GniCodes – Matlab Programs for Geometric Numerical Integration

Ernst Hairer [1] and Martin Hairer [2]

[1] Section de mathématiques, Univ. Genève, CH-1211 Genève 24, Switzerland
[2] Mathematics Institute, Univ. Warwick, Coventry CV4 7AL, England

Abstract. Geometric numerical integration is synonymous with structure-preserving integration of ordinary differential equations. These notes, prepared for the Durham summer school 2002, are complementary to the monograph of Hairer, Lubich and Wanner [12]. They give an introduction to the subject, and they discuss and explain the use of Matlab programs for experimenting with structure-preserving algorithms.

We start with presenting some typical classes of problems having properties that are important to be conserved by the discretization (Section 1). The flow of Hamiltonian differential equations is symplectic and possesses conserved quantities. Conservative systems have a time-reversible flow. Differential equations with first integrals and problems on manifolds are also considered. We then introduce in Section 2 simple symplectic and symmetric integrators, (partitioned) Runge-Kutta methods, composition and splitting methods, linear multistep methods, and algorithms for Hamiltonian problems on manifolds. We briefly discuss their symplecticity and symmetry. The improved performance of such geometric integrators is best understood with the help of a backward error analysis (Section 3). We explain some implications for the long-time integration of Hamiltonian systems and of completely integrable problems.

Section 4 is devoted to a presentation and explanation of Matlab codes for implicit Runge-Kutta, composition, and multistep methods. The final Section 5 gives a comparison of the different methods and illustrates the use of these programs at some typical interesting situations: the computation of Poincaré sections, and the simulation of the motion of two bodies on a sphere. The Matlab codes as well as their Fortran 77 counterparts can be downloaded at

http://www.unige.ch/math/folks/hairer

under the item "software".

1 Problems to be Solved

For the numerical solution of ordinary differential equations there exist well-developed theories, and excellent general purpose codes are available and widely used. If the flow of the differential equation has a particular structure, then its preservation by the discretization scheme can improve considerably its performance and its qualitative behaviour. This article focuses on structure-preserving algorithms for some important classes of problems – Hamiltonian systems and reversible differential equations.

1.1 Hamiltonian Systems

For a smooth function $H(p,q)$ defined on an open set $D \subset \mathbf{R}^d \times \mathbf{R}^d$ we consider the differential equation

$$\dot p_i = -\frac{\partial H}{\partial q_i}(p,q), \quad \dot q_i = \frac{\partial H}{\partial p_i}(p,q), \quad i = 1,\ldots,d. \tag{1.1}$$

The dimension d of the vectors p and q is called the 'degree of freedom' of the system. We also use the more compact notation

$$\dot p = -\nabla_q H(p,q), \quad \dot q = \nabla_p H(p,q),$$

or

$$\dot y = J^{-1}\nabla H(y), \quad y = \begin{pmatrix} p \\ q \end{pmatrix}, \quad J = \begin{pmatrix} 0 & I \\ -I & 0 \end{pmatrix}, \tag{1.2}$$

where $\nabla_p H, \nabla_q H, \nabla H$ denote the column vectors of partial derivatives with respect to the components of p, q, y, respectively. The matrix J is the structure matrix of Hamiltonian systems in canonical form.

Throughout this article we denote by $\varphi_t(y)$ the exact flow of the system (1.2), i.e., $\varphi_t(y_0) = y(t)$ is the solution at time t of the problem (1.2) with initial value $y(0) = y_0$.

Example 1.1 (Classical Mechanical Systems). Consider a mechanical system that can be described with (minimal) coordinates $q \in \mathbf{R}^d$. Denote its kinetic energy by $T(q,\dot q) = \frac{1}{2}\dot q^T M(q)\dot q$ (with a symmetric positive definite matrix $M(q)$) and its potential energy by $U(q)$. The motion of the system is then given as the solution of the Euler-Lagrange equations

$$\frac{d}{dt}\Big(M(q)\dot q\Big) = \frac{\partial}{\partial q}\Big(T(q,\dot q) - U(q)\Big) \tag{1.3}$$

corresponding to the variational problem $\int\big(T(q,\dot q) - U(q)\big)dt \to \min$. Introducing the new variables $p := M(q)\dot q$ (momenta or Poisson variables) the differential equation (1.3) is equivalent to the Hamiltonian system (1.1) with

$$H(p,q) = \frac{1}{2}p^T M(q)^{-1} p + U(q). \tag{1.4}$$

This is an immediate consequence from computing the partial derivatives of this function $H(p,q)$. A simple example, often used for illustrations, is the *mathematical pendulum* for which the Hamiltonian is

$$H(p,q) = \frac{1}{2}p^2 - \cos q. \tag{1.5}$$

Due to their special structure, Hamiltonian systems have several interesting properties:

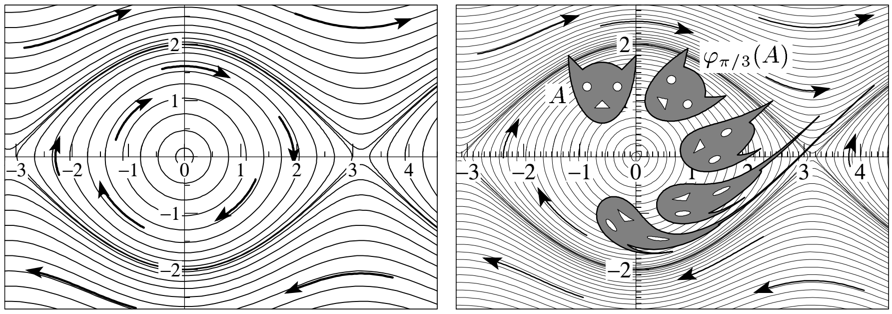


Fig. 1.1. Level curves $H(p,q) = \text{const}$ for the pendulum problem (left picture), and area-preservation of its exact flow (right picture).

- the Hamiltonian $H(p,q)$ is constant along solutions of (1.1); for classical mechanical systems this means that the total energy (sum of kinetic and potential energies) is a conserved quantity.
- for systems with one degree of freedom the flow φ_t is area-preserving; for the general case it is volume-preserving. This means that

$$\mu\big(\varphi_t(A)\big) = \mu(A) \quad \text{for } t \geq 0 \tag{1.6}$$

  for any compact set $A \subset \mathbf{R}^d \times \mathbf{R}^d$.
- the flow φ_t is a symplectic transformation, i.e.,

$$\varphi_t'(y)^T J\, \varphi_t'(y) = J \quad \text{for } t \geq 0, \tag{1.7}$$

  where the prime in $\varphi_t'(y)$ denotes the derivation with respect to y.

The first property is immediately verified by differentiating $\frac{d}{dt} H\big(p(t), q(t)\big) = \ldots = 0$. The solutions of the mathematical pendulum are therefore on the level curves of the Hamiltonian (1.5); see Fig. 1.1. The second property is a consequence of the third, because (1.7) and the continuous dependence of $\varphi_t'(y)$ on t imply $\det \varphi_t'(y) = 1$. This together with the transformation formula for multiple integrals proves (1.6). The right picture of Fig. 1.1 illustrates the area-preservation of the exact flow for the pendulum equation.

The symplecticity condition (1.7) has a nice geometric interpretation. It is equivalent to the property that

$$\omega\big(\varphi_t(A)\big) = \omega(A) \quad \text{for } t \geq 0$$

holds for any two-dimensional sub-manifold A of $\mathbf{R}^d \times \mathbf{R}^d$, where $\omega(A)$ denotes the sum of the oriented areas of the projections of A onto the (p_i, q_i)-plane. The important feature is that this property is characteristic for Hamiltonian systems (cf. [12, chap. VI]), which means that whenever the flow of a differential equation $\dot{y} = f(y)$ is symplectic for all t and all y, then $f(y)$ is locally of the form $f(y) = J^{-1} \nabla H(y)$. This characteristic property of Hamiltonian systems motivates the search for discretizations that are symplectic.

1.2 Reversible Differential Equations

Consider first a mechanical system for which the equations of motion are given by the second order differential equation (1.3). Since $T(q,\dot q)$ is quadratic in $\dot q$, they are equivalent to the system

$$\dot q = v, \quad \dot v = g(q,v), \tag{1.8}$$

satisfying $g(q,-v) = g(q,v)$. This implies the time-reversibility of the system and means that whenever $\big(q(t),v(t)\big)$ is a solution of (1.8), also $\big(q(-t),-v(-t)\big)$ is a solution. For example, in the study of planetary motion, the same differential equation permits us to investigate the future and the past, one only has to change the sign of the velocity vector v.

More generally, we consider a differential equation $\dot y = f(y)$ and a linear invertible transformation ρ. We call the differential equation *ρ-reversible* if

$$(\rho \circ f)(y) = -(f \circ \rho)(y). \tag{1.9}$$

For the previous situation we have $y = (q,v)$, $\rho(q,v) = (q,-v)$, and the vector field $f(y) = \big(v, g(q,v)\big)$ indeed satisfies (1.9) whenever $g(q,-v) = g(q,v)$. This is illustrated in the left picture of Fig. 1.2 at the hand of the perturbed pendulum equation $\dot q = v$, $\dot v = -\sin q - v^2/5$, which is still ρ-reversible with respect to $\rho(q,v) = (q,-v)$, but which is no longer Hamiltonian.

The flow of a ρ-reversible differential equation has a remarkable property:

- it is ρ-reversible, i.e., it satisfies (see the right picture of Fig. 1.2)

$$(\rho \circ \varphi_t)(y) = (\varphi_t^{-1} \circ \rho)(y) \quad \text{for all } t \text{ and all } y. \tag{1.10}$$

The proof of this statement is straightforward. One checks by differentiation that $(\rho \circ \varphi_t)(y)$ and $(\varphi_t^{-1} \circ \rho)(y) = (\varphi_{-t} \circ \rho)(y)$ are both solutions of the same differential equation $\dot z = -f(z)$, and are identical for $t = 0$. Formula (1.10)

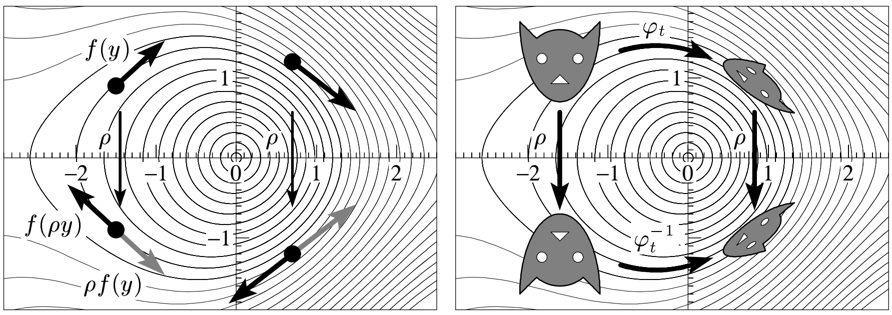


Fig. 1.2. The ρ-reversibility of the vector field $f(q,v) = (v, -\sin q - v^2/5)^T$ (left picture), and the ρ-reversibility of the corresponding flow.

thus follows from the uniqueness of the solution of an initial value problem. Analogous to the situation of Hamiltonian problems, this property is characteristic for ρ-reversible differential equations. This means that whenever the flow of a differential equation $\dot{y} = f(y)$ satisfies (1.10), then (1.9) holds. It is thus natural to look for numerical methods that share this property.

Example 1.2 (Kepler Problem). The relative motion of two bodies which attract each other is described by the differential equation

$$\dot{q}_1 = v_1, \quad \dot{q}_2 = v_2, \quad \dot{v}_1 = -\frac{q_1}{(q_1^2 + q_2^2)^{3/2}}, \quad \dot{v}_2 = -\frac{q_2}{(q_1^2 + q_2^2)^{3/2}}. \tag{1.11}$$

Since it can be considered as a classical mechanical system, it is ρ-reversible for $\rho(q_1, q_2, v_1, v_2) = (q_1, q_2, -v_1, -v_2)$. However, there are more symmetries in this problem, and it is seen to be ρ-reversible also for $\rho(q_1, q_2, v_1, v_2) = (q_1, -q_2, -v_1, v_2)$.

Example 1.3 (Second Order Differential Equations). Many problems of practical applications lead to $\ddot{q} = g(q)$, or equivalently,

$$\dot{q} = v, \quad \dot{v} = g(q). \tag{1.12}$$

For example, all classical mechanical systems for which $M(q) = M$ is a constant matrix are of this form. The differential equation (1.12) is ρ-reversible for $\rho(q, v) = (q, -v)$ independent of the form of $g(q)$. It is Hamiltonian only if $g(q) = -\nabla_q U(q)$ for some potential function $U(q)$.

1.3 Hamiltonian and Reversible Systems on Manifolds

It is often difficult to find suitable minimal coordinates for describing the motion of mechanical systems. Moreover, minimal coordinates are in general defined only locally and frequent changes of charts may be necessary. To avoid this difficulty we consider coordinates $q \in \mathbf{R}^d$ that are subject to constraints $g(q) = 0$. Expressing the Euler-Lagrange equations and their Hamiltonian formulation in terms of these coordinates, we are led to a system of the form

$$\begin{aligned} \dot{p} &= -\nabla_q H(p,q) - \nabla_q g(q)\lambda \\ \dot{q} &= \nabla_p H(p,q), \qquad 0 \;=\; g(q), \end{aligned} \tag{1.13}$$

where the additional term with the Lagrange multiplier λ forces the solution to satisfy $g(q) = 0$. Here, p and q are vectors in $\mathbf{R}^d$, $g(q) = (g_1(q), \ldots, g_m(q))^T$ is the vector of constraints, and $\nabla_q g = \big(\nabla_q g_1, \ldots, \nabla_q g_m\big)$ is the transposed Jacobian matrix of $g(q)$.

Differentiating the constraint $0 = g\big(q(t)\big)$ with respect to time yields

$$0 = \nabla_q g(q)^T \nabla_p H(p,q) \tag{1.14}$$

(the so-called hidden constraint) which is an invariant of the flow of (1.13). A second differentiation gives the relation

$$
\begin{aligned}
0 = \frac{\partial}{\partial q}\Big(\nabla_q g(q)^T \nabla_p H(p,q)\Big)\nabla_p H(p,q) \\
-\nabla_q g(q)^T \nabla_p^2 H(p,q)\Big(\nabla_q H(p,q) + \nabla_q g(q)\lambda\Big),
\end{aligned}
\tag{1.15}
$$

which allows us to express λ in terms of (p,q), if the matrix

$$
\nabla_q g(q)^T \nabla_p^2 H(p,q) \nabla_q g(q) \qquad \text{is invertible} \tag{1.16}
$$

($\nabla_p^2 H$ denotes the Hessian matrix of H). Inserting the so-obtained function $\lambda(p,q)$ into (1.13) gives a differential equation for (p,q) on the manifold

$$
\mathcal{M} = \big\{(p,q) \mid g(q) = 0,\ \nabla_q g(q)^T \nabla_p H(p,q) = 0\big\}. \tag{1.17}
$$

This interpretation allow us to deduce the existence and uniqueness of the solution from the standard theory for ordinary differential equations, provided that the initial values satisfy $(p_0, q_0) \in \mathcal{M}$.

