Simulation system TKSL/MA

Miloň Papežík, Pavel Pospíšil, Viktor Němec

Department of Computer Science and Engineering,
Faculty of Electrical Engineering and Computer Science,
Technical University of Brno,
Božetěchova 2, 612 66 BRNO, The Czech Republic,
Telephone: +42-5-7275277 (+42-5-41212219),
Fax:+42-5-41211141
E-Mail `pospispa@dcse.fee.vutbr.cz`

Introduction

Many real problems can be described by a system of the first order ordinary differential equations. Therefore, many numerical methods have been studied since the end of the last century. This work aims at Modern Taylor series method. A new simulation system based on the Modern Taylor series method and on a multibyte arithmetic has been created. Results from experiments are shown.

1 Modern Taylor Series Method

The best-known and most accurate method of calculating a new value of a numerical solution of a differential equation is to construct the Taylor series.

The Modern Taylor Series Method [4] has proved to be both very accurate and fast. It is based on a direct use of the Taylor series.

The main idea behind the Modern Taylor Series Method is an automatic integration method order setting, i.e. using as many Taylor series terms for computing as needed to achieve the required accuracy.

The idea above requires software capable of automatically performing the decomposition of the right-hand sides of ordinary differential equations.

This approach has been implemented in a simulation language TKSL/MA (an implementation of the Taylor Kunovsky Simulation Language on an multibyte arithmetic).

In fact, the well-known rules of differential and integral calculus have been used.

2 Simulation Language TKSL/MA

The simulation language TKSL/MA has been created to test an algorithm for Taylor series method.

TKSL/MA has the following features

- user friendly environment,
- adjustable computation accuracy,
- adjustable method order,
- computation with variable integration step h,

TKSL offers the most practical features available today and it also presents an entirely new approach to solving continuous systems:

- on analysing a particular continuous system only the required precision of the solution is defined,
- the order of the implicitly built-in integration method can interactively be increased (yielding a new computation quality),

3 Programming in TKSL/MA

Programming in TKSL/MA is very easy. Using TKSL/MA is demonstrated on a stiff system which is mentioned in [3]. The following example of Robertson's (1966) has become very popular in numerical studies (Willoughby

1974):

$$
\begin{array}{rcll}
A & \xrightarrow{0.04} & B & \text{(slow)} \\
B+B & \xrightarrow{3\cdot 10^7} & C+B & \text{(very fast)} \\
B+C & \xrightarrow{10^4} & A+C & \text{(fast)}
\end{array}
\qquad (1)
$$

which leads to the equations

$$
\begin{array}{lrclll}
\text{A:} & y_1' & = & -0.04y_1 \quad +10^4 y_2 y_3 & & y_1(0) = 1 \\
\text{B:} & y_2' & = & 0.04y_1 \quad -10^4 y_2 y_3 & -3\cdot 10^7 y_2^2 & y_2(0) = 0 \\
\text{C:} & y_3' & = & & 3\cdot 10^7 y_2^2 & y_3(0) = 0.
\end{array}
\qquad (2)
$$

The corresponding source text in TKSL/MA is

```
var t,y1,y2,y3,a,b;

const
  nula=0,
  jedna=1,
  min_ctyri_set=-0.04,
  ctyri_set=0.04,
  des_na_4=1e4,
  tri_kr_des_na_7=3e7;

system
t'=jedna                                          &nula;
y1'=min_ctyri_set*y1+des_na_4*a                   &jedna;
y2'=ctyri_set*y1-des_na_4*a-tri_kr_des_na_7*b    &nula;
y3'=tri_kr_des_na_7*b                             &nula;
a=y2*y3;
b=y2*y2;
sysend.

simul
  arithmetic=16;
  eps=1e-20;
  maxord=80;
  minord=1;
```

```
  step=0.3;
  tmax=0.3;
simend.

graph
  y2=f(t);
  step=f(t);
  y1,y2,y3=f(t);
graphend.
```

All variables that will be needed are declared in the line starting with **var**. All necessary constants are declared in the line starting with **const**. The equations being solved are declared in block mode (between **system** and **sysend**). The initial conditions for differential equations are written in the form & *initial condition*. The required experiment is written between **simul** and **simend**, **arithmetic** means number of bytes used, **eps** is the required accuracy, **maxord** is the maximum number of Taylor series terms used, **minord** is the minimum number of Taylor series terms used, **step** is the step size and **tmax** is the maximal computation time. The required graphs are specified between **graph** and **graphend**.

