Modification of Ordinary Differential Equations MATLAB Solver

Elena COCHEROVÁ

Department of Radio Electronics, FEI SUT, Ilkovičova 3, 812 19 Bratislava, Slovak Republic

cocherov@elf.stuba.sk

Abstract. *Various linear or nonlinear electronic circuits can be described by the set of ordinary differential equations (ODEs). The ordinary differential equations can be solved in the MATLAB environment in analytical (symbolic toolbox) or numerical way. The set of nonlinear ODEs with high complexity can be usually solved only by use of numerical integrator (solver).*

The modification of ode23 MATLAB numerical solver has been suggested in this article for the application in solution of some special cases of ODEs. The main feature of this modification is that the solution is found at every prescribed point, in which the special behavior of system is anticipated. The extrapolation of solution is not allowed in those points.

Keywords

Ordinary differential equation (ODE), MATLAB, nonlinear, numerical solver, ode23.

1. Introduction

The effective ODE solver should exert some adaptive control over the progress of solution, making frequent changes in its evaluation step size [1]. The set of ODEs can be solved numerically in the MATLAB environment by the use of different solvers: `ode23`, `ode45`, etc.

The `ode23` is an implementation of the explicit Runge-Kutta method of Bogacki and Shampine [2]. It uses a "free" interpolant of order 3 and local extrapolation. Since the extrapolation is allowed, the actual value of evaluation step *h* is changing during the solution of ODE, depending on the shape of solution and on prescribed accuracy of solution. If the estimated error of solution is small enough, *h* is increased. Therefore, if the system is going to the steady-state, step *h* becomes larger and larger. If in this state comes short external input signal, i.e. duration of external input signal is shorter than step size *h*, it may happen, that `ode23` jump-over this period of time in which external signal is applied. There is no response on that external signal in such case, the external signal is "ignored".

If user recognizes, that some part of response is omitted, he can increase prescribed accuracy of solution (decrease error tolerance), so evaluation step *h* will be less increased; that is usually sufficient to obtain the correct solution.

Another way, how to overcome this problem, is to insure that solution will be evaluated (not only extrapolated) in those points in which external signal comes. Such way is suggested in this article, and modified function is named `ode23mod`.

2. Description of the Original ode23 and Modified ode23mod Solver

The basic information about `ode23` function can be found in MATLAB help (e.g. version 5.3.1). Writing in command line of the MATLAB Command Window:

```
» help ode23
```

MATLAB shows following information:

```
ODE23  Solve non-stiff differential equations, low order method.

[T,Y] = ODE23('F',TSPAN,Y0) with TSPAN = [T0 TFINAL] integrates the system of differential equations y' = F(t,y) from time T0 to TFINAL with initial conditions Y0.  'F' is a string containing the name of an ODE file.  Function F(T,Y) must return a column vector. Each row in solution array Y corresponds to a time returned in column vector T. To obtain solutions at specific times T0, T1, ..., TFINAL (all increasing or all decreasing), use TSPAN = [T0 T1 ... TFINAL].

[T,Y] = ODE23('F',TSPAN,Y0,OPTIONS) solves as above with default integration parameters replaced by values in OPTIONS, an argument created with the ODESET function. See ODESET for details.  Commonly used options are scalar relative error tolerance 'RelTol' (1e-3 by default) and vector of absolute error tolerances 'AbsTol' (all components 1e-6 by default). Etc.
```

The user writes to the *tspan* vector those points, in which he wants to obtain the solution of ODE.

The `ode23` solver changes the actual step *h* according estimated error of solution that depends on settings of "odeset properties" (*RelTol* - Relative error tolerance,

AbsTol - Absolute error tolerance, *MaxStep* - Upper bound on step size, etc.) and actual value of solution *y*.

