Preserving multiple first integrals by discrete gradients

Morten Dahlby Brynjulf Owren Takaharu Yaguchi

November 3, 2010

Abstract

We consider systems of ordinary differential equations with known first integrals. The notion of a discrete tangent space is introduced as the orthogonal complement of an arbitrary set of discrete gradients. Integrators which exactly conserve all the first integrals simultaneously are then defined. Two approaches are presented, one based on projection and one based on local coordinates, both allowing for integrators of arbitrary order of convergence. The methods are tested on the Kepler problem.

1 Introduction

A system of ordinary differential equations which preserves a first integral $H(y)$ can be written in the form

$$\dot{y} = f(y) = S(y)\nabla H(y),\ y \in \mathbb{R}^m, \tag{1.1}$$

where $S(y)$ is an antisymmetric matrix. An approximate numerical solution, $y^n \approx y(t^n),\ n \geq 1$, is said to be integral preserving if $H(y^n) = H(y^0),\ n \geq 1$. There are several ways to obtain integral preserving numerical methods, and one of the most prevalent approaches in the literature is that of discrete gradients, first systematically treated by Gonzalez [2] and McLachlan et al. [5]. The idea is to introduce a discrete approximation to the gradient, letting $\overline{\nabla} H : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}^m$ be a continuous map satisfying

$$\begin{aligned} H(u) - H(v) &= \overline{\nabla} H(v,u)^\top (u - v), \\ \overline{\nabla} H(u,u) &= \nabla H(u). \end{aligned}$$

The existence of such discrete gradients is well established in the literature, see for instance the monograph by Hairer et al. [3]. Their construction is not unique, we give here two different examples. The Averaged Vector Field (AVF) gradient is defined as

$$\overline{\nabla}_{\mathrm{AVF}} H(v,u) = \int_0^1 \nabla H(\xi u + (1-\xi)v)\, \mathrm{d}\xi. \tag{1.2}$$

The coordinate increment method [4] is defined in terms of the coordinates of the vectors v and u, the ith component of $\overline{\nabla} H(v,u)$ is then given as

$$(\overline{\nabla}_{\mathrm{CI}} H(v,u))_i = \frac{H(u_1,\ldots,u_i,v_{i+1},\ldots,v_m) - H(u_1,\ldots,u_{i-1},v_i,\ldots,v_m)}{u_i - v_i}. \quad (1.3)$$

An important difference between these two discrete gradients is that (1.2) is symmetric $\overline{\nabla}_{\mathrm{AVF}} H(v,u) = \overline{\nabla}_{\mathrm{AVF}} H(u,v)$, while (1.3) is not. However, note that a symmetric version of the coordinate increment discrete gradient can be constructed by

$$\overline{\nabla}_{\mathrm{SCI}} H(v,u) = \frac{1}{2}\left(\overline{\nabla}_{\mathrm{CI}} H(v,u) + \overline{\nabla}_{\mathrm{CI}} H(u,v)\right). \quad (1.4)$$

Once a discrete gradient has been found, one immediately obtains an integral preserving method by simply letting

$$\frac{y^{n+1}-y^n}{h} = \overline{S}(y^n, y^{n+1})\,\overline{\nabla} H(y^n, y^{n+1}).$$

Here h is the time step, and $\overline{S}(y^n, y^{n+1})$ is some approximation to the matrix S in, one would normally require that $S(y) = \overline{S}(y,y)$. We remark that discrete gradient methods are implicit.

In this note we consider the case where there are more than one first integral and the objective is to preserve any number of such invariants simultaneously. Some earlier attempts to achieve this include the papers [5, 7] in which the antisymmetric matrix $S(y)$ is replaced by an antisymmetric tensor taking discrete gradients of all integrals to be preserved as input. A formula for this antisymmetric tensor is given. Another approach is an integrator for a class of separable Hamiltonian systems ([6] and references therein), where the integrator which preserves all integrals is designed based on separation of variables. We shall instead present an approach which does not rely on finding such a tensor nor a structure of the equation, the method only assumes knowledge of the first integrals to be preserved as well as the ODE vector field itself. The algorithm we develop makes use of discrete gradients for each of the first integrals. We prove that the methods have pth order of convergence, regardless of the underlying choice of discrete gradient.

In Section 3 we solve the Kepler problem using the methods developed in this paper. We show that preserving multiple invariants gives a qualitatively better solution than just preserving one.

