COMBINED EXPLICIT-IMPLICIT TAYLOR SERIES METHODS

S.N. Dimova [1] , I.G. Hristov [1, a], R.D. Hristova [1], I V. Puzynin [2], T.P. Puzynina [2], Z.A. Sharipov [2, b], N.G. Shegunov [1], Z.K. Tukhliev [2]

[1] *Sofia University, Bulgaria*

[2] *JINR, LIT, Dubna, Russia*

E-mails: [a] ivanh@fmi.uni-sofia.bg [b] *zarif@jinr.ru*

We investigate numerically a class of combined Explicit-Implicit Taylor series methods of various order of accuracy for solving Hamiltonian systems. The purpose of the investigation is to confirm our expectations that these methods behave as symplectic ones in terms of energy conservation, and that in some cases they may overmatch the standard second order Verlet method.
Indeed, the numerical results show that our methods conserve the energy for long-time integration. This indicates that we have a tool to construct easily energy conservation methods of any order of accuracy. Also, when a very high accuracy is needed, they show substantially better performance than the Verlet method. The comparison between our methods and the Verlet method is done in terms of a standard "CPU-time – Error" diagram on a classical Hamiltonian system, namely the Henon-Heiles problem.
In addition we test an OpenMP approach for computing multiple independent trajectories using our methods. The results are very promising. We achieve a significant speedup up to ~ 37, when we use the whole resource of one computational CPU node in the HybriLIT education and testing polygon.

Keywords: Hamiltonian systems, Taylor Series methods, Energy conservation methods, OpenMP parallel technology.

1. Introduction

We solve numerically the initial value problem

$$\dot{\boldsymbol{u}}(\boldsymbol{t}) = \boldsymbol{f}(\boldsymbol{u}), \boldsymbol{u}(\boldsymbol{0}) = \boldsymbol{u_0} \tag{1}$$

with $u, u_0 \in \mathbb{R}^n$ by using a class of combined Explicit-Implicit methods proposed in [1]. This class is based on Taylor series expansion and consists of methods of various orders of accuracy. The idea is to combine Taylor expansions about the forward and current time levels. Let y be the approximate solution, then for given N = 1,2,3, ... we have the method:

$$\boldsymbol{y}\left(\boldsymbol{t}+\frac{\boldsymbol{\tau}}{2}\right) = \sum_{\boldsymbol{k}=\boldsymbol{0}}^{\boldsymbol{N}} \frac{\boldsymbol{y}^{(\boldsymbol{k})}(\boldsymbol{t})}{\boldsymbol{k}!}\left(\frac{\boldsymbol{\tau}}{2}\right)^{\boldsymbol{k}} \quad (\textbf{explicit step}) \tag{2a}$$

$$\boldsymbol{y}(\boldsymbol{t}+\boldsymbol{\tau}) = \boldsymbol{y}\left(\boldsymbol{t}+\frac{\boldsymbol{\tau}}{2}\right) - \sum_{\boldsymbol{k}=\boldsymbol{1}}^{\boldsymbol{N}} \frac{\boldsymbol{y}^{(\boldsymbol{k})}(\boldsymbol{t}+\boldsymbol{\tau})}{\boldsymbol{k}!}\left(-\frac{\boldsymbol{\tau}}{2}\right)^{\boldsymbol{k}} \quad (\textbf{implicit step}) \tag{2b}$$

- Our first goal is to compare the standard Verlet method with the combined Explicit-Implicit methods of 4-th (***N*** = 3), 6-th (***N*** = 5), 8-th (***N*** = 7) orders in terms of a "CPU-time – Error" diagram on a test Hamiltonian problem.
- Our second goal is to test the computer performance scalability inside one CPU-node of the HybriLIT education and testing polygon, when scheduling onto OpenMP threads multiple independent trajectories.

2. Properties and realization details of the numerical methods

2.1. Symmetry and Energy conservation properties of the methods

The explicit and implicit Taylor methods are adjoint to each other [2], consequently methods (2) are symmetric and hence of an even order. For efficiency we consider methods for odd N. For $N = 1$ we obtain a 2-nd order method, which is the well known trapezoidal method. For $N = 3{,}5{,}7$ we obtain respectively 4-th, 6-th, 8-th order methods. Because we restrict ourselves to double precision arithmetic, it is not reasonable to consider methods above 8-th order.