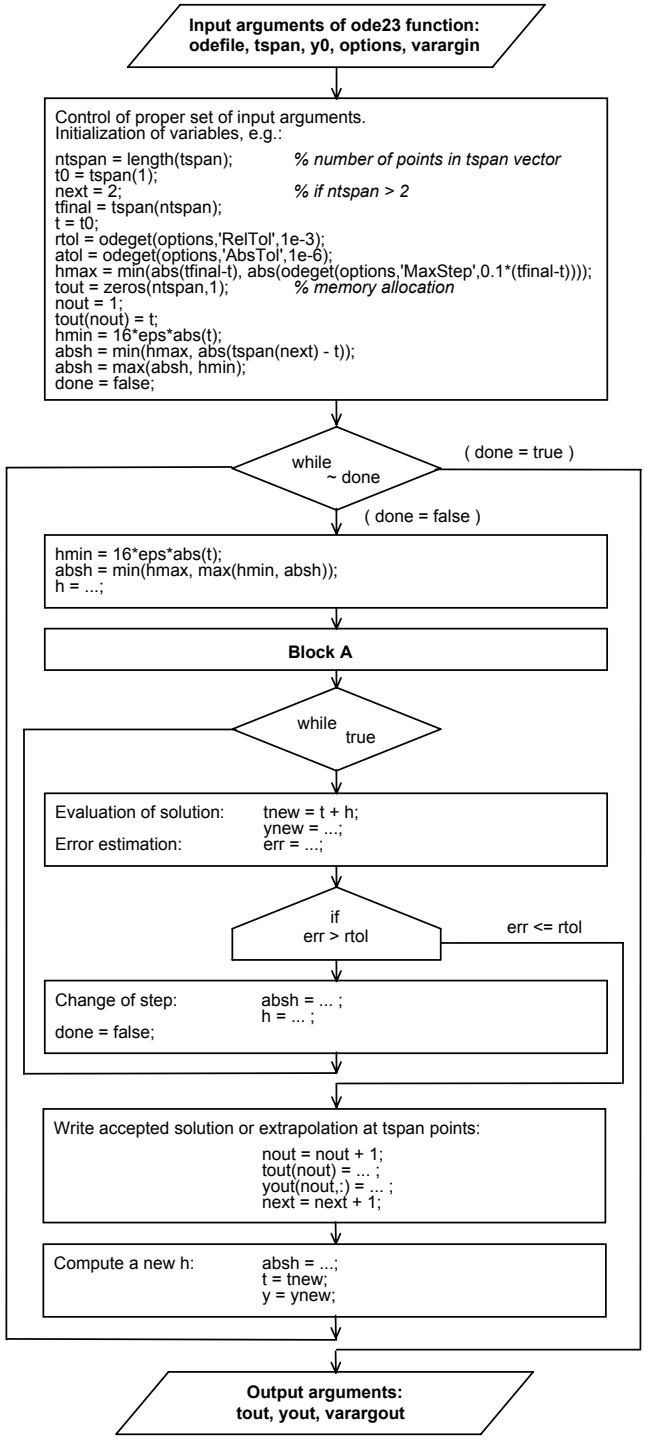

Fig. 1. The flow-diagram of most important parts of `ode23`.

In fact, `ode23` solver doesn't solve *y*' at every point from *tspan*. Actual solver step *h* may be larger than distance between subsequent points from the *tspan* vector, and then local extrapolation for those points is made.

The simplified flow-diagram of most important parts of `ode23` function is shown in Fig. 1. More detailed flow-diagram can be found in [3].

The flow-diagram of Block A is shown in Fig. 2.

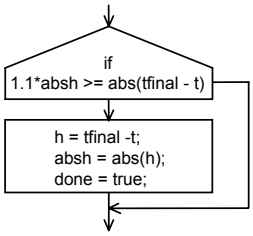

Fig. 2. The flow-diagram of Block A of `ode23` solver.

Listing of `ode23` function file can be displayed writing in the MATLAB Command Window:

```
» type ode23
```

e.g. Block A has the following listing:

```
% Stretch the step if within 10% of tfinal-t
   if 1.1*absh >= abs(tfinal - t)
     h = tfinal - t;
     absh = abs(h);
     done = true;
   end
```

The original Block A is suggested to be replaced in `ode23mod` by:

```
% Stretch the step if within 10%
% of tspan(next)-t

   if 1.1*absh >= abs(tspan(next) - t)
     h = tspan(next) - t;
     absh = abs(h);
     if  next == ntspan
                 done = true;
     end
   end
```

and the flow-diagram of Block A of `ode23mod` solver is shown in Fig. 3.

This modification insures, that the solution *y*' is performed at every point given in *tspan* vector (or more dense, if it is required according accuracy set by *RelTol* and *AbsTol*). Therefore no extrapolation is needed to be done in `ode23mod` solver.

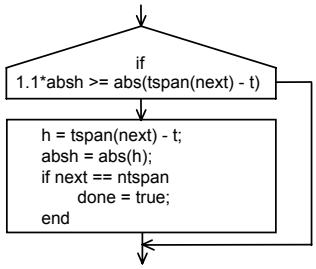

Fig. 3. The flow-diagram of Block A of `ode23mod` solver.