Important properties of the system (1.13) that should be conserved by a discretization are the following:

- for $(p_0, q_0) \in \mathcal{M}$ the solution stays on the manifold $\mathcal{M}$ for all t; hence, the flow is a mapping $\varphi_t : \mathcal{M} \to \mathcal{M}$.
- the flow φ_t is a symplectic transformation on $\mathcal{M}$ which means that

$$
(\varphi_t'(y)\xi)^T J\, \varphi_t'(y)\eta = \xi^T J\, \eta \quad \text{for } \xi, \eta \in T_y\mathcal{M}; \tag{1.18}
$$

  here, the product $\varphi_t'(y)\xi$ has to be interpreted as the directional derivative.
- for Hamiltonians satisfying $H(-p,q) = H(p,q)$ the flow φ_t is ρ-reversible for $\rho(p,q) = (-p,q)$ in the sense that (1.10) holds for $y = (p,q) \in \mathcal{M}$.

Example 1.4 (Two-Body Problem on the Sphere). We are interested in the motion of two bodies which attract each other, but which are restricted to stay on a sphere. Using Cartesian coordinates $q_1, q_2 \in \mathbf{R}^3$ for the positions of the two bodies and $p_1, p_2 \in \mathbf{R}^3$ for their velocities, the Hamiltonian becomes (after a suitable normalization)

$$
H(p_1, p_2, q_1, q_2) = \frac{1}{2}\big(p_1^T p_1 + p_2^T p_2\big) + U(q_1, q_2), \tag{1.19}
$$

and the constraint equations $g(q_1, q_2) = 0$ with $g : \mathbf{R}^6 \to \mathbf{R}^2$ are given by

$$
q_1^T q_1 - 1 = 0, \quad q_2^T q_2 - 1 = 0. \tag{1.20}
$$

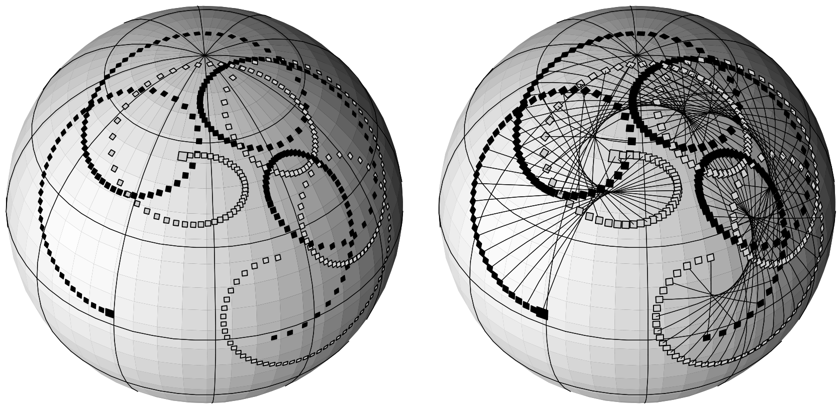


Fig. 1.3. A solution of the two-body problem on the sphere; initial values are indicated by larger symbols; the geodesic connection between the two bodies is plotted at every second time step in the right picture.

According to Kozlov and Harin [18], we choose $U(q_1, q_2) = -\cos\vartheta/\sin\vartheta$ as potential, where ϑ is the distance between the two bodies along a geodesics. We have $\cos\vartheta = q_1^T q_2$, so that the equations of motion become

$$\begin{aligned} \dot q_1 &= p_1, \quad & \dot p_1 &= f(q_1^T q_2) q_2 - \lambda_1 q_1, \\ \dot q_2 &= p_2, \quad & \dot p_2 &= f(q_1^T q_2) q_1 - \lambda_2 q_2, \end{aligned} \tag{1.21}$$

together with (1.20), where

$$f(c) = \frac{1}{(1-c^2)^{3/2}}.$$

The initial values have to lie on the manifold

$$\mathcal{M} = \big\{(p_1, p_2, q_1, q_2) \; ; \; q_1^T q_1 = 1, \; q_2^T q_2 = 1, \; q_1^T p_1 = 0, \; q_2^T p_2 = 0\big\},$$

and the solution stays on $\mathcal{M}$ for all t.

A particular solution is plotted in Fig. 1.3. We have chosen

$$q_i = \big(\cos\phi_i \sin\theta_i, \sin\phi_i \sin\theta_i, \cos\theta_i\big)^T$$

with $(\phi_1, \theta_1) = (0.8, 0.6)$ and $(\phi_2, \theta_2) = (0.5, 1.5)$ as initial values for the positions, and

$$p_i = \big(-\dot\phi_i \sin\phi_i \sin\theta_i + \dot\theta_i \cos\phi_i \cos\theta_i, \dot\phi_i \cos\phi_i \sin\theta_i + \dot\theta_i \sin\phi_i \cos\theta_i, -\dot\theta_i \sin\theta_i\big)$$

with $(\dot\phi_1, \dot\theta_1) = (1.1, -0.2)$ and $(\dot\phi_2, \dot\theta_2) = (-0.8, 0.0)$ as initial values for the velocities. The two bodies are indicated by small squares in different colors. The right picture of Fig. 1.3 also shows the geodesic connection between the two bodies.

Example 1.5 (Rigid Body Simulation). The motion of a rigid body with a fixed point chosen at the origin can be described by an orthogonal matrix $Q(t)$. Denoting by I_1, I_2, I_3 the moments of inertia of the body, its kinetic energy is

$$T = \frac{1}{2}\big(I_1 \Omega_1^2 + I_2 \Omega_2^2 + I_3 \Omega_3^2\big),$$

where the angular velocity $\Omega = (\Omega_1, \Omega_2, \Omega_3)^T$ of the body is defined by

$$\widehat{\Omega} = \begin{pmatrix} 0 & -\Omega_3 & \Omega_2 \\ \Omega_3 & 0 & -\Omega_1 \\ -\Omega_2 & \Omega_1 & 0 \end{pmatrix} = Q^T \dot{Q},$$

(see [2, Chap. 6]). In terms of Q, the kinetic energy on the manifold $\{Q \mid Q^T Q = I\}$ becomes

$$T = \frac{1}{2}\,\mathrm{trace}\,(\widehat{\Omega} D \widehat{\Omega}^T) = \frac{1}{2}\,\mathrm{trace}\,(Q^T \dot{Q} D \dot{Q}^T Q) = \frac{1}{2}\,\mathrm{trace}\,(\dot{Q} D \dot{Q}^T),$$

where $D = \mathrm{diag}\,(d_1, d_2, d_3)$ is given by the relations $I_1 = d_2 + d_3$, $I_2 = d_3 + d_1$, and $I_3 = d_1 + d_2$. With $P = \partial T / \partial \dot{Q} = \dot{Q} D$, we are thus concerned with

$$H(P, Q) = \frac{1}{2}\,\mathrm{trace}\,(P D^{-1} P^T) + U(Q),$$

and the constrained Hamiltonian system becomes

$$\begin{aligned} \dot{P} &= -\nabla_Q U(Q) - Q\Lambda, \\ \dot{Q} &= P D^{-1}, \qquad 0 \;=\; Q^T Q - I, \end{aligned} \tag{1.22}$$

where Λ is a symmetric matrix consisting of Lagrange multipliers. This is of the form (1.13) and satisfies the regularity condition (1.16).

2 Symplectic and Symmetric Integrators

A numerical integrator is a family $\Phi_h(y)$ of maps on the phase space that approximates the exact flow $\varphi_h(y)$ of the differential equation. It is the aim of 'geometric integration' to construct and to study methods for which the numerical solution, given by $y_{n+1} = \Phi_h(y_n)$, preserves the structure of the problem. We are mainly interested in methods for which Φ_h is symplectic or ρ-reversible, when it is applied to a Hamiltonian or ρ-reversible differential equation, respectively.

2.1 Simple Symplectic Methods

The simplest numerical methods for general differential equations $\dot{y} = f(y)$ are the *explicit Euler method*

$$y_{n+1} = y_n + h f(y_n) \tag{2.1}$$

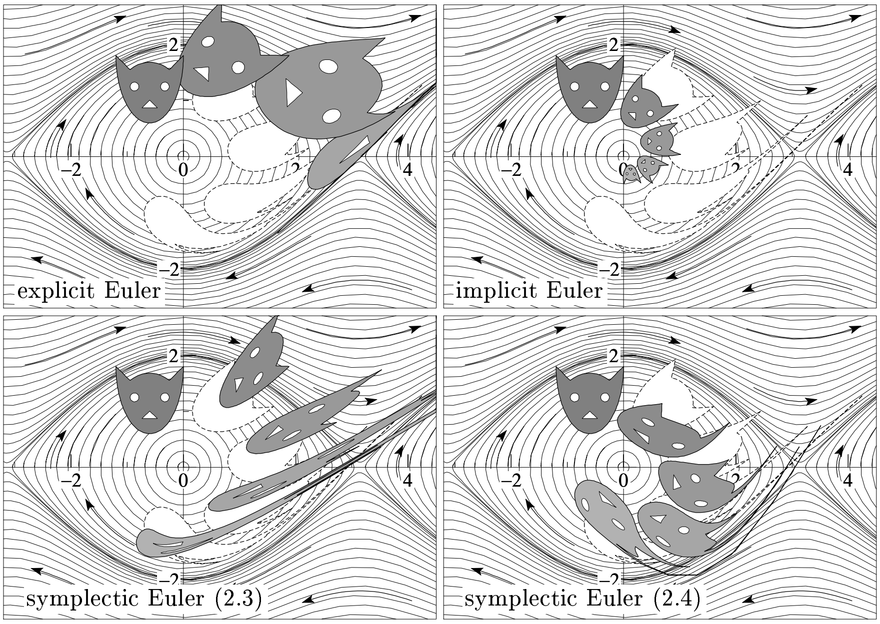


Fig. 2.1. Numerical flow with step size $h = \pi/3$ for the four 'Euler methods' of Sect. 2.1. The exact flow is included as a white shadow.

and the *implicit Euler method*

$$y_{n+1} = y_n + hf(y_{n+1}). \tag{2.2}$$

Here, h is the step size, and y_n is an approximation to the solution $y(t)$ at time $t = nh$. For Hamiltonian systems (1.1) we consider the method

$$p_{n+1} = p_n - h\nabla_q H(p_{n+1}, q_n), \quad q_{n+1} = q_n + h\nabla_p H(p_{n+1}, q_n), \tag{2.3}$$

which treats the p-variable by the implicit Euler method and the q-variable by the explicit Euler method. Similarly, we also consider

$$p_{n+1} = p_n - h\nabla_q H(p_n, q_{n+1}), \quad q_{n+1} = q_n + h\nabla_p H(p_n, q_{n+1}). \tag{2.4}$$

Both methods are called *symplectic Euler method*.

Example 2.1. We apply all four methods to the pendulum problem which is Hamiltonian with $H(p,q)$ given by (1.5), and we consider initial values in the set A of Fig. 1.1. The numerical solution obtained with the large step size $h = \pi/3$ is illustrated in Fig. 2.1. Neither the explicit nor the implicit Euler methods are area-preserving (i.e., symplectic). We shall see in the following theorem that both 'symplectic Euler methods' are area-preserving (hence the name symplectic). Due to the large step size, however, the numerical solution differs significantly from the exact solution which is included as a white shadow in the pictures (compare with Fig. 1.1).

Theorem 2.1. *For the numerical schemes (2.3) and (2.4) the mapping*

$$\Phi_h \ : \ \begin{pmatrix} p_n \\ q_n \end{pmatrix} \mapsto \begin{pmatrix} p_{n+1} \\ q_{n+1} \end{pmatrix}$$

is a symplectic transformation.

The *proof* of this theorem is straightforward (de Vogelaere [42] and [12, p. 176]). One computes the Jacobian of Φ_h by implicit differentiation, and one checks the identity (1.7).

2.2 Simple Reversible Methods

We next consider ρ-reversible differential equations (i.e., $\rho \circ f = -f \circ \rho$) as discussed in Sect. 1.2.

Theorem 2.2. *If a numerical method Φ_h satisfies*

$$\rho \circ \Phi_h = \Phi_{-h} \circ \rho \quad \textit{and} \quad \Phi_h = \Phi_{-h}^{-1}, \tag{2.5}$$

then it is ρ-reversible, i.e., $\rho \circ \Phi_h = \Phi_h^{-1} \circ \rho$.

This statement is obvious. The interest of this theorem lies in the fact that the second condition of (2.5) is independent of ρ, whereas the first condition of (2.5) is easy to check and satisfied by all 'reasonable' methods. For example, the explicit Euler discretization (2.1) yields

$$(\rho \circ \Phi_h)(y_n) = \rho y_{n+1} = \rho y_n + h\rho f(y_n) = \rho y_n - hf(\rho y_n) = (\Phi_{-h} \circ \rho)(y_n),$$

and a similar simple computation shows that the implicit Euler method and all (explicit and implicit) Runge-Kutta methods satisfy the first condition of (2.5). For partitioned Runge-Kutta methods, such as the symplectic Euler scheme, this is true for transformations ρ which are of the form $\rho(q,v) = (\rho_1(q), \rho_2(v))$.

If $\Phi_h(y)$ represents a numerical method of order at least one, i.e., $\Phi_h(y) = y + hf(y) + \mathcal{O}(h^2)$, then also $\Phi_{-h}^{-1}(y) = y + hf(y) + \mathcal{O}(h^2)$ and

$$\Phi_h^* := \Phi_{-h}^{-1} \tag{2.6}$$

is a numerical method of order at least one. It is called the *adjoint method* of Φ_h. Whenever an integrator satisfies

$$\Phi_h^* = \Phi_h, \tag{2.7}$$

it is called a *symmetric method*. The second condition in (2.5) of Theorem 2.2 is thus equivalent to the symmetry of the method Φ_h.

