Chapter 13
Semi-Analytical ERKN Integrators for Solving High-Dimensional Nonlinear Wave Equations

Incorporating the operator-variation-of-constants formula for high-dimensional nonlinear wave equations with Fast Fourier Transform techniques in this chapter, we present a class of semi-analytical ERKN integrators, which can nearly preserve the spatial continuity as well as the oscillations of the underlying nonlinear waves equations. Standard ERKN methods require, in every time step, the computation of the matrix-vector product whose computational complexity, in terms of basic multiplication is $\mathscr{O}(N^2)$, once a direct calculation procedure is implemented, where N is the dimension of the underlying differentiation matrix. We design and analyse efficient algorithms which are incorporated with the Fast Fourier Transform in the implementation of ERKN integrators, so that these algorithms reduce the computational cost from $\mathscr{O}(N^2)$ to $\mathscr{O}(N \log N)$ in terms of basic multiplication.

13.1 Introduction

This chapter concerns the numerical simulation of high-dimensional nonlinear wave equations. Although all of the ideas, algorithms and analysis in this chapter can be straightforwardly extended to the solution of nonlinear wave equations in a moderate number of space dimensions, we begin with the nonlinear one-dimensional Hamiltonian wave equation

$$u_{tt} - a^2 u_{xx} = f(u), \tag{13.1}$$

with 2π-periodic boundary condition ($x \in \Omega = \mathbb{R}/(2\pi\mathbb{Z})$) and initial values

$$u(x, t_0) = \varphi(x) \in H^{s+1}(\Omega), \quad u_t(x, t_0) = \psi(x) \in H^s(\Omega), \tag{13.2}$$

X. Wu, B. Wang, *Geometric Integrators for Differential Equations with Highly Oscillatory Solutions*, https://doi.org/10.1007/978-981-16-0147-7_13

where $s \geqslant 0$, and $H^s(\Omega)$ is the Sobolev space on Ω. We consider the domain $\Omega = [0, 2\pi]$ for simplicity. (13.1) is a conservative system due to the conservation of the Hamiltonian energy

$$H = H(t) = \frac{1}{2}\int_{\Omega}\Big((u_t)^2 + a^2(u_x)^2 + 2V(u(x,t))\Big)\mathrm{d}x, \tag{13.3}$$

where $f(u) = -\dfrac{\mathrm{d}V(u)}{\mathrm{d}u}$.

We first define the formal series

$$\phi_j(x) := \sum_{k=0}^{\infty}\frac{(-1)^k x^k}{(2k+j)!}, \quad j = 0, 1, \cdots, \tag{13.4}$$

for any $x \geqslant 0$, and the differential operator $\mathscr{A}$

$$(\mathscr{A}v)(x) = -a^2 v_{xx}(x).$$

This leads to the following operator-variation-of-constants formula for the initial-boundary-value problem of (13.1) (see, e.g. [1])

$$\begin{cases} u(x,t) = \phi_0\big((t-t_0)^2\mathscr{A}\big)\varphi(x) + (t-t_0)\phi_1\big((t-t_0)^2\mathscr{A}\big)\psi(x) \\ \qquad\qquad + \displaystyle\int_{t_0}^{t}(t-\zeta)\phi_1\big((t-\zeta)^2\mathscr{A}\big)f(u(x,\zeta))\mathrm{d}\zeta, \\ u_t(x,t) = -(t-t_0)\mathscr{A}\phi_1\big((t-t_0)^2\mathscr{A}\big)\varphi(x) + \phi_0\big((t-t_0)^2\mathscr{A}\big)\psi(x) \\ \qquad\qquad + \displaystyle\int_{t_0}^{t}\phi_0\big((t-\zeta)^2\mathscr{A}\big)f(u(x,\zeta))\mathrm{d}\zeta, \end{cases} \tag{13.5}$$

where both $\phi_0\big((t-t_0)^2\mathscr{A}\big)$ and $\phi_1\big((t-t_0)^2\mathscr{A}\big)$ are bounded operators as stated in Chap. 1 (see also [2]), although $\mathscr{A}$ is a linear, unbounded positive semi-definite operator.

We remark that the formula (13.5) exactly provides an implicit expression for the solution to (13.1). In particular, for the special case where $f(u) = 0$, (13.5) yields the closed-form solution to (13.1). Moreover, with the help of (13.5), we are hopeful of obtaining semi-analytical integrators for (13.1), which preserve the continuity of the spatial variable x, and only discretise the time variable t. An interesting example is the energy-preserving and symmetric scheme presented in Chap. 9 (see also [3]), which can exactly preserve the true continuous energy (13.3), not a discrete energy after spatial discretisations as is typically the case for other methods. It is noted that the extended Runge–Kutta–Nyström (ERKN) methods have been well developed for highly oscillatory systems of ordinary differential equations (see, e.g. [4–8])

$$\begin{cases} y'' + My = f(y), \quad t \in [t_0, T], \\ y(t_0) = y_0, \;\; y'(t_0) = y_0', \end{cases} \tag{13.6}$$

where M is a (symmetric) positive semi-definite matrix and $\|M\| \gg \max\left\{1, \left\|\dfrac{\partial f}{\partial y}\right\|\right\}$. This line of research for (13.6) will assist in the design and development of numerical schemes for (13.1).

For the formulation of semi-analytical ERKN integrators for (13.1), we first rewrite (13.5) as

$$\begin{cases} u(x, t_n + \tau h) = \phi_0\big(\tau^2 V\big)u(x, t_n) + \tau h\phi_1\big(\tau^2 V\big)u_t(x, t_n) \\ \qquad\qquad + h^2 \displaystyle\int_0^\tau (\tau - \zeta)\phi_1\big((\tau - \zeta)^2 V\big) f(u(x, t_n + \zeta h))\mathrm{d}\zeta, \\ u_t(x, t_n + \tau h) = -\tau hM\phi_1\big(\tau^2 V\big)u(x, t_n) + \phi_0\big(\tau^2 V\big)u_t(x, t_n) \\ \qquad\qquad + h \displaystyle\int_0^\tau \phi_0\big((\tau - \zeta)^2 V\big) f(u(x, t_n + \zeta h))\mathrm{d}\zeta, \end{cases} \tag{13.7}$$

where $V = h^2\mathscr{A}$ and $h > 0$ is the time stepsize. We assume that $U_i \approx u(x, t_n + C_i h)$, $u_{n+1} \approx u(x, t_n + h)$ and $u'_{n+1} \approx u_t(x, t_n + h)$. We set $\tau = C_i$ satisfying $0 < C_i < 1$ for $i = 1, \cdots, s$, and approximate the first integral appearing in (13.7) by a suitable quadrature formula with the weights $A_{ij}(V)$. This leads to the internal stages of ERKN integrators for (13.1). Likewise, by setting $\tau = 1$, the updates of ERKN integrators for (13.1) follow from the approximations to the two integrals appearing in (13.7) by suitable quadrature formulae with the weights $\overline{B}_i(V)$ and $B_i(V)$, respectively. Then we are in a position to define a semi-analytical ERKN integrator for (13.1).

An s-stage semi-analytical ERKN integrator for (13.1) reads

$$\begin{cases} U_i = \phi_0(C_i^2 V)u_n + C_i h\phi_1(C_i^2 V)u_n' + h^2 \displaystyle\sum_{j=1}^{s} A_{ij}(V) f(U_j), \quad i = 1, \cdots, s, \\ u_{n+1} = \phi_0(V)u_n + h\phi_1(V)u_n' + h^2 \displaystyle\sum_{i=1}^{s} \bar{B}_i(V) f(U_i), \\ u'_{n+1} = -h\mathscr{A}\phi_1(V)u_n + \phi_0(V)u_n' + h \displaystyle\sum_{i=1}^{s} B_i(V) f(U_i), \end{cases} \tag{13.8}$$

where the constants $C_1, \cdots, C_s$, and the operator-argument coefficients $A_{ij}(V)$, $\overline{B}_i(V)$ and $B_i(V)$ for $i, j = 1, \cdots, s$ are determined to ensure that the numerical scheme (13.8) is convergent and stable. It can be observed from (13.8) that the ERKN integrator defines a time-stepping procedure, and the initial conditions are exactly (13.2), i.e., $u_0 = u(x, t_0)$ and $u_0' = u_t(x, t_0)$. This class of integrators possesses the superior property of preserving spatial continuity. Therefore, we call them semi-analytical integrators (see, e.g. [9]). It should be noted that the ERKN method for (13.6) can also be expressed in the form of (13.8) by replacing the operator $\mathscr{A}$ in V with a suitable differentiation matrix M, and remember that the ERKN method for (13.6) is oscillation preserving as stated in Chap. 1. For convenience, we denote this ERKN integrator corresponding to (13.8) by a partitioned Butcher tableau

$$\begin{array}{c|c} C & A(V) \\ \hline & \bar{B}(V)^{\intercal} \\ \hline & B(V)^{\intercal} \end{array} = \begin{array}{c|ccc} C_1 & A_{11}(V) & \cdots & A_{1s}(V) \\ \vdots & \vdots & & \vdots \\ C_s & A_{s1}(V) & \cdots & A_{ss}(V) \\ \hline & \bar{B}_1(V) & \cdots & \bar{B}_s(V) \\ \hline & B_1(V) & \cdots & B_s(V) \end{array} \cdot \tag{13.9}$$

