Conjugate-symplecticity properties of Euler–Maclaurin methods and their implementation on the Infinity Computer[☆]

F. Iavernaro[a], F. Mazzia[b,*], M.S. Mukhametzhanov[c,d], Ya.D. Sergeyev[c,d]

[a] *Dipartimento di Matematica, Università degli Studi di Bari Aldo Moro, Italy*
[b] *Dipartimento di Informatica, Università degli Studi di Bari Aldo Moro, Italy*
[c] *DIMES, Università della Calabria, Italy*
[d] *Department of Software and Supercomputing Technologies Lobachevsky State University of Nizhni Novgorod, Russia*

Abstract

Multi-derivative one-step methods based upon Euler–Maclaurin integration formulae are considered for the solution of canonical Hamiltonian dynamical systems. Despite the negative result that simplecticity may not be attained by any multi-derivative Runge–Kutta methods, we show that the Euler–MacLaurin method of order p is conjugate-symplectic up to order $p + 2$. This feature entitles them to play a role in the context of geometric integration and, to make their implementation competitive with the existing integrators, we explore the possibility of computing the underlying higher order derivatives with the aid of the Infinity Computer.

Keywords: Ordinary differential equations, Hamiltonian systems, multi-derivative methods, numerical infinitesimals, Infinity Computer.
2010 MSC: 65L06, 65P10, 65D25

1. Introduction

In the present work, we will consider the application of multi-derivative one-step methods for the numerical solution of canonical Hamiltonian problems

$$y' = J\nabla H(y), \qquad y(t_0) = y_0 \in \mathbb{R}^{2m}, \tag{1}$$

with

$$y = \begin{pmatrix} q \\ p \end{pmatrix}, \quad q, p \in \mathbb{R}^m, \qquad J = \begin{pmatrix} O & I \\ -I & O \end{pmatrix}, \tag{2}$$

where q and p are the generalized coordinates and conjugate momenta,
$H : \mathbb{R}^{2m} \to \mathbb{R}$ is the Hamiltonian function and I stands for the identity matrix of dimension m. It is well-known that the flow $\varphi_t : y_0 \to y(t)$ associated with the dynamical system (1) is

[☆]This work was funded by the INdAM-GNCS 2018 Research Project "Numerical methods in optimization and ODEs" (the authors are members of the INdAM Research group GNCS). Research of F. Iavernaro and F. Mazzia was also supported by the Università degli Studi di Bari, project "Equazioni di Evoluzione: analisi qualitativa e metodi numerici".

[*]Corresponding author
Email addresses: felice.iavernaro@uniba.it (F. Iavernaro), francesca.mazzia@uniba.it (F. Mazzia), m.mukhametzhanov@dimes.unical.it (M.S. Mukhametzhanov), yaro@dimes.unical.it (Ya.D. Sergeyev)

symplectic, namely its Jacobian satisfies

$$\frac{\partial \varphi_t(y)}{\partial y}^{\top} J \frac{\partial \varphi_t(y)}{\partial y} = J, \quad \text{for all } y \in \mathbb{R}^{2m}. \tag{3}$$

Symplecticity is a characterizing property of canonical Hamiltonian systems and has relevant implications on the geometric properties of the orbits in the phase space. Consequently, the search of symplectic methods for the numerical integration of (1) forms a prominent branch of research. We recall that a one-step method $y_1 = \Phi_h(y_0)$ (h is the stepsize of integration) is called symplectic if its Jacobian matrix is symplectic, i.e. Φ_h satisfies the analog of (3) with $\Phi_h(y)$ in place of $\varphi_t(y)$. One prominent feature of symplectic integrators is, by definition, the conservation of the symplectic differential 2-form associated with matrix J in (2) which, in turn, implies the conservation of all quadratic first integrals of a Hamiltonian system. Though they fail to conserve non quadratic Hamiltonian functions, a backward error analysis shows that, when implemented with constant stepsize and under regularity assumptions, they provide a near conservation of the Hamiltonian function over exponentially long times [1] (see also [2, page 366]).

The study of symplecticity in combination with multi-derivative R-K methods was initiated by Lasagni [3] who provided a sufficient algebraic condition for a multi-derivative Runge–Kutta method to be symplectic. The brief investigation culminated with the work of Hairer, Murua, and Sanz Serna [4] who showed that, for irreducible multi-derivative R–K methods, Lasagni's condition is also necessary but it may only be satisfied by standard R–K formulae.

Given this background, it does make sense to wonder whether one-step multi-derivative formulae may share some weaker conditions related to symplecticity. A method $y_1 = \Phi_h(y_0)$ is conjugate to a symplectic method $y_1 = \Psi_h(y_0)$ if a global change of coordinates $\chi_h(y) = y + O(h)$ exists such that $\Phi_h = \chi_h \circ \Psi_h \circ \chi_h^{-1}$. We observe that the solution $\{y_n\}$ of a symplectic conjugate method satisfies $y_n = \Phi_h^n(y_0) = (\chi_h \circ \Psi_h \circ \chi_h^{-1})^n(y_0) = \chi_h \circ \Psi_h^n \circ \chi_h^{-1}(y_0)$. Consequently, symplectic conjugate methods inherit the long-time behavior of symplectic integrators. In the present work we are interested in a generalization of the conjugate-symplecticity property, introduced in [5]. A method $y_1 = \Phi_h(y_0)$ of order p is conjugate-symplectic up to order $p + r$, with $r \geq 0$, if a global change of coordinates $\chi_h(y) = y + O(h^p)$ exists such that $\Phi_h = \chi_h \circ \Psi_h \circ \chi_h^{-1}$, with the map Ψ_h satisfying

$$\Psi_h'(y)^T J \Psi_h'(y) = J + O(h^{p+r+1}). \tag{4}$$

A consequence of property (4) is that the method $\Phi_h(y)$ nearly conserves all quadratic first integrals and the Hamiltonian function over time intervals of length $O(h^{-r})$ (see [5]).

This path of investigation is further motivated by the recent studies concerning the implementation of methods involving higher derivatives of the vector field on the Infinity Computer, a new type of a supercomputer allowing one to work *numerically* with infinite and infinitesimal numbers [6, 7, 8, 9, 10]. The final goal of this new approach is to improve the computational effort associated with the evaluation of the involved derivatives and make them competitive with more standard integrators. In this paper, the Infinity Computer is used for this purpose. It is based on the positional numeral system with the infinite radix ① (called *grossone* and introduced as the number of elements of the set of natural numbers $\mathbb{N}$) introduced in [11, 12, 13] (see also recent surveys [14, 15]). The first ideas that can be considered as predecessors to the Infinity Computing and based on the principle "the part is less than the whole" were studied by Bernard Bolzano (see [16] and a detailed analysis in [17]). It should be noted that the Infinity Computing theory is not related either to Cantor's cardinals and ordinals ([18]) or non-standard analysis and Levi-Civita field ([19, 20, 21]).

In the Infinity Computing, with the introduction of ① in the mathematical language, all other symbols (like ∞, Cantor's ω, $\aleph_0, \aleph_1, \ldots$, etc.) traditionally used to deal with infinities

and infinitesimals in different situations are excluded from the language, because ① and other numbers constructed with its help not only can replace all of them but can be used with a higher accuracy. The ①-based numeral system avoids indeterminate forms and situations similar to $\infty + 1 = \infty$ and $\infty - 1 = \infty$ providing results ensuring that if a is a numeral written in this numeral system then for any a (i.e., a can be finite, infinite, or infinitesimal) it follows $a + 1 > a$ and $a - 1 < a$.

To construct a number C in the ①-based numeral system, we subdivide C into groups corresponding to powers of ①:

$$C = c_{p_m}①^{p_m} + \ldots + c_{p_1}①^{p_1} + c_{p_0}①^{p_0} + c_{p_{-1}}①^{p_{-1}} + \ldots + c_{p_{-k}}①^{p_{-k}}. \tag{5}$$

Then, we can write down the number C as follows:

$$C = c_{p_m}①^{p_m} \ldots c_{p_1}①^{p_1} c_{p_0}①^{p_0} c_{p_{-1}}①^{p_{-1}} \ldots c_{p_{-k}}①^{p_{-k}}, \tag{6}$$

where all numerals $c_i \neq 0$ belong to a traditional numeral system and are called *grossdigits*, while numerals p_i are sorted in the decreasing order with $p_0 = 0$

$$p_m > p_{m-1} > \ldots > p_1 > p_0 > p_{-1} > \ldots p_{-(k-1)} > p_{-k},$$

and called *grosspowers*.

The term having $p_0 = 0$ represents the finite part of C since $c_0①^0 = c_0$. Terms having finite positive grosspowers represent the simplest infinite parts of C. Analogously, terms having negative finite grosspowers represent the simplest infinitesimal parts of C. For instance, the simplest infinitesimal used in this work as the integration step in the Euler method for computing the derivatives is $①^{-1} = \frac{1}{①}$.

The ①-based methodology has been successfully applied in several areas of Mathematics and Computer Science: single and multiple criteria optimization (see [22, 23, 24]), handling ill-conditioning (see [25, 26]), cellular automata (see [27, 28]), Euclidean and hyperbolic geometry (see [29]), percolation (see [30, 31]), fractals (see [32, 33, 31]), infinite series and the Riemann zeta function (see [14, 15, 34]), the first Hilbert problem and supertasks (see [35, 36, 15, 37]), Turing machines and probability (see [38, 15, 39, 40]), etc.