Exchanging $h \leftrightarrow -h$ and $y_n \leftrightarrow y_{n+1}$ in (2.1) shows that the adjoint of the explicit Euler method is the implicit Euler method and vice versa. Similarly, the adjoint of the symplectic Euler method (2.3) is the method (2.4). None of these methods is symmetric.

Using the notion of the adjoint method it is easy to construct symmetric methods: let Ψ_h be an arbitrary method of order at least one, then the compositions

$$\Psi_{h/2} \circ \Psi^*_{h/2} \quad \text{and} \quad \Psi^*_{h/2} \circ \Psi_{h/2} \tag{2.8}$$

are symmetric methods of order at least two. The symmetry follows from the properties $(\Phi_h \circ \Psi_h)^* = \Psi^*_h \circ \Phi^*_h$ and $(\Phi^*_h)^* = \Phi_h$, and order at least two is a consequence of the fact that symmetric method always have an even order.

For example, if we let Ψ_h be the explicit Euler method, then the methods of (2.8) are

$$y_{n+1} = y_n + h\, f\Big(\frac{y_n + y_{n+1}}{2}\Big), \tag{2.9}$$

the *implicit midpoint rule*, and

$$y_{n+1} = y_n + \frac{h}{2}\Big(f(y_n) + f(y_{n+1})\Big), \tag{2.10}$$

the *trapezoidal rule*, respectively.

2.3 Störmer/Verlet Scheme

We next consider Hamiltonian systems (1.1) and the symplectic Euler method (2.3) in the role of Ψ_h. The compositions (2.8) then yield

$$\begin{aligned} q_{n+1/2} &= q_n + \frac{h}{2}\,\nabla_p H(p_n, q_{n+1/2}) \\ p_{n+1} &= p_n - \frac{h}{2}\,\Big(\nabla_q H(p_n, q_{n+1/2}) + \nabla_q H(p_{n+1}, q_{n+1/2})\Big) \\ q_{n+1} &= q_{n+1/2} + \frac{h}{2}\,\nabla_p H(p_{n+1}, q_{n+1/2}) \end{aligned} \tag{2.11}$$

and

$$\begin{aligned} p_{n+1/2} &= p_n - \frac{h}{2}\,\nabla_q H(p_{n+1/2}, q_n) \\ q_{n+1} &= q_n + \frac{h}{2}\,\Big(\nabla_p H(p_{n+1/2}, q_n) + \nabla_p H(p_{n+1/2}, q_{n+1})\Big) \\ p_{n+1} &= p_{n+1/2} - \frac{h}{2}\,\nabla_q H(p_{n+1/2}, q_{n+1}) \end{aligned} \tag{2.12}$$

respectively. For the important special case $H(p,q) = \frac{1}{2}p^2 + U(q)$, method (2.12) reduces to (after elimination of the p-variable)

$$q_{n+1} - 2q_n + q_{n-1} = -h^2 \nabla_q U(q_n). \tag{2.13}$$

This discretization of $\ddot{q} = -\nabla_q U(q)$ is attributed to Newton (cf. [13]), Delambre (cf. [25]), Encke, Störmer [36], and Verlet [41]. The methods (2.11) and (2.12) are nowadays often called *Störmer/Verlet scheme*.

Let us collect the most important properties of the Störmer/Verlet scheme:

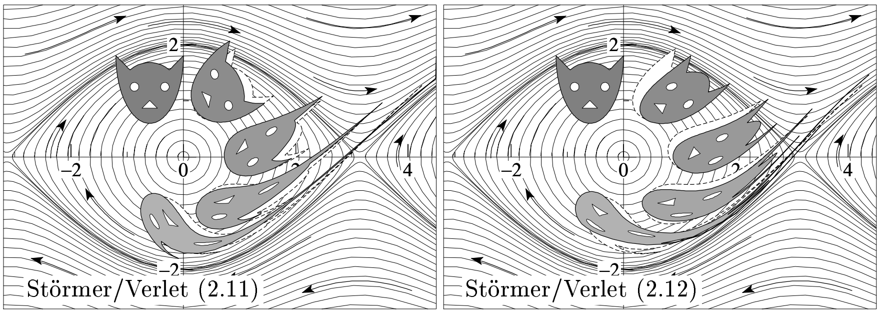


Fig. 2.2. Numerical flow with step size $h = \pi/3$ for the two versions of the Störmer/Verlet method. The exact flow is included as a white shadow.

- the method is of order two,
- it is a symplectic method,
- it is a symmetric method,
- for separable Hamiltonians $T(p) + U(q)$ the method is explicit,
- the method exactly conserves quadratic first integrals $p^T Cq$, e.g., the angular momentum in N-body problems.

The first four statements are immediate consequences of the above discussions. A proof of the last property is given in [12, p. 98].

In Fig. 2.2 we repeat the experiment of Example 2.1, but this time with the two versions of the Störmer/Verlet method. We clearly observe the higher accuracy (compared to the first order methods) and the area-preservation.

The Störmer/Verlet scheme is an excellent geometric integrator and it is widely used, in particular in molecular dynamics where a correct qualitative simulation is of utmost importance. For long-time computations in astronomy, where a very high accuracy is demanded, the order two of the Störmer/Verlet scheme is too low.

2.4 Splitting Methods

A different approach for constructing simple geometric integrators is based on the idea of splitting the vector field as

$$\dot{y} = f^{[1]}(y) + f^{[2]}(y). \tag{2.14}$$

If by chance the exact flows $\varphi_t^{[1]}$ and $\varphi_t^{[2]}$ of the systems $\dot{y} = f^{[1]}(y)$ and $\dot{y} = f^{[2]}(y)$ can be calculated exactly, we can consider

$$\Phi_h = \varphi_h^{[1]} \circ \varphi_h^{[2]} \tag{2.15}$$

as simple numerical integrator. It follows from Taylor expansion that this method is of order one. Even more important is the symmetric (second order) composition

$$\Phi_h = \varphi_{h/2}^{[1]} \circ \varphi_h^{[2]} \circ \varphi_{h/2}^{[1]} \tag{2.16}$$

which is usually called *Strang splitting*. These splitting methods have the following obvious properties:

- if both, $f^{[1]}(y)$ and $f^{[2]}(y)$, are Hamiltonian vector fields, then the compositions (2.15) and (2.16) are *symplectic* integrators;
- if both, $f^{[1]}(y)$ and $f^{[2]}(y)$, are ρ-reversible, then the symmetric method (2.16) is *ρ-reversible*.

For some situations the splitting (2.14) is obvious. For example, if a Hamiltonian system has $H(p,q) = T(p) + U(q)$ as Hamiltonian, then the flows corresponding to $H^{[1]}(p,q) = T(p)$ and $H^{[2]}(p,q) = U(q)$ are given explicitly by

$$\varphi_t^{[1]}(p,q) = \big(p, q + t\nabla_p T(p)\big), \quad \varphi_t^{[2]}(p,q) = \big(p - t\nabla_q U(q), q\big).$$

The resulting splitting methods (2.15) and (2.16) are then equivalent to the symplectic Euler method (2.3) and to the Störmer/Verlet scheme (2.12), respectively. In general, however, it is an art to find a suitable splitting (cf. [25]).

2.5 High Order Geometric Integrators

We start this section with a numerical experiment that motivates the search for high order symplectic and symmetric numerical integrators. We consider the Kepler problem which is Hamiltonian with

$$H(p_1, p_2, q_1, q_2) = \frac{1}{2}\,(p_1^2 + p_2^2) - \frac{1}{\sqrt{q_1^2 + q_2^2}}, \tag{2.17}$$

and we take as initial values

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

such that the solution is an ellipse with eccentricity $e = 0.6$. Figure 2.3 shows the work precision diagrams (global error at the endpoint after 200 revolutions against the required number of function evaluations and the computer time, respectively) for the second order Störmer/Verlet scheme as well as for various methods of order eight. It clearly demonstrates that for high accuracy requirements (say 10 digits) the low order method cannot compete with the high order ones. It would need about 1000 times more cpu time. The irregularities at the right bottom corner of the pictures are due to round-off.

Implicit Runge-Kutta Methods In the end of the 19th and the beginning of the 20th century Runge [30] and Kutta [19] introduced generalizations of the explicit Euler method with the aim of getting more accurate numerical approximations. These *explicit* methods can neither be symplectic nor symmetric as follows from the characterizations given below. Much more important for our purpose are *implicit* Runge-Kutta methods, introduced mainly

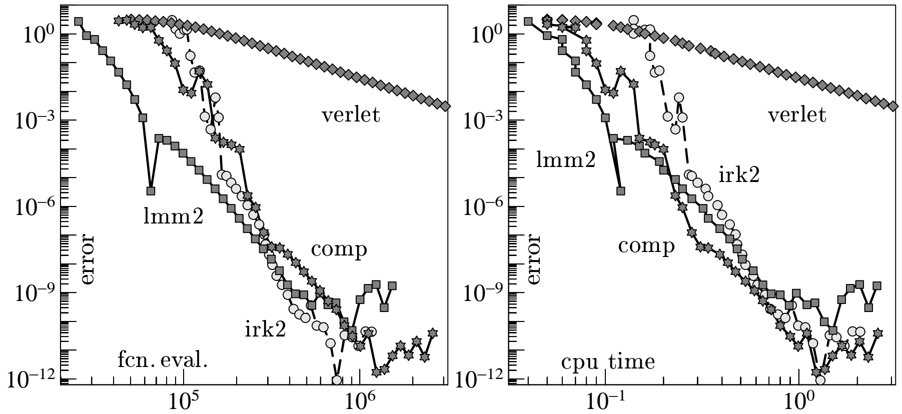


Fig. 2.3. Work precision diagrams for the Störmer/Verlet scheme and for three methods of order eight; implicit Runge-Kutta method (irk2), composition method (comp), and linear multistep method (lmm2).

in the work of Butcher [5]. For Hamiltonian systems or for general partitioned differential equations

$$\dot{q} = f(q,v), \quad \dot{v} = g(q,v) \tag{2.18}$$

we consider so-called *partitioned Runge-Kutta methods*, which treat the components of q and those of v by possibly different implicit Runge-Kutta methods. They are defined by

$$\begin{aligned} k_i &= f\Big(q_n + h\sum_{j=1}^{s} a_{ij}k_j,\, v_n + h\sum_{j=1}^{s}\widehat{a}_{ij}\ell_j\Big),\\ \ell_i &= g\Big(q_n + h\sum_{j=1}^{s} a_{ij}k_j,\, v_n + h\sum_{j=1}^{s}\widehat{a}_{ij}\ell_j\Big),\\ q_{n+1} &= q_n + h\sum_{i=1}^{s} b_i k_i, \qquad v_{n+1} \,=\, v_n + h\sum_{i=1}^{s}\widehat{b}_i\ell_i. \end{aligned} \tag{2.19}$$

The equations for k_i, ℓ_i $(i = 1,\ldots,s)$ are nonlinear and have to be solved by fixed-point iteration, provided that the step size h is sufficiently small.

It turns out that the method (2.19) is *symplectic* for general Hamiltonian systems, if the following relations are satisfied:

$$\begin{aligned} b_i\widehat{a}_{ij} + \widehat{b}_j a_{ji} \,&=\, b_i\widehat{b}_j \qquad \text{for } \; i,j = 1,\ldots,s,\\ b_i \,&=\, \widehat{b}_i \qquad \text{for } \; i = 1,\ldots,s. \end{aligned} \tag{2.20}$$

It is *symmetric*, if

$$\begin{aligned} a_{s+1-i,s+1-j} + a_{ij} &= b_j \qquad \text{for all } i,j,\\ \widehat{a}_{s+1-i,s+1-j} + \widehat{a}_{ij} &= \widehat{b}_j \qquad \text{for all } i,j. \end{aligned} \tag{2.21}$$

If the method does not contain superfluous stages and if the stages are suitably ordered, the conditions (2.20) and (2.21) are also necessary for symplecticity and symmetry, respectively. These characterizations have been obtained originally by Lasagni [22], Sanz-Serna [32] and Suris [37] for symplecticity, and by Stetter [35] and Wanner [43] for symmetry. They are discussed in detail in Chapters V and VI of [12].

For the important special case $\ddot{q} = g(q)$, i.e., $f(q,v) = v$ in (2.18) and $g(q,v)$ independent of v, the variables k_i can be eliminated explicitly and the method (2.19) reduces to

$$\begin{aligned} \ell_i &= g\Big(q_n + hc_i v_n + h^2 \sum_{j=1}^{s} \widetilde{a}_{ij}\ell_j\Big), \quad i = 1,\ldots s, \\ q_{n+1} &= q_n + hv_n + h^2 \sum_{i=1}^{s} \widetilde{b}_i \ell_i, \qquad v_{n+1} \;=\; v_n + h\sum_{i=1}^{s} \widehat{b}_i\ell_i. \end{aligned} \tag{2.22}$$

where $c_i = \sum_{j=1}^{s} a_{ij}$, and $\widetilde{b}_i, \widetilde{a}_{ij}$ are the coefficients of $\widetilde{b}^T = b^T\widehat{A}$ and $\widetilde{A} = A\widehat{A}$.

Example 2.2 (Method Used as 'irk2' in Fig. 2.3). The most important symplectic implicit Runge-Kutta methods are the so-called *Gauss methods.* They are built on the Gaussian quadrature $(b_i, c_i)_{i=1}^{s}$, which is interpolatory and for which $c_1, \ldots, c_s$ are the zeros of the sth shifted Legendre polynomial

$$\frac{d^s}{dx^s}\Big(x^s(x-1)^s\Big).$$

The coefficients a_{ij} are computed from the linear system

$$\sum_{j=1}^{s} a_{ij}c_j^{k-1} = \frac{c_i^k}{k} \qquad \text{for } i,k = 1,\ldots,s.$$

We let $\widehat{b}_i = b_i$ and $\widehat{a}_{ij} = a_{ij}$ in (2.19), so that all components of the differential equation are treated by the same method.

The method obtained in this way has originally been introduced by Butcher [6], and it has many nice properties. It is of order $2s$ (which is maximal among all s-stage Runge-Kutta methods), it is symplectic and symmetric, so that it is extremely well suited in the context of geometric integration. The only disadvantage is that even for simple situations such as $\ddot{q} = g(q)$, it gives an implicit discretization. In the experiment of Fig. 2.3 we use this method with $s = 4$. The '2' in 'irk2' indicates that the code is for second order differential equations $\ddot{q} = g(q)$ only, and that it is implemented as (2.22).