Despite the superior properties we have mentioned, the semi-analytical ERKN integrators (13.8) as well as the energy-preserving and symmetric scheme in [3] could not be easily applied to (13.1) for general nonlinear cases, since it is difficult to calculate and implement the operator-argument functions involved in these integrators. This fact greatly confines the potential and further application of these semi-analytical integrators, although they have been successfully applied to some linear or homogeneous wave equations (see, e.g. [10]). A feasible approach to implementing (13.8) in practice is approximating the operator $\mathscr{A}$ by a suitable differentiation matrix M, once the spatial discretisation is carried out. When sufficient spatial mesh grids are appropriately chosen, the spatial discretisation error will be smaller than the roundoff error in theory (see, e.g. [11, 12]). Hence, in the sense of numerical computation, we can expect and consider that the spatial precision and continuity are nearly preserved by the ERKN integrators (13.8), because it turns out that the global error of ERKN integrators is independent of the spatial refinement (see, e.g. [13]). Moreover, it has been emphasised that the global error bounds of the ERKN integrators are completely independent of the differentiation matrix M in Chap. 3.

However, for the sake of the near preservation of spatial continuity, the dimension of the differentiation matrix M will be selected so large that the error of spatial discretisations can be almost ignored. This results in the following three difficulties in the practical implementation of (13.8). First, the computation of matrix-valued functions $A_{ij}(V)$, $B_i(V)$, and $\bar{B}_i(V)$ will be of high complexity for such a high-dimensional matrix M, since they are in fact expressed in the series of M. Second, the multiplication at each time step between these matrices and vectors is also highly costly. For instance, if we denote N as the dimension of the matrix M, the multiplication between $A_i(V)$ and $f(Y_i)$ contains N^2 basic scalar multiplication and

$N(N-1)$ basic scalar addition, which are an order of magnitude $\mathscr{O}(N^2)$. Hence, the computational cost of basic operation in each time step will be rapidly increased in the magnitude of $\mathscr{O}(N^2)$ as N increases. Third, the necessary computer memory to store $A_{ij}(V)$, $B_i(V)$ and $\bar{B}_i(V)$ will also sharply increase, which may result in the computer running out of memory, in particular for high-dimensional problems. In order to obtain the near preservation of the spatial continuity in the application of semi-analytical ERKN integrators (13.8), and overcome the above mentioned obstacles in the practical implementation after a possibly highly refined spatial discretisation, it is wise to avoid the calculation and storage of such matrix-valued functions, as well as the direct multiplication between matrix-valued functions and the corresponding vectors. Consequently, in this chapter, we consider solving the nonlinear wave equations in Fourier space. Then the system of ordinary differential equations with respect to the Fourier coefficients will have a natural harmony with the ERKN method, i.e., the matrix-vector multiplication will disappear when the ERKN method is used to solve this system. Since the Fourier coefficients can be obtained by the Fast Fourier Transform (FFT) with $\mathscr{O}(N \log N)$ operations (see, e.g. [14]), we are hopeful of obtaining a fast implementation approach to ERKN integrators when applied to the nonlinear wave equations, even if N is very large. This motivates the presentation of Algorithm 1 in this chapter. Furthermore, making use of the equivalence between the splitting method and an important class of symplectic ERKN methods, we present Algorithm 2, which is shown to be more efficient than Algorithm 1.

The finite difference method could also yield the near preservation of spatial continuity, in theory, for the nonlinear wave equation (13.1), once the spatial stepsize $\Delta x \to 0$ and the convergence of the numerical solution to the exact solution is satisfied. Unfortunately, however, it follows from the Courant–Friedrichs–Lewy (CFL) condition in the literature (see, e.g. [15–17]) that the mesh ratio should satisfy $h/\Delta x \leqslant \gamma$ for the sake of numerical stability, where γ is a positive constant depending only on the selected difference scheme. This implies that the time stepsize h would be restricted to a very tiny magnitude, once the Δx is selected as so small that it can ensure near preservation of the spatial continuity. This fact greatly confines the application of the finite difference method. Fortunately, the stability analysis of the ERKN integrator (13.8) in [18] shows that the time stepsize is independent of the spatial stepsize, but dependent only on the coefficients of ERKN integrator (13.8) and the Lipschitz constant of the function $f(u)$. This advantage admits the use of a large time stepsize even though Δx is very small after the requirement of spatial-mesh refinement, once the semi-analytical ERKN integrator (13.8) is applied to solve the nonlinear wave equations.

Another noteworthy aspect of scientific research related to the theme of this chapter is the application of the Fourier spectral method and the FFT techniques. In the literature (see, e.g. [19–22]), the authors respectively discussed the applications of Fourier spectral discretisation for different types of partial differential equations. In particular, in [19, 20, 22] and Chap. 2 in [23], the authors also incorporated the FFT into the implementation to try to achieve smaller computational cost and lower memory storage. However, all such methods used a difference scheme or a Runge–Kutta-type discretisation for the time derivative. It is very clear that these procedures are not suitable for oscillatory nonlinear wave equations due to the high oscillation of the semidiscrete system. Moreover, the CFL condition is also required and crucial for these methods (see, e.g. [22]), which results in the same fatal defect of a tiny time stepsize for the finite difference method. On noticing that ERKN methods can efficiently solve a highly oscillatory system of ordinary differential equations, the two proposed algorithms combined with the FFT technique are really useful and promising in the implementation of semi-analytical ERKN integrator (13.8) for efficiently solving a nonlinear wave equations. Apart from the exponential integrators studied in this chapter we also note that there exist alternative approaches for effectively providing numerical solutions for (13.1) in the literature (see, e.g. [24–29]).

13.2 Preliminaries

We begin by considering initial value problems of second-order differential equations

$$\begin{cases} y'' = f(y), \quad t \in [t_0, T], \\ y(t_0) = y_0, \ y'(t_0) = y_0'. \end{cases} \tag{13.10}$$

The standard Runge–Kutta–Nyström (RKN) method (see, e.g. [30, 31]) for (13.10) is given by

$$\begin{cases} Y_i = y_n + c_i h y_n' + h^2 \sum_{j=1}^{s} a_{ij} f(Y_j), \quad i = 1, \cdots, s, \\ y_{n+1} = y_n + h y_n' + h^2 \sum_{i=1}^{s} \bar{b}_i f(Y_i), \\ y_{n+1}' = y_n' + h \sum_{i=1}^{s} b_i f(Y_i), \end{cases} \tag{13.11}$$

where $a_{ij}, \bar{b}_i, b_i, c_i$ for $i, j = 1, \cdots, s$ are real constants. An intrinsic relation between ERKN methods and RKN methods has been explored in [4], in which the authors revealed the underlying extension from the RKN method to the ERKN method. We summarise the following three theorems, which are useful for our subsequent analysis and the details can be found in [4].

Theorem 13.1 (See [4]) *Let a RKN method be of order r for (13.10) with coefficients c_i, b_i, $\bar{b}_i$ and a_{ij} for $i, j = 1, \cdots, s$. Then the ERKN method determined by* *the mapping:*

$$\begin{cases} C_i = c_i, \\ A_{ij}(V) = a_{ij}\phi_1((c_i - c_j)^2 V), \\ \bar{B}_i(V) = \bar{b}_i\phi_1((1 - c_i)^2 V), \\ B_i(V) = b_i\phi_0((1 - c_i)^2 V), \end{cases} \tag{13.12}$$

is also of order r for (13.6).

Theorem 13.2 (See [4]) *Let a RKN method be symplectic for* (13.10) *with coefficients c_i, b_i, $\bar{b}_i$ and a_{ij} for $i, j = 1, \cdots, s$. Then the ERKN method determined by* (13.12) *is also symplectic for* (13.6).

Theorem 13.3 (See [4]) *Let a RKN method be symmetric for* (13.10)*, whose coefficients c_i, b_i, $\bar{b}_i$ and a_{ij} satisfy the simplifying assumption $\bar{b}_i = b_i(1 - c_i)$ for $i, j = 1, \cdots, s$. Then the ERKN method determined by* (13.12) *is also symmetric for* (13.6).