An interesting peculiarity of the Infinity Arithmetic methodology in the context of this paper is that it allows one to work with black-box functions, namely the analytical expression of the function $f(y)$ may be unknown. In other words, the function f can be given by a code or formula which are unknown to the user. He/she provides an argument y and obtains a result $f(y)$ without any knowledge about how this result has been obtained. In particular, this means that the user can not calculate exact derivatives either analytically or symbolically. It has been shown that the Infinity Arithmetic methodology can successfully handle this situation in the context of numerical differentiation and solution of ordinary differential equations (see [6, 8, 10, 41]).

The paper is organized as follows. In Section 2 we introduce Euler–Maclaurin methods while in Section 3 we show their conjugate symplecticity properties. Section 4 is devoted to the efficient computation of the derivatives on the Infinity Computer. Finally, some numerical illustrations are presented in Section 5 while Section 6 contains some concluding remarks.

2. Euler–MacLaurin methods

Euler–Maclaurin methods are higher derivative collocation methods belonging to the class of Hermite-Obrechkov methods [42, page 277]. When applied to the general initial value problem

$$y'(t) = f(y(t)), \quad y(t_0) = y_0 \in \mathbb{R}^m, \tag{7}$$

they yield a polynomial $\sigma(t_0+ch)$ approximating the true solution $y(t)$ in the interval $[t_0, t_0+h]$ (h is the stepsize of integration) defined by means of the following collocation conditions at the ends of the interval

$$\begin{cases} \sigma(t_0) = y_0, \\ \sigma^{(j)}(t_0) = D_{j-1}f(\sigma(t_0)), \quad j = 1, \ldots, s, \\ \sigma^{(j)}(t_0+h) = D_{j-1}f(\sigma(t_0+h)), \quad j = 1, \ldots, s. \end{cases} \tag{8}$$

where, for any given vector z, $D_j f(z)$ denotes the total j-th time derivative of $f(y(t))$ evaluated at $y(t) = z$, with $y(t)$ formally satisfying (7):

$$D_j f(z) = \left.\frac{\mathrm{d}^j}{\mathrm{d}t^j} f(y(t))\right|_{y(t)=z}. \tag{9}$$

We have used here the subscript j to distinguish the operator defined in (9) (Lie derivative) from the classical time-derivative operator of order j which will be denoted, as usual, by D^j. The approximation at time $t_1 = t_0 + h$ is then yielded by $y_1 = \sigma(t_0+h) \simeq y(t_0+h) + O(h^{p+1})$, with $p = 2s$. Notice that the right-hand side of (9) may be expressed in terms of $y(t)$ via the relation (7). For example, for a given $z \in \mathbb{R}^m$,

$$D_1 f(z) = \left. f'(y(t))y'(t)\right|_{y(t)=z} = \left. f'(y(t))f(y(t))\right|_{y(t)=z} = f'(z)f(z),$$

where f' denotes the Jacobian matrix of the function f. More in general, the analytical computation of $D_j f(z) = D_1(D_{j-1}f(z))$ involves a tensor of order $j+1$. This considerably raises the computational cost associated with the implementation of the method as long as higher derivatives are considered. We will see that the use of the Infinity Computer circumvents this issue by producing a precise value of $D_j f(z)$ without explicitly evaluating its analytical expression in terms of the derivatives of f.

These integrators derive their name from the well-known Euler–Maclaurin integration formula: if m and n are natural numbers and $g(x)$ with $x \in \mathbb{R}$ is a regular function defined on $[m, n]$,

$$\begin{aligned} \int_m^n g(x)\mathrm{d}x &= \frac{g(m)+g(n)}{2} + \sum_{i=m+1}^{n-1} g(i) \\ &\quad - \sum_{k=1}^{s-1} \frac{B_{2k}}{(2k)!}\left(g^{(2k-1)}(n) - g^{(2k-1)}(m)\right) + R, \end{aligned} \tag{10}$$

where $s \geq 1$ and B_{2k} is the $2k$-th Bernoulli number.[1] The remainder R is bounded by

$$|R| \leq \frac{2}{(2\pi)^{2s-2}} \int_m^n \left|g^{(2s-1)}(x)\right| \mathrm{d}x.$$

We now consider the integral form of (7) in the interval $[t_0, t_0+h]$, namely

$$y(t_0+ch) = y(t_0) + h\int_0^c y'(t_0+\tau h)\mathrm{d}\tau \equiv y(t_0) + h\int_0^c f(y(t_0+\tau h))\mathrm{d}\tau.$$

[1] In formula (10) we have used $s-1$ in place of s to make the argument consistent with formulae (8). When $s = 1$ we get the standard composite trapezoidal quadrature rule. The first even Bernoulli number are $B_2 = 1/6$, $B_4 = -1/30$, $B_6 = 1/42$, $B_8 = -1/30, \ldots$.

Setting $c = 1$ and evaluating the integral by means of (10) with $m = 0$ and $n = 1$ yields

$$\begin{aligned} y(t_1) &= y(t_0) + h\int_0^1 y'(t_0+\tau h)\mathrm{d}\tau \\ &= y(t_0) + \frac{h}{2}\left(f(y(t_1)) + f(y(t_0))\right) - \sum_{k=1}^{s-1} \frac{h^{2k}B_{2k}}{(2k)!}\left(D^{2k-1}f(y(t_1)) - D^{2k-1}f(y(t_0))\right) + R. \end{aligned}$$

An approximation $y_1 \simeq y(t_1)$ is obtained by neglecting the remainder term $R = O(h^{2s+1})$. Taking into account (9) and considering that $y(t_0) = y_0$, we arrive at the one-step Euler-Maclaurin family of methods

$$y_1 = y_0 + \frac{h}{2}\left(f(y_1) + f(y_0)\right) - \sum_{k=1}^{s-1} \frac{h^{2k}B_{2k}}{(2k)!}\left(D_{2k-1}f(y_1) - D_{2k-1}f(y_0)\right). \tag{11}$$

When $s = 1$, (11) becomes the trapezoidal method while, for $s = 2$ and $s = 3$ we get the fourth and sixth order methods

$$y_1 = y_0 + \frac{h}{2}\left(f(y_1) + f(y_0)\right) - \frac{h^2}{12}\left(y_1'' - y_0''\right), \tag{12}$$

$$y_1 = y_0 + \frac{h}{2}\left(f(y_1) + f(y_0)\right) - \frac{h^2}{12}\left(y_1'' - y_0''\right) + \frac{h^4}{720}\left(y_1^{(4)} - y_0^{(4)}\right), \tag{13}$$

where, to simplify the notation, we have set $y_i^{(2k)} = D_{2k-1}f(y_i)$, $i = 0, 1$.

3. Conjugate symplecticity properties

To prove that the Euler-Maclaurin method (11) is conjugate to a symplectic method up to order $2s + 2$, we show that the map $y_1 = \Psi_h(y_0)$ associated with (11) is such that $\Psi_h(y) = \Phi_h(y) + O(h^{2s+3})$, where $y_1 = \Phi_h(y_0)$ is a suitable B-series integrator satisfying property (a) of the following lemma.

Lemma 1. *[43] Assume that problem (7) admits a quadratic first integral $Q(y) = y^\top S y$ (with S a symmetric matrix) and is solved by a B-series integrator $\Phi_h(y)$. The following properties are equivalent:*

(a) $\Phi_h(y)$ has a modified first integral of the form $\widetilde{Q}(y) = Q(y) + O(h)$;
(b) $\Phi_h(y)$ is formally conjugate to a symplectic B-series method.

Here, *formally conjugate* means that the power series yielding the conjugacy needs not be convergent. This result was stated in [43] and subsequently used in [44] to derive the conjugate symplecticity property of symmetric multistep methods.

In terms of the characteristic polynomials

$$\rho(z) = z - 1, \qquad \sigma(z) = \frac{1}{2}(z+1)$$

formula (11) reads, for a generic time t_n,

$$\rho(E)y_n = h\sigma(E)f(y_n) - \sum_{k=1}^{s-1} \frac{h^{2k}B_{2k}}{(2k)!}\rho(E)D_{2k-1}f(y_n), \tag{14}$$

where E denotes the shift operator: $E(y_n) = y_{n+1}$.

