij}=0$ for $j \geq i$, and semi-explicit if $a_{ij}=0$ for $j>i$, otherwise,the process is said to be implicit. In this paper ,we are interested only in the second and fourth order Runge- Kutta formulas, which have the following forms and tables

0	0	0
1	1	0
	1/2	1/2

$$k_1 = f(t_n, y_n)$$
$$k_2 = f(t_n + h, y_n + k_1)$$
$$y_{n+1} = y_n + \frac{h}{2}(k_1 + k_2) \qquad n = 1,2,...$$

0	0	0	0	0
1/2	1/2	0	0	0
1/2	0	1/2	0	0
1	0	0	1	0
	1/6	2/6	2/6	1/6

$$k_1 = f(t_n, y_n)$$
$$k_2 = f(t_n + \frac{h}{2}, y_n + \frac{k_1}{2})$$
$$k_3 = f(t_n + \frac{h}{2}, y_n + \frac{k_2}{2})$$
$$y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \qquad n = 1,2,...$$

3. Solution of DDE using explicit Runge-kutta method

To find a numerical solution for for equation.(1), we can classify the problem into two types:-

1. The problem with no delay term (i.e.,d(t,y(t))=0) which is a first order ordinary differential equations, and the analytic solution can be obtained directly.
2. The problem with constant delay.
3. the problem with variable delay.

In case (2) and (3), we need to evaluate y(t-d(t,y(t)) and since Runge-Kutta methods produce an approximate results at mesh points only, and the approximate solution between the mesh points is a matter of interpolation. Besides the other good qualities o f this method, Lagrange or Hermite interpolation, between mesh points provides a numerical solution just accurate

as the solution of mesh points. The more accurate results we get if we use Chebyshev polynomials.[4]

When using an interpolation scheme to evaluate the delay term, one must aware of the following:-

1. The number of solution values to be retained at any one time.
2. Selection of the points for interpolation and which interpolating scheme need to be used and at which order [1,2]

Suppose we have progressed through the i-th point in our mesh, t_i. In order to take the next step, we must compute $f_i = f(t_i, y(t_i), y(u(t_i)))$.

The computation is requiring a value for $y(u(t_j))$ and since $u(t_i)$ will in general be an on-mesh point, some approximation must be needed.

We first find a positive integer j such that $t_{j-1} \prec u(t_i) \leq t_j$.

Since we are dealing with retarded problem $j \leq i$,a sufficient past data need to be available to construct an approximate interpolation for y, which is then be used to evaluate $u(t_i)$ and this value is used to help approximating f_i

It is possible with interpolation schemes using both function and derivative values that an extrapolation can be done.

This occurs if j=i, since in that case, f_i is not known and hence the last full set of data is available only at t_{i-1}. Such situation was infrequent but never the less did occur.

Once j is found we use the past data at mesh points t_{j-3}, t_{j-1} and t_j for a four-point interpolation scheme and the data at t_{j-2}, t_{j-1} and t_j for a three point interpolation scheme. Since the problems were solved using a variety of step sizes h the interpolation routines had to be written to accommodate this scheme

4. Numerical examples

Example (1)

Our first example is

$$y'(t) = -y(t-1) \cdots t \geq 0$$

$$y(o) = 1$$

$$y(t) = 0, -1 \leq t \leq 0$$

Results are given for $t \in [0,6]$ in the following Tables:

Adaptive Runge-kutta with Lagrange interpolation

$\in$	2^{rd} order		4^{th} order	
	GE	ND	GE	ND
10^{-3}	4.310×10^{-3}	160	4.312×10^{-3}	159
10^{-6}	4.101×10^{-6}	432	4.231×10^{-6}	429
10^{-9}	3.810×10^{-9}	882	3.821×10^{-1}	875

Adaptive Runge-kutta with Hermite interpolation

$\in$	2rd order		4th order	
	GE	ND	GE	ND
10^{-3}	4.616×10^{-3}	155	4.1667×10^{-3}	151
10^{-6}	4.561×10^{-6}	450	4.0690×10^{-6}	411
10^{-9}	4.123×10^{-9}	884	3.9737×10^{-4}	871

Adaptive Runge-kutta with Chebyshev interpolation

$\in$	2rd order		4th order	
	GE	ND	GE	ND
10^{-3}	4.124×10^{-3}	150	4.026×10^{-3}	149
10^{-6}	3.156×10^{-6}	405	4.125×10^{-6}	401
10^{-9}	4.321×10^{-9}	870	3871×10^{-9}	872