2 Preserving multiple invariants

Suppose that an ODE system (1.1) possesses $q \geq 1$ independent first integrals, $H_1(y),\ldots,H_q(y)$. These invariants foliate $\mathbb{R}^m$ into $(m-q)$-dimensional submanifolds (leaves)

$$M = M_c = \{y \in \mathbb{R}^m : H_1(y) = c_1, H_2(y) = c_2, \ldots, H_q(y) = c_q\}.$$

The tangent space $T_y M$ of M at y is the orthogonal complement to

$$\mathrm{span}\{\nabla H_1(y),\ldots,\nabla H_q(y)\}.$$

For simplicity we write only M for M_c for the rest of this paper.

Definition 2.1. *Let $\overline{\nabla}$ be a fixed discrete gradient operator and let $H_1, \ldots, H_q$ be independent first integrals. The discrete tangent space at $(u, v) \in \mathbb{R}^m \times \mathbb{R}^m$ is*

$$T_{(v,u)}M = \{\eta \in \mathbb{R}^m : \langle \overline{\nabla} H_j(v, u), \eta \rangle = 0,\ 1 \leq j \leq q\}.$$

A vector $\eta = \eta_{(v,u)} \in T_{(v,u)}M$ is called a discrete tangent vector.

Note that this definition causes $T_{(y,y)}M = T_yM$.

Lemma 2.2. *Any integrator satisfying*

$$y^{n+1} - y^n = \eta_{(y^n, y^{n+1})} \in T_{(y^n, y^{n+1})}M$$

preserves all integrals, in the sense that $H_i(y^{n+1}) = H_i(y^n),\ 1 \leq i \leq q$.

Proof. For any i we compute

$$\begin{aligned} H_i(y^{n+1}) - H_i(y^n) &= \overline{\nabla} H_i(y^n, y^{n+1})^\top (y^{n+1} - y^n) \\ &= \overline{\nabla} H_i(y^n, y^{n+1})^\top \eta_{(y^n, y^{n+1})} \\ &= 0. \end{aligned}$$

□

In practice there are at least two ways of ensuring that the condition of Lemma 2.2 is satisfied. One is to use projection another one is local coordinates.

2.1 Projection

We consider (1.1) with the first integrals $H_1(y), \ldots, H_q(y)$. We propose the projection scheme

$$u^{n+1} = \phi_h(y^n), \quad y^{n+1} = y^n + \mathcal{P}(y^n, y^{n+1})(u^{n+1} - y^n) \tag{2.1}$$

where ϕ_h is an arbitrary integrator of order p

$$y(t + h) - u^{n+1} = y(t + h) - \phi_h(y(t)) = \mathcal{O}(h^{p+1}),$$

and $\mathcal{P}(y^n, y^{n+1})$ is a smooth projection operator onto the discrete tangent space $T_{(y^n, y^{n+1})}M$. An alternative method is

$$y^{n+1} = y^n + h\mathcal{P}(y^n, y^{n+1})\psi_h(y^n, y^{n+1}) \tag{2.2}$$

where ψ_h is an integrator which can be written on the form

$$y^{n+1} = y^n + h\psi_h(y^n, y^{n+1}).$$

This method is itself assumed to be of order p, that is

$$y(t + h) - y(t) - h\psi_h(y(t), y(t + h)) = \mathcal{O}(h^{p+1}). \tag{2.3}$$

Example 2.3. *Using Runge-Kutta as the underlying scheme ϕ_h we can construct examples of* (2.1)*. The unprojected solution is given as*

$$u^{n+1} = y^n + h\sum_{i=1}^{s} b_i k_i,$$

where $k_1, \ldots, k_s$ are the solutions to the (possibly implicit) equations

$$k_i = f\left(y^n + h\sum_{j=1}^{s} a_{ij}k_j\right).$$

Even if the Runge-Kutta scheme is explicit the scheme (2.1) *will be implicit since y^{n+1} appears in the projection operator*

$$y^{n+1} = y^n + h\mathcal{P}(y^n, y^{n+1})\left(\sum_{i=1}^{s} b_i k_i\right).$$

The difference between (2.1) *and* (2.2) *is subtle, but to illustrate that they are in fact distinct we consider the implicit midpoint method as the underlying scheme and we get for the two methods*

$$y^{n+1} = y^n + h\mathcal{P}(y^n, y^{n+1})f\left(\frac{y^n + u^{n+1}}{2}\right),$$
$$y^{n+1} = y^n + h\mathcal{P}(y^n, y^{n+1})f\left(\frac{y^n + y^{n+1}}{2}\right),$$

where in the former method u^{n+1} is computed by solving

$$u^{n+1} = y^n + hf\left(\frac{y^n + u^{n+1}}{2}\right).$$

Theorem 2.4. *The schemes* (2.1) *and* (2.2) *are of order p, that is*

$$\begin{aligned} y(t+h) - y(t) - \mathcal{P}(y(t), y(t+h))(u^{n+1} - y(t)) &= \mathcal{O}(h^{p+1}), \\ u^{n+1} &= \phi_h(y(t)), \end{aligned} \tag{2.4}$$

and

$$y(t+h) - y(t) - h\mathcal{P}(y(t), y(t+h))\psi_h(y(t), y(t+h)) = \mathcal{O}(h^{p+1}). \tag{2.5}$$