For an analytic function $u(t)$ and a stepsize $h > 0$, we introduce the shift operator $E_h(u(t)) = u(t+h)$ and recall the relation

$$E_h = e^{hD} = \sum_{k=0}^{\infty} \frac{h^k}{k!} D^k. \tag{15}$$

Theorem 1. *The map $y_1 = \Psi_h(y_0)$ associated with the one-step method (14) admits a B-series expansion and is conjugate-symplectic up to order $2s+2$.*

Proof. The existence of a B-series expansion for $y_1 = \Psi_h(y_0)$ may be directly deduced from [4], where a B-series representation of a generic multi-derivative Runge-Kutta method has been obtained. From the generating function of Bernoulli numbers (see, for example [45])

$$\frac{z}{e^z - 1} = \sum_{k=0}^{\infty} B_k \frac{z^k}{k!},$$

we get, considering that $B_1 = -1/2$ and $B_k = 0$ for k odd,

$$\frac{z\sigma(e^z)}{\rho(e^z)} = \frac{1}{2}\frac{z(e^z+1)}{e^z-1} = \frac{z}{e^z-1} + \frac{z}{2} = 1 + \sum_{k=1}^{\infty} B_{2k} \frac{z^{2k}}{(2k)!}. \tag{16}$$

In the spirit of backward analysis, we look for an analytical function $u(t)$ formally satisfying the difference equation (14) that is, by virtue of (15),

$$\rho(e^{hD})u(t) = h\sigma(e^{hD})f(u(t)) - \sum_{k=1}^{s-1} \frac{B_{2k}}{(2k)!} h^{2k} \rho(e^{hD}) D_{2k-1} f(u(t)).$$

Multiplying both sides of the previous equation by $D\rho(e^{hD})^{-1}$ yields

$$\dot{u}(t) = hD\rho(e^{hD})^{-1}\sigma(e^{hD})f(u(t)) - \sum_{k=1}^{s-1} \frac{B_{2k}}{(2k)!} h^{2k} D D_{2k-1} f(u(t)),$$

and, by taking into account (16), we finally arrive at

$$\dot{u}(t) = \left(1 + \sum_{k=s}^{\infty} \frac{B_{2k}}{(2k)!} h^{2k} D^{2k}\right) f(u(t)) + \sum_{k=1}^{s-1} \frac{B_{2k}}{(2k)!} h^{2k} D(D^{2k-1} - D_{2k-1}) f(u(t)). \tag{17}$$

Equation (17) coupled with the initial condition $u(t_0) = y_0$ is nothing but the *modified differential equation* associated with the Euler–MacLaurin method of order $2s$, so that $u(t_0 + nh) = y_n$.

Since $u(t) = y(t) + O(h^{2s})$, we see that $(D^{2k-1} - D_{2k-1})f(u(t)) = O(h^{2s})$ and hence the solution $u(t)$ of (17) is $O(h^{2s+2})$-close to the solution of the following initial value problem

$$\dot{u}(t) = \left(1 + \sum_{k=s}^{\infty} \frac{B_{2k}}{(2k)!} h^{2k} D^{2k}\right) f(u(t)), \qquad u(t_0) = y_0. \tag{18}$$

We may interpret (18) as the modified equation of a one-step method $y_1 = \Phi_h(y_0)$, where Φ_h is evidently the time-h flow associated with (18). Expanding the solution of (18) in Taylor series, we get

$$\begin{aligned} \Phi_h(y_0) \quad = \quad & y_1 = u(t_0+h) = y_0 + hf(y_0) + \sum_{k=s}^{\infty} \frac{B_{2k}}{(2k)!} h^{2k+1} D^{2k} f(y_0) \\ & + \frac{h^2}{2!} f'(y_0) f(y_0) + \sum_{k=s}^{\infty} \frac{B_{2k}}{(2k)!} h^{2k+2} D^{2k+1} f(y_0) + \dots . \end{aligned}$$

where $f'(y)$ is the Jacobian of $f(y)$ and we have set $D^r f(y_0) = D^r f(u(t))|_{t=t_0}$. Collecting like powers of h in the above expression yields a formal power series expansion in the stepsize h, that is a B-series expansion.

To show that $\Phi_h(y)$ admits a modified first integral $\widetilde{Q}(y) = Q(y) + O(h^{2s})$, we follow the same flow of computation appearing in [2, Theorem 4.10 on page 591], that states an analogous property for symmetric linear multistep methods. We first notice that

$$\left(1 + \sum_{k=s}^{\infty} \frac{B_{2k}}{(2k)!} z^{2k}\right)^{-1} = 1 + \sum_{k=s}^{\infty} \gamma_k z^{2k},$$

for suitable coefficients γ_k. Thus (18) is tantamount to

$$\left(1 + \sum_{k=s}^{\infty} \gamma_k h^{2k} D^{2k}\right) \dot{u}(t) = f(u(t)). \tag{19}$$

Multiplying both sides of (19) by the term $u(t)^\top S$ yields

$$\frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}t} Q(u(t)) + \sum_{k=s}^{\infty} \gamma_k h^{2k} u(t)^\top S u^{(2k+1)}(t) = 0, \tag{20}$$

where we have taken into account that $2u(t)^\top S\dot{u}(t) = \dot{Q}(u(t))$ and $z^\top S f(z) = 0$ for any $z \in \mathbb{R}^m$, since $Q(y)$ is a first integral of the original system (7). A repeated use of the property (the explicit dependence on the time t is omitted to simplify the notation)

$$u^{(i)\top} S u^{(j)} = \frac{\mathrm{d}}{\mathrm{d}t}\left(u^{(i)\top} S u^{(j-1)}\right) - u^{(i+1)\top} S u^{(j-1)},$$

with, in particular,

$$u^{(i)\top} S u^{(i+1)} = \frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}t}\left(u^{(i)\top} S u^{(i)}\right),$$

allows us to cast each term $u(t)^\top S u^{(2k+1)}(t)$ in (20) as

$$u^\top S u^{(2k+1)} = \frac{\mathrm{d}}{\mathrm{d}t}\left(u^\top S u^{(2k)} - \dot{u}^\top S u^{(2k-1)} + \cdots + (-1)^k \frac{1}{2} u^{(k)\top} S u^{(k)}\right).$$

We observe that the sum in brackets on the right hand side may be formally cast as a function of $u(t)$ by replacing all the derivatives with the aid of the modified differential equation (18). After this substitution, denoting by

$$Q_k(u(t)) = 2\gamma_k u(t)^\top S u^{(2k+1)}(t)$$

and $\widetilde{Q}(u) = Q(u) + \sum_{k=s}^{\infty} h^{2k} Q_k(u)$, from (20) we finally obtain $\frac{\mathrm{d}}{\mathrm{d}t}\widetilde{Q}(u(t)) = 0$ which concludes the proof, since $\Psi_h(y) = \Phi_h(y) + O(h^{2s+3})$. □

4. Computation of the derivatives

One drawback with these implicit methods is the computation of high order derivatives. Symbolic or automatic differentiation are often preferred to finite differences techniques involving terms in y and y', which suffer from numerical instability when the increment becomes small.

This drawback is overcome on the Infinity Computer and hereafter we illustrate two possible approaches in order to compute the k-th derivative of $y(t)$ at time t_i.

Strategy (a). This strategy was first proposed in [8]. We perform k infinitesimal steps starting at time t_i using the explicit Euler formula with stepsize $\bar{h} = ①^{-1}$ as follows:

$$y_{i,1} = y_i + ①^{-1} f(y_i), \quad y_{i,2} = y_{i,1} + ①^{-1} f(y_{i,1}), \ldots, \quad y_{i,k} = y_{i,k-1} + ①^{-1} f(y_{i,k-1}).$$

Then, the values of the needed derivatives can be obtained by means of the forward differences $F^k_{\bar{h}}[y_{i,0}, y_{i,1}, \ldots, y_{i,k}]$, with $\bar{h} = ①^{-1}$ as follows

$$y^{(k)}(t_i) = \frac{F^k_{①^{-1}}[y_{i,0}, y_{i,1}, \ldots, y_{i,k}]}{①^{-k}} + O(①^{-1}) \tag{21}$$

where

$$F^k_{①^{-1}}[y_{i,0}, y_{i,1}, \ldots, y_{i,k}] = \sum_{j=0}^{k} (-1)^j \binom{k}{j} y_{i,k-j}, \quad y_{i,0} = y_i. \tag{22}$$

As was proven in [8], since the error of the approximation is $O(①^{-1})$, the finite part of the value $\frac{F^k_{①^{-1}}[y_{i,0}, y_{i,1}, \ldots, y_{i,k}]}{①^{-k}}$ gives the *exact* derivative $y^{(k)}(t_i)$. For a more detailed description of the numerical computation of exact derivatives on the Infinity Computer, see [8].

Strategy (b). Let us propose another strategy for computing the exact derivatives, where finite differences may be employed directly on the value of f as follows:

$$y^{(k)}(t_i) = D^{k-1} f(y_i) = \frac{F^{k-1}_{①^{-1}}[f(y_{i,0}), f(y_{i,1}), \ldots, f(y_{i,k-1})]}{①^{-(k-1)}} + O(①^{-1}) \tag{23}$$

Now, let us prove that formulae (21) and (23) are equivalent.