During the implementation of ERKN integrators, the matrix-valued coefficients $A_{ij}(V)$, $\bar{B}_i(V)$ and $B_i(V)$ should be calculated in advance. Due to their complicated computation, we note that the coefficients of the ERKN integrators in this chapter share the form in (13.12), since the special cases where $\phi_0(x) = \cos\sqrt{x}$ and $\phi_1(x) = \sin\sqrt{x}/\sqrt{x}$ can highly simplify the calculation of $A_{ij}(V)$, $\bar{B}_i(V)$ and $B_i(V)$. Another advantage of the formula (13.12) is that the ERKN integrator obtained from (13.12) and the corresponding RKN method nearly has the best structure-preserving properties among its congruence class, which will reduce to the same RKN method (see [4]).

A preliminary step to simplifying the calculation of the computational cost of ERKN integrators will be made, provided we carry out the following transformation. If we set $F_i = f(U_i)$, then the first formula of (13.8) can be rewritten as

$$U_i = \phi_0(C_i^2 V)u_n + C_i h\phi_1(C_i^2 V)u_n' + h^2 \sum_{j=1}^{s} A_{ij}(V)F_j, \quad i = 1, \cdots, s,$$

by replacing $f(U_i)$ with F_i. Using $F_i = f(U_i)$ once again with the above equation, we obtain

$$F_i = f\big(\phi_0(C_i^2 V)u_n + C_i h\phi_1(C_i^2 V)u_n' + h^2 \sum_{j=1}^{s} A_{ij}(V)F_j\big), \quad i = 1, \cdots, s. \tag{13.13}$$

Finally, replacing $f(U_i)$ with F_i in the last two equations of (13.8) and combining with (13.13) we can deduce the equivalent formulation

$$\begin{cases} F_i = f\big(\phi_0(C_i^2 V)u_n + C_i h\phi_1(C_i^2 V)u_n' + h^2 \sum\limits_{j=1}^{s} A_{ij}(V)F_j\big), \quad i = 1, \cdots, s, \\ u_{n+1} = \phi_0(V)u_n + h\phi_1(V)u_n' + h^2 \sum\limits_{i=1}^{s} \bar{B}_i(V)F_i, \\ u_{n+1}' = -h\mathscr{A}\phi_1(V)u_n + \phi_0(V)u_n' + h \sum\limits_{i=1}^{s} B_i(V)F_i. \end{cases} \tag{13.14}$$

It is clear that, in comparison with (13.8), the ERKN integrator rewritten in the form of (13.14) reduces the number of function evaluations of $f(u)$ from $s^2 + 2s$ to s for each time step, while it maintains the same number $(s + 2)^2$, of matrix-vector multiplications. Therefore, the practical formulation of ERKN integrators in applications should be (13.14), rather than (13.8) for the nonlinear wave equation (13.1).

Since the error analysis of ERKN integrators as well as of Gauschi-type methods for nonlinear wave equations has been made in [13, 18, 32, 33], we will not consider this issue further, but pay attention to the practical implementation of these semi-analytical ERKN integrators for nonlinear wave equations.

13.3 Fast Implementation of ERKN Integrators

For $s \geqslant 0$, we have $H^{s+1}(\Omega) \subset L^2(\Omega)$, where $L^2(\Omega)$ is the complex Hilbert space equipped with the inner product and the norm

$$(u, v) = \frac{1}{2\pi} \int_{\Omega} u(x)\bar{v}(x)\mathrm{d}x, \quad \|u\| = (u, u). \tag{13.15}$$

Consider the Fourier series of $u(x, t)$ as follows

$$u_*(x, t) = \sum_{k=-\infty}^{+\infty} \hat{u}_k(t)\mathrm{e}^{\mathrm{i}kx}, \tag{13.16}$$

where the Fourier coefficients $\hat{u}_k(t)$ are determined by

$$\hat{u}_k(t) = (u, \mathrm{e}^{\mathrm{i}kx}) = \frac{1}{2\pi} \int_{\Omega} u(x, t)\mathrm{e}^{-\mathrm{i}kx}\mathrm{d}x, \quad k \in \mathbb{Z}. \tag{13.17}$$

Taking account of the completeness of the Fourier/trigonometric system $\{e^{ikx} : k \in \mathbb{Z}\}$ in $L^2(\Omega)$ (see, e.g. [34]), we obtain that $||u(x,t) - u_*(x,t)||_{L^2(\Omega)} = 0$. In the sense of $L^2(\Omega)$-norm, it is sufficient to find $u_*(x,t)$ instead of the exact $u(x,t)$. Note that the complete orthonormal set $\{e^{ikx} : k \in \mathbb{Z}\}$ constitutes the set of orthogonal eigenfunctions of the operator $\mathscr{A}$ in the Hilbert space $L^2(\Omega)$, i.e.,

$$\mathscr{A}(e^{ikx}) = a^2k^2e^{ikx}, \quad k \in \mathbb{Z}. \tag{13.18}$$

With the formulae (13.4) and (13.18), we consequently have that

$$\phi_j(h^2\mathscr{A})e^{ikx} = \phi_j(a^2k^2h^2)e^{ikx}. \tag{13.19}$$

Two special cases of (13.19) are $j = 0$ and $j = 1$:

$$\phi_0(h^2\mathscr{A})e^{ikx} = \cos(akh)e^{ikx}, \quad \phi_1(h^2\mathscr{A})e^{ikx} = \frac{\sin(akh)}{akh}e^{ikx}. \tag{13.20}$$

Let N denote the number of spatial mesh grids after spatial discretisation, and we only consider the even integer case for N. Since u is a real function, the Fourier coefficients $\hat{u}_k$ satisfy $\hat{u}_{-k} = \overline{\hat{u}_k}$. Let $X_N = \text{span}\{e^{ikx} : -N/2 \leqslant k \leqslant N/2\}$, and $P_N : L^2(\Omega) \to X_N$ be the L^2-orthogonal projection. It is obvious that $P_N u$ will be the truncated Fourier series

$$\big(P_N u\big)(x) = \sum_{k=-N/2}^{N/2} \hat{u}_k e^{ikx}, \tag{13.21}$$

which is also the best approximation to $u(x)$ in L^2-norm. The truncated Fourier series (13.21) provides us an efficient way to approximate $u(x)$. However, it is clear that the Fourier coefficients $\hat{f}_k(u(x))$ of $f(u)$ are hard to obtain, on noticing the integral in (13.17), since the expression of $f(u(x))$ with respect to x is always unknown. Therefore, we will not use the truncated Fourier series (13.21) in practice, but consider the following Fourier/trigonometric interpolation instead.

Let

$$x_j = jh = j\frac{2\pi}{N}, \quad 1 \leqslant j \leqslant N, \tag{13.22}$$

be N equispaced points in $[0, 2\pi]$. Since N is even, we set

$$Y_N = \Big\{u(x) = \sum_{k=-N/2}^{N/2} \tilde{u}_k e^{ikx} : \tilde{u}_{-N/2} = \tilde{u}_{N/2}\Big\}. \tag{13.23}$$

Then the Fourier/trigonometric interpolation polynomial on the equispaced points x_j $(1 \leqslant j \leqslant N)$ can be obtained by the interpolation operator $I_N : L^2(\Omega) \to Y_N$ as follows:

$$\big(I_N u\big)(x) = \sum_{k=-N/2}^{N/2} \tilde{u}_k \mathrm{e}^{\mathrm{i}kx}. \tag{13.24}$$

This satisfies

$$\big(I_N u\big)(x_j) = u(x_j), \quad 1 \leqslant j \leqslant N. \tag{13.25}$$

Differently from the Fourier coefficients $\hat{u}_k$ in (13.16), the interpolation coefficients $\tilde{u}_k$ can be effectively obtained from $u_j = u(x_j)$ $(1 \leqslant j \leqslant N)$ by the Discrete Fourier Transform (DFT)

$$\tilde{u}_k = \frac{1}{\omega_k N} \sum_{j=1}^{N} u_j \mathrm{e}^{-\mathrm{i}kx_j}, \quad k = -N/2, \cdots, N/2, \tag{13.26}$$

where $\omega_k = 1$ for $|k| < N/2$, and $\omega_k = 2$ for $k = \pm N/2$. Meanwhile, it follows from (13.24) and (13.25) that u_j can also be obtained by the inverse DFT

$$u_j = \sum_{k=-N/2}^{N/2} \tilde{u}_k \mathrm{e}^{\mathrm{i}kx_j}, \quad j = 1, \cdots, N. \tag{13.27}$$

The DFT (13.26) and inverse DFT (13.27) can be carried out by the Fast Fourier Transform (FFT) and inverse Fast Fourier Transform (IFFT) with only $\mathscr{O}(N \log N)$ operations (see, e.g. [14]), rather than by the direct matrix-vector multiplication with $\mathscr{O}(N^2)$ operations. This computational process for FFT and IFFT can be easily accomplished by using MATLAB (see, e.g. [12]).

Although the Fourier interpolation approximation $I_N u$ in (13.24) to $u(x)$ is usually not better than the truncation $P_N u$ in (13.21), the rigorous error analysis in [11] shows that '*the penalty for using interpolation instead of truncation is at worst a factor of two*'. Hence, we can still have the spectral accuracy of exponential convergence of the Fourier/trigonometric interpolation approximation (see, e.g. [12]). With regard to more details on the truncation error of the Fourier truncation and the interpolation error of the Fourier/trigonometric interpolation, readers are referred to [23, 35, 36].