Proof. We use the shorthand notation $\mathcal{P}$ for $\mathcal{P}(y(t), y(t+h))$ in this proof. To prove (2.4), we compute

$$\begin{aligned} &y(t+h) - y(t) - \mathcal{P}(u^{n+1} - y(t)) \\ &= y(t+h) - y(t) - (I - (I - \mathcal{P}))(u^{n+1} - y(t)) \\ &= y(t+h) - y(t) - (u^{n+1} - y(t)) + (I - \mathcal{P})(u^{n+1} - y(t)). \end{aligned}$$

Since ϕ_h is of order p, we have

$$y(t+h) - y(t) - (u^{n+1} - y(t)) = y(t+h) - u^{n+1} = \mathcal{O}(h^{p+1}). \tag{2.6}$$

Therefore if we have

$$(I - P)(u^{n+1} - y(t)) = \mathcal{O}(h^{p+1}),$$

the proof is completed. This estimate is obtained in the following way. Because the image of $I - \mathcal{P}(y(t), y(t+h))$ is spanned by

$$\left\{\overline{\nabla} H_1(y(t), y(t+h)), \ldots, \overline{\nabla} H_q(y(t), y(t+h))\right\},$$

it is enough to show

$$\overline{\nabla} H_i(y(t), y(t+h)) \cdot (u^{n+1} - y(t)) = \mathcal{O}(h^{p+1}).$$

From (2.6) we obtain

$$u^{n+1} - y(t) = y(t+h) - y(t) + \mathcal{O}(h^{p+1}),$$

and hence

$$\begin{aligned}
&\overline{\nabla} H_i(y(t), y(t+h)) \cdot (u^{n+1} - y(t)) \\
&= \overline{\nabla} H_i(y(t), y(t+h)) \cdot (y(t+h) - y(t) + \mathcal{O}(h^{p+1})) \\
&= H_i(y(t+h)) - H_i(y(t)) + \mathcal{O}(h^{p+1}) \\
&= \mathcal{O}(h^{p+1}).
\end{aligned}$$

The last equality is from the conservation property of the original equation.

The proof of (2.5) is almost identical and therefore omitted. □

Computing the projector. A simple and straightforward way of obtaining the projector $\mathcal{P}(y^n, y^{n+1})$ is as follows: Define the $(q \times m)$-matrix $Y = Y(y^n, y^{n+1})$ whose columns are the discrete gradients $\overline{\nabla} H_i(y^n, y^{n+1})$ for $i = 1, \ldots, q$. Compute a reduced QR-decomposition $QR = Y$ where $Q \in \mathbb{R}^{m \times q}$ and $R \in \mathbb{R}^{q \times q}$. Then define the projection matrix as $\mathcal{P}(y^n, y^{n+1}) = I - QQ^\top$.

2.2 Local coordinates

Inspired by [1], we consider local coordinates on a chart containing y^0 by defining a map $\eta \mapsto y = \chi(\eta)$. The map is defined implicitly by

$$\chi(\eta) = y : \ y - y^0 = T(y)\eta, \tag{2.7}$$

where $T(y)$ is a smooth $m \times (m - q)$-matrix whose columns form a basis for the left nullspace (orthogonal column complement) of the matrix $Y(y) = [\overline{\nabla} H_1(y^0, y), \ldots, \overline{\nabla} H_q(y^0, y)]$.

Lemma 2.5. *Suppose that $\nabla H_1(y^0), \ldots, \nabla H_q(y^0)$ are linearly independent for all $y^0 \in M$. Suppose also that for all $y^0 \in M$, $\overline{\nabla} H_1(y^0, y), \ldots, \overline{\nabla} H_q(y^0, y)$ are C^∞ with respect to y. Then the following statements hold.*

1. *(2.7) defines a one-to-one map ω_{y^0} in a neighborhood $N_{y^0} \subset M$. ω_{y^0} and $\omega_{y^0}^{-1}$ are C^∞.*
2. *The collection of the pairs $\{(N_{y^0}, \omega_{y^0}) \mid y^0 \in M\}$ forms an atlas of M.*