Usually an important property of a numerical method designed for Hamiltonian systems is its energy conservation. Although the trapezoidal method is not strictly symplectic, it has the property of energy conservation [2]. The methods (2) can be seen as a generalization of the trapezoidal method, so one could think of them as "high order trapezoidal methods". Intuitively, these high order methods should have the energy conservation property. We will not give here a strict proof of this property, but all numerical tests confirm this. The numerical tests show excellent long-time behavior of the total energy (the Hamiltonian) for all our methods, i.e. with respect to energy conservation our methods behave as symplectic ones.

The strength of methods (2) is the possibility to construct easily (at least theoretically) energy conservation methods of any order. Such methods can be useful for problems where very high accuracy is needed, of course by using extended precision - quadruple or higher.

2.2. Calculation of the derivatives - automatic differentiation

To use formulas (2) we have first to compute the coefficients of the Taylor polynomial (the normalized derivatives) and then for a given step τ to use the Horner evaluation of the Taylor polynomial. The evaluation of the derivatives is done by the classical automatic differentiation, avoiding symbolic derivation or numerical approximation of derivatives [4].

2.3. Solving the implicit step

To solve the implicit step of (2), a fixed-point iteration is applied.
The $\boldsymbol{m}-\mathbf{th}$ iteration is given by:

$$\boldsymbol{y}^{[\boldsymbol{m}]}(\boldsymbol{t}+\boldsymbol{\tau}) = \boldsymbol{y}(\boldsymbol{t}+\tfrac{\boldsymbol{\tau}}{\mathbf{2}}) - \sum_{\boldsymbol{k}=\mathbf{1}}^{\boldsymbol{N}} \frac{{\boldsymbol{y}^{(\boldsymbol{k})}}^{[\boldsymbol{m}-\mathbf{1}]}(\boldsymbol{t}+\boldsymbol{\tau})}{\boldsymbol{k}!}(-\tfrac{\boldsymbol{\tau}}{\mathbf{2}})^{\boldsymbol{k}} \tag{3}$$

The initial approximation $y^{[0]}(t+\tau)$ is calculated by an explicit step. To obtain a stopping criterion, we follow "Iteration until convergence" recommendations from [3]. Namely, we iterate until:

$$\textbf{either } \boldsymbol{\Delta}^{[\boldsymbol{m}]} = \mathbf{0} \textbf{ or } \boldsymbol{\Delta}^{[\boldsymbol{m}]} \geq \boldsymbol{\Delta}^{[\boldsymbol{m}-\mathbf{1}]}, \tag{4}$$

$$\Delta^{[m]} = \| y^{[m]} - y^{[m-1]} \|_{\infty}.$$

The inequality above indicates that the iteration increments begin to oscillate due to round-off errors. This criterion has the advantage that it does not require a problem or method dependent tolerance. The average number of iterations for a given method depends on the step size τ and converges to 1 as τ tends to 0. That means that for sufficiently small step size τ, the work for one step is fixed and small, like that in an explicit method.

2.4. Test setup: the Henon-Heiles problem

The Henon-Heiles problem, a classical Hamiltonian system, is considered as a test problem.
The Hamiltonian is:

$$\boldsymbol{H}(\boldsymbol{p},\boldsymbol{q}) = \tfrac{\mathbf{1}}{\mathbf{2}}(\boldsymbol{p}_{\mathbf{1}}{}^{\mathbf{2}} + \boldsymbol{p}_{\mathbf{2}}{}^{\mathbf{2}}) + \tfrac{\mathbf{1}}{\mathbf{2}}(\boldsymbol{q}_{\mathbf{1}}{}^{\mathbf{2}} + \boldsymbol{q}_{\mathbf{2}}{}^{\mathbf{2}}) + \boldsymbol{q}_{\mathbf{1}}{}^{\mathbf{2}}\boldsymbol{q}_{\mathbf{2}} - \tfrac{\mathbf{1}}{\mathbf{3}}\boldsymbol{q}_{\mathbf{2}}{}^{\mathbf{3}}. \tag{5}$$

The Hamiltonian equations $\dot{p} = -\nabla_q H(p,q), \dot{q} = \nabla_p H(p,q)$ give the following equations of the form (1):

$$\dot{\boldsymbol{q}}_{\mathbf{1}} = \boldsymbol{p}_{\mathbf{1}},\ \dot{\boldsymbol{q}}_{\mathbf{2}} = \boldsymbol{p}_{\mathbf{2}},\ \dot{\boldsymbol{p}}_{\mathbf{1}} = -\boldsymbol{q}_{\mathbf{1}} - \mathbf{2}\boldsymbol{q}_{\mathbf{1}}\boldsymbol{q}_{\mathbf{2}},\ \dot{\boldsymbol{p}}_{\mathbf{2}} = -\boldsymbol{q}_{\mathbf{2}} - \boldsymbol{q}_{\mathbf{1}}{}^{\mathbf{2}} + \boldsymbol{q}_{\mathbf{2}}{}^{\mathbf{2}}. \tag{6}$$

The initial conditions are taken from [2].