Now replacing $u(x, t)$ with the trigonometric interpolation polynomial $I_N u$ in (13.1) leads to the nonlinear system

$$\tilde{u}_k'' + a^2 k^2 \tilde{u}_k = \tilde{f}_k(t), \quad k = -N/2, \cdots, N/2. \tag{13.28}$$

Algorithm 1 Fast implementation of explicit ERKN integrator (13.14) with large N

```
1: set $U_0^j = \varphi(x_j)$, $V_0^j = \psi(x_j)$
2: for $n = 1$ to $N_T$ do
3:    FFT: $U_{n-1} \longrightarrow \widehat{U}_{n-1}$, $V_{n-1} \longrightarrow \widehat{V}_{n-1}$
4:    set $\widehat{U} = \cos(c_1 hK)\widehat{U}_{n-1} + K^{-1}\sin(c_1 hK)\widehat{V}_{n-1}$
5:    inverse FFT: $\widehat{U} \longrightarrow U$
6:    set $F_1 = f(U)$
7:    FFT: $F_1 \longrightarrow \widehat{F}_1$
8:    set $\widehat{U}_n = \cos(hK)\widehat{U}_{n-1} + K^{-1}\sin(hK)\widehat{V}_{n-1} + h^2\bar{b}_1 \cdot ((1-c_1)hK)^{-1}\sin((1-c_1)hK)\cdot\widehat{F}_1$
        $\widehat{V}_n = -K\sin(hK)\widehat{U}_{n-1} + \cos(hK)\widehat{V}_{n-1} + hb_1\cos((1-c_1)hK)\cdot\widehat{F}_1$
9:    for $i = 2$ to $s$ do
10:       set $\widehat{U} = \cos(c_i hK)\widehat{U}_{n-1} + K^{-1}\sin(c_i hK)\widehat{V}_{n-1}$
11:       for $k = 1$ to $i-1$ do
12:          $\widehat{U} = \widehat{U} + h^2 a_{ik}\cdot((c_i - c_k)hK)^{-1}\sin((c_i - c_k)hK)\cdot\widehat{F}_k$
13:       end for
14:       inverse FFT: $\widehat{U} \longrightarrow U$
15:       set $F_i = f(U)$
16:       FFT: $F_i \longrightarrow \widehat{F}_i$
17:       $\widehat{U}_n = \widehat{U}_n + h^2\bar{b}_i \cdot ((1-c_i)hK)^{-1}\sin((1-c_i)hK)\cdot\widehat{F}_i$
          $\widehat{V}_n = \widehat{V}_n + hb_i\cos((1-c_i)hK)\cdot\widehat{F}_i$
18:    end for
19:    inverse FFT: $\widehat{U}_n \longrightarrow U_n$, $\widehat{V}_n \longrightarrow V_n$
20: end for
```

where the $\tilde{f}_k(t)$ for $k = -N/2, \cdots, N/2$ are the trigonometric interpolation coefficients of $f(u(x))$ at time t. Since $\tilde{u}_k$ and $\tilde{f}_k$ are easily obtained by the FFT, by means of the variation-of-constants formula for (13.28) and the FFT, we propose an algorithm to implement the explicit ERKN integrator determined by (13.12). With the notation $x_j = j\Delta x, t_n = t_0 + nh$, $U_n^j \approx u(x_j, t_n)$, $V_n^j \approx u_t(x_j, t_n)$, $N_T = (T - t_0)/h$ and $K = |a \cdot (-N/2, \cdots, N/2)|^{\intercal}$, this algorithm is stated in Algorithm 1.

Note that in Algorithm 1, we do not directly apply the ERKN formula to thenonlinear system (13.28). That is, the wave equation (13.1) cannot be solved merely in Fourier space, since $\tilde{f}_k(t)$ cannot be directly expressed by $\tilde{u}_k$ for general nonlinear functions $f(u)$. This differs from [32], where the author considered the particular nonlinear case of $f(u) = u^p$ for $p \geqslant 2$. In that case, the wave equation (13.1) can be converted into a nonlinear system for $\tilde{u}_k$ with respect to t, where all the $\tilde{f}_k$ are obtained by the discrete convolution

$$\tilde{f}(u) = \underbrace{u * u * \cdots * u}_{p \text{ times}}, \quad (y * z)_j = \sum_{k+l \equiv j \ mod\ 2N} y_k z_l, \quad j = -N/2, \cdots, N/2. \tag{13.29}$$

That is why we carry out the FFT for each internal stage F_i in Algorithm 1. Here, it is important to note that we consider the general nonlinear function $f(u)$, which is the negative derivative of a potential energy $V(u)$. It is also worth mentioning that for the special case of $c_1 = 0$, we can simplify Step 4, 5, 6 in Algorithm 1 as $F_1 = f(U_{n-1})$, since $U = U_{n-1}$ under this situation.

13.4 The Case of Symplectic ERKN Integrators

As is known, the symplectic structure has many important physical and mathematical consequences, and it is therefore usually important to preserve it if possible.

We have presented Algorithm 1 in the previous section, which is used for the fast implementation of explicit ERKN integrators (13.8) whose coefficients are determined by (13.12). However, taking into account an important class of explicit symplectic ERKN integrators, we can design another fast implementation algorithm apart from Algorithm 1 on the basis of the equivalence between this important class of explicit symplectic ERKN methods and the corresponding splitting methods. The equivalence is stated below, and a similar result can be found in [37], where the author conducted the presentation and the proof in a different manner.

Theorem 13.4 *Let Ψ be an explicit symplectic ERKN method whose coefficients C_i, B_i, $\bar{B}_i$ and A_{ij} are determined by* (13.12)*, and c_i, b_i, $\bar{b}_i$ and a_{ij} satisfy*

$$\begin{cases} \bar{b}_i = b_i(1 - c_i), \\ a_{ij} = b_j(c_i - c_j), \\ c_i = \sum\limits_{k=1}^{i} b_k - \dfrac{1}{2} b_i, \end{cases} \tag{13.30}$$

for all $i, j = 1, \cdots, s$. Then the ERKN method is equivalent to a splitting method (see, e.g. [30])

$$\Phi_h = \varphi^{[1]}_{\alpha_{s+1}h} \circ \varphi^{[2]}_{\beta_s h} \circ \varphi^{[1]}_{\alpha_s h} \circ \cdots \circ \varphi^{[2]}_{\beta_2 h} \circ \varphi^{[1]}_{\alpha_2 h} \circ \varphi^{[2]}_{\beta_1 h} \circ \varphi^{[1]}_{\alpha_1 h}, \tag{13.31}$$

where

$$\begin{cases} \beta_i = b_i, \quad i = 1, \cdots, s, \\ \alpha_1 = \dfrac{1}{2} b_1, \quad \alpha_{s+1} = \dfrac{1}{2} b_s, \quad \alpha_j = \dfrac{1}{2}(b_j + b_{j-1}), \quad j = 2, \cdots, s, \end{cases} \tag{13.32}$$

$\varphi^{[1]}_t$ and $\varphi^{[2]}_t$ respectively denote the exact phase flows of the following first-order systems

$$\begin{cases} q' = p, \\ p' = -Mq, \end{cases} \tag{13.33}$$

and

$$\begin{cases} q' = 0, \\ p' = f(q), \end{cases} \tag{13.34}$$

by denoting $q = y$ and $p = y'$.

Proof We will complete the proof by showing that such a splitting method of the type (13.31) can be equivalently expressed by an ERKN method as stated in the theorem. Since $\varphi_t^{[1]}$ denotes the exact phase flow of (13.33), we then derive from the group property of the exact phase flow that

$$\varphi_{\alpha_1 h}^{[1]} = \varphi_{\beta_1 h/2}^{[1]}, \quad \varphi_{\alpha_{s+1} h}^{[1]} = \varphi_{\beta_s h/2}^{[1]}, \quad \varphi_{\alpha_i h}^{[1]} = \varphi_{\beta_{i+1} h/2}^{[1]} \circ \varphi_{\beta_i h/2}^{[1]}, \quad i = 2, \cdots, s, \tag{13.35}$$

due to the equalities in (13.32). Using the associativity of the combination operation $\circ$, we can write the splitting method Φ_h (13.31) as

$$\Phi_h = \Psi_{\beta_s h} \circ \Psi_{\beta_{s-1} h} \circ \cdots \circ \Psi_{\beta_2 h} \circ \Psi_{\beta_1 h}, \tag{13.36}$$

where $\Psi_{\beta_i h} = \varphi_{b_i h/2}^{[1]} \circ \varphi_{b_i h}^{[2]} \circ \varphi_{b_i h/2}^{[1]}$ for all $i = 1, \cdots, s$.