4 Results

Hairer has written in [3]:

> After a bad experience with explicit Euler just before, let's try a higher order method and a more elaborate code for this example: DOPRI5 (see [2]). The numerical solution obtained for y_2 with $Rtol$[1] $= 2 \cdot 10^{-2}$ (204 steps) as well as with $Rtol = 10^{-3}$ (203 steps) and $Atol$[2] $= 10^{-6}$.
>
> We observe that the solution y_2 rapidly reaches a quasi-stationary position in the vicinity of $y_2' = 0$, which in the beginning ($y_1 = 1, y_3 = 0$) is at $0.04 \approx 3 \cdot 10^7 y_2^2$, hence $y_2 \approx 3.65 \cdot 10^{-5}$, and then

[1] Rtol — relative tolerance

[2] Atol — Absolute tolerance

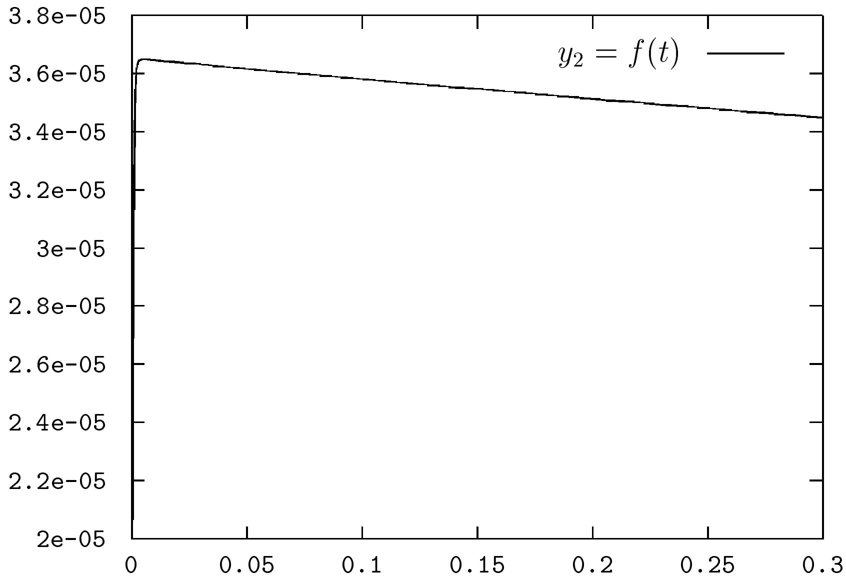

Figure 1: Chemical reaction — $y_2 = f(t)$

very slowly goes back to zero again. The numerical method, however, integrates this smooth solution by thousands of apparently unnecessary step. Moreover, the chosen step sizes are more or less independent of the chosen tolerance. Hence, they seem to be governed by stability rather than by precision requirements. It can also be seen that an implicit Runge-Kutta code (such as RADAU5 described in [3]) integrates this equation without any problem (13 steps, $Rtol = 10^{-2}$).

To solve system of equations (2) we have set following simulation parameters: $arithmetic = 16B$, $step = 0.3$, $tmax = 0.3$, $minord = 1$, $maxord = 90$ and $eps = 10^{-20}$. Results ($y_2 = f(t)$) are shown in Figure 1. We have obtained solution in 50 steps.

The step size is very small in the beginning but it is increasing later. The biggest step size which has been used is approximately 0.0094 (see Figure 2).

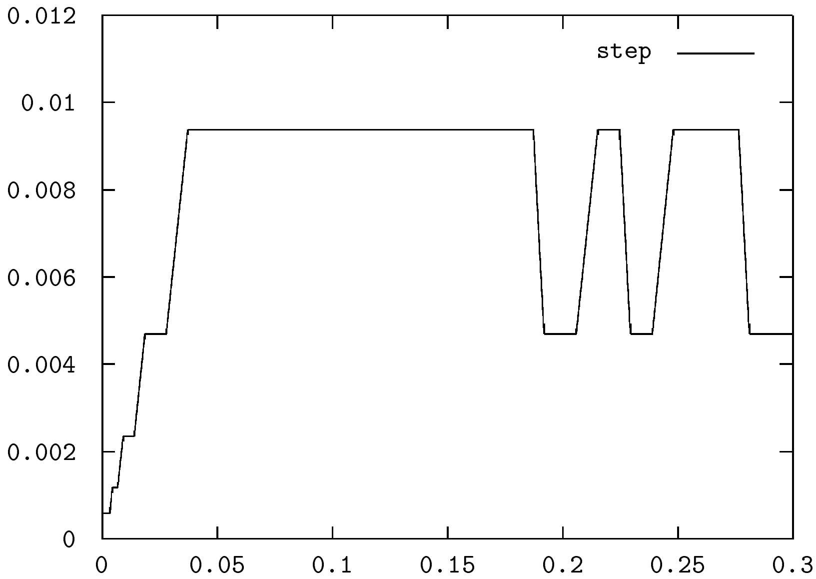

Figure 2: Chemical reaction — step size control

References

[1] Borrelli, R. L., Coleman, C., Boyce, W. E.: Differential Equations Laboratory Workbook, John Wiley & Sons, New York, 1992

[2] Hairer, E., Nørset, S. P., Wanner, G.: Solving Ordinary Differential Equations I—Nonstiff Problems, Springer-Verlag, Second revised edition, 1991.

[3] Hairer, E., Wanner, G.: Solving Ordinary Differential Equations II — Stiff and Differential-algebraic Problems, Springer-Verlag, 1991.

[4] Kunovský, J.: Modern Taylor Series Method. Habilitation work, FEI VUT Brno, 1995.