Proposition 1. *Let us suppose that for the solution $y(t)$ of the ordinary differential equation (7) it is known the value $y_i = y(t_i)$ at the point t_i. Then formulae (21) and (23) are equivalent.*

Proof. Since formulae (21) and (23) differ only in the forward differences, then in order to prove the proposition, it is sufficient to demonstrate that the following equality holds true

$$F^k_{①^{-1}}[y_{i,0}, y_{i,1}, \ldots, y_{i,k}] = ①^{-1} \cdot F^{k-1}_{①^{-1}}[f(y_{i,0}), \ldots, f(y_{i,k-1})], \tag{24}$$

where $y_{i,0} = y_i$. Let us use the mathematical induction to prove it. For $k = 1$, the assertion is trivial. However, since the first meaningful case is for $k = 2$ and in order to make the reader more acquainted with grossone based formalism, we consider this case in full detail. By using formulae (21)–(23) for $y_{i,0},\ y_{i,1}$, and $y_{i,2}$ we obtain

$$\begin{aligned} F^2_{①^{-1}}[y_{i,0}, y_{i,1}, y_{i,2}] &= y_{i,2} - 2y_{i,1} + y_{i,0} \\ &= y_{i,0} + ①^{-1} \cdot (f(y_{i,0}) + f(y_{i,1})) - 2(y_{i,0} + ①^{-1}) \cdot f(y_{i,0}) + y_{i,0} \\ &= ①^{-1} \cdot (f(y_{i,1}) - f(y_{i,0})) = ①^{-1} \cdot F^1_{①^{-1}}[f(y_{i,0}), f(y_{i,1})]. \end{aligned}$$

Suppose now that (24) holds for $k-1,\ k \geq 3$. We get

$$\begin{aligned} F^k_{①^{-1}}[y_{i,0}, \ldots, y_{i,k}] &= F^{k-1}_{①^{-1}}[y_{i,1}, \ldots, y_{i,k}] - F^{k-1}_{①^{-1}}[y_{i,0}, \ldots, y_{i,k-1}] \\ &= ①^{-1} \cdot F^{k-2}_{①^{-1}}[f(y_{i,1}), \ldots, f(y_{i,k-1})] \\ &\quad -①^{-1} \cdot F^{k-2}_{①^{-1}}[f(y_{i,0}), \ldots, f(y_{i,k-2})] \\ &= ①^{-1} \cdot F^{k-1}_{①^{-1}}[f(y_{i,0}), \ldots, f(y_{i,k-1})]. \end{aligned}$$

This completes the proof. □

The advantage of strategy (b) with respect to strategy (a) is that the use of formula (23) in place of (21) decreases the computational costs due to the following reasons:

1. It is not necessary to compute the value $y_{i,k}$ for (23), whereas it should be calculated in (21).
2. All the computations using (21) should be performed using the grosspowers up to $-k$. In contrast, formula (23) allows us to work with the numbers using only the grosspowers up to $-(k-1)$.

Let us consider the following example from [10] in order to illustrate these issues.

Example 1. *Let us find the first 3 derivatives $y'(t_0)$, $y''(t_0)$, and $y'''(t_0)$ of the solution $y(t)$ at the point $t_0 = 0$ of the following initial value problem:*

$$\frac{dy}{dt} = \frac{y - 2ty^2}{1+t},\ y(t_0) = 0.4, \tag{25}$$

whose exact solution is

$$y(t) = \frac{1+t}{2.5+t^2}. \tag{26}$$

Differentiating (26) we get the exact values of the following derivatives:

$$y'(t_0) = 0.4, \quad y''(t_0) = -0.32, \quad y'''(t_0) = -0.96.$$

Now, let us find these derivatives using strategy (a). First, we perform 3 iterations of the Euler method with the integration step $\bar{h} = ①^{-1}$, truncating all values after the grosspower -3:

$$\begin{aligned}
y_1 &= y_0 + ①^{-1} f(t_0, y_0) = 0.4 + 0.4①^{-1}, \\
y_2 &= y_1 + ①^{-1} f(t_0 + ①^{-1}, y_1) = 0.4 + 0.8①^{-1} - 0.32①^{-2} - 0.32①^{-3}, \\
y_3 &= y_2 + ①^{-1} f(t_0 + 2①^{-1}, y_2) = 0.4 + 1.2①^{-1} - 0.96①^{-2} - 1.92①^{-3}.
\end{aligned}$$

Applying formulae (21), (22), we obtain

$$\begin{aligned}
y'(t_0) &\simeq ① \cdot F^1_{①^{-1}}[y_0, y_1] = ① \cdot (y_1 - y_0) \\
&= ① \cdot (0.4 + 0.4①^{-1} - 0.4) = 0.4,
\end{aligned}$$

$$\begin{aligned}
y''(t_0) &\simeq ①^2 \cdot F^2_{①^{-1}}[y_0, y_1, y_2] = ①^2 \cdot (y_2 - 2y_1 + y_0) \\
&= ①^2 \cdot (0.4 + 0.8①^{-1} - 0.32①^{-2} - 0.32①^{-3} - 2(0.4 + 0.4①^{-1}) + 0.4) \\
&= -0.32 - 0.32①^{-1} = -0.32 + O(①^{-1}),
\end{aligned}$$

$$\begin{aligned}
y'''(t_0) &\simeq ①^3 \cdot F^3_{①^{-1}}[y_0, y_1, y_2, y_3] = ①^3 \cdot (y_3 - 3y_2 + 3y_1 - y_0) \\
&= ①^3 \cdot (0.4 + 1.2①^{-1} - 0.96①^{-2} - 1.92①^{-3} \\
&\quad -3(0.4 + 0.8①^{-1} - 0.32①^{-2} - 0.32①^{-3}) + 3(0.4 + 0.4①^{-1}) - 0.4) \\
&= -0.96,
\end{aligned}$$

from where we can extract the exact values of $y'(t_0)$, $y''(t_0)$, and $y'''(t_0)$ as finite parts of 0.4, $-0.32 - 0.32①^{-1}$, and -0.96, respectively.

Let us now apply strategy (b). Here, we need to perform $k-1$ iterations of the Euler method, obtaining the values y_1 and y_2, truncating them after the grosspower -2:

$$\begin{aligned}
f(t_0, y_0) &= 0.4, \\
f(t_0 + ①^{-1}, y_1) &= 0.4 - 0.32①^{-1} - 0.32①^{-2}, \\
f(t_0 + 2①^{-1}, y_2) &= 0.4 - 0.64①^{-1} - 1.6①^{-2}.
\end{aligned}$$

Applying formulae (23), we obtain

$$\begin{aligned} y'(t_0) &= ①^0 \cdot F^0_{①^{-1}}[f(t_0,y_0)] = f(t_0,y_0) = 0.4, \\ y''(t_0) &\approx ①^1 \cdot F^1_{①^{-1}}[f(t_0,y_0), f(t_0+①^{-1},y_1)] \\ &= ① \cdot (f(t_0+①^{-1},y_1) - f(t_0,y_0)) \\ &= ① \cdot (0.4 - 0.32①^{-1} - 0.32①^{-2} - 0.4) = -0.32 - 0.32①^{-1} \\ &= -0.32 + O(①^{-1}), \end{aligned}$$

$$\begin{aligned} y'''(t_0) &\approx ①^2 \cdot F^2_{①^{-1}}[f(t_0,y_0), f(t_0+①^{-1},y_1), f(t_0+2①^{-1},y_2)] \\ &= ①^2 \cdot (f(t_0+2①^{-1},y_2) - 2f(t_0+①^{-1},y_1) + f(t_0,y_0)) \\ &= ①^2 \cdot (0.4 - 0.64①^{-1} - 1.6①^{-2} - 2(0.4 - 0.32①^{-1} - 0.32①^{-2}) + 0.4) \\ &= -0.96, \end{aligned}$$

from where again we can extract the exact values of $y'(t_0)$, $y''(t_0)$, *and* $y'''(t_0)$ *as finite parts of* 0.4, $-0.32 - 0.32①^{-1}$, *and* -0.96, *respectively.*

It should be noticed that the value y_2 *cannot be truncated after the grosspower* -2 *using the first strategy, because the coefficient of* $①^{-3}$ *at the value* y_2 *is used also for computing* $y'''(t_0)$. *On the contrary, strategy (b) allows us to use the grosspowers up to* -2, *which decreases the computational cost of the procedures computing the* $2-nd$ *and the* $3-rd$ *derivatives.*

5. Numerical illustrations

In the present section, the Euler–Maclaurin formulae of order four (12) and six (13) are applied to a few well-known test problems to highlight their conservation properties. In particular, the long-time behavior of their numerical solutions is compared with that of the numerical solutions computed by the (symplectic) Gauss methods of order four and six.

To advance the solution at each integrations step, the nonlinear equations (12) and (13) have been solved by means of a modified Newton method, using the same Jacobian of the trapezoidal scheme, which is appropriate since the neglected terms are $O(h^2)$. In order to preserve the conservation properties, the nonlinear scheme must be iterated to attain the highest possible accuracy in double precision. Moreover, the derivatives have been computed using strategy (b) defined in the previous section.

