

Symplectic All-at-Once Method for Hamiltonian Systems

Bei-Bei Zhu ^{1,*} and Yong-Liang Zhao ²¹ School of Mathematics and Physics, University of Science and Technology Beijing, Beijing 100083, China² School of Mathematical Sciences, Sichuan Normal University, Chengdu 610068, China; ylzhaofde@sina.com

* Correspondence: zhubeibei@lsec.cc.ac.cn

Abstract: The all-at-once technique has attracted many researchers' interest in recent years. In this paper, we combine this technique with a classical symplectic and symmetric method for solving Hamiltonian systems. The solutions at all time steps are obtained at one-shot. In order to reduce the computational cost of solving the all-at-once system, a fast algorithm is designed. Numerical experiments of Hamiltonian systems with degrees of freedom $n \leq 3$ are provided to show that our method is more efficient than the classical symplectic method.

Keywords: hamiltonian systems; symplectic method; all-at-once technique; fast algorithm; structure-preserving methods

1. Introduction

Hamiltonian mechanics, along with Newtonian mechanics and Lagrangian mechanics, are three classical mechanics. It is a reformulation of the classic mechanics that evolved from Lagrangian mechanics. The Hamiltonian system is developed based on Hamiltonian mechanics. The Hamiltonian system is an important category of dynamical systems, which can represent all real physical processes with negligible dissipation. It has a very wide range of applications, such as biology [1], plasma physics [2–5] and celestial mechanics [6,7]. The basis of Hamiltonian system is symplectic geometry which is the phase space composed of n generalized coordinates and n generalized momentum in a system with n degrees of freedom. The remarkable property of a Hamiltonian system is that it has intrinsic symplectic structure which is preserved by its phase flow.

It is usually difficult to obtain the phase flow of a Hamiltonian system, even though it preserves the certain symplectic structure. Therefore, the numerical method that may preserve the symplectic structure of the Hamiltonian system is an alternative choice. Thus, the symplectic method [8] is developed to numerically simulate the Hamiltonian system, aiming to preserve the symplectic structure. Its structure-preservation property and symplectic method have a great advantage in computational stability and long-term simulation over conventional numerical schemes. Such an advantage afford the symplectic algorithm tremendous success in diverse areas, e.g., mechanics, biology and physics [1,4,7,9–13]. Particularly, symplectic algorithms have been widely applied in plasma physics in recent years. The charged particle dynamics and guiding center dynamics in electro-magnetic fields can be expressed as a Hamiltonian system and a non-canonical Hamiltonian system, respectively. Explicit symplectic methods based on generating functions and coordinate transformations are developed for charged particle dynamics [5,13]. For guiding center dynamics of charged particles, variational symplectic methods [2,14] and canonicalized symplectic method [4] have been proposed.

Traditionally, the symplectic method is a time-stepping scheme, which obtains the solutions step-by-step. The symplectic method is usually implicit, thus an iteration method, such as fixed point iteration method or Newton iteration method is needed to solve the method. If a large number of time steps are involved, the total number of iterations can be very large, and the symplectic method may become time-consuming. Therefore,



Citation: Zhu, B.-B.; Zhao, Y.-L. Symplectic All-at-Once Method for Hamiltonian Systems. *Symmetry* **2021**, *13*, 1930. <https://doi.org/10.3390/sym13101930>

Academic Editors: Liang-Jian Deng, Xian-Ming Gu and Tai-Xiang Jiang

Received: 22 September 2021

Accepted: 8 October 2021

Published: 14 October 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

to overcome the time-consuming problem of the time-stepping method, we propose an alternative method. The basic idea of our method is combining the all-at-once technique with a symplectic method. For convenience, we call this novel method the symplectic all-at-once method in this paper. More precisely, we stack the solutions of the symplectic method at all time steps in a vector, then an all-at-once system or a block lower triangular system will be obtained. Such a lower triangular structure facilitates the use of the fast algorithm. We will show in the numerical experiments that our method equipped with a fast algorithm takes less computational cost than the time-stepping method.

More and more researchers have paid attention to the all-at-once technique in recent years as it can obtain the solution at all time steps. The all-at-once method has been widely used in time-dependent optimal control problems [15,16] and fractional partial differential equations (FPDEs) [17–22]. Stoll and Wathen [15] used the all-at-once approach to solve the discretized unsteady Stokes problems. This approach was shown to be efficient and mesh-independent. The all-at-once method was also applied for the optimal control of the unsteady Burgers equation [16]. Furthermore, the all-at-once technique equipped with fast algorithms was utilized when using finite difference methods to solve FPDEs [17–22]. The reason why many researchers use the all-at-once approach is to reduce the computational cost. Zhao et al. [20] employed the all-at-once approach for the time–space fractional diffusion equation. They developed a block bi-diagonal Toeplitz preconditioner for efficiently solving their resulting system. Gu and Wu proposed a parallel-in-time iterative algorithm for solving the all-at-once system arising from partial integro-differential problems [17]. The fast implementations for solving block lower triangular systems arising from fractional differential equations can be found in [18,19] and the references therein.

In this paper, we develop a symplectic all-at-once method for Hamiltonian systems. The all-at-once system is solved by the Newton method equipped with a fast algorithm which greatly saves computational time. Numerical experiments in three Hamiltonian problems are carried out to show that our method is more efficient than the classical symplectic method.

The rest of this paper is organized as follows. In Section 2, we first give a review of symplectic methods for Hamiltonian systems. Then, we derive the all-at-once system based on the midpoint rule. Accordingly, we provide a framework of our method for solving Hamiltonian systems. In Section 3, several numerical examples are provided to show the performance of our method. Moreover, we also compare our method with the classical midpoint rule and the third order non-symplectic Runge–Kutta method. In this paper, we only conduct experiments on Hamiltonian systems with degrees of freedom $n \leq 3$. In Section 4, we summarize our work and make enlightening remarks.

2. All-at-Once System for Hamiltonian Systems

Firstly, we give a short introduction to Hamiltonian systems and symplectic methods.

2.1. Symplectic Methods for Hamiltonian Systems

It is well known that the Hamiltonian system [1,7,11] is an important category in ordinary differential equations. Hamiltonian systems exist in diverse areas of physics, mechanics and engineering. It is generally accepted that all the physical process with negligible dissipation may be expressed as suitable Hamiltonian systems. So, reliable numerical methods that are developed for the numerical solution of Hamiltonian system would have wide applications. A prominent property of Hamiltonian systems is the symplecticity of the phase flow. Therefore, symplectic method is developed to preserve the symplectic structure of the Hamiltonian system.

In this paper, we consider the autonomous Hamiltonian system

$$\frac{dp}{dt} = -\frac{\partial H(p, q)}{\partial q}, \quad \frac{dq}{dt} = \frac{\partial H(p, q)}{\partial p}, \quad (1)$$

where $H(p, q)$ is the Hamiltonian function and $p = (p_1, \dots, p_n)^\top$ and $q = (q_1, \dots, q_n)^\top$ represent the momentum and configuration variables, respectively. If we combine all the variables in (1) in a $2n$ -dimensional vector $y = (p^\top, q^\top)^\top$, then (1) takes the compact form

$$\dot{y} = J^{-1} \nabla H(y), \tag{2}$$

where J is a $2n \times 2n$ skew-symmetric matrix

$$J_{2n \times 2n} = \begin{pmatrix} 0 & I_n \\ -I_n & 0 \end{pmatrix}$$

with I_n and 0 representing the $n \times n$ identity and zero matrix, respectively. By differentiating H with respect to t along the solution of system, we find that

$$\frac{d}{dt} H(y(t)) = (\nabla H)^\top \dot{y}(t) = (\nabla H)^\top J^{-1} (\nabla H) = 0.$$

Then, Hamiltonian function H is a conserved quantity which remains constant along the solution of the system. It is also called the energy of the system.