Proof. 1. From the continuity of the discrete gradients, we deduce that for all $y^0 \in M$, there exists a neighborhood $\tilde{N}_{y^0}$ in $\mathbb{R}^m$ of y^0 in which $\overline{\nabla} H_1(y^0, y), \ldots, \overline{\nabla} H_q(y^0, y)$ are linearly independent. For $y \in \tilde{N}_{y^0}$, $T(y)$ admits the QR decomposition $T(y) = QR$ and η is obtained by $\eta = (R^\top R)^{-1} Q^\top (y - y^0) = (T^\top T)^{-1} T^\top (y - y^0)$. This is a C^∞ function. Conversely, the Jacobian matrix of the function $\eta(y)$ at $y = y^0$ is

$$\frac{\partial \eta}{\partial y}(y^0) = (T^\top T)^{-1} T^\top$$

and hence

$$\operatorname{rank} \frac{\partial \eta}{\partial y}(y^0) = \operatorname{rank} (T^\top T)^{-1} T^\top = m - q.$$

Thus ω is defined in a neighborhood $\bar{N}_{y^0}$ of y^0 by the implicit function theorem and is C^∞. The proof is completed by letting $N_{y^0} = \bar{N}_{y^0} \cap \tilde{N}_{y^0} \cap M$.

2. This is immediately obtained from the first statement.

□

Consider now the curve $\eta(t)$ and let $y(t) = \chi(\eta(t))$. We differentiate the curve to obtain from (2.7)

$$\dot{y}(t) = T'_{y(t)}(\dot{y}(t))\eta(t) + T(y(t))\dot{\eta}(t).$$

From this we compute

$$\dot{\eta} = -T^\top(\chi \circ \eta) T'_{\chi \circ \eta}(f(\chi \circ \eta))\eta + T^\top(\chi \circ \eta) f(\chi \circ \eta) \tag{2.8}$$

where the original ODE is $\dot{y} = f(y)$. The method we propose takes one step as follows

1. Let $\eta_0 = 0$.
2. Take a step with any pth order method applied the ODE (2.8) using $y^0 = y^n$ in (2.7). The result is η_1.
3. Compute $y^{n+1} = \chi(\eta_1)$.

We immediately obtain the next theorem from Lemma 2.5, because the solution curve lies in M and a pth order method is applied in a chart of M.

Theorem 2.6. *Under the assumptions of Lemma 2.5, the above scheme is of order p.*

The main difficulty in this approach is the computation of the derivative map $T'_y(\zeta)$ for arbitrary values of $y \in \mathbb{R}^m$ and $\zeta \in \mathbb{R}^m$. This is needed explicitly in the integration algorithm, but may also be a useful tool in computing the coordinate map (2.7). We may define $T(y)$ as the last $m - q$ columns of the $m \times m$-matrix $Q(y)$ defined through a QR-decomposition where $Y(y) = Q(y)R(y)$ and where we have used the shorthand notation

$$Y(y) = [\overline{\nabla} H_1(y^n, y), \ldots, \overline{\nabla} H_q(y^n, y)]$$

We realise the QR-decomposition by means of the Householder method, applying a sequence of q elementary orthogonal transformations to the matrix $Y(y)$ as described in most elementary text books in numerical linear algebra, see e.g [8]. Each transformation is of the form

$$Q_k = I - 2v_k v_k^\top, \qquad v_k \in \mathbb{R}^m,\ v_k^\top v_k = 1,$$

and its aim is to eliminate all elements under the diagonal in the kth column of the matrix to which it is applied.

In order to explain how we compute the derivative $Q'_y(\zeta) =: DQ(y,\zeta)$, we first review the Householder method.

$Y^{(1)} := Y$
for $k = 1 : q$,
  $w_k = \Pi_k Y_k^{(k)} - \|\Pi_k Y_k^{(k)}\| \mathrm{e}_k$
  $v_k = \frac{w_k}{\|w_k\|}$
  for $r = k : q$,
    $Y_r^{(k+1)} = (I - 2\, v_k v_k^\top) Y_r^{(k)}$
  end
end

where the following conventions have been used:

- $\|\cdot\|$ is the Euclidean norm.
- $Y^{(k+1)} = Q_k Y^{(k)}$, $Y_r^{(k)}$ is column r of $Y^{(k)}$.
- The projector Π_k puts zeros in the first $k-1$ components and leaves the rest of the components unchanged when applied to a vector in $\mathbb{R}^m$.
- e_k is the kth canonical unit vector in $\mathbb{R}^m$.