We now show that $\Psi_{\beta_i h}$ is equivalent to an ERKN method with the stepsize $b_i h$. Let (p_0, q_0) and (p_1, q_1) be initial values and the corresponding numerical solutions after applying $\Psi_{\beta_i h}$ to the initial values, respectively. We can derive the scheme from the formulation of $\Psi_{\beta_i h}$ as follows:

$$\begin{cases} Q_1 = \phi_0\left(\frac{1}{4}b_i^2 V\right) q_0 + \frac{1}{2} b_i h \phi_1\left(\frac{1}{4}b_i^2 V\right) p_0, \\ P_1 = -\frac{1}{2} b_i h M \phi_1\left(\frac{1}{4}b_i^2 V\right) q_0 + \phi_0\left(\frac{1}{4}b_i^2 V\right) p_0, \\ Q_2 = Q_1, \\ P_2 = P_1 + b_i h f(Q_2), \\ q_1 = \phi_0\left(\frac{1}{4}b_i^2 V\right) Q_2 + \frac{1}{2} b_i h \phi_1\left(\frac{1}{4}b_i^2 V\right) P_2, \\ p_1 = -\frac{1}{2} b_i h M \phi_1\left(\frac{1}{4}b_i^2 V\right) Q_2 + \phi_0\left(\frac{1}{4}b_i^2 V\right) P_2, \end{cases} \tag{13.37}$$

where $V \equiv h^2 M$. It follows from the identities

$$\begin{cases} \lambda \phi_0(\kappa^2 V)\phi_1(\lambda^2 V) + \kappa \phi_0(\lambda^2 V)\phi_1(\kappa^2 V) = (\lambda + \kappa)\phi_1((\lambda + \kappa)^2 V), \\ \phi_0(\lambda^2 V)\phi_0(\kappa^2 V) + \lambda\kappa V \phi_1(\kappa^2 V)\phi_1(\lambda^2 V) = \phi_0((\lambda - \kappa)^2 V), \end{cases} \tag{13.38}$$

that the direct calculation by eliminating P_1, P_2, and Q_1 from (13.37) leads to

$$\begin{cases} Q_1 = \phi_0\left(\frac{1}{4}b_i^2V\right)q_0 + \frac{1}{2}b_ih\phi_1\left(\frac{1}{4}b_i^2V\right)p_0, \\ q_1 = \phi_0(b_i^2V)q_0 + b_ih\phi_1(b_i^2V)p_0 + \frac{1}{2}(b_ih)^2\phi_1\left(\frac{1}{4}b_i^2V\right)f(Q_1), \\ p_1 = -b_ihM\phi_1(b_i^2V)q_0 + \phi_0(b_i^2V)p_0 + b_ih\phi_0\left(\frac{1}{4}b_i^2V\right)f(Q_1). \end{cases} \tag{13.39}$$

It can be easily verified that (13.39) is just a particular ERKN method with the stepsize b_ih, whose Butcher tableau reads

$$\begin{array}{c|c} 1/2 & 0 \\ \hline & \phi_1(V/4)/2 \\ \hline & \phi_0(V/4) \end{array}\,. \tag{13.40}$$

Let $(p_{n+1}^{(1)}, q_{n+1}^{(1)}) = \Psi_{\beta_1h}(p_n, q_n)$, $(p_{n+1}^{(i+1)}, q_{n+1}^{(i+1)}) = \Psi_{\beta_{i+1}h}(p_{n+1}^{(i)}, q_{n+1}^{(i)})$ for $i = 1, \cdots, s-1$, and $(p_{n+1}, q_{n+1}) = (p_{n+1}^{(s)}, q_{n+1}^{(s)})$. In consequence, we have $(p_{n+1}, q_{n+1}) = \Phi_h(p_n, q_n)$. In what follows, we aim at showing that the transformation $\Phi_h : (p_n, q_n) \mapsto (p_{n+1}, q_{n+1})$ can be explicitly expressed by an ERKN method, which has exactly the property required in the theorem. To complete the proof, we just need to prove the following proposition by induction on the superscript i.

The mapping $\Psi_{\beta_ih} \circ \Psi_{\beta_{i-1}h} \circ \cdots \circ \Psi_{\beta_2h} \circ \Psi_{\beta_1h} : (p_n, q_n) \mapsto (p_{n+1}^{(i)}, q_{n+1}^{(i)})$ can be expressed by an ERKN scheme as follows

$$\begin{cases} Q_k = \phi_0(c_k^2V)q_n + c_kh\phi_1(c_k^2V)p_n + h^2\sum_{j=1}^{k-1}A_{kj}^{(i)}f(Q_j), \quad k = 1, \cdots, i, \\ q_{n+1}^{(i)} = \phi_0(\mu_i^2V)q_n + \mu_ih\phi_1(\mu_i^2V)p_n + h^2\sum_{k=1}^{i}\bar{B}_k^{(i)}f(Q_k), \\ p_{n+1}^{(i)} = -\mu_ihM\phi_1(\mu_i^2V)q_n + \phi_0(\mu_i^2V)p_n + h\sum_{k=1}^{i}B_k^{(i)}f(Q_k), \end{cases} \tag{13.41}$$

where $\mu_i = \sum_{j=1}^{i} b_i$, $c_i = \mu_i - \frac{b_i}{2}$, $A_{kj}^{(i)} = b_j(c_k - c_j)\phi_1((c_k - c_j)^2V)$, $\bar{B}_k^{(i)} = b_k(\mu_i - c_k)\phi_1((\mu_i - c_k)^2V)$ and $B_k^{(i)} = b_k\phi_0((\mu_i - c_k)^2V)$.

For $i = 1$, (13.41) naturally holds on account of (13.39). Suppose that (13.41) holds for any $i < s$. Then we turn to showing that it also holds for the case of $i + 1$.

On noticing the fact that $(p_{n+1}^{(i+1)}, q_{n+1}^{(i+1)}) = \Psi_{\beta_{i+1}h}(p_{n+1}^{(i)}, q_{n+1}^{(i)})$ and $\Psi_{\beta_{i+1}h}$ is also an ERKN method, it follows from the composition law for ERKN method in [4] that the mapping $\Psi_{\beta_{i+1}h} \circ \Psi_{\beta_i h} \circ \cdots \circ \Psi_{\beta_2 h} \circ \Psi_{\beta_1 h} : (p_n, q_n) \mapsto (p_{n+1}^{(i+1)}, q_{n+1}^{(i+1)})$ really can be expressed in an ERKN method, whose coefficients read

$$\begin{cases} A_{kj}^{(i+1)} = A_{kj}^{(i)}, \quad k = 1, \cdots, i, \ j = 1, \cdots, k-1, \\ A_{i+1,j}^{(i+1)} = \phi_0\left(\dfrac{b_{i+1}^2}{4}V\right)\bar{B}_j^{(i)} + \dfrac{b_{i+1}}{2}\phi_1\left(\dfrac{b_{i+1}^2}{4}V\right)B_j^{(i)}, \quad j = 1, \cdots, i, \\ \bar{B}_j^{(i+1)} = \phi_0(b_{i+1}^2 V)\bar{B}_j^{(i)} + b_{i+1}\phi_1(b_{i+1}^2 V)B_j^{(i)}, \quad j = 1, \cdots, i, \\ \bar{B}_{i+1}^{(i+1)} = \dfrac{b_{i+1}^2}{2}\phi_1\left(\dfrac{b_{i+1}^2}{4}V\right), \\ B_j^{(i+1)} = \phi_0(b_{i+1}^2 V)B_j^{(i)} - b_{i+1}V\phi_1(b_{i+1}^2 V)\bar{B}_j^{(i)}, \quad j = 1, \cdots, i, \\ B_{i+1}^{(i+1)} = b_{i+1}\phi_1\left(\dfrac{b_{i+1}^2}{4}V\right). \end{cases} \tag{13.42}$$

Then with the help of the induction hypothesis and the identities in (13.38), it follows from (13.42) that

$$\begin{cases} A_{kj}^{(i+1)} = b_j(c_k - c_j)\phi_1((c_k - c_j)^2 V), \ k = 1, \cdots, i+1, \ j = 1, \cdots, k-1, \\ \bar{B}_j^{(i+1)} = b_j(\mu_{i+1} - c_j)\phi_1((\mu_{i+1} - c_j)^2 V), \quad j = 1, \cdots, i+1, \\ B_j^{(i+1)} = b_j\phi_0((\mu_{i+1} - c_j)^2 V), \quad j = 1, \cdots, i+1, \end{cases} \tag{13.43}$$

which confirms that the result also holds for the case of $i+1$. Moreover, the consistency of the splitting method means that $\mu_s = \sum_{j=1}^{s} b_j = 1$. By setting $i = s$ in (13.41) we finally conclude that the splitting method in (13.31) can be written as an ERKN method, whose coefficients satisfy (13.12) and (13.30). This completes the proof. □