Partitioned Multistep Methods Another extension of the Euler methods are linear multistep methods, originally introduced by Adams in 1855 and published in Bashforth [3]. Neither explicit nor implicit classical multistep methods have been successful in geometric integration. Lambert and Watson [21] considered special classes for second order differential equations $\ddot{q} = g(q)$, which have been revived by Quinlan and Tremaine [27] for the long-time integration of planetary orbits. For partitioned differential equations (2.18), which are more general than $\dot{q} = v$, $\dot{v} = g(q)$, these methods can be interpreted as partitioned linear multistep methods, defined by

$$\begin{aligned} \sum_{j=0}^{k} \alpha_j q_{n+j} &= h \sum_{j=0}^{k} \beta_j f(q_{n+j}, v_{n+j}), \\ \sum_{j=0}^{\widehat{k}} \widehat{\alpha}_j v_{n+j} &= h \sum_{j=0}^{\widehat{k}} \widehat{\beta}_j g(q_{n+j}, v_{n+j}). \end{aligned} \tag{2.23}$$

It is not evident to discuss symplecticity and symmetry of multistep methods, because we are concerned with an algorithm $(y_n, \ldots, y_{n+k-1}) \mapsto y_{n+k}$ and not with a one-step method $y_{n+1} = \Phi_h(y_n)$ which is a transformation on the phase space. However, Kirchgraber [17] showed that to every consistent strictly stable multistep method one can associate a so-called *underlying one-step method* Φ_h which has the same long-time dynamics. More precisely, it satisfies the following properties:

- for every y_0, the sequence defined by $y_{n+1} = \Phi_h(y_n)$ is a solution of the multistep method;
- for an arbitrary starting approximation $y_0, \ldots, y_{k-1}$, the numerical approximation of the multistep method tends exponentially fast to a particular solution obtained by the underlying one-step method.

The existence of an underlying one-step method (as a formal series in powers of h) satisfying the first of these properties, can be shown for general consistent methods (2.23); see [12, Chap. XIV]. The second property cannot be fulfilled by methods that are not strictly stable. Assuming that for arbitrary starting approximations the multistep solution remains close to that obtained by the underlying one-step method, it is natural to call a method (2.23) *symplectic* and *symmetric*, if the underlying one-step method is symplectic and symmetric, respectively.

Unfortunately, it turns out that partitioned multistep methods cannot be symplectic (Tang [39]). However, they can be symmetric. In terms of the coefficients of the method (2.23), the symmetry of the underlying one-step method is equivalent to (assuming irreducibility of the methods)

$$\begin{aligned} \alpha_j &= -\alpha_{k-j}, \quad \beta_j = \beta_{k-j} \qquad \text{for } j = 0, \ldots, k, \\ \widehat{\alpha}_j &= -\widehat{\alpha}_{\widehat{k}-j}, \quad \widehat{\beta}_j = \widehat{\beta}_{\widehat{k}-j} \qquad \text{for } j = 0, \ldots, \widehat{k}. \end{aligned} \tag{2.24}$$

For stable symmetric multistep methods the zeros of the generating polynomials $\rho(\zeta) = \sum_{j=0}^{k} \alpha_j \zeta^j$ and $\widehat{\rho}(\zeta) = \sum_{j=0}^{\widehat{k}} \widehat{\alpha}_j \zeta^j$ have to lie on the unit circle. Such methods cannot be strictly stable, and for this reason symmetric multistep methods have been disregarded for a long time.

Also for this class of methods we are mainly interested in the numerical solution of second order differential equations $\ddot{q} = g(q)$. Elimination of the v-variables in (2.23) yields the formula

$$\sum_{j=0}^{K} A_j q_{n+j} = h^2 \sum_{j=0}^{K} B_j g(q_{n+j}), \tag{2.25}$$

where the generating polynomials $R(\zeta) = \sum_{j=0}^{k} A_j \zeta^j$ and $S(\zeta) = \sum_{j=0}^{k} B_j \zeta^j$ are obtained from those of (2.23) by

$$R(\zeta) = \rho(\zeta) \cdot \widehat{\rho}(\zeta), \quad S(\zeta) = \sigma(\zeta) \cdot \widehat{\sigma}(\zeta).$$

Here, $\rho(\zeta), \widehat{\rho}(\zeta), \sigma(\zeta), \widehat{\sigma}(\zeta)$ are the generating polynomials of $\alpha_j, \widehat{\alpha}_j, \beta_j, \widehat{\beta}_j$, respectively. We recall that method (2.25) is of order p, if

$$R(e^h) - h^2 S(e^h) = \mathcal{O}(h^{p+2}) \quad \text{for} \quad h \to 0. \tag{2.26}$$

Formula (2.25) does not involve derivative approximations v_n. If they are needed, they can be obtained by finite differences from the position approximations q_n.

Example 2.3 (Method Used as 'lmm2' in Fig. 2.3). We put $K = 8$ and we let

$$R(\zeta) = (\zeta - 1)(\zeta^7 - 1) = (\zeta - 1)^2(\zeta^6 + \zeta^5 + \zeta^4 + \zeta^3 + \zeta^2 + \zeta + 1),$$

so that all zeros lie on the unite circle and, apart from $\zeta = 1$, all zeros are simple. To get a method of order $p = 8$, the polynomial $S(\zeta)$ has to satisfy

$$S(\zeta) = R(\zeta)/\log^2 \zeta + \mathcal{O}\big((\zeta - 1)^p\big)$$

(cf. condition (2.26)). Expanding the right-hand expression into a Taylor series at $\zeta = 1$ and truncating to get a polynomial of degree 7, we obtain the generating polynomial

$$S(\zeta) = \frac{13207}{8640}\big(\zeta^7 + \zeta\big) - \frac{8934}{8640}\big(\zeta^6 + \zeta^2\big) + \frac{42873}{8640}\big(\zeta^5 + \zeta^3\big) - \frac{33812}{8640}\,\zeta^4.$$

The resulting method (2.25) is of order 8 for problems $\ddot{q} = g(q)$, symmetric, and explicit (because $B_K = 0$). An approximation to the derivative is obtained by symmetric differences as

$$\begin{aligned}\dot{y}_n = \frac{1}{840h}\Big(&672\,(y_{n+1} - y_{n-1}) - 168\,(y_{n+2} - y_{n-2}) \\ &+ 32\,(y_{n+3} - y_{n-3}) - 3\,(y_{n+4} - y_{n-4})\Big).\end{aligned}$$

Composition Methods We consider the composition of a given basic one-step method $\Phi_h(y)$ with different step sizes:

$$\Psi_h = \Phi_{\gamma_s h} \circ \ldots \circ \Phi_{\gamma_2 h} \circ \Phi_{\gamma_1 h}. \tag{2.27}$$

The aim is to increase the order (and hence the accuracy) while preserving desirable properties (symplecticity, symmetry) of the basic method. This idea has mainly been developed in the papers of Suzuki [38], Yoshida [44], and McLachlan [24]. For a recent comprehensive survey see [25] and Chapters II, III, and V of [12].

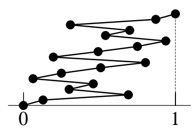
The reason of the success of composition methods within geometric integrators are the following properties:

- if Φ_h is symplectic, then the composition method Ψ_h is *symplectic*;
- if Φ_h is symmetric and if the step size parameters γ_i satisfy $\gamma_i = \gamma_{s+1-i}$, then the composition Ψ_h is *symmetric*.

The main problem consists in finding parameters γ_i such that the composition Ψ_h is of a given order. Suzuki [38] and Yoshida [44] propose general simple procedures that allow one to construct composition methods of arbitrarily high order. For orders higher than four they are, however, not very efficient. One is therefore obliged to investigate and to solve the set of order conditions for the γ_i which guarantee that the method Ψ_h of (2.27) has a certain order.

Example 2.4 (Method Used as 'comp' in Fig. 2.3). From the many published examples of composition methods, let us present the coefficients of a method of order 8 with $s = 17$ steps:

$$\begin{aligned}
\gamma_1 = \gamma_{17} &= 0.13020248308889008087881763 \\
\gamma_2 = \gamma_{16} &= 0.56116298177510838456196441 \\
\gamma_3 = \gamma_{15} &= -0.38947496264484728640807860 \\
\gamma_4 = \gamma_{14} &= 0.15884190655515560089621075 \\
\gamma_5 = \gamma_{13} &= -0.39590389413323757733623154 \\
\gamma_6 = \gamma_{12} &= 0.18453964097831570709183254 \\
\gamma_7 = \gamma_{11} &= 0.25837438768632204729397911 \\
\gamma_8 = \gamma_{10} &= 0.29501172360931029887096624 \\
\gamma_9 &= -0.60550853383003451169892108
\end{aligned}$$



This set of coefficients is due to Kahan and Li [15]. The little picture to the right illustrates the 17 steps necessary for obtaining order 8. The zig-zag behaviour is typical for composition methods. It is impossible to get high order without negative step sizes.

For the computations of Fig. 2.3 we use the Störmer/Verlet scheme (2.12) as basic integrator. The resulting composition method is symplectic and (due to $\gamma_i = \gamma_{18-i}$) symmetric.

2.6 Rattle for Constrained Hamiltonian Systems

Let us explain here, how the Störmer/Verlet method (2.12) can be generalized to solve constrained Hamiltonian systems of the form (1.13). Without taking much care of velocity approximations Ryckaert, Ciccotti and Berendsen [31] show how constraints $g(q) = 0$ can be included in the formulation (2.13). Anderson [1] reformulates their method and includes a velocity approximation that satisfies the hidden constraint (1.14). The resulting algorithm, still for separable Hamiltonians, is called 'Rattle'. Later, Jay [16] and Reich [28] observed that the Rattle algorithm can be extended to general Hamiltonians.

Recall that the exact flow of a constrained Hamiltonian system lies on the manifold $\mathcal{M}$, defined in (1.17). Assume therefore that an approximation $(p_n, q_n) \in \mathcal{M}$ is given. One step of the algorithm is defined as

$$\begin{aligned} p_{n+1/2} &= p_n - \frac{h}{2}\Big(\nabla_q H(p_{n+1/2}, q_n) + \nabla_q g(q_n)\lambda_n\Big) \\ q_{n+1} &= q_n + \frac{h}{2}\Big(\nabla_p H(p_{n+1/2}, q_n) + \nabla_p H(p_{n+1/2}, q_{n+1})\Big) \\ 0 &= g(q_{n+1}) \\ p_{n+1} &= p_{n+1/2} - \frac{h}{2}\Big(\nabla_q H(p_{n+1/2}, q_{n+1}) + \nabla_q g(q_{n+1})\mu_n\Big) \\ 0 &= \nabla_q g(q_{n+1})^T \nabla_p H(p_{n+1}, q_{n+1}). \end{aligned} \tag{2.28}$$

For fixed λ_n, the first two equations define uniquely $p_{n+1/2}$ and q_{n+1}, if h is sufficiently small. The parameter λ_n has to be chosen to satisfy $g(q_{n+1}) = 0$. This is possible if the matrix (1.16) is invertible. In the last two equations, μ_n has to be chosen to satisfy the constraint for p_{n+1}.

Similar to the Störmer/Verlet method for unconstrained Hamiltonian systems, this algorithm has many nice properties that are useful within geometric integration:

- the numerical solution stays on the manifold $\mathcal{M}$; i.e., the method (2.28) defines a numerical flow $\Phi_h : \mathcal{M} \to \mathcal{M}$;
- the numerical flow $\Phi_h : \mathcal{M} \to \mathcal{M}$ is a symplectic transformation on $\mathcal{M}$;
- the method is symmetric;
- the method is convergent of order two.

The symplecticity of the numerical flow has first been shown by Leimkuhler and Skeel [23]. The other properties are easy consequences of the definition of the method. This integrator is an ideal candidate as basic method for compositions of the form (2.27). For elaborate proofs and for extensions to higher orders we refer to Sect. VII.1 of [12].

3 Theoretical Foundation of Geometric Integrators

Intuitively, it is quite obvious that a symplectic method should be preferred for the integration of Hamiltonian systems. Similarly, symmetric (more precisely, ρ-reversible) integrators should be preferred for ρ-reversible differential equations. This is motivated by the fact that the symplecticity of the flow is characteristic for Hamiltonian systems, and the ρ-reversibility of the flow is characteristic for ρ-reversible differential equations.

In this section we give some more precise statements on the long-time behaviour of geometric integrators. In particular, we discuss the idea of *backward error analysis* which is the key for a deeper understanding of most numerical phenomena. This idea was common to many numerical analysts already before a systematic study started with the work of Feng [8], Sanz-Serna [33], Yoshida [45], Hairer [9] and many others.

3.1 Backward Error Analysis

Consider an ordinary differential equation

$$\dot{y} = f(y) \tag{3.1}$$

and a numerical method $y_{n+1} = \Phi_h(y_n)$. The idea of backward error analysis consists in searching and studying a *modified differential equation*

$$\dot{y} = f(y) + hf_2(y) + h^2 f_3(y) + \ldots, \tag{3.2}$$

such that the exact time-h flow $\widetilde{\varphi}_h(y)$ of (3.2) is equal to the numerical flow $\Phi_h(y)$. Already simple examples (e.g., trapezoidal rule applied to a quadrature problem $\dot{y} = f(t)$) show that the series in (3.2) cannot be expected to converge in general. The precise statement is the following:

Theorem 3.1. *Consider the differential equation (3.1) with an infinitely differentiable vector field $f(y)$. Assume that the numerical flow admits a Taylor series expansion of the form*

$$\Phi_h(y) = y + hf(y) + h^2 d_2(y) + h^3 d_3(y) + \ldots \ . \tag{3.3}$$

Then, there exist unique vector fields $f_j(y)$ such that for any $N \geq 1$

$$\Phi_h(y) = \widetilde{\varphi}_{h,N}(y) + \mathcal{O}(h^{N+1}),$$

where $\widetilde{\varphi}_{t,N}$ is the exact flow of the truncated modified equation

$$\dot{y} = f(y) + hf_2(y) + \ldots + h^{N-1} f_N(y)$$

(notice that the flow $\widetilde{\varphi}_{t,N}$ also depends on h, because h is a parameter in the modified differential equation).