The numerical experiments have been performed using Matlab R2017b. Though Gauss methods generally exhibit a better accuracy for a given stepsize and order, we stress that a fair comparison of the actual performance of the two classes of methods cannot take aside the computational complexity associated with their implementation and, in particular, the effort in solving the underlying nonlinear systems at each step of the integration procedure. In this respect, we notice that while the dimension of the nonlinear systems associated with the Gauss methods is proportional to the number of stages and hence to the considered order, this is not the case for the Euler–Maclaurin formulae, for which the dimension remains the same, i.e. that of the underlying continuous problem, independently of the order. This study is a delicate issue due to the non-homogeneous platforms the two codes have been run. At the moment, the only available emulator of the Infinity Computer Arithmetic is a c++ prototype callable in Matlab through a suitable interface which prohibits us to have an efficient implementation and therefore to execute a fair comparisons with other techniques. We stress that the c++ emulator has been only used for the computation of the derivatives, while all the other operations have been performed using

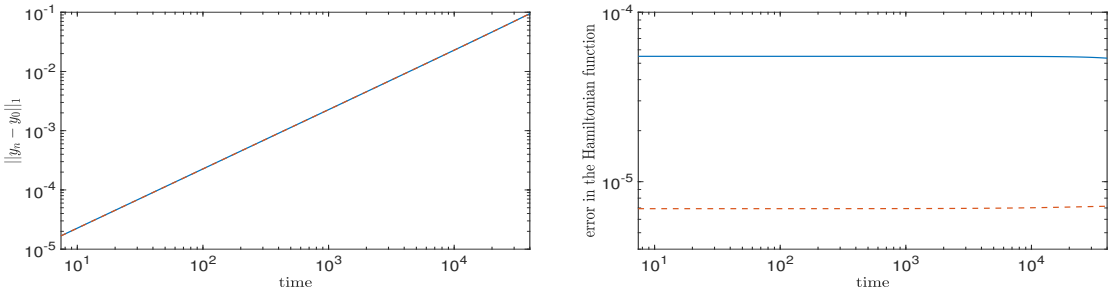

Figure 1: Results for the fourth-order Euler-Maclaurin method (solid lines) and Gauss method (dashed lines) applied to the pendulum problem.

the standard double precision floating point arithmetic available in Matlab. We checked the obtained results with those provided by computing the derivatives analytically.

With this premise, to gain some preliminary insight into the potentialities of multi-derivative methods, we have prepared a couple of experiments involving the pendulum and the Kepler problems, for which the periodic nature of the solution allows us to accurately estimate the error. In particular, taking into account the discussion above, we compare the fourth-order Euler-Maclaurin and Gauss methods on the basis of their computational complexity but under the assumption that the derivative $f'(y)f(y)$ needed by the the former integrator be analytically evaluated. The results have been discussed in the next two subsections and collected in Figure 3.

5.1. Nonlinear pendulum

As a first example, we consider the dynamics of a pendulum under influence of gravity. It is usually described in terms of the angle q that the pendulum forms with its stable rest position:

$$\ddot{q} + \sin q = 0, \tag{27}$$

where $p = \dot{q}$ is the angular velocity. The Hamiltonian function associated with (27) is

$$H(q, p) = \frac{1}{2}p^2 - \cos q. \tag{28}$$

An initial condition (q_0, p_0) such that $|H(q_0, p_0)| < 1$ gives rise to a periodic solution $y(t) = (q(t), p(t))^\top$ corresponding to oscillations of the pendulum around the straight-down stationary position. In particular, starting at $y_0 = (q_0, 0)^\top$, the period of oscillation may be expressed in terms of the complete elliptical integral of the first kind as

$$T(q_0) = \int_0^1 \frac{\mathrm{d}z}{\sqrt{(1 - z^2)(1 - \sin^2(q_0/2)z^2)}}.$$

We choose $q_0 = \pi/2$ to which there corresponds a period $T = 7.416298709205487$. We use the fourth and sixth order Euler–Maclaurin and Gauss methods with stepsize $h = T/28$ to integrate the problem over $5{\cdot}10^3$ periods for the fourth-order methods and $4{\cdot}10^5$ periods for the sixth-order methods. We then compute the errors $\|y_n - y_0\|_1$ in the solution and $\max_{1\le j\le 28} |H(y_{n+j}) - H(y_0)|$ in the energy function at times multiples of the period T, that is for $n = 28k$, with $k = 1, 2, \ldots$. Figures 1 and 2 report the obtained results. On the left plot, we can see that the error in the

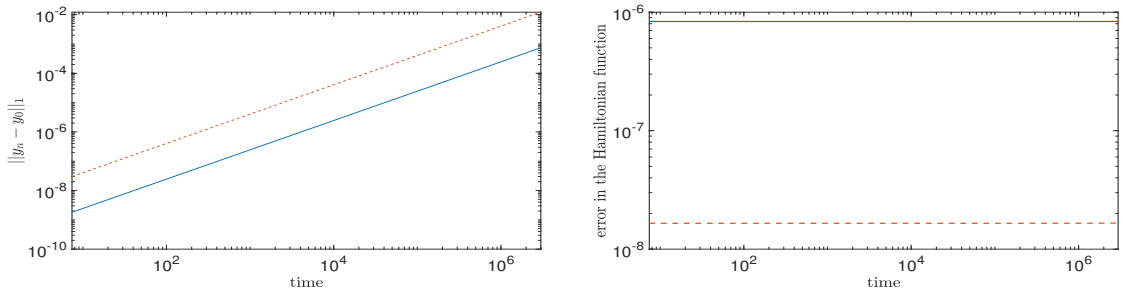

Figure 2: Results for the sixth-order Euler-Maclaurin method (solid lines) and Gauss method (dashed lines) applied to the pendulum problem.

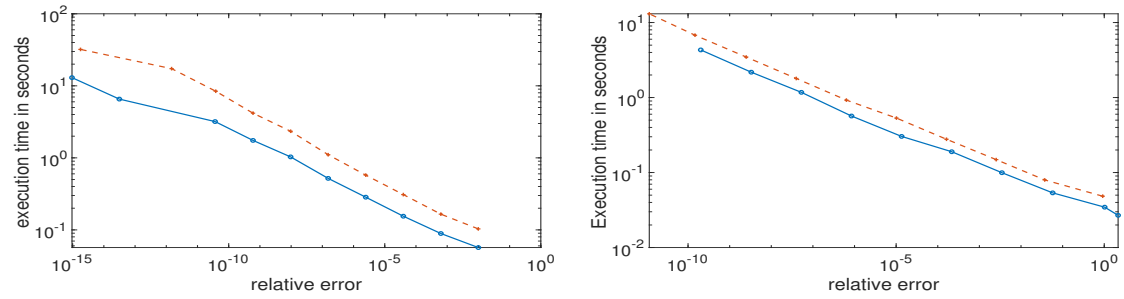

Figure 3: Work precision diagrams for the fourth-order Euler-Maclaurin method (solid lines) and Gauss method (dashed lines) applied to the pendulum problem (left picture) and to the Kepler problem (right picture).

solution as time increases is essentially the same for the fourth-order formulae and quite similar for the sixth-order formulae. A near conservation of the energy function is observable on the right of each figure.

The work precision diagram in the left picture of Figure 3 plots the execution times versus the accuracy in the numerical solutions obtained by the fourth-order Euler-Maclaurin and Gauss methods applied to integrate the pendulum problem on the interval $[0, 10^2T]$. The results show that the Euler-Maclaurin formula is indeed competitive. We emphasize that the Gauss method has been implemented by using the efficient techniques described in [46].

5.2. The Kepler problem

This classical problem describes the motion of two bodies subject to Newton's law of gravitation. As is well-known, the problem is a completely integrable Hamiltonian dynamical system with two degrees of freedom (see, for example, [47]). If the origin of the coordinate system is set on one of the two bodies, the Hamiltonian function

$$H(q_1, q_2, p_1, p_2) = \frac{1}{2}(p_1^2 + p_2^2) - \frac{1}{\sqrt{q_1^2 + q_2^2}},$$

describes the motion of the other body, namely an ellipse in the q_1-q_2 plane. Taking as initial conditions

$$q_1(0) = 1 - e, \quad q_2(0) = 0, \quad p_1(0) = 0, \quad p_2(0) = \sqrt{\frac{1+e}{1-e}},$$

	order 4		order 6	
N	error	rate	error	rate
32	8.47e-03	4.10	2.59e-03	6.40
64	4.92e-04	4.01	3.07e-05	6.08
128	3.04e-05	4.00	4.53e-07	5.99
256	1.90e-06	4.00	7.10e-09	6.00
512	1.18e-07	4.00	1.11e-10	5.99
1024	7.42e-09	4.00	1.73e-12	5.77

Table 1: Convergence rate of the error in the angular moment for the fourth-order and sixth-order Euler-Maclaurin methods. N is the number of mesh points in each period, the error is computed over 10 periods.

the trajectory describes an ellipse with eccentricity e and is periodic with period $T = 2\pi$. Besides the total energy H, further relevant first integrals are the angular momentum

$$M(q_1, q_2, p_1, p_2) = q_1 p_2 - q_2 p_1.$$

and the Lenz vector $A = (A_1, A_2, A_3)^\top$, whose components are

$$A_1(q,p) = p_2 M(q,p) - \frac{q_1}{||q||_2}, \quad A_2(q,p) = -p_1 M(q,p) - \frac{q_2}{||q||_2}, \quad A_3(q,p) = 0.$$

Of the four first integrals H, M, A_1, and A_2 only three are independent. Having set $e = 0.6$ and $h = T/400$, we integrate the problem over $8 \cdot 10^2$ periods for the fourth-order methods and 10^5 periods for the sixth-order methods and compute the error $\|y_n - y_0\|_1$ in the solution at specific times multiples of the period T, that is for $n = 400k$, with $k = 1, 2, \ldots$. Figures 4 and 5 report the obtained results for the fourth and sixth order Euler–Maclaurin (solid lines) and Gauss (dashed lines) methods. On the top-left picture is the absolute error of the numerical solution; the top-right picture shows the maximum error in the Hamiltonian function in each period; the error in the angular momentum, also evaluated in each period, is drawn in the bottom-left picture while the bottom-right picture concerns the maximum error in each period of the Lenz vector. As is expected, we can see a linear drift in the error $\|y_n - y_0\|_1$ as the time increases. The same linear growth is experienced in the Lenz invariant. Euler–Maclaurin methods assure a near conservation of the Hamiltonian function and angular momentum. This latter quadratic invariant is precisely conserved (up to machine precision) by Gauss methods due to their symplecticity property.