We set $q = y$, $p = y'$, and then the phase flow $\varphi_h^{[1]}$ and $\varphi_h^{[2]}$ can be respectively expressed as

$$\varphi_h^{[1]} : \ (y_0, y_0') \mapsto (\phi_0(V)y_0 + h\phi_1(V)y_0', \ -hM\phi_1(V)y_0 + \phi_0(V)y_0'), \tag{13.44}$$

and

$$\varphi_h^{[2]} : . \ (y_0, y_0') \mapsto (y_0, \ y_0' + hf(y_0)). \tag{13.45}$$

Algorithm 2 Fast implementation of special symplectic ERKN integrator

```
 1: set $U_0^j = \varphi(x_j)$, $V_0^j = \psi(x_j)$
 2: for $n = 1$ to $N_T$ do
 3:     set $U = U_{n-1}, \quad V = V_{n-1}$
 4:     for $k = 1$ to $s$ do
 5:         FFT:  $U \longrightarrow \widehat{U}, \quad V \longrightarrow \widehat{V}$
 6:         $\widehat{U}_{(0)} = \cos(\alpha_k hK)\widehat{U} + K^{-1}\sin(\alpha_k hK)\widehat{V}$,
            $\widehat{V}_{(0)} = -K\sin(\alpha_k hK)\widehat{U} + \cos(\alpha_k hK)\widehat{V}$
 7:         inverse FFT:  $\widehat{U}_{(0)} \longrightarrow U, \quad \widehat{V}_{(0)} \longrightarrow V$
 8:         $V = V + \beta_k h f(U)$
 9:     end for
10:     FFT:  $U \longrightarrow \widehat{U}, \quad U \longrightarrow \widehat{V}$
11:     $\widehat{U}_{(0)} = \cos(\alpha_{s+1} hK)\widehat{U} + K^{-1}\sin(\alpha_{s+1} hK)\widehat{V}$,
        $\widehat{V}_{(0)} = -K\sin(\alpha_{s+1} hK)\widehat{U} + \cos(\alpha_{s+1} hK)\widehat{V}$
12:     inverse FFT:  $\widehat{U}_{(0)} \longrightarrow U, \quad \widehat{V}_{(0)} \longrightarrow V$
13:     set $U_n = U$, $V_n = V$
14: end for
```

With the help of Theorem 13.4, if we solve (13.44) and (13.45) in the Fourier space with the FFT and IFFT, we can obtain Algorithm 2, which is specially designed for the implementation of the symplectic ERKN integrators stated in Theorem 13.4. Note that the multiplication of the two vectors occurring in both Algorithms 1 and 2 is in the componentwise sense.

13.5 Analysis of Computational Cost and Memory Usage

13.5.1 Computational Cost at Each Time Step

In this section, we focus on the analysis of computational cost at each time step for the three implementation approaches, i.e., the direct calculation approach of (13.14) with matrix-vector multiplication, Algorithm 1 for general ERKN integrators determined by (13.12) and Algorithm 2 for symplectic ERKN integrators that are equivalent to splitting methods. We estimate the computational cost for an explicit ERKN integrator denoted by N_d and N_f, which respectively denote the basic scalar operations (multiplication or addition) and function evaluations of $f(u)$. Note that the multiplication between an N-dimensional matrix-valued function and a corresponding vector contains N^2 basic scalar multiplications and $N(N-1)$ basic scalar additions. Since the FFT (or IFFT) can be accomplished with $\mathscr{O}(N\log N)$ operations for an N-dimensional vector, we assume that $\mathscr{O}(N\log N) = \tilde{C}\cdot N\log N$, where $\tilde{C}$ is a positive constant independent of N.

We assume that the underlying ERKN integrator is of s-stages. The computational cost for each approach is shown in Table 13.1. This shows that the number of function evaluations in one time step is the same for the three different approaches, i.e., s. However, they differ greatly in the number of basic scalar operations. The

Table 13.1 The number of floating point of calculation for the three implementation approaches

Approach	Number	
	N_d	N_f
Direct calculation	$(s^2+7s+6)N^2-(s+2)N$	s
Algorithm 1	$(2s+3)\tilde{C}\cdot N\log N+\left(\frac{3s^2}{2}+\frac{15s}{2}+4\right)N$	s
Algorithm 2	$4(s+1)\tilde{C}\cdot N\log N+(9s+8)N$	s

primary difference is that the cost of the direct calculation approach is $\mathcal{O}(N^2)$, whereas the other two algorithms proposed in this chapter are $\mathcal{O}(N\log N)$, which will sharply decrease the calculation cost once the spatial grid number N isso large that the spatial continuity can be nearly preserved by ERKN integrators, in the sense of numerical computation. A careful observation shows that this advantage essentially derives from the faster calculation of spectral derivatives by the FFT.

We now turn to the comparison between Algorithms 1 and 2. A rough estimate may give that Algorithm 2 takes more basic operations than Algorithm 1, since the coefficient of the dominant part $N\log N$ of the former is $4(s+1)$, which is more than that of the latter, i.e., $2s+3$. However, our numerical simulations in Sect. 13.6 show that for a symplectic ERKN integrator of the type stated in Theorem 13.4, Algorithm 2 consumes less CPU time than Algorithm 1. In order to explain this phenomenon, we make a detailed comparison between the two algorithms as follows.

For a fixed integer N, let $\Theta=\tilde{C}\log N>0$. Then a comparison between the computational cost of the two algorithms reduces to the comparison between $(2s+3)\Theta+\left(\frac{3s^2}{2}+\frac{15s}{2}+4\right)=\frac{3s^2}{2}+\left(\frac{15}{2}+2\Theta\right)s+(4+3\Theta)$ and $4(s+1)\Theta+(9s+8)=(9+4\Theta)s+(8+4\Theta)$. Since s is positive, the only zero point of $\left(\frac{3s^2}{2}+\left(\frac{15}{2}+2\Theta\right)s+(4+3\Theta)\right)-\left((9+4\Theta)s+(8+4\Theta)\right)=\frac{3s^2}{2}-\left(\frac{3}{2}+2\Theta\right)s-(4+\Theta)$ is $s_0=\left(\frac{1}{2}+\frac{2}{3}\Theta\right)+\frac{1}{3}\sqrt{\left(\frac{3}{2}+2\Theta\right)^2+(24+6\Theta)}$. Therefore, the computational cost for Algorithm 2 will be less than that for Algorithm 1 provided $s\geqslant s_0$. On the contrary, Algorithm 2 costs more once $s<s_0$. Here, we list some possible values of s_0 for different Θ in Table 13.2. On noticing that s denotes the stage of ERKN integrator and larger s always implies higher order, we can roughly conclude that Algorithm 2 will be more efficient than Algorithm 1 for high-order symplectic ERKN integrators. Though the constant $\tilde{C}$ of $\mathcal{O}(N\log N)$ cannot be precisely determined, the numerical experiments in the following section confirm that Algorithm 2 consumes less CPU time than Algorithm 1, even for the symplectic ERKN integrator of 3 stages, which clearly supports the higher efficiency of Algorithm 2.

Table 13.2 Values of s_0 for different Θ

Θ	1	2	3	4
s_0	3.3333	4.5465	5.8040	7.0860

Here it is remarked that we are concerned only with explicit ERKN integrators in Algorithm 1. For implicit integrators, a similar analysis can be made, in which iterative solutions are needed in the implementation. Likewise, the computational cost for implicit ERKN integrators will have an order of magnitude $\mathscr{O}(N \log N)$, which is also much smaller than that of direct calculation, i.e., an order of magnitude $\mathscr{O}(N^2)$.

13.5.2 *Occupied Memory and Maximum Number of Spatial Mesh Grids*

For the numerical experiments in Sect. 13.6, the program runs in MATLAB 2012a on a computer Lenovo Yangtian A6860f (CPU: Intel (R) Core (TM) i5-6500 CPU @ 3.20 GHz (4CPUs), Memory: 8 GB, Os: Microsoft Windows 7 with 64bit). Hence, the maximum possible occupied memory is set as 8 GB to avoid memory overflow. Besides, each real number is stored in the double-precision floating-point format, which occupies 8 Bytes of memory.

Concerning the direct calculation procedure, all the coefficients $A_{ij}(V)$, $B_i(V)$, $\bar{B}_i(V)$, $\phi_0(C_i^2V)$, $\phi_1(C_i^2V)$, $\phi_0(V)$ and $\phi_1(V)$ should be calculated and stored in advance. Since N denotes the number of spatial mesh grids, the numerical solutions u_{n+1} and u'_{n+1} are all N-dimensional vectors. This implies that each coefficient of the ERKN integrators will be an $N \times N$ matrix, which needs N times more memory storage than that of u_{n+1} or u'_{n+1}. Thus, we count up the occupied memory by mainly considering the coefficients of the ERKN integrator due to the large magnitude of N. We now estimate the maximum value of N under the environment of 8 GB memory storage for an s-stage explicit ERKN integrator determined by (13.12). It should be noted that both $\phi_0(C_1^2V)$ and $\phi_1(C_1^2V)$ do not need be calculated once $C_1 = 0$ for some explicit ERKN integrators.