It is worth mentioning that in this paper, we only consider the autonomous Hamiltonian system (1). For non-autonomous Hamiltonian systems, we can transform them to autonomous ones [10]. More precisely, for the following non-autonomous Hamiltonian system

$$\frac{dp}{dt} = -\frac{\partial H(p, q, t)}{\partial q}, \quad \frac{dq}{dt} = \frac{\partial H(p, q, t)}{\partial p}, \tag{3}$$

with momentum variables $p = (p_1, p_2, \dots, p_n)^\top$ and conjugate variables $q = (q_1, q_2, \dots, q_n)^\top$, it can be transformed to an autonomous Hamiltonian system by adding a conjugate pair $p_0 := h = -H, q_0 := t$. This approach is to consider the time t as an additional conjugate variable. The new momentum h , to interpret it physically, is the negative of the total energy. Then the corresponding phase space has $2n + 2$ dimension, $y_1 = (p_0, p_1, \dots, p_n, q_0, q_1, \dots, q_n)$, and the new Hamiltonian function $\tilde{H}(p_0, p, q_0, q) = H(p, q, t) + h = H(p, q, q_0) + p_0$. Accordingly, the system becomes

$$\begin{aligned} \frac{dp_0}{dt} &= -\frac{\partial H(p, q, t)}{\partial t} = -\frac{\partial \tilde{H}(p_0, p, q_0, q)}{\partial q_0}, \\ \frac{dp}{dt} &= -\frac{\partial H(p, q, t)}{\partial q} = -\frac{\partial \tilde{H}(p_0, p, q_0, q)}{\partial q}, \\ \frac{dq_0}{dt} &= 1 = \frac{\partial \tilde{H}(p_0, p, q_0, q)}{\partial p_0}, \\ \frac{dq}{dt} &= \frac{\partial H(p, q, t)}{\partial p} = \frac{\partial \tilde{H}(p_0, p, q_0, q)}{\partial p}. \end{aligned}$$

It is an autonomous Hamiltonian system

$$\dot{y}_1 = J^{-1} \nabla \tilde{H}(y_1), \quad y_1 = (p_0, p^\top, q_0, q^\top)^\top$$

with Hamiltonian function $\tilde{H}(y_1) = H(p, q_0, q) + p_0$.

As is mentioned above, the symplecticity of the phase flow is a characteristic of Hamiltonian systems. That means that the phase flow $\varphi_t(y)$ of the Hamiltonian system (2) preserves symplectic structure, i.e., $\varphi_t(y)$ satisfies

$$\left(\frac{\partial \varphi_t(y)}{\partial y} \right)^\top J \left(\frac{\partial \varphi_t(y)}{\partial y} \right) = J, \quad \forall t \in \mathbb{R}.$$

In most occasions, it is difficult to obtain the exact solution of (2) when the system is highly nonlinear. So, we resort to using numerical methods to solve (2), such as the

symplectic geometric methods [1,7,8,11].

Definition 1. A numerical method

$$\Phi_\tau : y \in \mathbb{R}^{2n} \rightarrow \tilde{y} \in \mathbb{R}^{2n}, \quad \tau > 0,$$

is called symplectic if it satisfies

$$\left(\frac{\partial \tilde{y}}{\partial y}\right)^\top J \left(\frac{\partial \tilde{y}}{\partial y}\right) = J,$$

for a given time stepsize τ .

Notice that the symplectic method still preserves the symplectic structure, just as the phase flow of Hamiltonian systems does. Furthermore, the symplectic method possesses another important property: long-term energy conservation.

Among all kinds of symplectic methods, symplectic Runge–Kutta method [1,11] is a commonly used method because of its stability. An s –stage Runge–Kutta method is given by

$$\begin{aligned} y_{k+1} &= y_k + \tau \sum_{i=1}^s b_i f(K_i), \\ K_i &= y_k + \tau \sum_{j=1}^s a_{ij} f(K_j), \quad i = 1, 2, \dots, s, \end{aligned} \quad (4)$$

where b_i and a_{ij} are real numbers, τ is the time stepsize and y_{k+1} is the forward value of the variable y based on the known value y_k .

Theorem 1. A Runge–Kutta method is symplectic if the coefficients satisfy [23,24]

$$b_i b_j = b_i a_{ij} + b_j a_{ji} \quad i, j = 1, \dots, s.$$

Generally, an explicit Runge–Kutta method does not satisfy the above symplectic condition. Thus, a symplectic Runge–Kutta method is usually implicit. Midpoint rule [1,7,11] is a second order one-stage symplectic Runge–Kutta method. Formally, we take the midpoint rule as

$$\Phi_\tau : y_k \rightarrow y_{k+1} = y_k + \tau J^{-1} \nabla H \left(\frac{y_{k+1} + y_k}{2} \right).$$

Definition 2. A numerical method

$$\Phi_\tau : y \in \mathbb{R}^{2n} \rightarrow \tilde{y} \in \mathbb{R}^{2n}, \quad \tau > 0,$$

is said to be symmetric if it satisfies

$$\Phi_\tau \circ \Phi_{-\tau} = id.$$

The above condition has an equivalent expression $\Phi_\tau = \Phi_{-\tau}^{-1}$ [1].

To obtain the adjoint method $\Phi_{-\tau}^{-1}$ from Φ_τ , we just need to exchange $y_k \leftrightarrow y_{k+1}$ and $\tau \leftrightarrow -\tau$. By comparing the midpoint rule with its adjoint method, we find that they are exactly the same. Thus, the midpoint rule is a symmetric method. The midpoint rule is widely used for simulating Hamiltonian systems because of its symmetry and stability [4,25,26]. In our work, the all-at-once technique based on the midpoint rule is derived for solving (2).

2.2. All-at-Once System

We assume that the ordinary differential Equation (ODE) is of particular form

$$\dot{y} = f(y), \quad y \in \mathbb{R}^d. \quad (5)$$

The case that f includes a slow explicit dependence on t , i.e., $f = f(y, t)$, may be trivially reduced to (5) by adding a component y_{d+1} to the state vector y and setting $\dot{y}_{d+1} = 1$. If the ODE system is a Hamiltonian system, then the case $f(y, t)$ is a non-autonomous Hamiltonian system. According to the preceding section, by adding a conjugate pair, the non-autonomous system can be transformed to an autonomous one. Thus, the case (5) we considered is of generality. To simulate the solutions of general ODEs, the traditional way is to proceed the solution step-by-step, i.e., the so called time-stepping method. The all-at-once method stacks the solutions at all time steps in a vector, which proceeds all the solutions simultaneously by solving the resulted all-at-once system.

By applying the all-at-once technique to the midpoint rule, a bi-diagonal block structure will be obtained, which greatly reduces the storage requirement. Precisely, let us take N time steps with $N\tau = T$, then the time discretization of the ODE system is

$$y_{k+1} = y_k + \tau f_{k+1}, \quad k = 0, \dots, N-1,$$

where $f_{k+1} = f((y_{k+1} + y_k)/2)$. Then, we can construct the following nonlinear system with the solutions at all time steps:

$$\underbrace{\begin{pmatrix} A_0 & & & & \\ A_1 & A_0 & & & \\ & & \ddots & \ddots & \\ & & & A_1 & A_0 \end{pmatrix}}_A \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}}_x = \underbrace{\begin{pmatrix} y_0 + \tau f_1 \\ \tau f_2 \\ \vdots \\ \tau f_N \end{pmatrix}}_b, \quad (6)$$

where $A_0 = I_d$ and $A_1 = -I_d$. This bi-diagonal block structure makes the fast computation possible. Aiming to solve this nonlinear system, the Newton iteration method is chosen in this article. It is well known that the Newton method has the property of second-order convergence, thus less iteration is required to obtain the satisfactory solution than other iteration methods. Therefore, the Newton method is suitable to solve the all-at-once system. In order to proceed the Newton method, the initial value and the Jacobian matrix in each iterative step are required. Therefore, a well-chosen initial value and an efficient method to solve the Jacobian equations are two essential ingredients to solve (6).