Finally, also for this problem, we have run the fourth-order Euler-Maclaurin and Gauss methods for decreasing values of the stepsize over 10 periods, and stored the corresponding errors and execution times. The work precision diagram in the right picture of Figure 3 collects the obtained results and shows a very similar behavior of the two integrators. The convergence rate of the maximum absolute error in the angular momentum, computed over the all integration interval, is consistent with the order of the methods, testifiyng the near conservation of this first quadratic integral, see Table 1.

5.3. Fermi-Pasta-Ulam problem

The Fermi-Pasta-Ulam problem models a physical system composed by $2m$ unit point masses disposed along a line and chained together by alternating weak nonlinear springs and stiff linear springs[47, 2]. The force exerted by the nonlinear springs are assumed proportional to the cube of the displacement of the associated masses. Denoting by q_1, q_2, ..., q_{2m} the displacements of the masses from their rest points and assuming the endpoints of the external springs to be fixed,

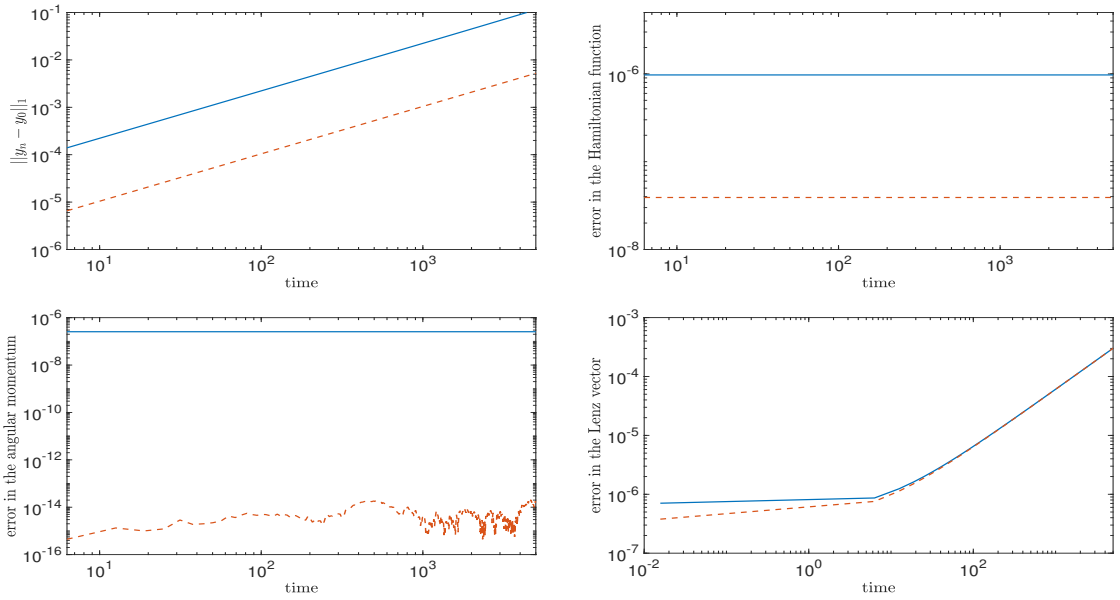

Figure 4: Results for the fourth-order Euler-Maclaurin method (solid lines) and Gauss method (dashed lines) applied to the Kepler problem.

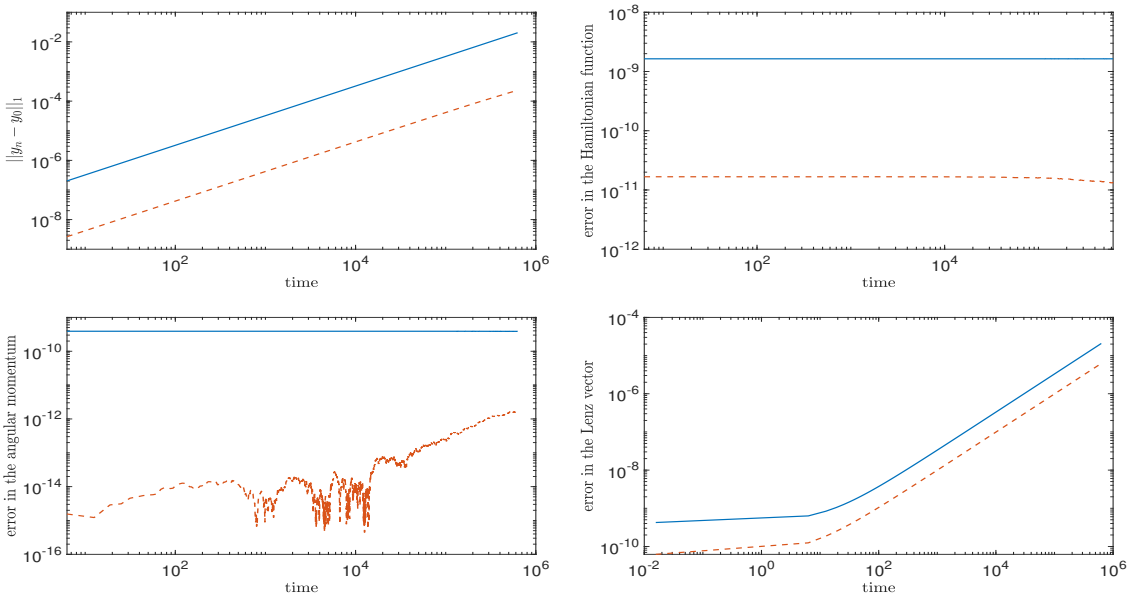

Figure 5: Results for the sixth-order Euler-Maclaurin method (solid lines) and Gauss method (dashed lines) applied to the Kepler problem.

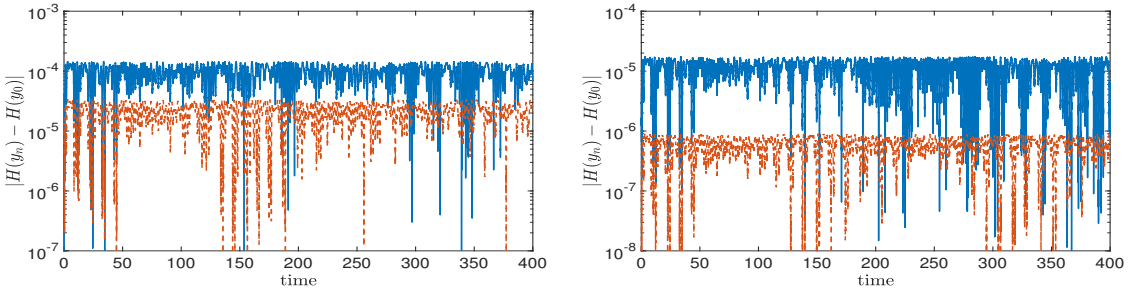

Figure 6: Error in the Hamiltonian function (29) generated by the Euler Maclaurin methods (solid line) and Gauss methods (dashed line) of order 4 (left picture) and order 6 (right picture) applied to the Fermi-Pasta-Ulam problem.

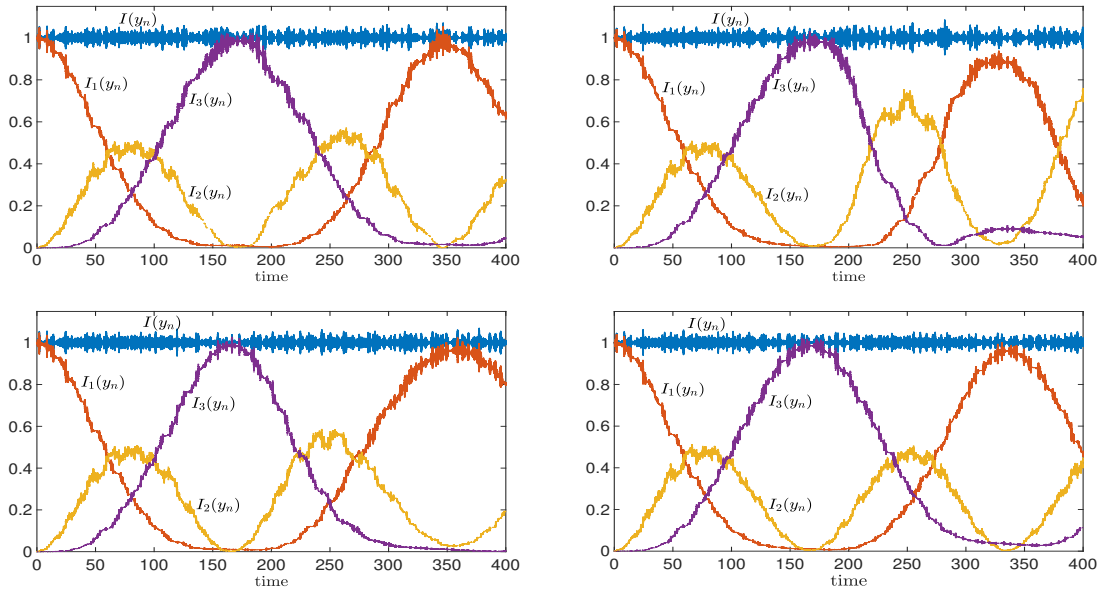

Figure 7: Fermi-Pasta-Ulam problem: energy functions $I_i(t)$ associated with each linear spring and their sum $I(t)$ computed by Euler-Maclaurin methods (left pictures) and Gauss methods (right pictures). The upper and bottom pictures refer to the formulae of order four and six respectively.