Let us outline a constructive *proof*. Without taking care of convergence we expand the exact flow of (3.2) into a Taylor series

$$\begin{aligned}
\widetilde{\varphi}_h(y) &= y + h\,\widetilde{y}\,'(0) + \frac{h^2}{2!}\,\widetilde{y}\,''(0) + \frac{h^3}{3!}\,\widetilde{y}\,'''(0) + \ldots \\
&= y + h\big(f(y) + hf_2(y) + h^2 f_3(y) + \ldots\big) \qquad (3.4)\\
&\quad + \frac{h^2}{2!}\big(f'(y) + hf_2'(y) + \ldots\big)\big(f(y) + hf_2(y) + \ldots\big) + \ldots
\end{aligned}$$

(where the prime denotes derivation with respect to time) and compare like powers of h in the expressions (3.4) and (3.3). This yields recurrence relations for the functions $f_j(y)$, namely,

$$\begin{aligned} f_2(y) &= d_2(y) - \frac{1}{2!} f'f(y) \\ f_3(y) &= d_3(y) - \frac{1}{3!}\Big(f''(f,f)(y) + f'f'f(y)\Big) - \frac{1}{2!}\Big(f'f_2(y) + f_2'f(y)\Big). \end{aligned} \tag{3.5}$$

Example 3.1. We consider the pendulum equation $\dot q = p$, $\dot p = -\sin q$ and apply the explicit Euler discretization (2.1). We have $d_j(y) = 0$ for all $j \geq 2$, so that (3.5) yields for the modified equation

$$\begin{pmatrix} \dot q \\ \dot p \end{pmatrix} = \begin{pmatrix} p \\ -\sin q \end{pmatrix} + \frac{h}{2}\begin{pmatrix} \sin q \\ p\cos q \end{pmatrix} + \frac{h^2}{12}\begin{pmatrix} -4\,p\cos q \\ (p^2 + 4\cos q)\sin q \end{pmatrix} + \ldots . \tag{3.6}$$

For the implicit Euler method (2.2) we get (3.6) with h replaced by $-h$. A similar computation yields for the symplectic Euler method (2.3) the modified differential equation

$$\begin{pmatrix} \dot q \\ \dot p \end{pmatrix} = \begin{pmatrix} p \\ -\sin q \end{pmatrix} + \frac{h}{2}\begin{pmatrix} -\sin q \\ p\cos q \end{pmatrix} + \frac{h^2}{12}\begin{pmatrix} 2\,p\cos q \\ (p^2 - 2\cos q)\sin q \end{pmatrix} + \ldots , \tag{3.7}$$

whereas the same equation with h replaced by $-h$ is obtained for the method (2.4). The four pictures of Fig. 3.1 show the exact flow of the modified differential equations (truncated after the $\mathcal{O}(h^2)$ term) corresponding to these four Euler methods together with the numerical solution for the initial value $(p_0, q_0) = (-1.2, 0.7)$. We observe a surprisingly good agreement. This figure should be compared to the exact flow of the unperturbed system (cf. Fig. 1.1).

The $\mathcal{O}(h)$ perturbation in (3.6) provokes the origin to become a source for the explicit Euler method, and a sink for the implicit Euler method. For the two symplectic discretizations we observe that the solutions of the modified equation are periodic, and that the numerical approximation lies near a closed curve. It has thus the correct qualitative behaviour. This is explained by the fact that the differential equation (3.7) is Hamiltonian with

$$\widetilde H(p,q) = \frac{1}{2}\,p^2 - \cos q - \frac{h}{2}\,p\sin q + \frac{h^2}{12}(p^2 - \cos q)\cos q + \ldots ,$$

so that the exact solutions stay on the level curves of $\widetilde H(p,q)$.

3.2 Properties of the Modified Equation

The previous example demonstrates that the numerical solution is extremely close to the exact solution of the modified differential equation. To study properties of the numerical solution, it is therefore justified to investigate instead the corresponding properties of the modified differential equation, and this is often much simpler. Let us collect some properties valid for general differential equations:

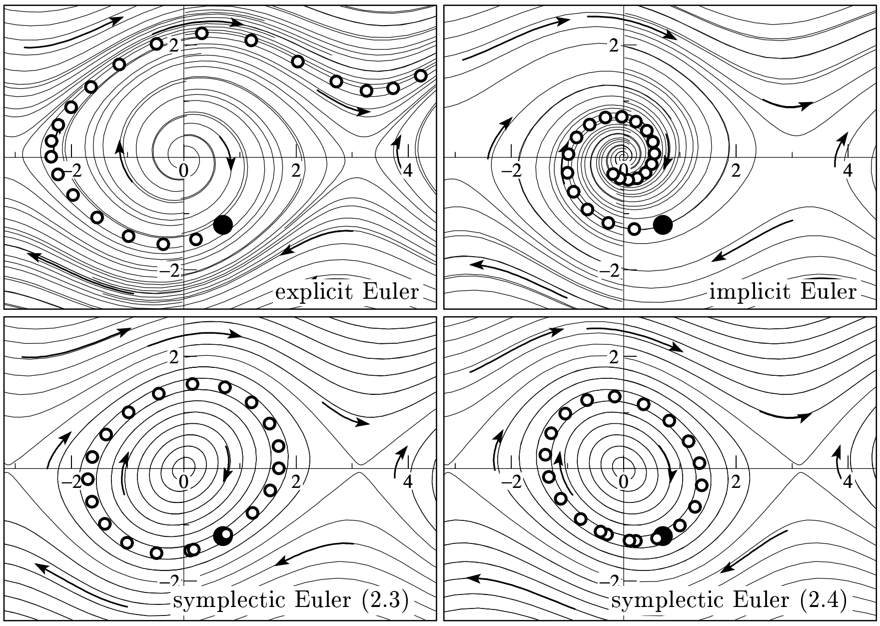


Fig. 3.1. Numerical solution with step size $h = 0.4$ for the four 'Euler methods' of Sect. 2.1 compared to the exact flow of their modified differential equations truncated after the $\mathcal{O}(h^2)$ term.

- if the method is of order r, i.e., $\Phi_h(y) - \varphi_h(y) = \mathcal{O}(h^{r+1})$, then we have $f_j(y) = 0$ for $j = 2, \ldots, r$;
- if $h^{r+1}\delta_{r+1}(y)$ is the leading term of the local truncation error, i.e., $\Phi_h(y) - \varphi_h(y) = h^{r+1}\delta_{p+1}(y) + \mathcal{O}(h^{r+2})$, then we have $f_{r+1}(y) = \delta_{r+1}(y)$;
- if $\Phi_h(y)$ has the modified equation (3.2), then the adjoint method has $f_j^*(y) = (-1)^{j+1} f_j(y)$ as coefficient functions of the modified equation;
- for symmetric methods the modified equation is an expansion in even powers of h; i.e., $f_{2k}(y) = 0$ for all k.

We now turn our attention to Hamiltonian systems and to ρ-reversible differential equations.

Theorem 3.2 (Local Modified Hamiltonian). *Consider a Hamiltonian system (1.1) with smooth Hamiltonian $H : D \to \mathbf{R}$ ($D \subset \mathbf{R}^{2d}$) and apply a symplectic numerical method $\Phi_h(y)$. Then, the vector fields $f_k(y)$ of the modified differential equation are locally Hamiltonian, i.e., locally we have $f_k(y) = J^{-1}\nabla H_k(y)$.*

The *proof* is by induction on k. Its ideas can be traced back to Moser [26], and it can be found in Benettin & Giorgilli [4], Tang [40], Reich [29], and in Chapter IX of [12]. Since the idea of proof is applicable to many other situations, we outline it shortly.

We assume (by induction) that the truncated modified equation

$$\dot{y} = f(y) + hf_2(y) + \ldots + h^{k-1} f_k(y) \tag{3.8}$$

is Hamiltonian. Its flow $\widetilde{\varphi}_{t,k}(y)$ satisfies

$$\Phi_h(y) = \widetilde{\varphi}_{h,k}(y) + h^{k+1} f_{k+1}(y) + \mathcal{O}(h^{k+2}).$$

Since Φ_h and $\widetilde{\varphi}_{h,k}$ are symplectic transformations,

$$J = \Phi_h'(y)^T J \Phi_h'(y) = J + h^{k+1} \Big(f_{k+1}'(y)^T J + J f_{k+1}'(y) \Big) + \mathcal{O}(h^{k+2})$$

holds. Consequently, the matrix $J f_{k+1}'(y)$ is symmetric and the existence of $H_{k+1}(y)$ satisfying $f_{k+1}(y) = J^{-1} \nabla H_{k+1}(y)$ follows from the integrability lemma. □

If H and Φ_h are both defined and smooth on the whole of $\mathbf{R}^{2d}$ or on a simply connected domain D, the functions H_k of the modified Hamiltonian are globally defined. However, as shown by the following example, the functions H_k are in general not globally defined, and the above theorem cannot be used for the study of the long-time behaviour of numerical solutions.

Example 3.2. For the harmonic oscillator $\dot{p} = -q$, $\dot{q} = p$, consider the discretization

$$p_{n+1} = p_n - hq_n - h^2 \gamma \, p_{n+1}, \quad q_{n+1} = q_n + hp_{n+1} - h^2 \gamma \, q_n \tag{3.9}$$

where $\gamma = 0.25/(p_{n+1}^2 + q_n^2)$. It is a $\mathcal{O}(h^2)$ perturbation of the symplectic Euler method and therefore it is a method of order 1. Its symplecticity follows from the fact that it can be written as

$$p_{n+1} = p_n - h \nabla_q S(p_{n+1}, q_n), \quad q_{n+1} = q_n + h \nabla_p S(p_{n+1}, q_n)$$

with $S(p,q) = \frac{1}{2}(p^2 + q^2) - \frac{h}{4} \arg(q + ip)$. Its numerical approximation, plotted in the right picture of Fig. 3.2, is disappointing and does not show the correct qualitative behaviour. This is due to the fact that $S(p,q)$, and hence $H_2(p,q)$, are not globally defined.

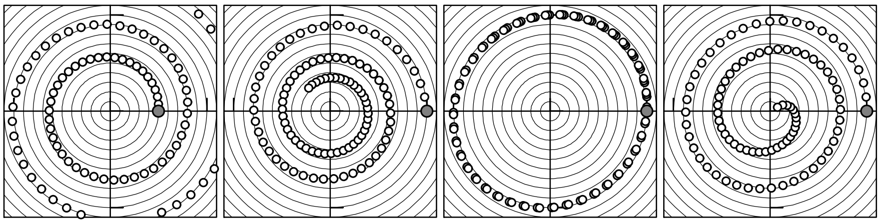
Theorem 3.3 (Global Modified Hamiltonian). *Consider a Hamiltonian system (1.1) with smooth Hamiltonian $H : D \to \mathbf{R}$ ($D \subset \mathbf{R}^{2d}$) and apply the symplectic method*

$$p_{n+1} = p_n - h \, \nabla_q S(p_{n+1}, q_n), \quad q_{n+1} = q_n + h \, \nabla_p S(p_{n+1}, q_n)$$

with generating function

$$S(p,q) = S_1(p,q) + h \, S_2(p,q) + h^2 S_3(p,q) + \ldots \, ,$$

where all $S_k(p,q)$ are globally defined on D. Then, the vector fields $f_k(y)$ of the modified differential equation are globally Hamiltonian, i.e., we have $f_k(y) = J^{-1} \nabla H_k(y)$ with smooth $H_k : D \to \mathbf{R}$.



explicit Euler implicit Euler symplectic Euler sympl. meth. (3.9)

Fig. 3.2. Numerical solution of different first order methods applied to the harmonic oscillator with step size $h = 0.15$.

The *proof* of this theorem is based on the Hamilton-Jacobi differential equation (cf. Sect. IX.3.2 of [12]). Let us mention that all previous methods (symplectic Euler, Störmer/Verlet, symplectic partitioned Runge-Kutta methods, composition methods) satisfy the assumption of Theorem 3.3.

Theorem 3.4 (ρ-Reversible Modified Vector Field). *Consider a ρ-reversible differential equation (cf. Sect. 1.2) and apply a ρ-reversible numerical method $\Phi_h(y)$. Then, the vector fields $f_k(y)$ of the modified differential equation are ρ-reversible, i.e., they satisfy (1.9).*

The *proof* uses the same ideas as that of Theorem 3.2.

3.3 Long-Time Behaviour of Geometric Integrators

Using backward error analysis and in particular the results of Theorems 3.3 and 3.4, we shall show that symplectic integrators (for Hamiltonian systems) and ρ-reversible integrators (for ρ-reversible differential equations) have an improved long-time behaviour. We study the conservation of the Hamiltonian and of general first integrals, and the error growth for integrable systems.

Conservation of the Hamiltonian We know that the Hamiltonian $H(p,q)$ is constant along exact solutions of the Hamiltonian system (energy conservation for mechanical systems). Since the local error of a rth order integrator is of size $\mathcal{O}(h^{r+1})$, we have $H(p_{n+1},q_{n+1}) - H(p_n,q_n) = \mathcal{O}(h^{r+1})$. Summing up these errors, we obtain $H(p_n,q_n) - H(p_0,q_0) = \mathcal{O}(nh^{r+1}) = \mathcal{O}(th^r)$ for $t = nh$, because no cancellation of errors can be expected for general integrators. For symplectic integrators, however, we have the much more favourable estimate

$$H(p_n,q_n) - H(p_0,q_0) = \mathcal{O}(h^r) \quad \text{for} \quad nh \le T \tag{3.10}$$

with an extremely large T (in practice it can be considered as infinity), provided that the numerical solution stays in a compact set. This can be explained with the help of Theorem 3.3 as follows: the modified differential equation is Hamiltonian with

$$\widetilde{H}(p,q) = H(p,q) + h^r H_{r+1}(p,q) + h^{r+1} H_{r+2}(p,q) + \dots . \tag{3.11}$$

The exact flow of the modified equation, and hence also the numerical solution, keep the modified Hamiltonian $\widetilde{H}(p,q)$ exactly constant. If the numerical solution stays in a compact set, the functions $H_j(p,q)$ are bounded along the numerical solution so that (3.10) holds. This argument is not yet rigorous, because the series (3.11) usually does not converge. If one truncates the series suitably, one can rigorously prove (3.10) on exponentially long time intervals, i.e., for $T = \mathcal{O}(e^{\gamma/h})$ with some positive γ (cf. [4], [11], and Sect. IX.7 of [12]).

It is natural to study whether also other first integrals can be well conserved by numerical integrators. Recall that $I(y)$ is a *first integral* of $\dot{y} = f(y)$, if it is constant along all solutions of the differential equations, i.e., if $I'(y)f(y) = 0$ vanishes identically.