Here is the strategy: We solve (6) on a coarse mesh based on using higher order symplectic methods. Then, we use the interpolate method to obtain the initial value on a fine mesh. More precisely,

Step 1: Let τ_c ($\tau_c > \tau$) be the large time stepsize and the time steps $N_c = T/\tau_c$. Using the fourth order Gauss' method [1,11] to integrate the Hamiltonian system (2) aiming at calculating the values y_1, \dots, y_{N_c} .

Step 2: Using the interpolation method, such as the piecewise cubic Hermite interpolating polynomial, to obtain the initial value $y_1^0, y_2^0, \dots, y_N^0$.

The expression of fourth order Gauss' method is

$$\begin{aligned} y_{k+1} &= y_k + \frac{1}{2}\tau f(K_1) + \frac{1}{2}\tau f(K_2), \\ K_1 &= y_k + \frac{1}{4}\tau f(K_1) + \left(\frac{1}{4} - \frac{\sqrt{3}}{6}\right)\tau f(K_2), \\ K_2 &= y_k + \left(\frac{1}{4} + \frac{\sqrt{3}}{6}\right)\tau f(K_1) + \frac{1}{4}\tau f(K_2). \end{aligned}$$

Algorithm 1 Compute $z = J_F^{-1}v$

- 1: Given a dN dimensional vector v , reshape v into $d \times N$ Matrix V
 - 2: Compute $\tilde{b}_1 = A_{11}^{-1}V(:, 1)$
 - 3: **for** $k = 2, \dots, N$ **do**
 - 4: $\psi = V(:, k) - A_{k,k-1}\tilde{b}_{k-1}$
 - 5: $\tilde{b}_k = A_{kk}^{-1}\psi$
 - 6: **end for**
 - 7: Stack \tilde{b}_k ($k = 1, \dots, N$) in the vector z
-

Remark 1. If the integration time is very long, we need to make a further partition of the time interval into several subintervals for the all-at-once system.

3. Numerical Experiments

All experiments were performed in MATLAB R2016b on a Windows 10 (64 bit) PC with the configuration: Intel(R) Core(TM) i7-8550U CPU 1.80 GHz and 16 GB RAM. The methods we used to perform numerical simulation are listed as follows.

AAO: symplectic all-at-once method which is a combination of all-at-once technique with the midpoint rule. Newton iteration method equipped with Thomas algorithm is utilized to solve the obtained all-at-once system. The midpoint rule is a second order symmetric and symplectic method.

TS: time-stepping method which represents the traditional way to deal with the midpoint rule. The solutions are obtained step-by-step.

RK3: the third-order Runge–Kutta method [4] which is neither symplectic nor symmetric. The method can be expressed as

$$\begin{aligned} y_{k+1} &= y_k + \frac{1}{2}\tau f(K_1) + \frac{1}{2}\tau f(K_2), \\ K_1 &= y_k + \frac{1}{2}\tau f(K_1) + \frac{\sqrt{3}}{6}\tau f(K_2), \\ K_2 &= y_k - \frac{\sqrt{3}}{6}\tau f(K_1) + \frac{1}{2}\tau f(K_2), \end{aligned} \quad (7)$$

3.1. Example 1: Harmonic Oscillator

First, we consider the well-know Hamiltonian system with one degree of freedom $n = 1$

$$H = T + V, \quad T = \frac{p^2}{2m}, \quad V = \frac{kq^2}{2}.$$

Here, q represents the configuration variable and p is the momentum. This system describes the motion of a particle of mass m attached to a spring with stiffness constant k . Here, T and V are kinetic energy and potential energy, respectively, [11].

As the harmonic oscillator is a Hamiltonian system, the equations can be written as

$$\frac{dp}{dt} = -kq, \quad \frac{dq}{dt} = \frac{p}{m}.$$

When we plot the phase plane (p, q) , the curves $(p(t), q(t))$ are ellipses

$$\frac{p^2}{2m} + \frac{kq^2}{2} = \text{Const.}$$

If $km = 1$, the curves are circles.

The simple harmonic oscillators constitute a Hamiltonian system with the conjugate variables (p, q) . We consider the case for $k = 1$ and $m = 1$, then the Hamiltonian system reads

$$\begin{pmatrix} \dot{p} \\ \dot{q} \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \nabla H(p, q) \quad (8)$$

with $H = \frac{p^2}{2} + \frac{q^2}{2}$ [11]. The exact solution is

$$\begin{pmatrix} p(t) \\ q(t) \end{pmatrix} = (\cos t I_2 - \sin t J_2) \begin{pmatrix} p_0 \\ q_0 \end{pmatrix}, \quad J_2 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

We first integrated the problem (8) with the symplectic all-at-once method under the initial condition $p_0 = 0.1, q_0 = 0.1$. The Newton iteration method for solving (6) terminates if the tolerance error is less than 10^{-12} or the iteration number is more than 2000. The coarse time stepsize for the symplectic all-at-once method is $\tau_c = 1$. Then we integrated the problem (8) with the time-stepping method and third order Runge–Kutta method.

Firstly we compare the symplectic all-at-once method with the third order Runge–Kutta method. We have displayed in Figure 1 the phase orbits and the energy errors of both methods over the interval $[0, 20,000]$. The phase orbit of the symplectic all-at-once method is accurate over a long time, but the phase orbit of the third order Runge–Kutta method spirals inwards over time. The energy error of the symplectic all-at-once method is preserved very well, but that of the the third order Runge–Kutta method increases over time. Therefore, the symplectic all-at-once method has significant advantage in preserving the phase orbit and the energy of the system over the third order Runge–Kutta method. The reason why the symplectic all-at-once method behaves better than the third order Runge–Kutta method is that the midpoint rule is symmetric and symplectic.

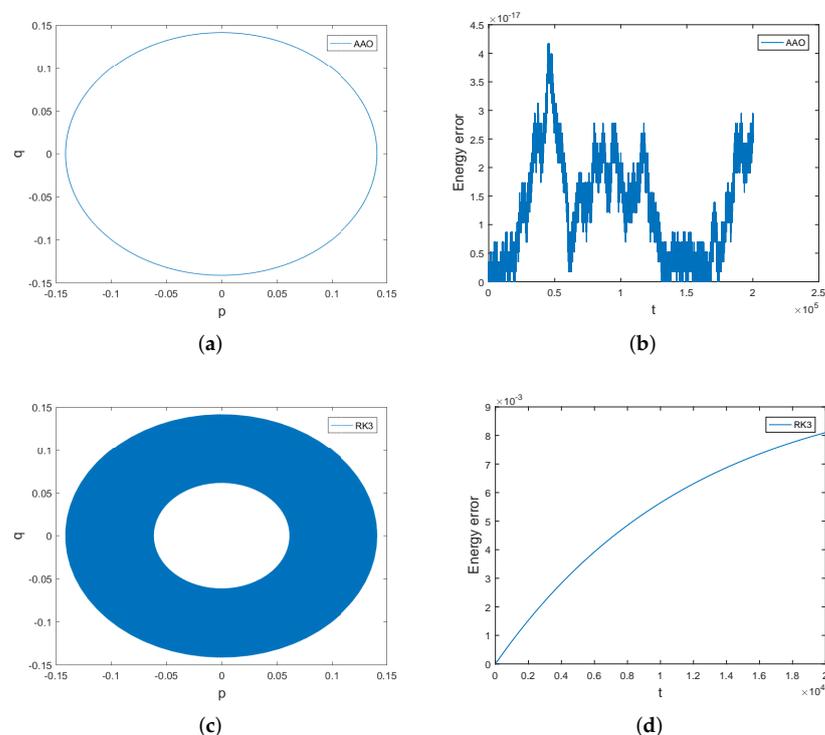


Figure 1. The phase orbits and the energy errors of the symplectic all-at-once method and third order Runge–Kutta method over the interval $[0, 20,000]$. The stepsize $\tau = 0.1$. Subfigure (a,b) are the phase orbit and energy error obtained by symplectic all-at-once method. Subfigure (c,d) are the phase orbit and energy error of third order Runge–Kutta method.