3. Comparison of the Original ode23 and Modified ode23mod Solver

As an example, the numerical solution of the ODEs given by the Hodgkin - Huxley (HH) model [3], [4] has

been performed by use of original `ode23` solver and modified `ode23mod` solver.

The HH model of nerve fibre membrane is described by the set of four nonlinear ODEs. The first equation includes the term $i_{st}(t)$, that represents an external input stimulation current density, e.g. see Fig. 4.

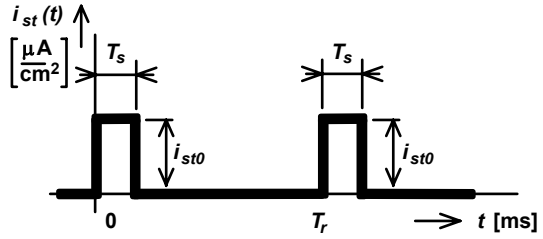

Fig. 4. The external input signal: a pair of impulses.

The solution of HH set of ODEs including stimulation input signal (shown in Fig. 4) is illustrated in Fig. 5 for the cases that `ode23` or `ode23mod` is used.

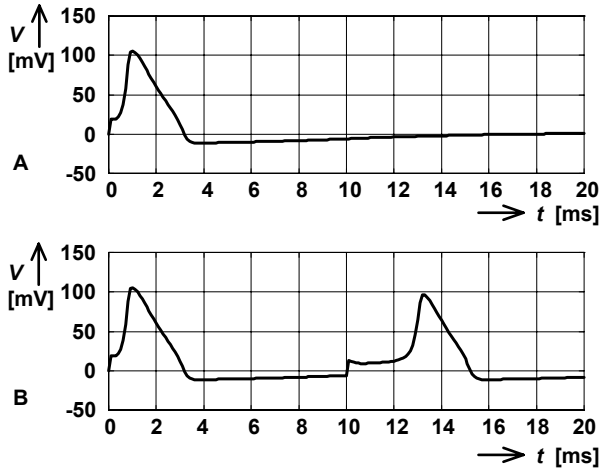

Fig. 5. Solution of HH model for stimulation given by use of: `ode23` - improper solution (A), and given by use of `ode23mod` - proper solution (B).

The HH system responds on first stimulation impulse (that starts in the time $t = 0$) and after certain time system is coming to the steady state.

If the further stimulus (second stimulus, that starts in the time $t = T_r$) comes in the time, when the system is in the steady state (or near), `ode23` solver uses larger step. The problem may occur if the actual evaluation step h is larger than width of stimulation signal (T_s). In such case `ode23` renders an improper solution (Fig. 5A), where the response on the second stimulus is omitted.

Used parameters are following: amplitude of stimulation current density $i_{st0} = 200$ μA/cm^2, duration of stimulation pulse $T_s = 0.1$ ms, $T_r = 10$ ms, *Tfinal* = 20 ms, *step* = = 0.1 ms, *tspan* = 0 : *step* : *Tfinal*, relative error tolerance *RelTol* = 10^{-3} and absolute error tolerance *AbsTol* = 10^{-6} (default values of *RelTol* and *AbsTol*).

The results in Tab. 1 and Tab. 2 are obtained for different number of points of *tspan* vector - *ntspan* (different time *step*) and for different *RelTol* (10^{-3} or 10^{-6}).

	RelTol = 10^{-3}			*RelTol* = 10^{-6}		
step **[ms]**	**0.1**	**0.01**	**0.001**	**0.1**	**0.01**	**0.001**
ntspan	**201**	**2001**	**20001**	**201**	**2001**	**20001**
nsteps	**110**	**110**	**110**	**1022**	**1022**	**1022**
nfailed	**30**	**30**	**30**	**51**	**51**	**51**
nfevals	**421**	**421**	**421**	**3220**	**3220**	**3220**
Evaluation time [ms]	**2.2**	**4.5**	**27.6**	**15.2**	**17.4**	**40.7**
Proper solution	**NO**	**NO**	**NO**	**YES**	**YES**	**YES**

Tab. 1. Results for `ode23`.