Example 3.3. Consider the Kepler problem (1.11). Besides the Hamiltonian (2.17), it has also the *angular momentum*

$$L(p_1,p_2,q_1,q_2) = q_1p_2 - q_2p_1. \tag{3.12}$$

and the so-called *Runge-Lenz-Pauli vector*

$$A(p,q) = \begin{pmatrix} p_1 \\ p_2 \\ 0 \end{pmatrix} \times \begin{pmatrix} 0 \\ 0 \\ q_1p_2 - q_2p_1 \end{pmatrix} - \frac{1}{\sqrt{q_1^2 + q_2^2}} \begin{pmatrix} q_1 \\ q_2 \\ 0 \end{pmatrix} \tag{3.13}$$

as first integrals. As numerical scheme we take the Störmer/Verlet method. We apply it to the Kepler problem with initial values as in Sect. 2.5, and we use the step size $h = 0.02$. Figure 3.3 shows the values of $H(p_n,q_n) - H(p_0,q_0)$ and of the first two components of $A(p_n,q_n) - A(p_0,q_0)$ along the numerical solution. The angular momentum $L(p,q)$ is exactly preserved by the method and therefore not visible in the figure. We see that, in agreement with (3.10), the error in the Hamiltonian is bounded by $\mathcal{O}(h^2)$ on the whole interval of integration. The Runge-Lenz-Pauli vector (3.13), however, is not preserved. The lower picture of Fig. 3.3, where the errors obtained with step size $h = 0.04$ are included in grey, indicates that they behave like $e(h^2t) + \mathcal{O}(h^2)$.

Completely Integrable Systems The example above demonstrated that it is difficult to predict the conservation of general first integrals by numerical methods (even when they are symplectic). There is, however, an important special case for which more information can be obtained. We mention some facts and refer the reader to Chapters X and XI of [12].

We call a Hamiltonian system (1.1) *completely integrable*, if there exists a symplectic transformation

$$(p,q) = \psi(a,\theta), \qquad 2\pi\text{-periodic in } \theta, \tag{3.14}$$

such that the Hamiltonian becomes

$$H(p,q) = H\big(\psi(a,\theta)\big) = K(a). \tag{3.15}$$

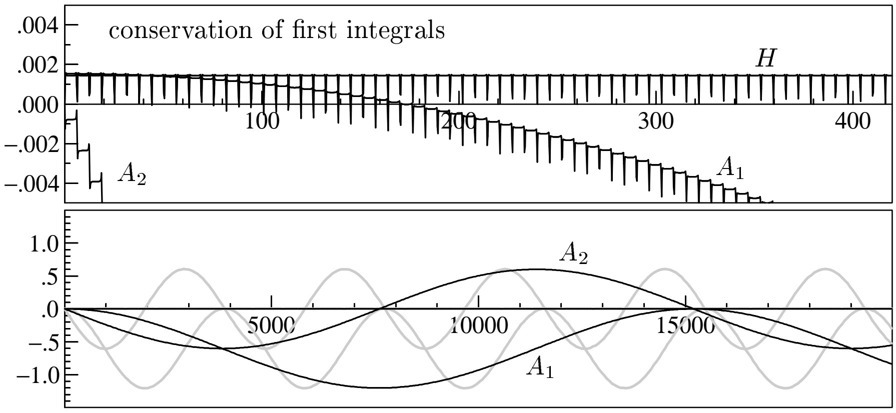


Fig. 3.3. Kepler problem: the Hamiltonian and the first two components of the Runge-Lenz-Pauli vector along the numerical solution of the Störmer/Verlet method with step sizes $h = 0.02$ (black) and $h = 0.04$ (grey).

The new variables (a, θ) are called *action-angle variables*. Suppose we know explicitly the transformation ψ. Since it is symplectic, the Hamiltonian system (1.1) becomes in the new variables

$$\dot a_i = 0, \quad \dot\theta = \omega_i(a), \quad i = 1, \ldots, d$$

with $\omega_i(a) = \partial K / \partial a_i(a)$. This system can be readily solved, and gives $a_i(t) = a_{i0}$, $\theta_i(t) = \theta_{i0} + \omega_i(a_0)t$, so that

$$\big(p(t), q(t)\big) = \psi\big(a_0, \theta_0 + \omega(a_0)t\big).$$

The gives a periodic or quasi-periodic flow on the torus defined by $a = \text{const}$. Among the problems seen in this survey article, Hamiltonian systems with one degree of freedom (harmonic oscillator, pendulum) and the Kepler problem are completely integrable. Under some additional technical assumptions (see the general reference [12]), *symplectic numerical integrators* applied to such completely integrable Hamiltonian systems have the following interesting properties:

- the global error grows at most linearly with time, more precisely, for $t = nh$ we have

$$p_n - p(t) = \mathcal{O}(h^r t), \quad q_n - q(t) = \mathcal{O}(h^r t);$$

- first integrals that only depend on the action variables are well preserved on exponentially long time intervals; i.e., if $I(p, q)$ is such that $I(\psi(a, \theta))$ is independent of θ, then

$$I(p_n, q_n) - I(p_0, q_0) = \mathcal{O}(h^r) \quad \text{for} \quad nh \le T$$

  with T as in (3.10).

This result has to be seen in contrast to general methods, where the global error increases typically quadratically with time, and where the error in first integrals drifts linearly from the correct value.

We finally mention that the notion of complete integrability can be reinterpreted for ρ-reversible differential equations (not necessarily Hamiltonian), and the same results (linear error growth, conservation of action variables) hold for ρ-reversible integrators applied to such systems. Let us illustrate this at an interesting example.

Example 3.4 (Toda lattice). Let us consider particles on a line interacting pairwise with exponential forces, and suppose periodic boundary conditions $q_{d+1} = q_1$. The Hamiltonian is given by

$$H(p,q) = \sum_{k=1}^{d} \Big(\frac{1}{2}\, p_k^2 + \exp(q_k - q_{k+1}) \Big).$$

The corresponding Hamiltonian system has the interesting property that the d eigenvalues of the matrix

$$L = \begin{pmatrix} a_1 & b_1 & & & b_d \\ b_1 & a_2 & b_2 & 0 & \\ & b_2 & \ddots & \ddots & \\ & 0 & \ddots & a_{d-1} & b_{d-1} \\ b_d & & & b_{d-1} & a_d \end{pmatrix}, \qquad \begin{aligned} a_k &= -\tfrac{1}{2} p_k \\ b_k &= \tfrac{1}{2} \exp\big(\tfrac{1}{2}(q_k - q_{k+1})\big) \end{aligned}$$

are first integrals. This Hamiltonian system is completely integrable within the class of Hamiltonian systems and also within the class of ρ-reversible systems, and the action variables are related to the eigenvalues of the matrix L.

We consider the case $d = 3$, and we apply an implicit Runge-Kutta method (Lobatto IIIB, $s = 3$, order $r = 4$) which is symmetric but not symplectic. The upper picture of Fig. 3.4 shows the Euclidean norm of the vector of errors in the eigenvalues of L. It is of size $\mathcal{O}(h^4)$ on the whole interval of integration. The lower picture shows the norm of the global error, and we nicely observed the linear error growth. This confirms the statement about integrable systems of this section.

4 Matlab Programs of '`GniCodes`'

We explain a few Matlab programs that implement the most important geometric integrators of the previous sections. They are collected in the Matlab package `GniCodes` which is available (together with short installing instructions) on the web at the address

http://www.unige.ch/math/folks/hairer

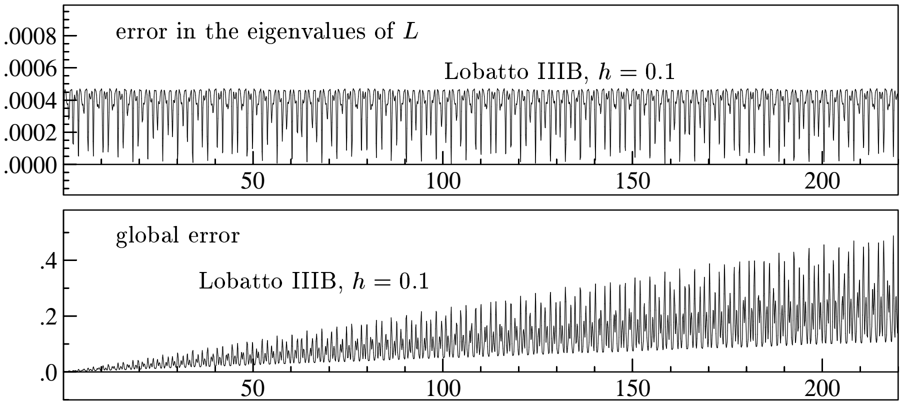


Fig. 3.4. The Toda lattice: error in the eigenvalues of L (upper picture), and global error of the numerical solution for an implicit Runge-Kutta method.

Fortran 77 versions of the programs are also available at the same address. Another Matlab package related to geometric integration is `DiffMan` of [7]. The philosophy of our package, however, is completely different and it is closer to the standard Matlab ODE suite (`ode45`, `ode23`, etc) of Shampine and Reichelt [34].

4.1 Standard Call of Integrators

We give an overview of how to use the three classes of geometric integrators that are implemented in the package `GniCodes` for the moment. For the solution of second order initial value problems

$$\ddot{q} = g(q), \quad q(0) = q_0, \quad \dot{q}(0) = \dot{q}_0 \tag{4.1}$$

all these methods have the same syntax, and are usually called as

```
[T,P,Q] = gni_meth('g',tspan,y0,options,...);
```

where `gni_meth` has to be replaced by `gni_irk2` for the implicit Runge-Kutta method, by `gni_lmm2` for the linear multistep method, and by `gni_comp` for the composition method based on the Störmer/Verlet scheme. The '2' in `irk2` and in `lmm2` expresses the fact that these programs are applicable only to second order differential equations (4.1). For the composition method `gni_comp` there is the possibility to define the basic integrator by altering the `options` structure, so that the method can be used for the solution of any differential equation. The relevant syntax will be explained in Sect. 4.6 below.

The meaning of the arguments in a call of `gni_meth` is as follows:

`'g'` This argument must be a string containing the name of a Matlab file describing the problem. The syntax for such a file is described in Sect. 4.2 below.

`tspan` should contain the time span over which the problem is to be solved. It has to be given in the form `[t0,tf]`.

`y0` This is a vector containing the initial values for q and $\dot{q}$. The initial values for q are given by the first d components and those for $\dot{q}$ by the remaining d components of `y0`.

`options` This argument should contain a GNI options structure created by `gniset` (the syntax of `gniset` is the same as for the standard `odeset` function). This option structure contains additional instructions for the integrator.

`...` After `options`, an arbitrary number of optional arguments can be given. These arguments are passed over to the function `F`.

The list of available options differs slightly from the standard ODE suite. Some of these options are also function-dependent and will be explained in the sequel. The following options are available for every integrator:

`'OutputFcn'` This is a string containing the name of an output function. The format for the output functions is the same as for the standard ODE suite, in particular the standard `odeplot` output function can be used (and will be used as the default output function). The vector passed to the output function contains in its d first components the value of the solution and in its d remaining components the values for its time-derivatives. If this parameter is set to `phaseplot`, the solution is drawn in the phase space corresponding to the first components of $(q, \dot{q})$.

`'OutputSel'` As for the standard ODE suite, this contains a vector of indices determining which components of the solution will be passed to the output function. By default, all the indices (including those corresponding to the time-derivative of the solution) are passed through.

`'OutputSteps'` tells the integrator which steps to take into account for the output. For example, if `OutputSteps` is equal to 10, only every 10th step generates some output. Putting `OutputSteps` equal to 0, output is made available only at the beginning and the end of the integration interval.

`'Vectorized'` has the same meaning as for the standard ODE suite.

`'Events'` If this option is set to `'on'`, event location is enabled. See Sect. 4.3 for an explanation of how to use event location.

`'StepSize'` Size h of one integration step. It is slightly altered by the code, if the length of the integration interval is not an integer multiple of h.

`'NumSteps'` is the number of integration steps. This option is only used when `StepSize` is not specified. If neither is specified, a warning is issued and the default step size $h = 0.01$ is used.

`'Method'` allows to select the type of method to use. The list of available methods depends on the integrator and is listed in the corresponding sections below.

Note that (like for `ode45` for example) the arguments `tspan`, `y0`, and `options` are optional and can be defined in the file `F.m` instead.

On output, gni_meth returns three vectors [T,Q,P], containing the times at which the solution was evaluated, as well as the values of q and $\dot{q}$ at these times. If event location is turned on, additional return values are given as described in Sect. 4.3 below.

4.2 Problem Description

The problem to be solved should be described in a .m file. In the most simple case, this file only returns the right-hand side of the second-order differential equation. E.g., in order to solve the equation $\ddot{q} = -q^3$, one may create a file trivial.m containing the following:

```
function out = trivial(t,q)
   out = -q^3;
```

From the command line, one would then use it for example as

```
options = gniset('StepSize', 0.1);
gni_meth('trivial', [0 10], [0 2.5], options);
```

An additional parameter flags can be used by the integrator to retrieve default parameters for the problem. Assume we want to solve the previous problem between $t = 0$ and $t = 10$, using a step size of 0.1 and with initial values $q(0) = 0$ and $\dot{q}(0) = 2.5$. We could then define the file trivial.m as

```
function [out,out2,out3] = trivial(t,q,flags)
if (nargin < 3) | isempty(flags)
   out = -q^3;
else
   switch flags
   case 'init',
      out = [0 10];
      out2 = [0 2.5];
      out3 = gniset('StepSize', 0.1);
   end
end
```

and call it from the command line in the most simple possible way as

```
gni_meth('trivial');
```

For a system of differential equations, out and q are column vectors. If the option 'Vectorized' is set to 'on' in the GNI options structure, the integrator may request to evaluate the right-hand side of the problem for several values of t and q in one call. If the problem is of dimension d and the integrator requests m values, t is a line vector of size m and q is a $d \times m$ matrix. The right-hand side is expected to be also a $d \times m$ matrix. If 'Vectorized' is set to 'off', one can safely assume that $m = 1$.

When vectorized correctly, the `.m` file for the Kepler problem with initial values as in Sect. 2.5 looks like

```
function [out,out2,out3] = kepler(t,q,flags,ecc)
if (nargin < 3) | isempty(flags)
   rad=q(1,:).*q(1,:)+q(2,:).*q(2,:);
   rad=rad.*sqrt(rad);
   out(1,:)=-q(1,:)./rad;
   out(2,:)=-q(2,:)./rad;
else
   switch flags
   case 'init',
      if (ecc < 0) | (ecc >= 1)
         error('The␣eccentricity␣must␣lie␣between␣0␣and␣1');
      end
      out = [0 2*pi];
      out2 = [1-ecc,0,0,sqrt((1+ecc)/(1-ecc))];
      out3 = gniset('NumSteps',50,'Vectorized','on','Events','off',...
         'OutputFcn','phaseplot','OutputSel',[1,2]);
   end
end
```

Notice that this problem depends on the eccentricity `ecc`, which has been appended to the end of the parameter list. To solve this problem with `ecc`$=0.6$, just type

```
gni_meth('kepler',[ ],[ ],[ ],0.6);
```

Entering `[ ]` in the parameter list tells the integrator to use the default values of the problem definition file instead. The parameter `ecc` is again simply appended at the end of the parameter list.