Then, we compare the computational efficiency of the symplectic all-at-once method and the time-stepping method. Table 1 reports, for varying time stepsize τ , the CPU times of the symplectic all-at-once method and the time-stepping method, over the interval $[0, 2000]$. By looking at the columns of Table 1, we find that solving the all-at-once system is a bit faster than solving the system (8) using the time-stepping method. When the stepsize is $0.1/64$ or $0.1/128$, the CPU time of the time-stepping method is nearly twice of that of the symplectic all-at-once method. The global errors of p and q of the symplectic all-at-once method are displayed in Table 1. Figure 2 indicates that the global error behaves as τ^2 which clearly shows the order of the midpoint rule. We have displayed in Table 2 the global errors of p and q of the time-stepping method and they are the same as those of the symplectic all-at-once method. The discrepancy between the solutions computed by both methods is also presented in Table 2. The maximum errors under different stepsizes are less than the value 1.7×10^{-14} .

Table 1. The CPU times of the symplectic all-at-once method (AAO) and the time-stepping method (TS) and global error of the symplectic all-at-once method over the time interval $[0, 2000]$.

τ	CPU (AAO)	CPU (TS)	Global Error of p (AAO)	Global Error of q (AAO)
0.1	0.097	0.117	2.090×10^{-1}	2.090×10^{-1}
0.1/2	0.164	0.230	5.845×10^{-2}	5.839×10^{-2}
0.1/4	0.309	0.456	1.472×10^{-2}	1.470×10^{-2}
0.1/8	0.573	0.909	3.681×10^{-3}	3.678×10^{-3}
0.1/16	1.119	1.843	9.202×10^{-4}	9.195×10^{-4}
0.1/32	2.189	3.638	2.300×10^{-4}	2.299×10^{-4}
0.1/64	4.364	7.565	5.751×10^{-5}	5.747×10^{-5}
0.1/128	8.874	15.381	1.434×10^{-5}	1.437×10^{-5}

Table 2. The global errors of p and q of the time-stepping method (TS) and the maximum errors of p and q between the symplectic all-at-once method and the time-stepping method over the time interval $[0, 2000]$.

τ	Global Error of p (TS)	Global Error of q (TS)	Maximum Error of p	Maximum Error of q
0.1	2.090×10^{-1}	2.090×10^{-1}	7.980×10^{-16}	8.049×10^{-16}
0.1/2	5.845×10^{-2}	5.839×10^{-2}	5.220×10^{-15}	5.239×10^{-15}
0.1/4	1.472×10^{-2}	1.470×10^{-2}	8.756×10^{-15}	8.778×10^{-15}
0.1/8	3.681×10^{-3}	3.678×10^{-3}	1.004×10^{-14}	1.004×10^{-14}
0.1/16	9.202×10^{-4}	9.195×10^{-4}	1.176×10^{-14}	1.174×10^{-14}
0.1/32	2.300×10^{-4}	2.299×10^{-4}	8.344×10^{-15}	8.420×10^{-15}
0.1/64	5.751×10^{-5}	5.747×10^{-5}	1.428×10^{-14}	1.389×10^{-14}
0.1/128	1.434×10^{-5}	1.437×10^{-5}	1.632×10^{-14}	1.643×10^{-14}

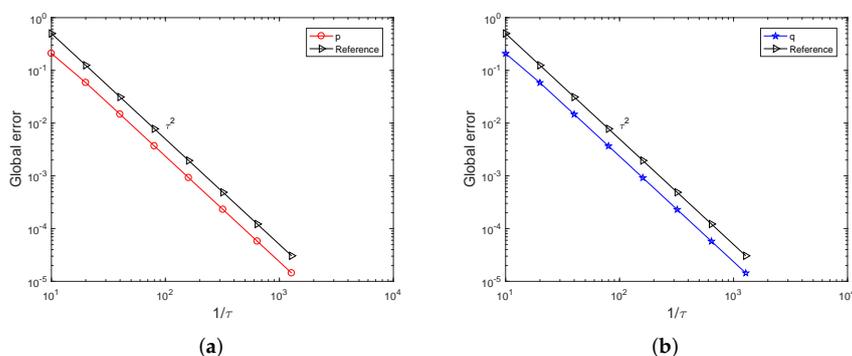


Figure 2. The global errors of p and q over the interval $[0, 2000]$. Subfigure (a) presents the global error of p of symplectic all-at-once method while Subfigure (b) presents the global error of q . Both subfigures clearly show the order of symplectic all-at-once method is 2.

We have also integrated the system over long time interval $[0, 20,000]$ and the comparison results are displayed in Table 3. At all time stepsizes, the CPU times of the symplectic all-at-once method are less than those of the time-stepping method. The difference in CPU time becomes larger when the stepsize turns smaller. The maximum energy errors between the two methods at all stepsizes τ are very close to the machine accuracy. In conclusion, the symplectic all-at-once method is more efficient than the time-stepping method.

Table 3. The comparison of CPU times and maximum energy errors (in the variable p) between the symplectic all-at-once method (AAO) and the time-stepping method (TS) over the interval $[0, 20,000]$.

τ	CPU (AAO)	CPU (TS)	Energy Error
0.1	1.012	1.118	1.214×10^{-17}
0.1/2	1.706	2.543	2.255×10^{-17}
0.1/4	3.949	4.786	5.204×10^{-18}
0.1/8	6.611	10.904	8.674×10^{-18}
0.1/16	11.285	22.096	1.041×10^{-17}
0.1/32	23.435	38.254	1.041×10^{-17}
0.1/64	46.830	76.264	5.204×10^{-18}
0.1/128	94.054	155.335	5.204×10^{-18}

3.2. Example 2: Hénon–Hiles Model

The Hénon–Hiles Model describes stellar motion in celestial mechanics [1,28]. More precisely, it describes the star’s motion under the action of a gravitational potential of a galaxy with cylindrical symmetry. Hénon and Hiles reduced the dimension of the system, and they show that the model actually describes the motion of a particle in a plane with the total energy [1,28]

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

where the potential function $U(q_1, q_2)$ is arbitrary. They chose the potential as

$$U(q_1, q_2) = \frac{1}{2}(q_1^2 + q_2^2) + q_2\left(q_1^2 - \frac{q_2^2}{3}\right).$$

When the potential energy $U(q_1, q_2)$ approaches $\frac{1}{6}$, the level curves of $U(q_1, q_2)$ tends to an equivalent triangle and the triangle finally forms the region

$$R := \left\{ (q_1, q_2)^\top : U(q_1, q_2) \leq \frac{1}{6} \right\}.$$

If one choose a starting point $(q_1^0, q_2^0, p_1^0, p_2^0)^\top$ such that the total energy

$$H(q_1^0, q_2^0, p_1^0, p_2^0) < \frac{1}{6},$$

then the solution stays in the region R . However, if one choose a starting point that the energy H approaches $\frac{1}{6}$, the chaotic behavior will occur.