The vectors v_k computed in the algorithm contain all information needed to reconstruct the factor Q, whereas $R := Y^{(q+1)}$. For simplicity, and to avoid the loss of regularity in $Q(y)$ viewed as a matrix valued function of y, we have here ignored the sign convention which is usually applied in the definition of w_k [8]. The idea is now to differentiate the variables in the algorithm with respect to y, writing for any object, say $X(y)$, its derivative as $DX = DX(y,\zeta) = \frac{\mathrm{d}}{\mathrm{d}\varepsilon}\big|_{\varepsilon=0} X(y+\epsilon\zeta)$ for any $y, \zeta \in \mathbb{R}^m$. The dependence on y, ζ will usually be suppressed in the notation when no confusion is at risk. Notice that D commutes with Π_k for any k. The following recursion formulae are easily derived

$$\begin{aligned}
Dw_k &= \Pi_k DY_k^{(k)} - \frac{(\Pi_k Y_k^{(k)})^\top \Pi_k DY_k^{(k)}}{\|\Pi_k Y_k^{(k)}\|}\mathrm{e}_k,\\
Dv_k &= \left(Dw_k - \frac{w_k^\top Dw_k}{\|w_k\|^2}\, w_k\right)\|w_k\|^{-1},\\
DY_r^{(k+1)} &= DY_r^{(k)} - 2\left(v_k^\top Y_r^{(k)} Dv_k + Dv_k^\top Y_r^{(k)} v_k + v_k^\top DY_r^{(k)} v_k\right).
\end{aligned}$$

The initial $DY^{(1)} = DY^{(1)}(y,\zeta)$ should be computed by differentiating the q discrete gradients $\overline{\nabla} H_1(p,y), \ldots, \overline{\nabla} H_q(p,y)$ with respect to y. When the discrete

gradients are given by the AVF formula, we may derive the following expressions for the rth column of $DY^{(1)}$

$$DY_r^{(1)}(y,\zeta) = D\overline{\nabla} H_r(y,\zeta) = \left(\int_0^1 \xi \nabla^2 H_r(\xi y + (1-\xi)p) \mathrm{d}\xi \right) \cdot \zeta$$

where $\nabla^2 H_r$ is the Hessian of the integral H_r. The expression for the coordinate increment cases are given in the appendix.

In the present application we only make use of the Q-part of the QR-decomposition, thus we need only store $v_1, \ldots, v_q$ and $Dv_1, \ldots, Dv_q$ for subsequent use. We may summarize the extended algorithm for computing these quantities as follows, using a Matlab inspired indexing notation where the submatrix $Y_{a:b,c:d}$ of Y means

$$Y_{a:b,c:d} = \begin{pmatrix} Y_{a,c} & \cdots & Y_{a,d} \\ \vdots & \ddots & \vdots \\ Y_{b,c} & \cdots & Y_{b,d} \end{pmatrix}.$$

Given Y and DY as $m \times q$-matrices
for $k = 1 : q$
$w = Y_{k:m,k} - \|Y_{k:m,k}\| \mathrm{e}_1$
$Dw = DY_{k:m,k} - \dfrac{Y_{k:m,k}^\top DY_{k:m,k}}{\|Y_{k:m,k}\|} \mathrm{e}_1$
$\tilde{v}_k = w/\|w\|$
$D\tilde{v}_k = (Dw - \dfrac{w^\top Dw}{\|w\|^2} w)\|w\|^{-1}$
$DY_{k:m,k+1:q} = DY_{k:m,k+1:q} - 2\tilde{v}_k \tilde{v}_k^\top DY_{k:m,k+1:q} - 2D\tilde{v}_k \tilde{v}_k^\top Y_{k:m,k+1:q}$
$\qquad -2\tilde{v}_k\, D\tilde{v}_k^\top Y_{k:m,k+1:q}$
$Y_{k:m,k+1:q} = Y_{k:m,k+1:q} - 2\tilde{v}_k \tilde{v}_k^\top Y_{k:m,k+1:q}$
end

The first $k-1$ entries of the vectors v_k, Dv_k are zeros, and the remaining $m-k+1$ entries are contained in $\tilde{v}_k$, $D\tilde{v}_k$ on exit. The complexity of this algorithm is $\mathcal{O}(mq^2 + q^3)$.