	RelTol = 10^{-3}			*RelTol* = 10^{-6}		
step **[ms]**	**0.1**	**0.01**	**0.001**	**0.1**	**0.01**	**0.001**
ntspan	**201**	**2001**	**20001**	**201**	**2001**	**20001**
nsteps	**301**	**2023**	**20011**	**1104**	**2375**	**20032**
nfailed	**45**	**13**	**3**	**68**	**45**	**33**
nfevals	**1039**	**6109**	**60043**	**3517**	**7261**	**60196**
Evaluation time [ms]	**4.8**	**28.8**	**282.9**	**16.2**	**33.9**	**284.7**
Proper solution	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**

Tab. 2. Results for `ode23mod`.

Output statistics (output argument *varargout*) from `ode23` function contain:

- *nsteps* - number of points in which *y*' is evaluated;
- *nfailed* - number of points in which estimated error (err) is larger than *RelTol* and step size *h* has been changed smaller;
- *nfevals* - how many times is evaluated *y*' (since the interpolant of order 3 is used, *y*' is evaluated 3 times at every time point).

Evaluation time depends mainly on the number of points, where *y*' is evaluated (*nsteps*), extrapolated (relates partially to *ntspan* for `ode23` solver), number of points where solution was not accepted for insufficient accuracy (*nfailed*) and on PC configuration.

Obviously, for default value of *RelTol* = 10^{-3} `ode23` solver renders improper solution independent of *step* size in *tspan* vector (since *nsteps* of solver is independent on *ntspan* given by user).

For *step* = 0.1 ms and *RelTol* = 10^{-6} solution is proper for both `ode23` and the `ode23mod` solvers, and evaluation times are comparable (15.2 ms and 16.2 ms, resp.). The evaluation time increases enormously for smaller *step* values (larger *ntspan*) when the `ode23mod` solver is used.

4. Conclusion

The `ode23mod` solver is proper choice for such cases,

when user wants insure himself, that the solution of differential equations is made at certain special evaluation points, e.g. in which external input signal comes. Those points must be written in *tspan* vector.

Since the suggested `ode23mod` solver is quite time consumptive at certain conditions (small *step*) in comparison with original `ode23` solver, therefore the user must decide when to exploit it and must find the optimal number of points in *tspan* vector.

Acknowledgements

This work was supported by the grant VEGA 1/9045/02 and by the Ministry of Education of the Slovak Republic under Grant VTP 102/2000 "Digital Processing of Audio, Video and Biomedical Signals".

References

[1] PRESS, W. H., FLANNERY, B. P., TEUKOLSKY, S. A., VETTERLING, W. T. *Numerical recipes in C*. Cambridge: Cambridge University Press, 1992.

[2] SHAMPINE, L. F., REICHELT, M. W. The MATLAB ODE suite. *SIAM Journal on Scientific Computing*. 1997, vol. 18, no. 1, p. 1-22.

[3] http://kre.elf.stuba.sk/~cocherova.

[4] COCHEROVÁ, E. Refractory period determination in the Hodgkin - Huxley model of the nerve fibre membrane. In *Proceedings of the 4th Electronic Circuits and System Conference*. Bratislava, 2003, p. 171 to 174.

About Author...

Elena COCHEROVÁ was born in Trenčín in 1970. She received Ing. degree in radioelectronics (1993) and Ph.D. degree in electronics (2002) from the Faculty of Electrical Engineering and Information Technology (FEI), the Slovak University of Technology (SUT) in Bratislava. She works as an university teacher at the Department of Radio Electronics, FEI SUT in Bratislava.

The Board of the Radioengineering Society: Results of Elections

The Editorial Board of the Radioengineering Journal is pleased to inform the readers about the results of the postal elections of the Board of the Radioengineering Society. The Radioengineering Society is the publisher of the Radioengineering Journal.

The elections were held in June, 2003, and 39 members of the Radioengineering Society took part in. The elected members of the Board of the Radioengineering Society are:

- **Prof. Miloš Mazánek**
  Dept. of Electromagnetic Field, Czech Technical University, Prague,
- **Prof. Zbyněk Škvor**
  Dept. of Electromagnetic Field, Czech Technical University, Prague,
- **Prof. Jiří Svačina**
  Dept. of Radio Electronics, Brno University of Technology, Brno.

The members of the auditing committee of the Radioengineering Society are:

- **Prof. Vladimír Šebesta**
  Dept. of Radio Electronics, Brno University of Technology, Brno
- **Prof. Karel Hoffmann**
  Dept. of Electromagnetic Field, Czech Technical University, Prague.

The Editorial Board of the Radioengineering Journal thanks them for their goodwill to take the responsibility for prolonging the good tradition of the Society, and wishes them good luck in their activities.