We have displayed in Figure 3 that the Poincare section [1,11] of (p_2, q_2) with the $q_1 = 0$ plane obtained by the symplectic all-at-once method. Figure 3a shows a quasi-periodic solution with the initial energy $H_0 = \frac{1}{12}$. The initial condition is $p_1 = \frac{1}{6}, p_2 = q_1 = q_2 = 0$. Figure 3b shows a chaotic solution with the initial energy $H_0 = 0.15925$. The initial condition is $p_1 = \sqrt{0.3185}, p_2 = q_1 = q_2 = 0$. The intersections depicted in Figure 3a correspond to one trajectory; they lie on a curve which indicates the quasi-periodic behavior. We can see from Figure 3b that the intersections do not stay in a curve, but are scattered in the two-dimensional (q_2, p_2) plane. This trajectory is called chaotic. The energy errors of

the symplectic all-at-once method for the quasi-periodic trajectory and the chaotic trajectory are also displayed in Figure 3c,d. Energy error is presented by $|H - H_0|$. In both cases, the energy error is bounded by a very small interval. Thus, the symplectic all-at-once method can not only accurately depict the chaotic motion, but also nearly preserves the energy over long time.

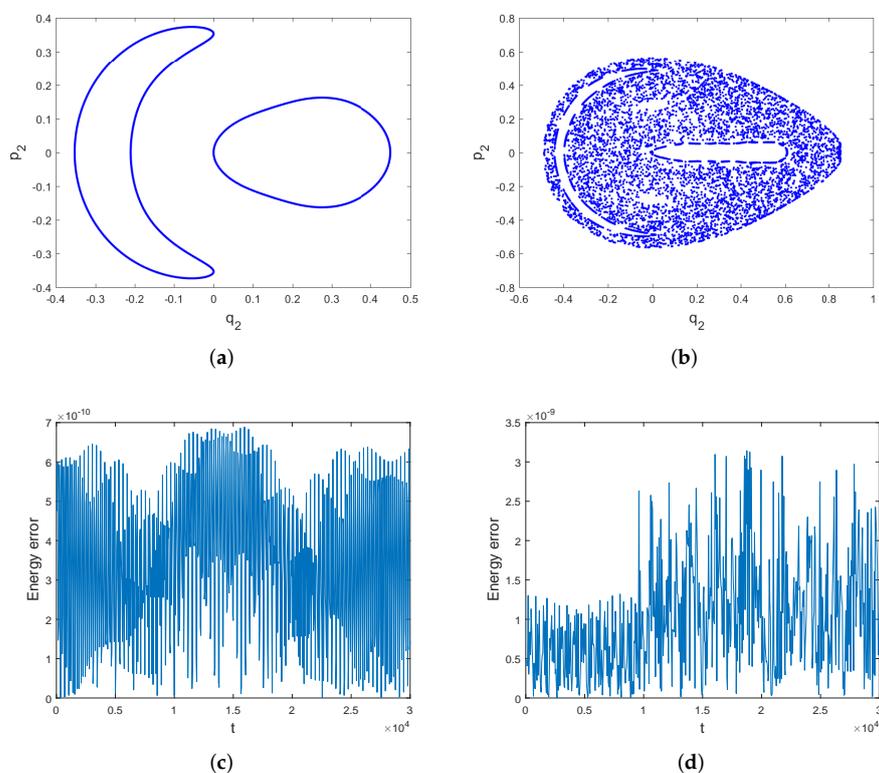


Figure 3. Subfigure (a) shows the Poincaré section of (q_2, p_2) with the $q_1 = 0$ plane. The initial energy is $H_0 = \frac{1}{12}$. The integration is performed by the symplectic all-at-once method for $0 \leq t \leq 30,000$ with stepsize $\tau = 0.0005$. Subfigure (b) shows the Poincaré section of (q_2, p_2) with the $q_1 = 0$ plane. The initial energy is $H_0 = 0.15925$. The integration is performed by the symplectic all-at-once method for $0 \leq t \leq 30,000$ with stepsize $\tau = 0.0005$. Subfigure (c,d) show the energy errors of the symplectic all-at-once method for both trajectories.

We choose the total energy as above $H(p_1, p_2, q_1, q_2) = \frac{1}{2}(p_1^2 + p_2^2) + \frac{1}{2}(q_1^2 + q_2^2) + q_1^2 q_2 - \frac{1}{3} q_2^3$, then the Hénon–Hiles model can be expressed as a four dimensional Hamiltonian system [1,28]

$$\begin{pmatrix} \dot{\mathbf{p}} \\ \dot{\mathbf{q}} \end{pmatrix} = \begin{pmatrix} 0 & -I_2 \\ I_2 & 0 \end{pmatrix} \nabla H(\mathbf{p}, \mathbf{q}), \quad \mathbf{p} = (p_1, p_2)^\top, \quad \mathbf{q} = (q_1, q_2)^\top. \quad (9)$$

We integrated the system (9) with the symplectic all-at-once method and the time-stepping method over the time interval $[0, 10,000]$. The initial condition is chosen as $p_1 = 0$, $p_2 = 0.1$, $q_1 = 0.2$, $q_2 = 0$ such that $H(p_1, p_2, q_1, q_2) < \frac{1}{6}$. The coarse time stepsize for the symplectic all-at-once method is $\tau_c = 0.1/2$.

The Poincaré section of (p_2, q_2) with the $q_1 = 0$ plane and the energy errors of the symplectic all-at-once system and the third order Runge–Kutta method are shown in Figure 4. The Poincaré sections of both methods do not show much difference. However, the symplectic all-at-once method behaves better in energy conservation than the third order Runge–Kutta method. We can see from Figure 4 that the energy error of the symplectic all-at-once method is bounded by a very small number, but that of the third order Runge–Kutta method increases without bound.

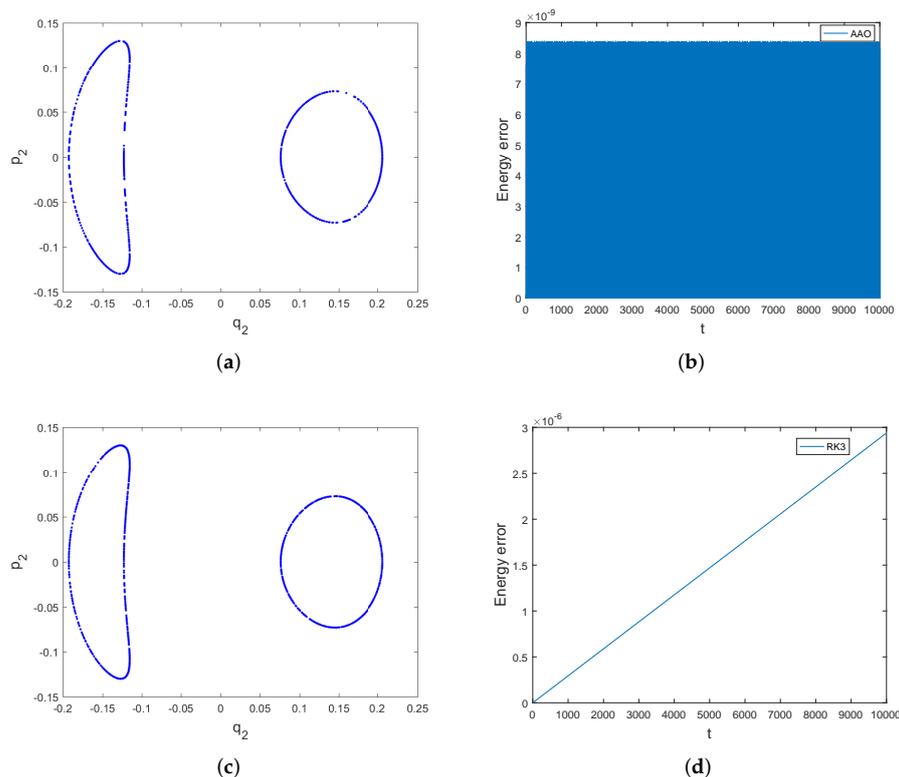


Figure 4. The Poincaré section of (q_2, p_2) and energy errors of symplectic all-at-once method and third order Runge–Kutta method over the interval $[0, 10,000]$. The stepsize is $\tau = 0.01/2$. Subfigure (a,b) display the Poincaré section and energy error of symplectic all-at-once method, while Subfigure (c,d) display the Poincaré section and energy error of third order Runge–Kutta method.