One should also note that we always multiply Q (DQ resp) by vectors $\bar{\eta} \in \mathbb{R}^m$ whose first q columns are zero, this may be taken advantage of in the implementation. The procedure for computing $Q\bar{\eta}$ by means of $v_1, \ldots, v_k$ is described in [8, Algorithm 10.3], the cost is $\mathcal{O}(mq)$. We may extend this algorithm so that it computes also $DQ\,\bar{\omega}$ given $Dv_1, \ldots, Dv_q$. Defining the $m \times m$-matrix $P_k = Q_k \cdots Q_q$, $k = 1, \ldots, q$, we get the downwards recursion

$$P_{k-1} = Q_{k-1} P_k, \qquad P_q = Q_q, \quad P_1 = Q = Q_1 \cdots Q_q,$$

and differentiation yields

$$\begin{aligned} DP_{k-1} &= DQ_{k-1} P_k + Q_{k-1} DP_k \\ &= -2(Dv_{k-1}\, v_{k-1}^\top + v_{k-1}\, Dv_{k-1}^\top) P_k + (I - 2v_{k-1} v_{k-1}^\top) DP_k. \end{aligned}$$

The following algorithm results for computing $\phi = DQ\,\bar{\omega}$

$\psi = \bar{\omega},\ \phi = 0$
for $k = q : -1 : 1$
$\quad \phi = \phi - 2(v_k^\top \psi\, D v_k + D v_k^\top \psi\, v_k + v_k^\top \phi\, v_k)$
$\quad \psi = \psi - 2 v_k^\top \psi\, v_k$
end

The complexity of this algorithm is $\mathcal{O}(mq)$.

3 Numerical integration of the Kepler problem

The Kepler two-body problem describes the motion of two bodies which attract each other. By placing the first body in the origin, the position (y_1, y_2) and the velocity (y_3, y_4) of the other body are given by the following four-dimensional ODE

$$\begin{aligned} \dot{y}_1 &= y_3, \\ \dot{y}_2 &= y_4, \\ \dot{y}_3 &= -\frac{y_1}{(y_1^2 + y_2^2)^{3/2}}, \\ \dot{y}_4 &= -\frac{y_2}{(y_1^2 + y_2^2)^{3/2}}. \end{aligned} \tag{3.1}$$

This system preserves the Hamiltonian

$$H_1(y) = \frac{1}{2}\left(y_3^2 + y_4^2\right) - \frac{1}{\sqrt{y_1^2 + y_2^2}},$$

the angular momentum

$$H_2(y) = y_1 y_4 - y_2 y_3,$$

and the Runge-Lenz-Pauli vector

$$\begin{aligned} H_3(y) &= y_2 y_3^2 - y_1 y_3 y_4 - \frac{y_2}{\sqrt{y_1^2 + y_2^2}}, \\ H_4(y) &= y_1 y_4^2 - y_2 y_3 y_4 - \frac{y_1}{\sqrt{y_1^2 + y_2^2}}. \end{aligned}$$

Since $q = m = 4$ the invariants will be dependent. By constructing a scheme that preserves, say, the first three invariants one gets a scheme which exactly preserves all four. We want to compare schemes that preserve none, one, two, and all of the invariants above. We use the projection method (2.1) with the standard fourth order explicit Runge-Kutta method as the underlying scheme. The discrete gradients are calculated using (1.4). The schemes that preserve one of H_1, H_3 are denoted as RK4Proj1 and RK4Proj3, respectively. The scheme that preserves both H_1 and H_3 is called RK4Proj13. The original Runge-Kutta scheme preserves neither and is denoted RK4. The RK4Proj123 scheme is omitted from the plot since it produces exactly the ellipsis of the Kepler problem.

The resulting plots from these methods in Figure 3.1 and 3.2 are arranged according to the table

RK4	RK4Proj1
RK4Proj3	RK4Proj13

Table 3.1: The location of the schemes in the plots of Figure 3.1 and 3.2.

The initial values are taken from section I.2.3 of [3]

$$y_1^0 = 1 - e, \quad y_2^0 = 0, \quad y_3^0 = 0, \quad y_4^0 = \sqrt{\frac{1+e}{1-e}},$$

where the eccentricity is $e = 0.6$ and the exact solution has period 2π. The time step is $h = 0.2$ and we integrate for 50000 steps.

Figure 3.1 shows the numerical solutions and Figure 3.2 shows the preservation error. We see that all the schemes preserve the invariants they were constructed to preserve. Also, RK4Proj123 did indeed preserve all the four invariants, however this is not shown in the plots. RK4 spirals inwards until it eventually blows up. RK4Proj1 has a counterclockwise precession effect. The Runge-Lenz-Pauli vector has to do with the orientation of the ellipse and RK4Proj3 does therefore not exhibit this effect, it will however converge to a small circle around the origin. The solution of RK4Proj13 shows an improvement compared to RK4Proj1 and RK4Proj3.

This example illustrates that there are cases where the preservation of one or more invariants are important to get a numerical solution with good long term properties. Not surprisingly, one observes a gradual improvement in the quality of the solution as the number of preserved first integrals increases. The extra computational effort needed to preserve multiple integrals compared to one is almost negligible, in this example the computation took less than 10% longer.