3. Numerical results

3.1. The HybriLIT education and testing polygon

All of the numerical tests are performed on the HybriLIT education and testing polygon, a part of the HybriLIT Heterogeneous Platform, which itself is a part of the Multipurpose information and computing complex (MICC) of the Laboratory of Information Technologies of JINR [5]. The OS Scientific Linux 7.4 is installed in the HybriLIT platform. Our computations are performed on one computational CPU-node consisting of 2 x Intel(R) Xeon(R) Processor E5-2695v3 (28 cores, 56 hardware threads). For compilations the Intel (R) Fortran Compiler 18.0 is used. The best performance for all of the methods was obtained when we have used optimization options -O3 -xhost.

3.2. CPU-time - Error diagram

The larger memory demand of methods (2) in comparison to the Verlet method is compensated by their higher convergence order. As expected, a higher order method performs better, when the accuracy requirements become sufficiently high. As it is seen from Figure 1, the "intersection points" for our test problem are between 10^{-7} and 10^{-4}. For accuracy 10^{-10} for example our methods behaves substantially better then the Verlet method. Our conclusion is that, in general, the methods (2) should show better behavior than the Verlet method for problems requiring high accuracy.

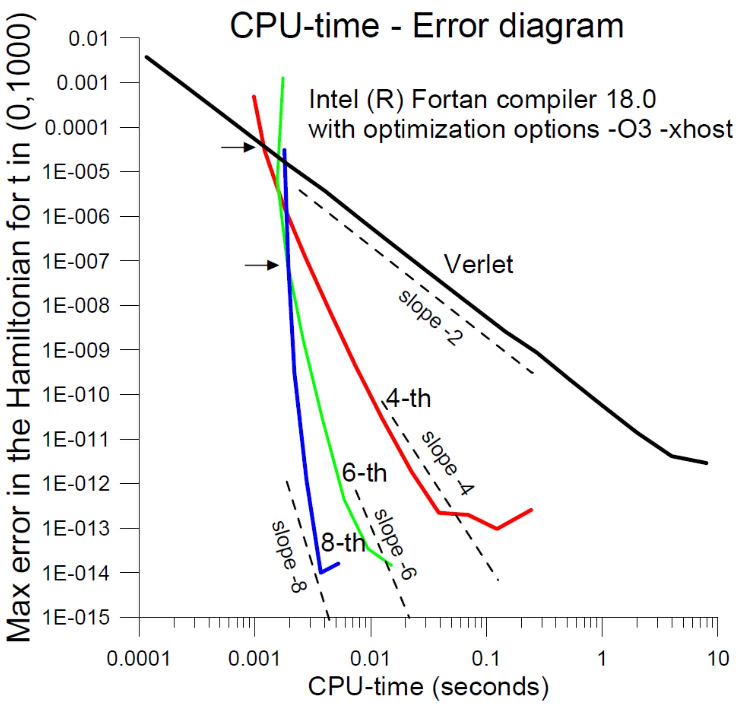

Figure 1. CPU-time – Error diagram

3.3 Computing multiple independent trajectories with OpenMP

There are two main sources of large scale problems when solving systems of ODEs and thus, requiring parallel computing. The first source are systems of very large number of equations, for example, coming from the molecular dynamics. The second source are small systems that need to be solved for a large number of independent sets of initial conditions or for many sets of parameters. This is the case when we calculate different indicators for a given dynamical system, for example the Fast Lyapunov Indicator [6]. Similar is the situation when we solve system of ODEs in Monte Carlo framework.

Since in this work we are focused only on solving small systems with high accuracy, we propose an approach of parallelization with OpenMP [7] of the second type of problems, i.e. when we simulate multiple independent trajectories. We schedule the independent trajectories onto OpenMP threads by a simple OpenMP "DO loop" with the "schedule" clause with parameter "guided". The results are very promising. A strong scalability result is shown in Figure 2. The simulation of 1000 independent trajectories by the 6-th order method shows significant speedup up to ~ 37, when we use the whole resource of one computational CPU node.