We report in Table 4 for the global errors of p_1 and q_1 of the symplectic all-at-once method under different time stepsize $\tau = 0.01/2^i (i = 1, 2, \dots, 7)$ over the interval $[0, 1000]$. It can be seen that the convergence order of the symplectic all-at-once method is nearly 2. This clearly shows that the symplectic all-at-once method is of order 2. The global errors of the time-stepping method under different stepsizes are displayed in Table 5, and they are the same as those of the symplectic all-at-once method. The maximum errors of p_1 and q_1 between two methods are also presented in Table 5. We can see that the errors are of small value (less than 2.6×10^{-11}).

Table 4. The global errors of variables p_1 and q_1 of the symplectic all-at-once method (AAO) over the interval $[0, 1000]$.

τ	Global Error of p_1	Convergence Order	Global Error of q_1	Convergence Order
0.01/2	4.074×10^{-4}	–	4.198×10^{-4}	–
0.01/4	1.019×10^{-4}	1.999	1.049×10^{-4}	2.001
0.01/8	2.546×10^{-5}	2.001	2.624×10^{-5}	1.999
0.01/16	6.366×10^{-6}	2.000	6.559×10^{-6}	2.000
0.01/32	1.591×10^{-6}	2.000	1.640×10^{-6}	2.000
0.01/64	3.979×10^{-7}	1.999	4.100×10^{-7}	2.000
0.01/128	9.947×10^{-8}	2.000	1.025×10^{-7}	2.000

Table 5. The global errors of variables p_1 and q_1 of the time-stepping method (TS) and the maximum errors of p_1 and q_1 between the symplectic all-at-once method and the time-stepping method over the interval $[0, 1000]$.

τ	Global Error of p_1	Global Error of q_1	Maximum Error of p_1	Maximum Error of q_1
0.01/2	4.074×10^{-4}	4.198×10^{-4}	3.160×10^{-14}	3.369×10^{-14}
0.01/4	1.019×10^{-4}	1.049×10^{-4}	3.096×10^{-14}	3.060×10^{-14}
0.01/8	2.546×10^{-5}	2.624×10^{-5}	1.371×10^{-13}	1.676×10^{-13}
0.01/16	6.366×10^{-6}	6.559×10^{-6}	7.212×10^{-14}	8.903×10^{-14}
0.01/32	1.591×10^{-6}	1.640×10^{-6}	2.429×10^{-11}	2.534×10^{-11}
0.01/64	3.979×10^{-7}	4.100×10^{-7}	1.641×10^{-12}	1.549×10^{-12}
0.01/128	9.947×10^{-8}	1.025×10^{-7}	2.550×10^{-13}	3.165×10^{-13}

Furthermore, the CPU times of the symplectic all-at-once method and the time-stepping method for varying τ are reported in Table 6. From the table, we see that the CPU times of the symplectic all-at-once method are less than those of the time-stepping method. The advantage of the symplectic all-at-once method in CPU time becomes more obvious when τ is smaller. This clearly shows that the symplectic all-at-once method is more efficient than the time-stepping method. The maximum energy errors between the two methods are also presented in Table 6. The energy errors approximately reach the machine accuracy.

Table 6. The comparison of CPU times and maximum energy errors between the symplectic all-at-once method (AAO) and the time-stepping method (TS) over the interval $[0, 10,000]$.

τ	CPU (AAO)	CPU (TS)	Energy Error
0.01/2	17.779	19.199	2.134×10^{-15}
0.01/4	28.628	37.953	1.672×10^{-15}
0.01/8	52.044	76.525	6.158×10^{-15}
0.01/16	99.407	154.227	1.122×10^{-14}
0.01/32	188.392	271.944	4.212×10^{-12}
0.01/64	383.799	473.621	1.356×10^{-13}
0.01/128	746.694	953.738	2.353×10^{-14}

3.3. Example 3: Charged Particle System

Theoretically, there are three methods to describe the motion of plasma while the single charged particle motion is the simplest one. We describe the motion of one single charged particle in given electro-magnetic field without taking account of the reaction of the charged particle motion to the field and the interactions between charged particles. The motion of charged particles is the most fundamental equation and it helps people have a better understanding of many important phenomena in plasma physics research. The dynamics of one single charged particle in a magnetized plasma is a multi-scale problem because it contains two components: the fast gyromotion and the slow guiding center motion. On one hand, by averaging out the fast gyromotion, the guiding center motion can be obtained. The behaviour of guiding centers is governed by gyrokinetics and related theories. On the other hand, by rewriting the equations of the single charged particle motion, it is found that the motion can be expressed as a Hamiltonian system if the Cartesian coordinate (\mathbf{x}, \mathbf{p}) is chosen. Here, the variable $\mathbf{x} = (x, y, z)$ is the configuration variable and \mathbf{p} is the momentum.

The dynamics of a charged particle with the Lorentz force [3,5,13] in the coordinate (\mathbf{x}, \mathbf{p}) is

$$\begin{aligned}\frac{d\mathbf{x}}{dt} &= \frac{1}{m}[\mathbf{p} - q\mathbf{A}(\mathbf{x})], \\ \frac{d\mathbf{p}}{dt} &= -q\nabla\phi(x) + \frac{q}{m}\left(\frac{\partial\mathbf{A}(\mathbf{x})}{\partial\mathbf{x}}\right)^\top[\mathbf{p} - q\mathbf{A}(\mathbf{x})]\end{aligned}\quad (10)$$

which can be written as a Hamiltonian system

$$\frac{d\mathbf{y}}{dt} = J\nabla H(\mathbf{y}),$$

where $\mathbf{y} = (\mathbf{x}^\top, \mathbf{p}^\top)^\top$ is a six-dimensional vector and the Hamiltonian function $H(\mathbf{y}) = \frac{1}{2m}[\mathbf{p} - q\mathbf{A}(\mathbf{x})]^2 + q\phi(x)$. Here q, m are constants, $\mathbf{A}(\mathbf{x})$ is the vector potential. The relation of the magnetic field $\mathbf{B}(\mathbf{x})$ and the vector potential is $\mathbf{B}(\mathbf{x}) = \nabla \times \mathbf{A}(\mathbf{x})$. We integrated the system (10) with the symplectic all-at-once method and the time-stepping method over the time interval $[0, 2000\pi]$. The initial condition for the position variables is $\mathbf{x} = (x_0, y_0, z_0) = [0, -0.65, 0]$ and the velocity is $\mathbf{v} = (\dot{x}_0, \dot{y}_0, \dot{z}_0) = [0.1, 0.01, 0]$. The vector potential $A(x)$ is chosen to be $(0, x, 0)^\top$.