In Figure 3.3 we plot the global error of the four schemes (RK2Proj123, RK4Proj123, RK5Proj123, and RK7Proj123) that preserve the four invariants. The underlying schemes are four RK-schemes of order 2, 4, 5, and 7. We see that the schemes attain the order of the underlying scheme, which is what we expected from Theorem 2.4.

Conclusion and further work. We have presented a new methodology for preserving multiple first integrals in systems of ordinary differential equations, using discrete gradients as the underlying tool. By using the notion of a discrete tangent space, two methodologies for designing numerical schemes are easily derived, projection and local coordinates. The resulting algorithms are relatively inexpensive compared to well-known algorithms preserving precisely one first integral. Although the present paper considers only systems of ordinary differential equations, the approach taken may be easily adapted to partial differential equations to be considered in future work.

References

[1] E. Celledoni and B. Owren. A class of intrinsic schemes for orthogonal integration. *SIAM J. Numer. Anal.*, 40(6):2069–2084 (electronic) (2003), 2002.

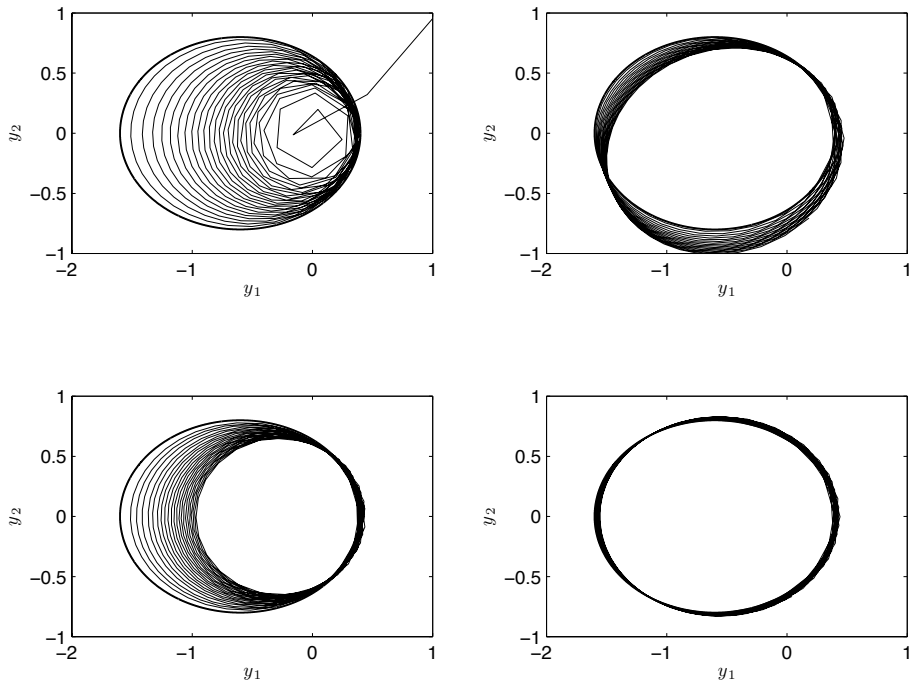

Figure 3.1: The numerical solution (thin line) of the Kepler problem (3.1) using the schemes of Table 3.1 with $h = 0.2$. The first 500 steps are shown. The exact solution (thick line) is an ellipse with eccentricity $e = 0.6$.