4. Conclusions

- The numerical tests show excellent long-time behavior of the total energy for the combined Explicit-Implicit Taylor series methods. This means that with respect to the energy conservation our methods behave as symplectic ones.
- The average number of iterations for the implicit step depends on the step size τ and converges to 1 as τ tends to 0. For sufficiently small step size the work for one step is fixed and small and our methods behave as explicit ones.
- For problems, requiring high accuracy, the considered methods show substantially better performance than the standard second order Verlet method.

- The results for computing multiple independent trajectories with our methods using OpenMP parallel technology show significant speedup up to ~ 37, when we use the whole resource of one computational CPU node in the HybriLIT education and testing polygon.

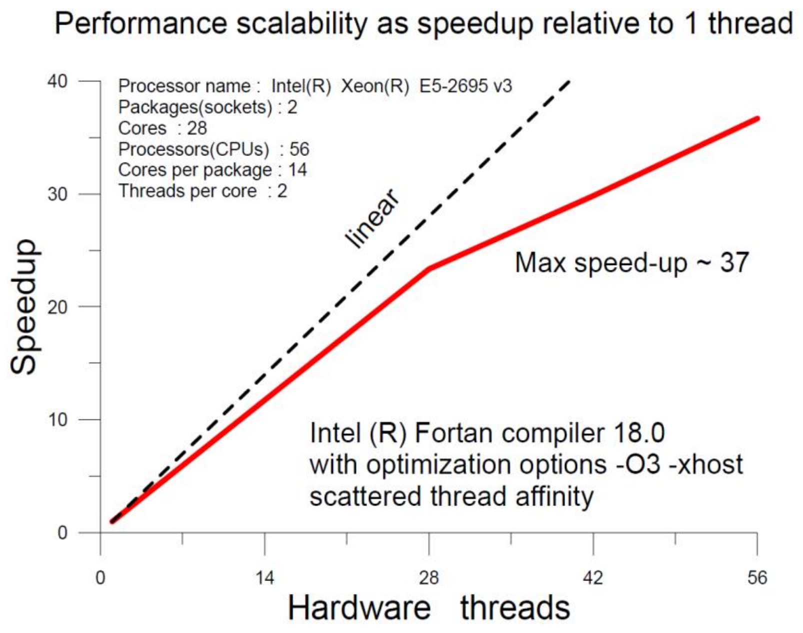

Figure 2. Performance scalability

Thanks and Acknowledgements

We thank the Laboratory of Information Technologies of JINR for the opportunity to use the computational resources of the HybriLIT Heterogeneous Platform, a part of the Multipurpose information and computing complex (MICC).

The work is financially supported by RFBR grant No. 17-01-00661-a and by a grant of the Plenipotentiary Representative of the Republic of Bulgaria at the JINR. The work of the first two authors is partially supported by the SU Grant 80-10-139/25.04.2018.

References

[1] Akishin, P. G., Puzynin, I. V., Vinitsky, S. I. (1997). *A hybrid numerical method for analysis of dynamics of the classical Hamiltonian systems*. Computers and Mathematics with Applications, 34 (2-4), 45-73.

[2] Hairer, E., Lubich, C., Wanner, G. (2006). *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations* (Vol. 31). Springer Science and Business Media.

[3] Hairer, E., McLachlan, R. I., Razakarivony, A. (2008). *Achieving Brouwer's law with implicit Runge–Kutta methods*. BIT Numerical Mathematics, 48(2), 231-243.

[4] Jorba, A., Zou, M. (2005). *A software package for the numerical integration of ODEs by means of high-order Taylor methods* Experimental Mathematics, 14(1), 99-117.

[5] Multipurpose Information and Computing Complex of the Laboratory of IT of JINR. Available at: http://hlit.jinr.ru (accessed 10.10.2018)

[6] Rodriguez, M., Blesa, F., Barrio, R. (2015). *OpenCL parallel integration of ODEs: Applications in computational dynamics*. Computer Physics Communications, 192, 228-236.

[7] Chapman, B., Jost, G., Van Der Pas, R. (2008). *Using OpenMP: portable shared memory parallel programming* (Vol. 10). MIT press.