Note that:-
$\in$=the required error tolerance
ND=number of derivative evaluation
GE=maximum global error

Example (2):-

$$y'(t) = -y(t-1+e^{-t}) + \sin(t-1+e^{-t}) + \cos t, t \geq 0$$
$$y(0) = \sin t, t \leq 0$$

Results are for $t \in [0,10]$ with delay=$1-e^{-t}$ in the following Tables:

Adaptive Runge-kutta with lagrange interpolation

$\in$	2rd order		4th order	
	GE	ND	GE	ND
10^{-3}	6.0532×10^{-4}	4	6.0031×10^{-4}	4
10^{-6}	1.1261×10^{-6}	17	1.046×10^{-6}	17
10^{-9}	4.2561×10^{-10}	730	4.065×10^{-10}	720

Adaptive Runge-kutta with Hermite interpolation

$\in$	2rd order		4th order	
	GE	ND	GE	ND
10^{-3}	6.0053×10^{-6}	3	6.0001×10^{-6}	3
10^{-6}	1.321×10^{-6}	6	1.031×10^{-6}	16
10^{-9}	4.123×10^{-11}	710	4.061×10^{-11}	705

Adaptive Runge-kutta with Chebyshev interpolation

$\in$	2^{rd} order		4^{th} order	
	GE	ND	GE	ND
10^{-3}	$5.312\times10^{--5}$	3	6.120×10^{-5}	3
10^{-6}	1.021×10^{-6}	5	2.0123×10^{-6}	15
10^{-9}	3.914×10^{-10}	702	4.123×10^{-10}	995

5. Step size control:[6]

The step size has chosen as small as necessary to get an accurate approximation. On the other hand it is chosen as big as possible to reach the end of the internal in a few steps as possible. For this reason, and to maintain the stability we can use the following adaptive procedure:

Assume that we are given required the error tolerance ($\in$), and that local truncation error is estimated, by E then:-

a-If $E\geq\in$, then reject the computed solution. For choosing the next step size, we have to take account of the possibility of a point of jump discontinuity in the k-th derivative of the solution, where $k\leq p+1$. hence, we register the point $t^*=t_n+h$ as a possible point of discontinuity and then we choose the next step size as $\frac{h}{2}$.Using the step size $\frac{h}{2}$,the solution and its derivative value at t_n we calculate the next approximation of solution and LTE estimate E and repeat the test in (a).

b-If $E\leq\in$ then we accept me the next approximation to the solution and let the next step mesh point to be $t_n=t_n+h$. For choosing the next step size, we make the following tests:-

i-If $E \geq \frac{\in}{2^{p+1}}$, keep the same step size and go to (ii)

ii-If $E \geq \frac{\in}{2^{p+1}}$, then if $t^* \leq t_n$ then evaluate the step size, otherwise keep the Same step size and go to (iii)

iiii-If $(t_n+h)^T$ where T is a point where the solution is required, then we take next step size to be $h=T-t_n$ .using the step size h ,the solution and its derivative values, we calculate the next approximation and repeat the test in (a).

6. Conclusions:

From the above results, we can notice that the fourth order Runge-Kutta method gave better results than the second order. Also, the results obtained by using Chebyshev polynomials are better than those obtained by Lagrange or Hermite interpolation.

References:-

[1] A.N. Al-Multib "Numerical Methods for solving Delay Differential Equations" ph.D.Thesis, university of Manchester,(1977).

[2] C.A.H.paul, "Developing a Delay Differential Equation Solver", Numerical Analysis Report, No 204, December, (1991)

[3] C.A.H.paul and C.T.H. Balcer"Explicit Runge-Kutta Methods for Numerical Solution of Singular Delay Differential Equations", Numerical Analysis, No.212, April,(1992).
[4] J.C. Moson and DCHandscomb,"Chebyshev polynomials",ACRC press Company,(2003).
[5] L.F. Shawpine and s. Thompson, "Solving delay differential equations with delay 23", March23, (2000).
[6] S.D.Conte and Carl de Boor ,"Numerical Analysis, An Algorithmic Approach", Third Edition, McGraw-Hill book(1980).

تقنية جبيجيف لحل المعادلات التفاضلية التباطؤية عددياً

الخلاصة

في هذه البحث تم استخدام طريقة رانج–كتا الصريحة لحل المعادلات التفاضلية التباطؤية عددياً واستخدمنا في طريق تقريب الحد التباطئ d(t,y(t)) متعددة حدود جبيجيف بدلا من متعددة حدود لاكرانج وهرمت.