The global errors of variables x and p_1 of the symplectic all-at-once method under different time stepsize $\tau = \pi/10/2^i (i = 1, 2, \dots, 6)$ over the interval $[0, 200\pi]$ are displayed in Table 7. The global errors computed by the symplectic all-at-once method in Table 7 and those computed by the time-stepping method in Table 8 are the same. It is shown in Table 7 the convergence order of the symplectic all-at-once method is 2 which means that the method is of order 2. We have also computed the maximum errors of x and p_1 between two methods, they are all of very small value (less than 3.3×10^{-15}). The maximum errors nearly reach the machine accuracy.

Table 7. The global errors of variables x and p_1 of the symplectic all-at-once method (AAO) over the interval $[0, 200\pi]$.

τ	Global Error of x	Convergence Order	Global Error of p_1	Convergence Order
$\pi/20$	2.879×10^{-1}	–	2.885×10^{-1}	–
$\pi/40$	7.704×10^{-2}	1.902	7.724×10^{-2}	1.901
$\pi/80$	1.939×10^{-2}	1.990	1.941×10^{-2}	1.993
$\pi/160$	4.859×10^{-3}	1.997	4.854×10^{-3}	2.000
$\pi/320$	1.215×10^{-3}	2.000	1.213×10^{-3}	2.001
$\pi/640$	3.038×10^{-4}	2.000	3.034×10^{-4}	1.999

Table 8. The global errors of variables x and p_1 of the time-stepping method (TS) and the maximum errors of x and p_1 between the symplectic all-at-once method and time-stepping method over the interval $[0, 200\pi]$.

τ	Global Error of x (TS)	Global Error of p_1 (TS)	Maximum Error of x	Maximum Error of p_1
$\pi/20$	2.879×10^{-1}	2.885×10^{-1}	1.221×10^{-15}	1.194×10^{-15}
$\pi/40$	7.704×10^{-2}	7.724×10^{-2}	8.188×10^{-16}	8.340×10^{-16}
$\pi/80$	1.939×10^{-2}	1.941×10^{-2}	1.443×10^{-15}	1.443×10^{-15}
$\pi/160$	4.859×10^{-3}	4.854×10^{-3}	1.943×10^{-15}	1.887×10^{-15}
$\pi/320$	1.215×10^{-3}	1.213×10^{-3}	2.012×10^{-15}	2.026×10^{-15}
$\pi/640$	3.038×10^{-4}	3.034×10^{-4}	3.247×10^{-15}	3.109×10^{-15}

We first compare the phase orbits and the energy errors of the symplectic all-at-once method and the third order Runge–Kutta method. Displayed in Figure 5 are the orbits of two components (x, y) of \mathbf{x} and the energy errors of both methods over the interval $[0, 2000\pi]$. We can see that the orbit of the symplectic all-at-once method is very accurate over long time, but the orbit of the third order Runge–Kutta method spirals inward over

time. The energy error of the symplectic all-at-once method is bounded by a very small number, while the energy error of the third order Runge–Kutta method is not bounded and increase along time. The energy errors of the third order Runge–Kutta method are much larger than those of the symplectic all-at-once method. Therefore, the symplectic all-at-once method has overwhelming advantage in tracking the orbit and preserving the energy than the third order Runge–Kutta method.

Table 9 reports the CPU times of the symplectic all-at-once method and the time-stepping method over the interval $[0, 2000\pi]$ with different stepsizes $\tau = \pi/10/2^i, i = 0, 1, \dots, 6$. We can see from Table 9 that the CPU times of the symplectic all-at-once method are all less than those of the time-stepping method. Especially, when the stepsize τ is less than $\pi/80$, the CPU time of the time-stepping method is three times of that of the symplectic all-at-once method. Therefore, this clearly shows that the symplectic all-at-once method is more efficient than the time-stepping method. We have also reported the Newton iteration of the symplectic all-at-once method and the iterations are all 3, which shows the efficiency of our method.

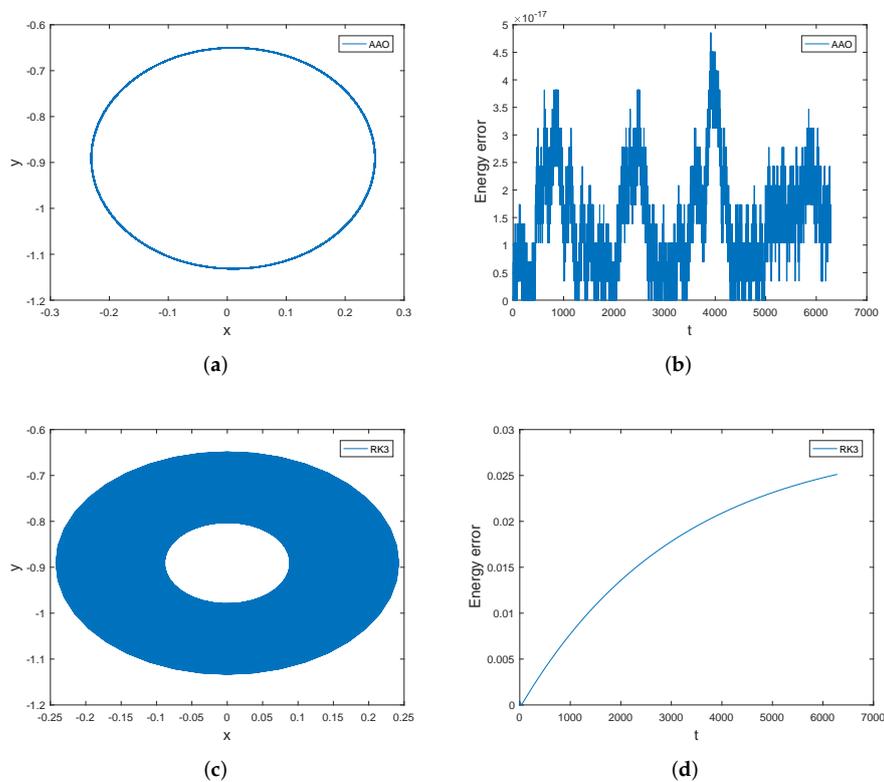


Figure 5. The orbits of (x, y) and the energy errors of the symplectic all-at-once method and the third order Runge–Kutta method over the interval $[0, 2000\pi]$. The stepsize $\tau = \pi/20$. Subfigure (a,b) present the orbit of (x, y) and energy error of symplectic all-at-once method, while Subfigure (c,d) present the orbit of (x, y) and energy error of third order Runge–Kutta method.

Table 9. The comparison of CPU times between the symplectic all-at-once method (AAO) and the time-stepping method (TS) over the interval $[0, 2000\pi]$.

τ	CPU (AAO)	CPU (TS)	Iteration (AAO)
$\pi/10$	0.882	1.112	3
$\pi/20$	1.221	2.192	3
$\pi/40$	1.875	4.363	3
$\pi/80$	3.216	8.724	3
$\pi/160$	5.860	17.555	3
$\pi/320$	11.144	36.371	3
$\pi/640$	21.947	74.532	3

4. Conclusions

We have proposed a symplectic all-at-once method to integrate Hamiltonian systems. The symplectic all-at-once method is a combination of the all-at-once technique and the symplectic midpoint rule. When applying the all-at-once technique to the midpoint rule, we obtain a lower bi-diagonal block system. This bi-diagonal block structure facilitates the use of the fast algorithm. The Newton method equipped with Thomas algorithm are used to rapidly solve this system. The numerical results of the three Hamiltonian problems show that our method is more efficient than the time-stepping method. It is also shown that our method has a significant advantage in tracking the phase orbit and preserving the energy of the system compared to the third order non-symplectic Runge–Kutta method. Thus, the symplectic all-at-once method provides a good choice for solving the Hamiltonian system (2).