$q_0 = q_{2m+1} = 0$, the resulting Hamiltonian problem is defined by the energy function

$$H(q,p) = \frac{1}{2}\sum_{i=1}^{m}(p_{2i-1}^2 + p_{2i}^2) + \frac{\omega^2}{4}\sum_{i=1}^{m}(q_{2i} - q_{2i-1})^2 + \sum_{i=0}^{m}(q_{2i+1} - q_{2i})^4, \tag{29}$$

where $p_i = \dot{q}_i$, $i = 1, \ldots, 2m$ are the conjugate momenta, and ω is the stiffness coefficient of the linear springs. Following the discussion in [2, page 22], we introduce the energy I_i associated with the ith linear spring

$$I_i = \frac{1}{4}\left((p_{2i} - p_{2i-1})^2 + \omega^2(q_{2i} + q_{2i-1})\right).$$

The total energy $I(t) = I_1 + \cdots + I_m$ brought by the linear springs satisfies $I(t) = I(t_0) + O(\omega^{-1})$, so that it is almost conserved for large values of the stiffness coefficient ω. In our experiments we choose $m = 3$ and $\omega = 50$ and integrated the problem on the interval $[0, 400]$ with stepsize $h = 0.03$ and the initial values $p_0 = (0, \sqrt{2}, 0, 0, 0, 0)$ and $q_0 = (\frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2w}, \frac{\sqrt{2}}{w} + \frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2w}, 0, 0, 0, 0)$. The two pictures in Figure 6 display the absolute error in the Hamiltonian function (29) evaluated along the numerical solution produced by Euler–Maclaurin (solid line) and Gauss (dashed line) methods. On the left are the results for the fourth-order formulae, while on the right are the results for the sixth order formulae. We can see that, for both methods, the Hamiltonian function is nearly conserved. The pictures in Figure 7 suggest that the very same conclusions apply to the nearly conserved quantity $I(t)$ above defined: there is an exchange of energy among the linear modes but the total energy does not exhibit any drift.

5.4. A non-separable Hamiltonian problem

As last example, we illustrate the behavior of the Euler-Maclaurin formulae on the non-separable problem defined by the Hamiltonian function

$$H(q,p) = (q^2 + p^2)^2 - 2a^2(q^2 - p^2). \tag{30}$$

The level curves defining the trajectories in the phase plane are the well-known Cassini ovals (see, for example, [47, page 53]). The shape of the orbit originating from a point (q_0, p_0) depends on the value of the quantity $r = (1 + H(q_0, p_0)/a^2)^{1/4}$. The three possible cases are summarized in the left picture of Figure 8. For $r < 1$, the orbit is a loop surrounding one of the foci $F_1 = (-a, 0)$ or $F_2 = (a, 0)$ and entirely lying on the semi-plane $q > 0$ or $q < 0$, respectively. For $r > 1$, the orbit is an oval or bone-shaped loop embracing both the foci. The figure-eight level curve corresponding to $r = 1$ is the lemniscate of Bernoulli and acts as a separatrix between the two possible dynamics described above.

Consequently, a correct reproduction of the orbit when $H(q_0, p_0) \approx 0$ requires the use of an integrators with good geometric properties. To show that this is indeed the case, in the right picture of Figure 8 we display the orbits originating from the point $(q_0, p_0) = (0, 10^{-2})$ computed by the Euler-Maclaurin formula of order 4, and by the explicit and implicit Taylor methods of the same order. This latter multi-derivative method is defined by considering a truncated Taylor expansion of $f(y(t))$ at time $t_{n+1} = t_n + h$ thus generalizing the implicit Euler method. The value of the Hamiltonian function associated with the true solution is $H(q_0, p_0) \approx 5 \cdot 10^{-4}$ corresponding to a value of $r \approx 1.00005 > 1$; the integration time interval is $[0, 45]$, while the used stepsize is $h = 1.5 \cdot 10^{-2}$. Both Taylor methods exhibit a dramatic drift in the Hamiltonian function thus failing to yield the correct shape of the orbit even for short times. On the contrary, the near conservation of (30) along the numerical solution generated by the Euler-Maclaurin integrator guarantees a very accurate orbit determination over long simulation times.

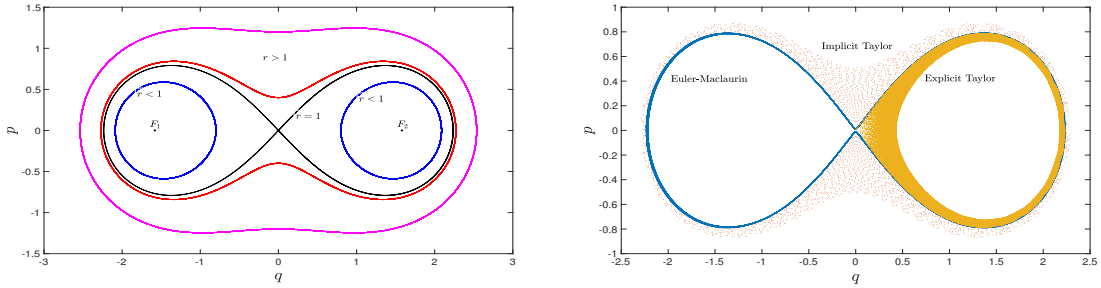

Figure 8: Left picture: possible shapes of Cassini ovals for different values of the parameter r. Right picture: orbits generated by the fourth-order Euler-Maclaurin, explicit Taylor and implicit Taylor methods.

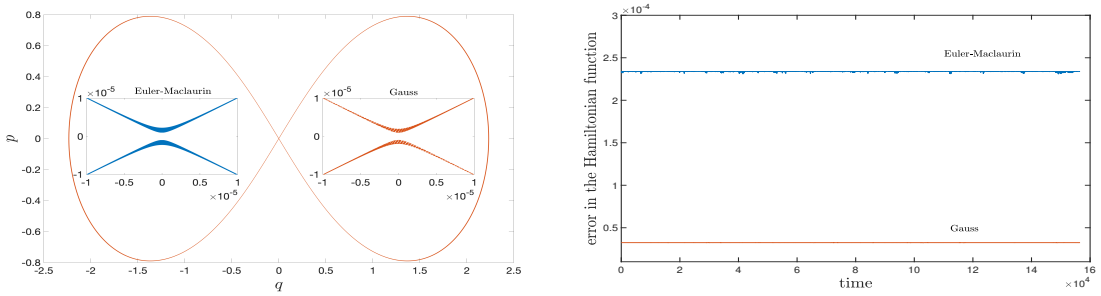

Figure 9: Comparison between the Euler-Maclaurin and Gauss methods of order four applied to problem (30). Left picture: the orbits computed by the two integrators overlap with each other. The two close-ups show that their behavior is very similar in a neighbourhood of the origin. Right picture: error in the Hamiltonian function evaluated as the maximum absolute error in each period.

This feature is further confirmed by the results displayed in Figure 9 where we compare the qualitative behavior of the fourth-order Euler-Maclaurin and Gauss methods in reproducing the orbit originating at $(q_0, p_0) = (0, 10^{-6})$. This orbit is characterized by a period $T \approx 3.131990057003955$ and a value of $r \approx 1 + 10^{-13}$ and thus is very close (but external) to the lemniscate (see [47, page 135]). We solved the problem over the time interval $[0, 5 \cdot 10^4 T]$ by using a stepsize $h = 2.5 \cdot 10^{-3} T$, corresponding to 400 mesh points in each period. The left picture of Figure 9 contains the two undistinguishable numerical trajectories in the phase plane together with two close-ups showing that, for both methods, the computed orbit is correctly bounded away from the origin. As it is clear from the right picture of Figure 9, this good asymptotic behavior is a consequence of the fact that the numerical Hamiltonian function $H(q_n, p_n)$ undergoes very small and bounded oscillations which prevent the trajectory to cross the lemniscate at any observed time.

6. Conclusions

This paper studies the conservation properties of Euler–Maclaurin formulae and their implementation on the Infinity Computer. These are a family of multi-derivative one-step methods containing the classical trapezoidal method as seed formula. The higher-order Euler–Maclaurin methods have even order $p = 2s$, where s denotes the maximum index of the involved derivatives of the vector field, and are topologically conjugate to a B-series symplectic formula up to the order $2s + 2$. This property makes them suitable for integrating canonical Hamiltonian systems

over long times. A similar result, exploiting Theorem 1, has been recently derived for a class of Hermite-Obreshkov one-step methods [48].

A new approach to compute the exact higher order derivatives using numerical infinities and infinitesimals is proposed. This new technique is simple, is able to work with black-box representations of the function $f(y)$ and avoids hard evaluations with tensors related to the function $f(y)$. A comparison among this new approach and other known techniques, such as automatic differentiation, is beyond the scope of this paper and will be considered in a future work.