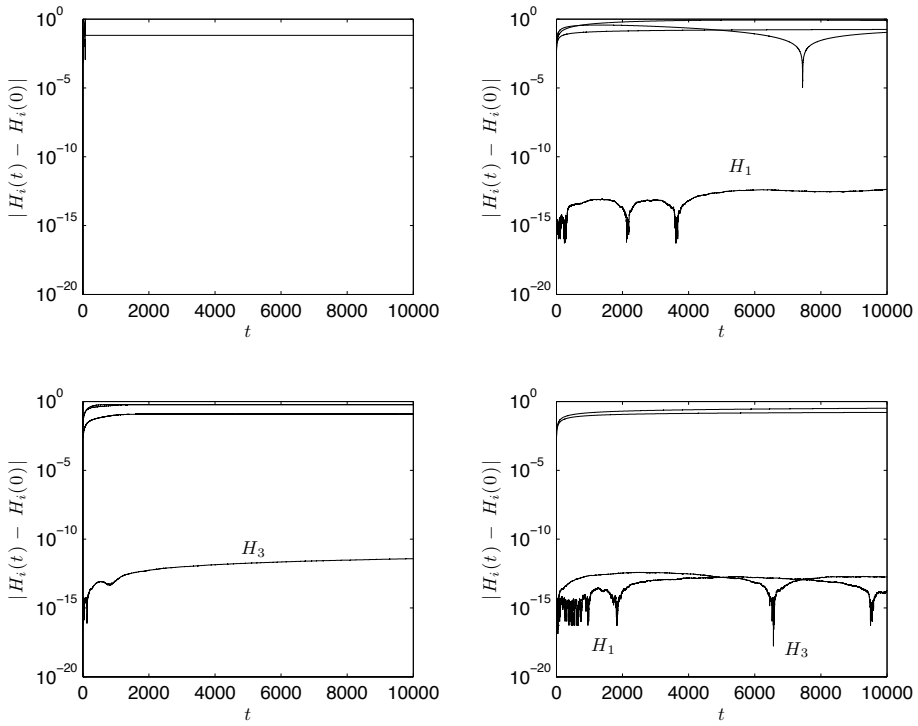

Figure 3.2: The error in the first integrals for the solutions of the four schemes of Table 3.1. There is one line for each of the four invariants. 50000 steps are shown.

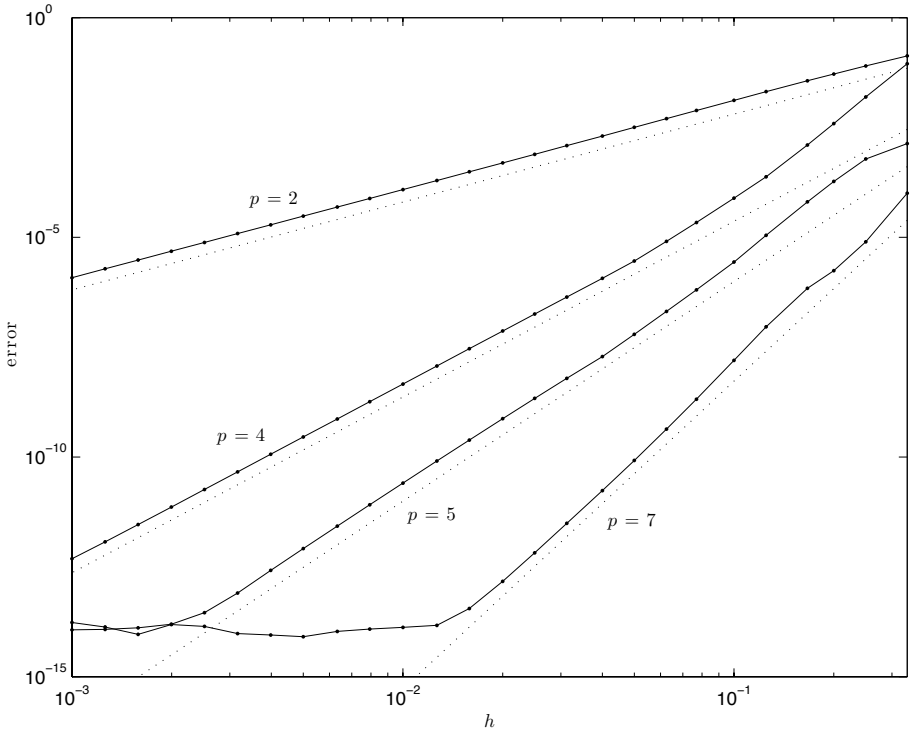

Figure 3.3: The global error of the four schemes RK2Proj123, RK4Proj123, RK5Proj123, and RK7Proj123. The dotted lines are reference lines of exact order.

[2] O. Gonzalez. Time integration and discrete Hamiltonian systems. *J. Nonlinear Sci.*, 6(5):449–467, 1996.

[3] E. Hairer, C. Lubich, and G. Wanner. *Geometric numerical integration*, volume 31 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, second edition, 2006. Structure-preserving algorithms for ordinary differential equations.

[4] T. Itoh and K. Abe. Hamiltonian-conserving discrete canonical equations based on variational difference quotients. *J. Comput. Phys.*, 76(1):85–102, 1988.

[5] R. I. McLachlan, G. R. W. Quispel, and N. Robidoux. Geometric integration using discrete gradients. *R. Soc. Lond. Philos. Trans. Ser. A Math. Phys. Eng. Sci.*, 357(1754):1021–1045, 1999.

[6] Y. Minesaki and Y. Nakamura. New numerical integrator for the Stäckel system conserving the same number of constants of motion as the degree of freedom. *J. Phys. A*, 39(30):9453–9476, 2006.

[7] G.R.W. Quispel and H. Capel. Solving ODE's numerically while preserving all first integrals, 1999. Preprint.

[8] L. N. Trefethen and D. Bau. *Numerical linear algebra*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.