In this work, we only consider $2n$ dimensional Hamiltonian systems with $n \leq 3$. For $n = 1$ or 2 , the inverse of A_{kk} , $k = 1, 2, \dots, N$ in lines 2 and 5 of Algorithm 1 can be calculated directly. Thus, the computational complexity is highly reduced. For $n \geq 4$, we suggest using Krylov subspace methods to perform fast computation because the Krylov subspace method does not need to compute A_{kk}^{-1} in Algorithm 1. We only consider the combination of the all-at-once technique and symplectic methods. Apart from symplectic methods, volume-preserving methods and energy-preserving methods are also two important categories of structure-preserving algorithms. They can preserve one of the invariant of Hamiltonian system, the volume and the energy of the system, respectively. Thus, they are also good numerical methods to simulate the Hamiltonian system. In the future work, we will consider combining the all-at-once technique with energy-preserving methods and volume-preserving methods, and then designing fast algorithms for these all-at-once systems.

Author Contributions: Conceptualization, B.-B.Z., Y.-L.Z.; Formal Analysis, Y.-L.Z.; Funding acquisition, B.-B.Z.; Methodology, B.-B.Z., Y.-L.Z.; Software, B.-B.Z.; Writing—original draft, B.-B.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (Grant No. 11901564).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AAO All-at-once
TS Time-stepping

References

1. Hairer, E.; Lubich, C.; Wanner, G. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*; Springer: Berlin/Heidelberg, Germany, 2002.
2. Qin, H.; Guan, X. Variational symplectic integrator for long-time simulations of the guiding-center motion of charged particles in general magnetic fields. *Phys. Rev. Lett.* **2008**, *100*, 035006. [[CrossRef](#)]
3. Tu, X.B.; Zhu, B.B.; Tang, Y.F.; Qin, H.; Liu, J.; Zhang, R.L. A family of new explicit, revertible, volume-preserving numerical schemes for the system of Lorentz force. *Phys. Plasma* **2016**, *23*, 122514. [[CrossRef](#)]
4. Zhang, R.L.; Liu, J.; Tang, Y.F.; Qin, H.; Xiao, J.Y.; Zhu, B.B. Canonailization and symplectic simulation of the gyrocenter dynamics in time-independent magnetic fields. *Phys. Plasma* **2014**, *21*, 03504.
5. Zhou, Z.Q.; He, Y.; Sun, Y.J.; Liu, J.; Qin, H. Explicit symplectic methods for solving charged particle trajectories. *Phys. Plasma* **2017**, *24*, 052507. [[CrossRef](#)]
6. Emel'yanenko, V.V. A method of symplectic integrations with adaptive time-steps for individual Hamiltonians in the planetary N-body problem. *Celest. Mech. Dyn. Astr.* **2007**, *98*, 191–202. [[CrossRef](#)]
7. Feng, K. *Collected Works of Feng Kang: II*; National Dence Industry Press: Beijing, China, 1995.
8. Feng, K. On difference schemes and symplectic geometry. In *International Symposium on Differential Geometry and Differential Equations*; Science Press: Beijing, China, 1984; pp. 42–58.
9. Channell, P.J.; Scovel, J.C. Symplectic integration of Hamiltonian systems. *Nonlinearity* **1990**, *3*, 231–259. [[CrossRef](#)]
10. Feng, K.; Qin, M.Z. *Symplectic Geometric Algorithms for Hamiltonian System*; Springer: New York, NY, USA, 2009.
11. Sanz-Serna, J.M.; Calvo, M.P. *Numerical Hamiltonian Problems*; Chapman and Hall: London, UK, 1994.
12. Tang, Y.F. Formal energy of a symplectic scheme for Hamiltonian systems and its applications (I). *Comput. Math. Appl.* **1994**, *27*, 31–39.
13. Zhang, R.L.; Qin, H.; Tang, Y.F.; Liu, J.; He, Y.; Xiao, J.Y. Explicit symplectic algorithms based on generating functions for charged particle dynamics. *Phys. Rev. E* **2016**, *94*, 013205. [[CrossRef](#)]
14. Qin, H.; Guan, X.; Tang, W.M. Variational symplectic algorithm for guiding center dynamics and its application in tokamak geometry. *Phys. Plasmas* **2009**, *16*, 042510. [[CrossRef](#)]
15. Stoll, M.; Wathen, A. All-at-once solution of time-dependent Stokes control. *J. Comput. Phys.* **2013**, *232*, 498–515. [[CrossRef](#)]
16. Yilmaz, F.; Karasözen, B. An all-at-once approach for the optimal control of the unsteady Burgers equation. *J. Comput. Appl. Math.* **2014**, *259*, 771–779. [[CrossRef](#)]
17. Gu, X.M.; Wu, S.L. A parallel-in-time iterative algorithm for Volterra partial integro-differential problems with weakly singular kernel. *J. Comput. Phys.* **2020**, *417*, 109576. [[CrossRef](#)]
18. Ke, R.; Ng, M.K.; Sun, H.W. A fast direct method for block triangular Toeplitz-like with tri-diagonal block systems from time-fractional partial differential equations. *J. Comput. Phys.* **2016**, *303*, 203–211. [[CrossRef](#)]
19. Lu, X.; Pang, H.K.; Sun, H.W. Fast apprioximate inversion of a block triangular Toeplitz matrix with applications to fractional sub-diffusion equations. *Numer. Linear Algebra Appl.* **2015**, *22*, 866–882. [[CrossRef](#)]
20. Zhao, Y.L.; Zhu, P.Y.; Gu, X.M.; Zhao, X.L.; Cao, J. A limited-memory block bi-diagonal Toeplitz preconditioner for block lower triangular Toeplitz system from time-space fractional diffusion equation. *J. Comput. Appl. Math.* **2019**, *362*, 99–115. [[CrossRef](#)]
21. Zhao, Y.L.; Gu, X.M.; Ostermann, A. A Preconditioning technique for an all-at-once system from Volterra subdiffusion equations with graded time steps. *J. Sci. Comput.* **2021**, *88*, 11. [[CrossRef](#)]
22. Gu, X.M.; Zhao, Y.L.; Zhao, X.L.; Carpentieri B.; Huang Y.Y. A note on parallel preconditioning for the all-at-once solution of Riesz fractional diffusion equations. *Numer. Math. Theor. Meth. Appl.* **2021**, *14*, 893–919.
23. Lasagni, F.M. Canonical Runge-Kutta methods. *ZAMP* **1988**, *39*, 952–953. [[CrossRef](#)]
24. Sanz-Serna, J.M. Runge-Kutta Schemes for Hamiltonian Systems. *BIT Numer. Math.* **1988**, *28*, 877–883. [[CrossRef](#)]
25. Tang, Y.F.; Pérez-García, V.; Vázquez, L. Symplectic methods for the Ablowitz-Ladik model. *Appl. Math. Comput.* **1997**, *82*, 17–38. [[CrossRef](#)]
26. Zhu, B.B.; Tang, Y.F.; Zhang, R.L.; Zhang, Y.H. Symplectic simulation of dark solitons motion for nonlinear Schrödinger equation. *Numer. Algorithms* **2019**, *81*, 1485–1503. [[CrossRef](#)]
27. Saad, Y. *Iterative Methods for Sparse Linear Systems*, 2nd ed.; SIAM: Philadelphia, PA, USA, 2003.
28. Brugnano, L.; Iavernaro, F. *Line Integral Methods for Conservative Problems*; Chapman and Hall/CRC: Boca Raton, FL, USA, 2016.