4.3 Event Location

In many situations (for example the computation of Poincaré sections), it is useful to know at which times some *event function* $g(t, q(t), p(t))$ vanishes. This is usually referred to as *event location*.

Event location is implemented in the GNI suite in a way that is again very similar to the standard ODE suite implementation. It can be enabled by specifying the value `'on'` for the `'Events'` selector of the GNI options structure.

When event location is turned on, the integrator can be called as

```
[T,P,Q,TE,PE,QE,IE] = gni_meth('g',tspan,y0,options,...);
```

The output vector `TE` contains the times at which events occurred. The vectors `PE` and `QE` contain the values of the solution and its derivative at these times. If more than one event function is defined, the vector `IE` contains the index of the event function that triggered the event.

When event location is turned on, the problem description file is expected to respond to the `flags` set to `'events'` by returning in the first output argument a vector of event functions. Furthermore it is supposed to return in the second and third output arguments vectors telling the integrator whether the corresponding event is terminal or not and which types of zero-crossings to consider. When a terminal event is encountered, the integration stops, whether the end of the integration interval has been reached or not. The following example shows how to define a problem description file that allows to retrieve the times at which the solution either crosses 1 upwards or 0 in any direction. The integration stops whenever the solution crosses -2 downwards.

```
function [out,out2,out3] = trivial(t,q,flags)
if (nargin < 3) | isempty(flags)
   out = -q^3;
else
   switch flags
   case 'init',
      out = [0 10];
      out2 = [-1 5];
      out3 = gniset('StepSize', 0.1,'Events','on');
   case 'events',
      out = [q(1)-1,q(1),q(1)+2];
      out2 = [0 0 1];
      out3 = [1 0 -1];
   end
end
```

4.4 Program `gni_irk2`

The program `gni_irk2` uses an implicit Runge-Kutta scheme to solve second order differential equations $\ddot{q} = g(q)$. The following selectors of the GNI options structure are specific to `gni_irk2`:

`'Method'` This selector allows to specify which scheme is to be used. The accepted values for `'Method'` are `'G4'`, `'G8'`, and `'G12'`. The letter 'G' refers to the fact that all of these methods are Gauss methods (cf. Example 2.2), and the number that follows indicates the order of the corresponding method. If no method is specified, `'G12'` is used.

`'MaxIter'` Since the schemes are implicit, a non-linear system of equation has to be solved at every integration step. This is achieved through a fixed point iteration. This selector allows to specify the maximal number of iterations that are performed. The default value is 50.

The coefficients of the different methods are contained in the file `coeff_irk2`. New methods can easily be incorporated.

Let us shortly explain the meaning of the required coefficients. The arrays `C`, `B`, `BC` correspond to the vectors with coefficients c_i, $\widehat{b}_i$, $\widetilde{b}_i$ of (2.22), the 2-dimensional array `AA` to the matrix $\widetilde{A}$. Further coefficients are needed for an efficient solution of the nonlinear Runge-Kutta equations of (2.22), which are equivalent to

$$Q_i = q_n + hc_i v_n + h^2 \sum_{j=1}^{s} \widetilde{a}_{ij}\, g(Q_j), \quad i = 1, \dots, s. \tag{4.2}$$

We solve this system by fixed point iteration and we use

$$Q_i^0 = q_n + hc_i v_n + h^2 \sum_{j=1}^{s+3} e_{ij}\, g(Q_{j,n-1}) \tag{4.3}$$

as starting guess, where $Q_{1,n-1}, \dots, Q_{s,n-1}$ are the internal stage values of the previously computed step, $Q_{s+1,n-1} = q_{n-1}$, $Q_{s+2,n-1} = q_n$, and

$$Q_{s+3,n-1} = q_n + \sum_{i=1}^{s} \mu_i \big(Q_{i,n-1} - q_{n-1}\big) + h\mu_{s+1} v_{n-1} + h\mu_{s+2} v_n$$

is an approximation to the solution at $t = t_{n-1} + \mu h$. The coefficients μ, μ_i, e_{ij} (stored in the arrays `SM`, `AM`, and `E`) are determined such that (4.3) coincides as far as possible with the Taylor series of the solution of (4.2). We refer to [12, Sect. VIII.6.1] and [20] for more details.

4.5 Program `gni_lmm2`

Linear multistep methods (2.25) for second order differential equations are implemented in the code `gni_lmm2`. Since these methods are not self-starting, we have to provide starting approximations. This is done by a call to `gni_irk2` with `'Method'` set to `'G12'`. For the moment we have implemented the three methods of Table 1, and the `'Method'` options are `'801'`, `'802'`, and `'803'`, respectively.

Table 1. Symmetric multistep methods for second order problems.

	method 801		method 802		method 803	
i	C_{i-1}	$12096\, B_i$	C_{i-1}	$120960\, B_i$	C_{i-1}	$8640\, B_i$
1	1	17671	1	192481	1	13207
2	0	-23622	2	6582	1	-8934
3	1	61449	3	816783	1	42873
4	1	-50516	3.5	-156812	1	-33812

The coefficients of the methods are stored in the separate file `coeff_lmm2` as follows. The generating polynomial $R(\zeta)$ has $\zeta = 1$ as a double zero, and therefore it can be written as

$$R(\zeta) = (\zeta - 1)^2 (C_0 + C_1\zeta + C_2\zeta^2 + \dots + C_{k-2}\zeta^{k-2}).$$

Since for explicit symmetric methods these coefficients satisfy $B_{K-i} = B_i$ (with $B_0 = 0$) and $C_{K-2-i} = C_i$, only those given in Table 1 have to be specified. The coefficients C_i and B_i uniquely determine the method (2.25).

4.6 Program `gni_comp`

This program allows to easily implement general composition methods. A composition method (2.27) is characterized by the set of coefficients $\{\gamma_i\}$ and by the basic method Φ_h. They are controlled by the following two options:

`'Method'` This option allows to choose between several predefined sets of coefficients γ_i. The available methods are `'21'`, `'43'`, `'45'`, `'67'`, `'69'`, `'815'`, `'817'`, and `'1033'`. These methods are of order 2, 4, 6, 8, and 10 respectively.

`'PartialFlow'` This option allows the user to specify the name of a Matlab file defining the basic method Φ_h. The default method is the Störmer/Verlet method.

For reasons of efficiency we assume the basic method to be of the form

$$\Phi_h = \omega_{h/2} \circ \beta_{h/2,h,h/2} \circ \alpha_{h/2} \ , \tag{4.4}$$

where the following simplification formula holds:

$$\beta_{*,*,ha} \circ \alpha_{ha} \circ \omega_{hc} \circ \beta_{hc,*,*} = \beta_{*,*,ha+hc} \circ \beta_{ha+hc,*,*} \ . \tag{4.5}$$

Every one-step method can be written in the form (4.4) by choosing $\omega_{h/2} = \alpha_{h/2} = id$ (the identity), and $\beta_{hc,hb,ha} = \Phi_{hb}$. But this is not the reason for writing Φ_h in this apparently complicated form. The advantage of the representation (4.4) is that in many important situations a large part of the work for evaluating Φ_h can be put into $\alpha_{h/2}$ and $\omega_{h/2}$ and, by the simplification formula (4.5), this part can be avoided unless at grid points where an output of the solution is required.

The code for the basic method must have the following structure:

```
function [outP,outQ] = basic(t,P,Q,ode,ha,hb,hc,first,last,flags,args)
if isempty(flags)
   if (first)
      apply α_ha to the vectors P and Q
   end
   apply β_hc,hb,ha
   if (last)
      apply ω_hc
   end
else
   switch flags
   case 'init',
      perform some initialization
   case 'done',
```

```
      perform some cleanup
   end
end
```

For example, the Störmer/Verlet method (2.12) for $\ddot{q} = g(q)$, considered as a splitting method (2.16), can be written as (4.4) with $\omega_{h/2} = id$, $\beta_{hc,hb,ha} = \varphi^{[1]}_{hc} \circ \varphi^{[2]}_{hb}$, and $\alpha_{ha} = \varphi^{[1]}_{ha}$. This presentation satisfies condition (4.5), because $\varphi^{[1]}_t$ has the group property. The Matlab program for this basic method is

```
function [outP,outQ] = stverl(t,P,Q,ode,ha,hb,hc,first,...
   last,flags,varargin)
if isempty(flags)
   if (first)
      Q = Q + ha*P;
   end
   F = feval(ode,t,Q,varargin{:});
   outP = P + hb*F;
   outQ = Q + hc*outP;
end
```

The actual implementation uses compensated summation (to reduce round-off error) and it is the default method used by `gni_comp`.

5 Some Typical Applications

Let us finally illustrate the use of our programs at some typical examples, where geometric integrators are recommended. We start with a comparison of geometric integrators for second order ordinary differential equations. We then show how Poincaré sections can be computed, and we terminate with a slightly more sophisticated use of composition methods.

5.1 Comparison of Geometric Integrators

Often it is difficult to decide which integrator is the best for a given problem. The implicit Runge-Kutta and composition methods have a sound theoretical basis, but they typically need more function evaluations per step than linear multistep methods. On the other hand, multistep methods have larger local error, so that smaller step sizes are required. The best choice is in general problem dependent.

Consider first the Kepler problem with eccentricity `ecc`= 0.6. The file `kepler.m`, containing the problem description, is explained in Sect. 4.2. We compute the solution over the interval $[0, 400\pi]$ with many different step sizes. As we have seen in Fig. 2.3 the efficiency of all three classes of integrators (implicit Runge-Kutta, multistep, composition) is about the same for this problem. This need not always be the case.

As another example consider the 6-body problem (sun and the five outer planets) with data and initial values as in Chap. I of [12] on a relatively short time interval $[0, 500\,000]$. Similar as in Fig. 2.3 we show in Fig. 5.1 the work precision diagram of the different methods. It is somewhat surprising that for this problem (with orbits of very small eccentricity) the linear multistep method is the most efficient integrator. The cpu times in Fig. 2.3 and in Fig. 5.1 are obtained with Fortran implementations of the codes.

5.2 Computation of Poincaré Sections

Consider the *Hénon-Heiles* Hamiltonian

$$H(p_1, p_2, q_1, q_2) = \frac{1}{2}(p_1^2 + p_2^2) + \frac{1}{2}(q_1^2 + q_2^2) + q_1^2 q_2 - \frac{1}{3} q_2^3. \tag{5.1}$$

The corresponding Hamiltonian system is integrable for sufficiently small energy, e.g., for the initial values $p_1(0) = p_2(0) = q_1(0) = q_2(0) = 0.18$, which we take for our computations. This means that the solution stays on a two-dimensional torus in the four-dimensional phase space. Its intersection with the hyperplane $q_1 = 0$ (Poincaré section) thus gives a closed curve in the phase space. We study the projection of this curve onto the (q_2, p_2)-plane.

The left picture of Fig. 5.2 shows the Poincaré section for the numerical solution obtained by `dop853` (an explicit Runge-Kutta method of order eight with step size control, see [14, Appendix]) and tolerance $Atol = Rtol = 10^{-5}$ on the interval $[0, 100\,000]$. The picture clearly demonstrates that the numerical solution is qualitatively wrong as it does not remain on a closed curve. The same experiment with the three geometric integrators `gni_lmm2`, `gni_irk2`, and `gni_comp` gives a correct simulation of the system, and it cannot be distinguished from a picture for the exact solution (right picture of Fig. 5.2). If we use step sizes such that the error of the Hamiltonian remains

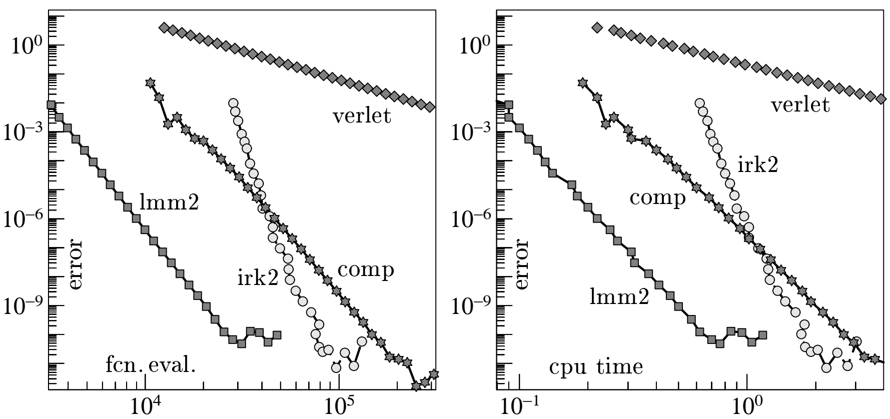


Fig. 5.1. Work precision diagrams for the Störmer/Verlet scheme and for three methods of order eight; implicit Runge-Kutta method (irk2), composition method (comp), and linear multistep method (lmm2), applied to the outer solar system.

below 10^{-5}, the code gni_lmm2 (with $h = 0.22$) requires 454 716 function evaluations, gni_irk2 (with $h = 1.5$) needs 3 731 867 function evaluations, and gni_comp (with $h = 1.2$) 1 416 661 function evaluations. For comparison, the code dop853 requires 1 216 680 evaluations of the vector field, but the error in the Hamiltonian increases linearly with time. The high number of function evaluations for gni_irk2 is due to the fact that for low accuracy requirements (large step size) the convergence of the fixed point iterations for solving the nonlinear Runge-Kutta equations is rather slow.