For the one-dimensional case of the nonlinear wave equation (13.1), the least required Bytes of memory storage is $8\left(\dfrac{s(s-1)}{2} + 4s\right)N^2 = 4s(s+7)N^2$. This implies that N should satisfy

$$4s(s+7)N^2 \leqslant 8 \times 1024^3, \tag{13.46}$$

which yields

$$N \leqslant N_{\max} = 32768\sqrt{\frac{2}{s(s+7)}}. \tag{13.47}$$

For the two-dimensional case, we admit the assumption that the number N_x of grids in the x-direction equals to the number N_y of grids in the y-direction, i.e., $N_x = N_y$. In this case, u_{n+1} and u'_{n+1} will be $N_x N_y$-dimensional vectors, and V is an $N_x N_y \times N_x N_y$ matrix. Then we can obtain that $N_x (= N_y) \leqslant \sqrt{N_{\max}}$. Likewise, the result $N_x(= N_y = N_z) \leqslant \sqrt[3]{N_{\max}}$ can be obtained in a similar manner under the assumption $N_x = N_y = N_z$.

However, it follows from the description of Algorithms 1 and 2 that the only stored values are u_{n+1}, u'_{n+1} and some other intermediate variables. Thus, we can obtain the estimations as follows:

$$\begin{aligned} &N \leqslant \widetilde{N}_{\max} = \frac{1024^3}{2+\Lambda}, \\ &N_x(= N_y) \leqslant \sqrt{\widetilde{N}_{\max}}, \\ &N_x(= N_y = N_z) \leqslant \sqrt[3]{\widetilde{N}_{\max}}, \end{aligned} \tag{13.48}$$

respectively for the one-dimensional, two-dimensional and three-dimensional cases. Here the positive integer Λ denotes the number of intermediate variables during the implementation of the two algorithms, and $\Lambda = s + 2$ for Algorithm 1 while $\Lambda = 4$ for Algorithm 2.

In Table 13.3, we list some values of $N_{\max}$ (or $\widetilde{N}_{\max}$) for different approaches in all the three-dimensional cases with $s = 4, 10$, and 16. Some points can be concluded from this table. First, the value of $\widetilde{N}_{\max}$ is much larger than that of $N_{\max}$, which indicates that the two algorithms presented in this chapter can admit more dense spatial grids than the direct calculation procedure in order to nearly preserve the spatial continuity. Second, for the one-dimensional case, all the values of $N_{\max}$ and $\widetilde{N}_{\max}$ are larger than 1024, which means that all the three approaches can nearly preserve the spatial continuity with sufficient spatial mesh grids. Third, for the two-dimensional and three-dimensional cases, the direct calculation procedure onlyallows a mesh grid number of which is no more than 100. In particular, for the three-dimensional case, the admissible value of $N_{\max}$ is less than 20. This implies that ERKN integrators can hardly preserve the spatial continuity once the direct calculation procedure is applied. However, for the algorithms presented in this chapter, $\widetilde{N}_{\max}$ is at least of magnitude of 512, which is nearly sufficient for the most nonlinear wave equations to nearly preserve the spatial continuity. Finally, a comparison between Algorithms 1 and 2 shows that the former will occupy a little more memory than the latter.

13.6 Numerical Experiments

In this section, we conduct the numerical experiments with different ERKN integrators in order to show the remarkable efficiency of the algorithms presented

Table 13.3 The value of $N_{\max}$ (or $\widetilde{N}_{\max}$) for different approaches in all the three-dimensional cases with $s = 4, 10, 16$

Approach		Dimension		
		One	Two	Three
$s = 4$	Direct calculation	6986	83	19
	Algorithm 1	128×1024^2	11.31×1024	512
	Algorithm 2	170.67×1024^2	13.06×1024	1.10×512
$s = 10$	Direct calculation	3554	59	15
	Algorithm 1	73.14×1024^2	8.55×1024	0.83×512
	Algorithm 2	170.67×1024^2	13.06×1024	1.10×512
$s = 16$	Direct calculation	2415	49	13
	Algorithm 1	51.20×1024^2	7.16×1024	0.74×512
	Algorithm 2	170.67×1024^2	13.06×1024	1.10×512

in this chapter when applied to nonlinear wave equations. The selected ERKN integrators are as follows:

- ERKN3s4: the 3-stage fourth-order symmetric and symplectic ERKN integrator [4, 38] that can be written as a splitting method ;
- ERKN3s4b: the 3-stage fourth-order ERKN integrator obtained by the RKN method in [31];
- ERKN4s5: the 4-stage fifth-order ERKN integrator obtained by the RKN method in [31];
- ERKN7s6: the 7-stage sixth-order symmetric and symplectic ERKN integrator [38] that can be written as a splitting method;
- ERKN7s6b: the 7-stage sixth-order symplectic ERKN integrator proposed in [4];
- ERKN17s8: the 17-stage eighth-order symmetric and symplectic ERKN integrator derived in [4, 38] that can be written as a splitting method.

We remark that, except for ERKN3s4, ERKN7s6 and ERKN17s8, the other three methods cannot be written as splitting methods. For the three implementation approaches, we respectively use the symbols **D**, **A**, and **F** to denote the direct calculation procedure, Algorithms 1 and 2. For instance, the three implementations of ERKN3s4 are respectively denoted by **D3s4**, **A3s4** and **F3s4**. During the numerical experiments, the numerical solution computed by ERKN16s10 (16-stage ERKN method of order 10 derived in [4]) with sufficiently small stepsize is thought of as the reference solution, when the analytical solution is not available. Note that the Fourier spectral discretisation is used for all the problems considered in this section. Hence, the discrete Hamiltonian energy corresponding to (13.3) has the following form

$$H_n = \frac{1}{2}{u'}_n^{\intercal} u'_n + \frac{1}{2}u_n^{\intercal} M u_n + V(u_n),$$

where M is the spectral differentiation matrix. In this sense, the global Hamiltonian energy error is measured by $\mathrm{GHE}_n = |H_n - H_0|$.

Problem 13.1 (Breather Soliton) We first consider the well-known sine-Gordon equation (see, e.g. [13, 18])

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} - \sin u,$$

on the region $(x, t) \in [-L, L] \times [0, T]$, with the initial conditions

$$u(x, 0) = 0, \quad u_t(x, 0) = 4\kappa\ \mathrm{sech}(\kappa x),$$

and the boundary conditions

$$u(-L, t) = u(L, t) = 4 \arctan\left(c^{-1}\ \mathrm{sech}(\kappa L) \sin(c\kappa t)\right),$$

where $\kappa = 1/\sqrt{1 + c^2}$. The exact solution is given by

$$u(x, t) = 4 \arctan\left(c^{-1}\ \mathrm{sech}(\kappa x) \sin(c\kappa t)\right), \tag{13.49}$$

which is known as the breather solution of the sine-Gordon equation.

We set $L = 40$, $T = 40$ and $c = 0.5$ for this problem in the numerical experiment. We use this problem having the exact solution (13.49) to verify and show that the algorithms presented in this chapter work very well. The global error (GE) results for each ERKN integrator with $N = 512$ and the time stepsize $h = 0.2$ are shown in Fig. 13.1, which are implemented by Algorithm 1. It confirms that these algorithms perform perfectly for this problem. Meanwhile, the CPU times taken by **A3s4**, **A3s4b**, **A4s5**, **A7s6**, **A7s6b**, and **A17s8** are respectively 0.112, 0.114, 0.126, 0.178, 0.184, 0.429 s. These small values really support the theoretical prediction of the high efficiency of Algorithm 1. Furthermore, for the three ERKN integrators that are equivalent to splitting methods, we also implement them by Algorithm 2. The difference between a numerical solution obtained by Algorithm 2 and its counterpart Algorithm 1 is in the magnitude of $\mathscr{O}(10^{-11})$ for all the three ERKN integrators. This fact confirms the consistency between the two algorithms. The CPU times of **F3s4**, **F7s6**, and **F17s8** are respectively 0.105, 0.147, and 0.242 s, which are less than that of Algorithm 1. This fact strongly supports the earlier claim of the higher efficiency of Algorithm 2 than Algorithm 1. In addition, the global Hamiltonian energy errors (GHE) corresponding to Fig. 13.1 are shown in Fig. 13.2. It can be observed from Fig. 13.2 that these ERKN integrators preserve the energy very well.

Note that since the mesh ratio $h/\Delta x = 1.28 > 1$, the finite difference scheme may be numerical unstable for such a large mesh ratio. To illustrate this point, we conduct further numerical experiments with the compact fourth-order centraldifference scheme [39] for the spatial derivative. Meanwhile, we use the 3-stage fourth-order RKN method in [31] to discretise the temporal derivative.