Acknowledgement

We wish to thank two anonymous reviewers for careful reading our manuscript and for the valuable comments and suggestions they posted which improved the quality of the paper. We are also grateful to Ernst Hairer for helping us in fixing an issue in Theorem 1.

References

[1] G. Benettin, A. Giorgilli, On the Hamiltonian interpolation of near to the identity symplectic mappings with application to symplectic integration algorithms, J. Statist. Phys. 74 (1994) 1117–1143.

[2] E. Hairer, C. Lubich, G. Wanner, Geometric Numerical Integration. Structure-Preserving Algorithms for Ordinary Differential Equations, Second ed., Springer, Berlin, 2006.

[3] F. M. Lasagni, Integration methods for Hamiltonian differential equations, unpublished manuscript, 1990.

[4] E. Hairer, A. Murua, J. Sanz-Serna, The non-existence of symplectic multi-derivative Runge-Kutta methods, BIT 34(1) (1994) 80–87.

[5] E. Hairer, C.J. Zbinden, On conjugate symplecticity of B-series integrators, IMA J. Numer. Anal. 33(1) (2013), 57–79.

[6] P. Amodio, F. Iavernaro, F. Mazzia, M. S. Mukhametzhanov, Y. D. Sergeyev, A generalized Taylor method of order three for the solution of initial value problems in standard and infinity floating-point arithmetic, Mathematics and Computers in Simulation 141 (2016) 24–39.

[7] F. Mazzia, Y. D. Sergeyev, F. Iavernaro, P. Amodio, M. S. Mukhametzhanov, Numerical methods for solving ODEs on the Infinity Computer, in: Proc. of the 2nd Intern. Conf. "Numerical Computations: Theory and Algorithms", Vol. 1776, AIP Publishing, New York, 2016, p. 090033.

[8] Y. D. Sergeyev, Solving ordinary differential equations by working with infinitesimals numerically on the infinity computer, Applied Mathematics and Computation 219(22) (2013) 10668–10681.

[9] Y. D. Sergeyev, Numerical infinitesimals for solving ODEs given as a black-box, in: Proc. of the International Conference on Numerical Analysis and Applied Mathematics 2014 (ICNAAM-2014), Vol. 1648, AIP Publishing, New York, 2015, p. 150018.

[10] Y. D. Sergeyev, M. S. Mukhametzhanov, F. Mazzia, F. Iavernaro, P. Amodio, Numerical methods for solving initial value problems on the Infinity Computer, International Journal of Unconventional Computing 12(1) (2016) 3–23.

[11] Y. D. Sergeyev, Arithmetic of Infinity, Edizioni Orizzonti Meridionali, CS, 2003, 2nd ed. 2013.

[12] Y. D. Sergeyev, A new applied approach for executing computations with infinite and infinitesimal quantities, Informatica 19(4) (2008) 567–596.

[13] Y. D. Sergeyev, Lagrange Lecture: Methodology of numerical computations with infinities and infinitesimals, Rendiconti del Seminario Matematico dell'Università e del Politecnico di Torino 68(2) (2010) 95–113.

[14] Y. D. Sergeyev, Un semplice modo per trattare le grandezze infinite ed infinitesime, Matematica nella Società e nella Cultura: Rivista della Unione Matematica Italiana 8(1) (2015) 111–147.

[15] Y. D. Sergeyev, Numerical infinities and infinitesimals: Methodology, applications, and repercussions on two Hilbert problems, EMS Surveys in Mathematical Sciences 4(2) (2017) 219–320.

[16] B. Bolzano, Paradoxien des Unendlichen, C.H. Reclam (German original), 1851.

[17] K. Trlifajová, Bolzano's infinite quantities, Foundations of Science 23 (2018) 681–704.

[18] G. Cantor, Contributions to the founding of the theory of transfinite numbers, Dover Publications, New York, 1955.

[19] A. Robinson, Non-standard Analysis, Princeton Univ. Press, Princeton, 1996.

[20] T. Levi-Civita, Sui numeri transfiniti, Rend. Acc. Lincei, Series 5a 113 (1898) 7–91.

[21] Y. D. Sergeyev, Independence of the grossone-based infinity methodology from non-standard analysis and comments upon logical fallacies in some texts asserting the opposite, Foundations of Science 24(1) (2019) 153–170.

[22] M. Cococcioni, M. Pappalardo, Ya. D. Sergeyev, Lexicographic multi-objective linear programming using grossone methodology: Theory and algorithm, Applied Mathematics and Computation 318 (2018) 298–311.

[23] S. De Cosmis, R. De Leone, The use of grossone in mathematical programming and operations research, Applied Mathematics and Computation 218(16) (2012) 8029–8038.

[24] R. De Leone, Nonlinear programming and grossone: Quadratic programming and the role of constraint qualifications, Applied Mathematics and Computation 318 (2018) 290–297.

[25] M. Gaudioso, G. Giallombardo, M. S. Mukhametzhanov, Numerical infinitesimals in a variable metric method for convex nonsmooth optimization, Applied Mathematics and Computation 318 (2018) 312–320.

[26] Y. D. Sergeyev, D. E. Kvasov, M. S. Mukhametzhanov, On strong homogeneity of a class of global optimization algorithms working with infinite and infinitesimal scales, Communications in Nonlinear Science and Numerical Simulation 59 (2018) 319–330.

[27] L. D'Alotto, Cellular automata using infinite computations, Applied Mathematics and Computation 218(16) (2012) 8077–8082.

[28] L. D'Alotto, A classification of two-dimensional cellular automata using infinite computations, Indian Journal of Mathematics 55 (2013) 143–158.

[29] M. Margenstern, Fibonacci words, hyperbolic tilings and grossone, Communications in Nonlinear Science and Numerical Simulation 21(1–3) (2015) 3–11.

[30] D. I. Iudin, Y. D. Sergeyev, M. Hayakawa, Infinity computations in cellular automaton forest-fire model, Communications in Nonlinear Science and Numerical Simulation 20(3) (2015) 861–870.

[31] M. Vita, S. D. Bartolo, C. Fallico, M. Veltri, Usage of infinitesimals in the Menger's Sponge model of porosity, Applied Mathematics and Computation 218(16) (2012) 8187–8196.

[32] F. Caldarola, The Sierpinski curve viewed by numerical computations with infinities and infinitesimals, Applied Mathematics and Computation 318 (2018) 321–328.

[33] Y. D. Sergeyev, The exact (up to infinitesimals) infinite perimeter of the Koch snowflake and its finite area, Communications in Nonlinear Science and Numerical Simulation 31(1–3) (2016) 21–29.

[34] A. Zhigljavsky, Computing sums of conditionally convergent and divergent series using the concept of grossone, Applied Mathematics and Computation 218(16) (2012) 8064–8076.

[35] D. Rizza, Supertasks and numeral systems, in: Proc. of the 2nd Intern. Conf. "Numerical Computations: Theory and Algorithms", Vol. 1776, AIP Conference Proceedings, New York, 2016, 090005.

[36] Y. D. Sergeyev, Counting systems and the First Hilbert problem, Nonlinear Analysis Series A: Theory, Methods & Applications 72(3-4) (2010) 1701–1708.

[37] D. Rizza, Numerical methods for infinite decision-making processes, Int. Journ. of Unconventional Computing 14 (2) (2019), 139-158.

[38] D. Rizza, A study of mathematical determination through Bertrand's Paradox, Philosophia Mathematica 26(3) (2017) 375–395.

[39] Y. D. Sergeyev, A. Garro, Observability of Turing machines: A refinement of the theory of computation, Informatica 21(3) (2010) 425–454.

[40] Y. D. Sergeyev, A. Garro, Single-tape and multi-tape Turing machines through the lens of the Grossone methodology, Journal of Supercomputing 65(2) (2013) 645–663.

[41] Y. D. Sergeyev, Higher order numerical differentiation on the Infinity Computer, Optimization Letters 5(4) (2011) 575–585.

[42] E. Hairer, S. P. Nørsett, G. Wanner, Solving ordinary differential equations. I. Nonstiff problems., Second ed., Springer Series in Computational Mathematics **8**, Springer-Verlag, Berlin, 1993.

[43] P. Chartier, E. Faou, A. Murua, An algebraic approach to invariant preserving integrators: the case of quadratic and hamiltonian invariants, Numer. Math. 103(4) (2006) 575–590.

[44] E. Hairer, Conjugate-symplecticity of linear multistep methods, J. Comput. Math. 26(5) (2008) 657–659.

[45] E. Hairer, G. Wanner, Analysis by its history, Undergraduate Texts in Mathematics. Readings in Mathematics, Springer-Verlag, New York, 2008.

[46] L. Brugnano, G. Frasca Caccia, F. Iavernaro, Efficient implementation of Gauss collocation and Hamiltonian boundary value methods, Numer. Algorithms 65(3) (2014) 633–650.

[47] L. Brugnano, F. Iavernaro, Line Integral Methods for Conservative Problems, Monographs and Research Notes in Mathematics, CRC Press, Boca Raton, FL, 2016.

[48] F. Mazzia, A. Sestini, On a class of conjugate symplectic Hermite-Obreshkov one-step methods with continuous spline extension, Axioms 7(3) (2018) 58.