For the computation of the Poincaré section (Fig. 5.2) we have used the following program:

```
function [out,out2,out3] = henon(t,q,flags)
if (nargin < 3) | isempty(flags)
   out(1,:)=-q(1,:).*(1+2*q(2,:));
   out(2,:)=-q(2,:).*(1-q(2,:)) - q(1,:).^2;
else
   switch flags
   case 'init',
      out = [0 100000];
      out2 = [0.18 0.18 0.18 0.18];
      out3 = gniset('StepSize',0.22,'Vectorized','on',...
         'Events','on','OutputSteps',0);
   case 'events',
      out = [q(1)];
      out2 = [0];
      out3 = [0];
   end
end
```

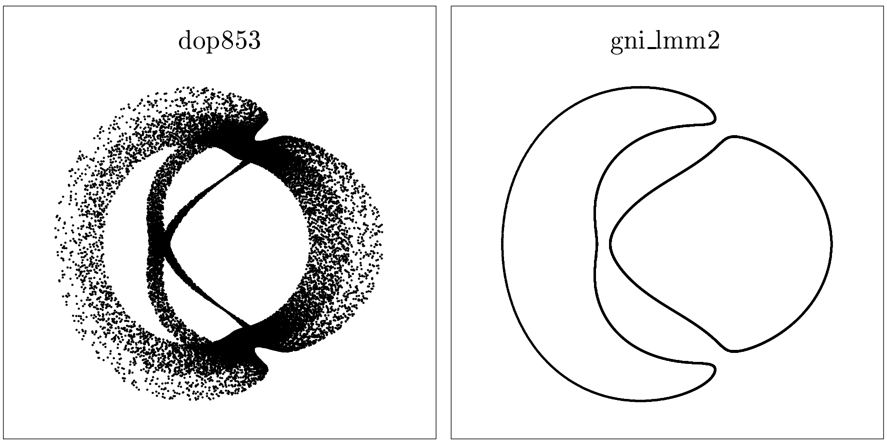


Fig. 5.2. Poincaré section for the Hénon-Heiles problem: dop853 with tolerance $Atol = Rtol = 10^{-5}$ (left picture) and gni_lmm2 with step size $h = 0.22$ (right picture); integration interval $[0, 100\,000]$.

The plot of the Poincaré section is then obtained with

```
[T,Q,P,TE,QE,PE]=gni_lmm2('henon');
plot(QE(:,2),PE(:,2),'.');
```

5.3 'Rattle' as Basic Integrator for Composition

As a final example, we present a Matlab implementation of the Rattle algorithm (2.28) applied to the *two-body problem on the sphere* as introduced in Example 1.4. We follow the description of Sect. 4.6 and we do it in such a way that it can be used as basic integrator for the composition method gni_comp.

A possible implementation is the following program:

```
function [outP,outQ] = rattwo(t,P,Q,gradpot,ha,hb,hc,first,last,...
   flags,varargin)
if isempty(flags)
   F = feval(gradpot,t,Q,varargin{:});
   EP = P - ha*F;
   EQ = Q + hb*EP;
   EE1 = EQ(1:3)'*EQ(1:3);
   EQ1 = EQ(1:3)'*Q(1:3);
   EE2 = EQ(4:6)'*EQ(4:6);
   EQ2 = EQ(4:6)'*Q(4:6);
   BET1 = 1 - EE1;
   ALAM1 = -BET1/(hb*(EQ1+sqrt(BET1+EQ1^2)));
   BET2 = 1 - EE2;
   ALAM2 = -BET2/(hb*(EQ2+sqrt(BET2+EQ2^2)));
   outP = EP - [ALAM1*Q(1:3);ALAM2*Q(4:6)];
   outQ = Q + hb*outP;
   if (last)
      F = feval(gradpot,t,outQ,varargin{:});
      outP = outP - hc*F;
      AMU1 = sum(outP(1:3).*outQ(1:3));
      AMU2 = sum(outP(4:6).*outQ(4:6));
      outP = outP - [AMU1*outQ(1:3);AMU2*outQ(4:6)];
   end
end
```

We remark that, due to the simple structure of the Hamiltonian, the method is explicit in $p_{n+1/2}$, q_{n+1} and p_{n+1}, and it is implicit only in the Lagrange multipliers. Since the constraints are quadratic, we are only concerned with the solution of a scalar quadratic equation for each of the components of λ_n. This is why no iterations are involved in the above program. Since $\nabla_q H(p,q)$ does not depend on p, the first equation of (2.28) can be combined with the fourth equation of the preceeding step into one formula to give $p_{n+1/2} = p_{n-1/2} + \dots$. This is the reason for putting the computation of p_{n+1} into $\omega_{h/2}$ of the decomposition (4.4).

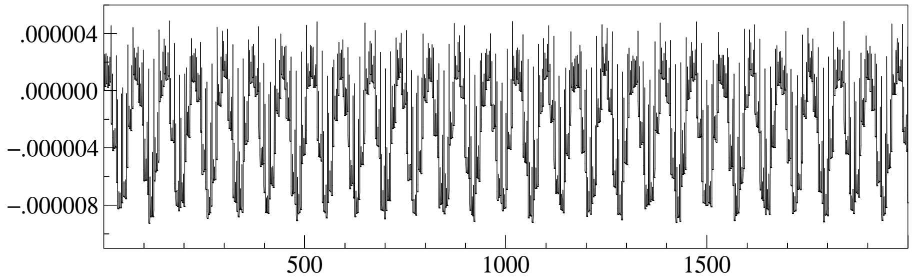


Fig. 5.3. Error in the Hamiltonian of an 8th order composition method with Rattle as basic integrator; step size $h = 0.15$

The argument `gradpot` in the function `rattwo` is a function that computes the gradient of the potential (i.e., $\nabla_q U(q) = \nabla_q H(p,q)$). For the two-body problem on the sphere it is given by

```
function [out,out2,out3] = twobodysphere(t,q,flags)
if (nargin < 3) | isempty(flags)
   prod = q(1:3)'*q(4:6);
   out = -q([4:6,1:3])/(1-prod^2)^(3/2);
else
   switch flags
   case 'init',
      out = [0 10];
      phi = [1.3 -2.1];
      theta = [2.1 -1.1];
      out2([1 4]) = cos(phi).*sin(theta);
      out2([2 5]) = sin(phi).*sin(theta);
      out2([3 6]) = cos(theta);
      dphi = [1.2 0.1];
      dtheta = [0.1 -0.5];
      out2([7 10]) = -dphi.*sin(phi).*sin(theta) ...
         + dtheta.*cos(phi).*cos(theta);
      out2([8 11]) = dphi.*cos(phi).*sin(theta) ...
         + dtheta.*sin(phi).*cos(theta);
      out2([9 12]) = -dtheta.*sin(theta);
      out3 = gniset('StepSize',0.02,'Vectorized','off',...
         'Events','off','PartialFlow','rattwo','OutputFcn',...
         'sphereplot','OutputSteps',5,'Method','817');
   end
end
```

Here, the option `sphereplot` permits us to get a 3-dimensional plot of the solution. The problem is then simply solved by calling

```
gni_comp('twobodysphere');
```

The experiment of Fig. 5.3 confirms the statement of Theorem 3.3 for the Rattle algorithm applied to constrained Hamiltonian systems. We have plotted the error of the Hamiltonian for the composition method of Example 2.4, applied with step size $h = 0.15$ to the interval $[0, 2000]$. As expected for a symplectic integrator, there is no drift in the error of the Hamiltonian. This is also confirmed by integrations over much longer time intervals.

Hints for the implementation of the *rigid body* problem of Example 1.5 can be found in [10], where also the extension of Theorem 3.3 to numerical methods for Hamiltonian systems on manifolds (including the Rattle algorithm) is proved.

Acknowledgement

E.H. is grateful to the organizers of the 2002 Durham summer school, in particular to James F. Blowey, to Alan W. Craig, and to the local expert Sebastian Reich, for providing a sympathical and stimulating athmosphere. We also wish to thank Christian Lubich, Gerhard Wanner and numerous other colleagues for their useful discussions on the topic of this work.

References

1. H.C. Andersen (1983), Rattle: a “velocity” version of the Shake algorithm for molecular dynamics calculations, *J. Comput. Phys.* 52:24–34.
2. V.I. Arnold (1989), *Mathematical Methods of Classical Mechanics*, Second Edition, Springer-Verlag.
3. F. Bashforth (1883), *An attempt to test the theories of capillary action by comparing the theoretical and measured forms of drops of fluid. With an explanation of the method of integration employed in constructing the tables which give the theoretical form of such drops, by J.C.Adams.* Cambridge Univ. Press.
4. G. Benettin & A. Giorgilli (1994), On the Hamiltonian interpolation of near to the identity symplectic mappings with application to symplectic integration algorithms, *J. Statist. Phys.* 74:1117–1143.
5. J.C. Butcher (1963), Coefficients for the study of Runge-Kutta integration processes, *J. Austral. Math. Soc.* 3:185–201.
6. J.C. Butcher (1964), Implicit Rung-Kutta processes, *Math. Comput.* 18:50-64.
7. K. Engø, A. Marthinsen and H.Z. Munthe-Kaas (2001), DiffMan: an object-oriented MATLAB toolbox for solving differential equations on manifolds. Special issue: Themes in geometric integration. *Appl. Numer. Math.* 39:349–365.
8. K. Feng (1991), Formal power series and numerical algorithms for dynamical systems. In *Proceedings of international conference on scientific computation, Hangzhou, China, Eds. Tony Chan & Zhong-Ci Shi, Series on Appl. Math.* 1:28–35.
9. E. Hairer (1994), Backward analysis of numerical integrators and symplectic methods, *Annals of Numerical Mathematics* 1:107–132.
10. E. Hairer (2002), Global modified Hamiltonian for constrained symplectic integrators. *Numer. Math.* to appear.
11. E. Hairer and Ch. Lubich (1997), The life-span of backward error analysis for numerical integrators, *Numer. Math.* 76:441–462.

12. E. Hairer, C. Lubich and G. Wanner (2002), *Geometric Numerical Integration. Structure-Preserving Algorithms for Ordinary Differential Equations*. Springer Series in Computational Mathematics 31, Springer, Berlin.
13. E. Hairer, C. Lubich and G. Wanner (2003), Geometric numerical integration illustrated by the Störmer/Verlet method. *Acta Numerica* to appear.
14. E. Hairer, S.P. Nørsett and G. Wanner (1993), *Solving Ordinary Differential Equations I. Nonstiff Problems, 2nd edition*. Springer Series in Computational Mathematics 8, Springer, Berlin.
15. W. Kahan and R.-C. Li (1997), Composition constants for raising the orders of unconventional schemes for ordinary differential equations, *Math. Comput.* 66:1089–1099.
16. L. Jay (1994), *Runge-Kutta type methods for index three differential-algebraic equations with applications to Hamiltonian systems*, Thesis No. 2658, 1994, Univ. Genève.
17. U. Kirchgraber (1986), Multi-step methods are essentially one-step methods, *Numer. Math.* 48:85–90.
18. V.V. Kozlov and A.O. Harin (1992), *Kepler's problem in constant curvature spaces*, Celestial Mech. Dynam. Astronom. 54:393–399.
19. W. Kutta (1901), Beitrag zur näherungsweisen Integration totaler Differentialgleichungen, *Zeitschr. für Math. u. Phys.* 46:435–453.
20. M.P. Laburta (1998), Construction of starting algorithms for the RK-Gauss methods, *J. Comput. Appl. Math.* 90:239–261.
21. J.D. Lambert and I.A. Watson (1976), Symmetric multistep methods for periodic initial value problems, *J. Inst. Maths. Applics.* 18:189–202.
22. F.M. Lasagni (1988), Canonical Runge-Kutta methods, *ZAMP* 39:952–953.
23. B.J. Leimkuhler and R.D. Skeel (1994), Symplectic numerical integrators in constrained Hamiltonian systems, *J. Comput. Phys.* 112:117–125.
24. R.I. McLachlan (1995), On the numerical integration of ordinary differential equations by symmetric composition methods, *SIAM J. Sci. Comput.* 16:151–168.
25. R.I. McLachlan and G.R.W. Quispel (2002), Splitting methods. *Acta Numerica* 2002:341–434.
26. J. Moser (1968), Lectures on Hamiltonian systems, *Mem. Am. Math. Soc.* 81:1–60.
27. G.D. Quinlan and S. Tremaine (1990), Symmetric multistep methods for the numerical integration of planetary orbits, *Astron. J.* 100:1694–1700.
28. S. Reich (1994), Momentum conserving symplectic integrators, *Phys.* D 76:375–383.
29. S. Reich (1999), Backward error analysis for numerical integrators, *SIAM J. Numer. Anal.* 36:1549–1570.
30. C. Runge (1895), Ueber die numerische Auflösung von Differentialgleichungen, *Math. Ann.* 46:167–178.
31. J.-P. Ryckaert, G. Ciccotti and H.J.C. Berendsen (1977), Numerical integration of the cartesian equations of motion of a system with constraints: molecular dynamics of n-alkanes, *J. Comput. Phys.* 23:327–341.
32. J.M. Sanz-Serna (1988), Runge-Kutta schemes for Hamiltonian systems, *BIT* 28:877–883.
33. J.M. Sanz-Serna (1992), Symplectic integrators for Hamiltonian problems: an overview, *Acta Numerica* 1:243–286.

34. L.F. Shampine and M.W. Reichelt (1997), The MATLAB ODE suite. *SIAM J. Sci. Comput.* 18:1–22.
35. H.J. Stetter (1973), *Analysis of Discretization Methods for Ordinary Differential Equations*, Springer-Verlag, Berlin.
36. C. Störmer (1907), Sur les trajectoires des corpuscules électrisés. *Arch. sci. Phys. nat. Genève* 24:5–18, 113–158, 221–247.
37. Y.B. Suris (1988), On the conservation of the symplectic structure in the numerical solution of Hamiltonian systems (in Russian), In: *Numerical Solution of Ordinary Differential Equations*, ed. S.S. Filippov, Keldysh Institute of Applied Mathematics, USSR Academy of Sciences, Moscow, 1988, 148–160.
38. M. Suzuki (1990), Fractal decomposition of exponential operators with applications to many-body theories and Monte Carlo simulations, *Phys. Lett.* A 146:319–323.
39. Y.-F. Tang (1993), The symplecticity of multi-step methods, *Computers Math. Applic.* 25:83–90.
40. Y.-F. Tang (1994), Formal energy of a symplectic scheme for Hamiltonian systems and its applications (I), *Computers Math. Applic.* 27:31–39.
41. L. Verlet (1967), Computer "experiments" on classical fluids. I. Thermodynamical properties of Lennard-Jones molecules. *Phyiscal Review* 159:98–103.
42. R. de Vogelaere (1956), Methods of integration which preserve the contact transformation property of the Hamiltonian equations. Report Nr. 4, Dept. Math. Univ. of Notre Dame, Notre Dame, Ind.
43. G. Wanner (1973), Runge-Kutta-methods with expansion in even powers of h, *Computing* 11:81–85.
44. H. Yoshida (1990), Construction of higher order symplectic integrators, *Phys. Lett.* A 150:262–268.
45. H. Yoshida (1993), Recent progress in the theory and application of symplectic integrators, *Celestial Mech. Dynam. Astronom.* 56:27–43.