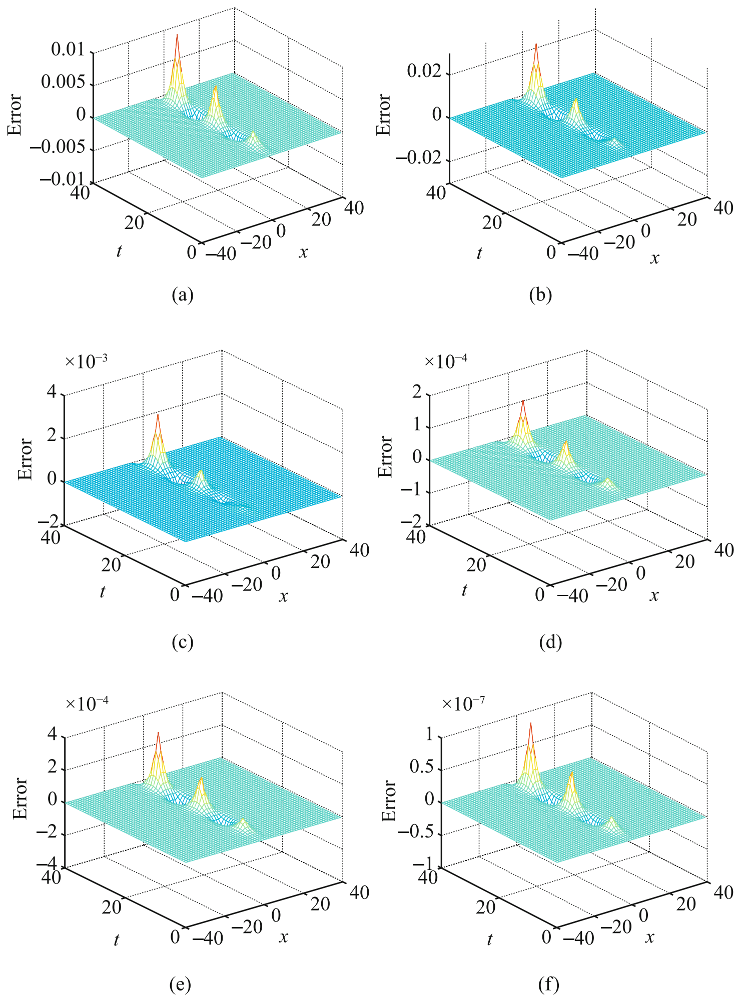

Fig. 13.1 Global errors (GE) of different methods on the region $(x, t) \in [-40, 40] \times [0, 40]$. (**a**) GE of ERKN3s4. (**b**) GE of ERKN3s4b. (**c**) GE of ERKN4s5. (**d**) GE of ERKN7s6. (**e**) GE of ERKN7s6b. (**f**) GE of ERKN17s8

Exactly as we predicted, this method is unstable for these fixed h and Δx due to the numerical overflow of solutions. This fact clearly shows the broader applicability of the algorithms presented in this chapter than the finite difference method in solving nonlinear wave equations, since the former admit large time stepsizes. To numerically check the convergence of the integrators presented in this chapter, we list their global errors at the final time $T = 40$ with different spatial stepsize Δx and time stepsize h in Table 13.4, from which it can be observed that more dense mesh grids indicate smaller global errors. This clearly supports the numerical convergence of the semi-analytical integrators.

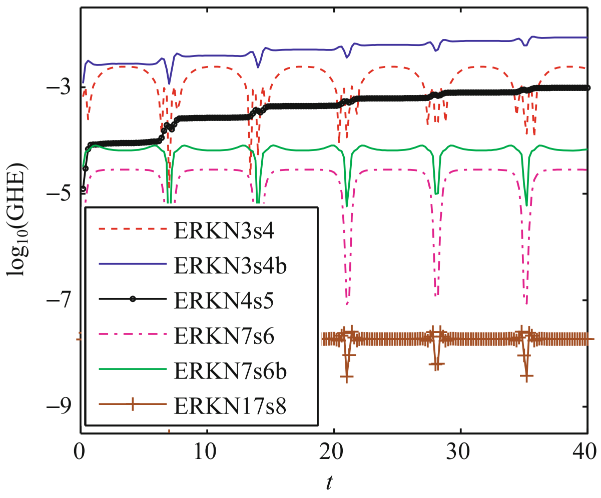

Fig. 13.2 Global Hamiltonian energy errors (GHE) of different methods

Table 13.4 Global errors at the final time $T = 40$ with different mesh grids

$(\Delta x, h)$	(5/4,1/5)	(5/8,1/10)	(5/16,1/20)	(5/32,1/40)
ERKN3s4	4.2905	5.0385×10^{-3}	1.0681×10^{-4}	1.4253×10^{-4}
ERKN3s4b	4.2633	5.0056×10^{-3}	2.1863×10^{-4}	1.6716×10^{-4}
ERKN4s5	4.3140	6.0142×10^{-3}	9.4580×10^{-6}	4.1583×10^{-7}
ERKN7s6	4.3101	5.8759×10^{-3}	8.7842×10^{-7}	2.8212×10^{-9}
ERKN7s6b	4.3104	5.8812×10^{-3}	9.5823×10^{-7}	6.6574×10^{-9}
ERKN17s8	4.3098	5.8720×10^{-3}	8.6836×10^{-7}	1.2319×10^{-11}

Problem 13.2 Consider the nonlinear Klein–Gordon equation (see, e.g. [40, 41])

$$\begin{cases} u_{tt} - u_{xx} + u + u^3 = 0, \quad 0 < x < L, \quad t \in (0, T), \\ u(0, t) = u(L, t), \end{cases}$$

with the periodic boundary condition. The initial conditions are given by

$$u(x, 0) = A\left[1 + \cos\left(\frac{2\pi}{L}x\right)\right], \quad u_t(x, 0) = 0,$$

where $L = 1.28$ and A is the amplitude.

We set $A = 20$ for this problem in the numerical experiment. Such a large amplitude makes this problem challenging for its numerical solution (see, e.g. [40, 41]), since the solution will have an abrupt change in both time and space directions. This phenomenon can be observed from Fig. 13.3, where we plot the reference numerical solution in the region $(x, t) \in [0, 1.28] \times [0, 10]$. To show the

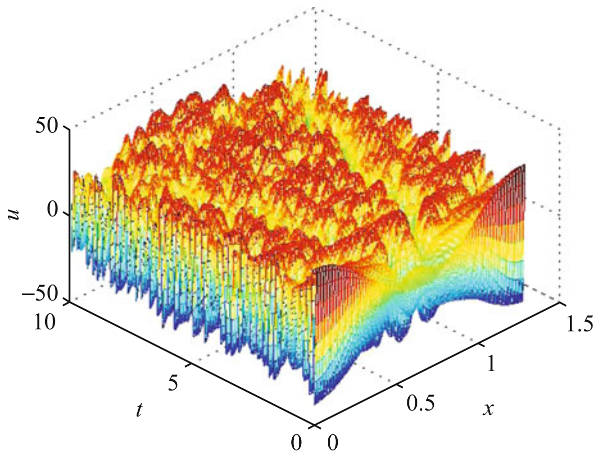

Fig. 13.3 Reference solution with ERKN10

effectiveness of the ERKN integrators, we plot the global errors of these methods in Fig. 13.4 with the time stepsize $h = 0.001$. It can be observed from Fig. 13.4 that ERKN integrators really can solve this problem with some accuracy. It is noted that, the difference between numerical solutions obtained from the three implementation approaches for each ERKN integrator are of magnitude $\mathscr{O}(10^{-12})$, which confirms that the two algorithms presented in this chapter are promising. The global Hamiltonian energy errors corresponding to Fig. 13.4 are presented in Fig. 13.5, which show the good preservation of the ERKN integrator presented in this chapter.

To compare the efficiency of the three implementation approaches, we carry out these methods with different numbers N of spatial grid points and the fixed time stepsize $h = 0.001$. The numerical results for the consumed CPU time are shown in Tables 13.5, and 13.6 indicates the detailed ratio of CPU time of the direct calculation procedure to that of Algorithm 1 (or Algorithm 2). It is clear from the two tables that the two algorithms cost much less time than the direct calculation procedure for all the six ERKN integrators. This fact supports the theoretical analysis that the algorithms described in this chapter really can nearly preserve the spatial continuity with a large number N of spatial grid points and reasonable computational cost. In Table 13.6, for the underlying ERKN methods larger N always implies a larger ratio, which confirms that the superiority of these algorithms over the direct calculation procedure will be more marked for a larger grid number N. Moreover, the comparison between Algorithms 1 and 2 in the two tables shows that Algorithm 2 always consumes less CPU time than Algorithm 1. In particular, for a fixed N, Algorithm 2 becomes more efficient (larger ratio in Table 13.6) than Algorithm 1 for ERKN methods of higher order (hence with more stages, i.e., a bigger s).

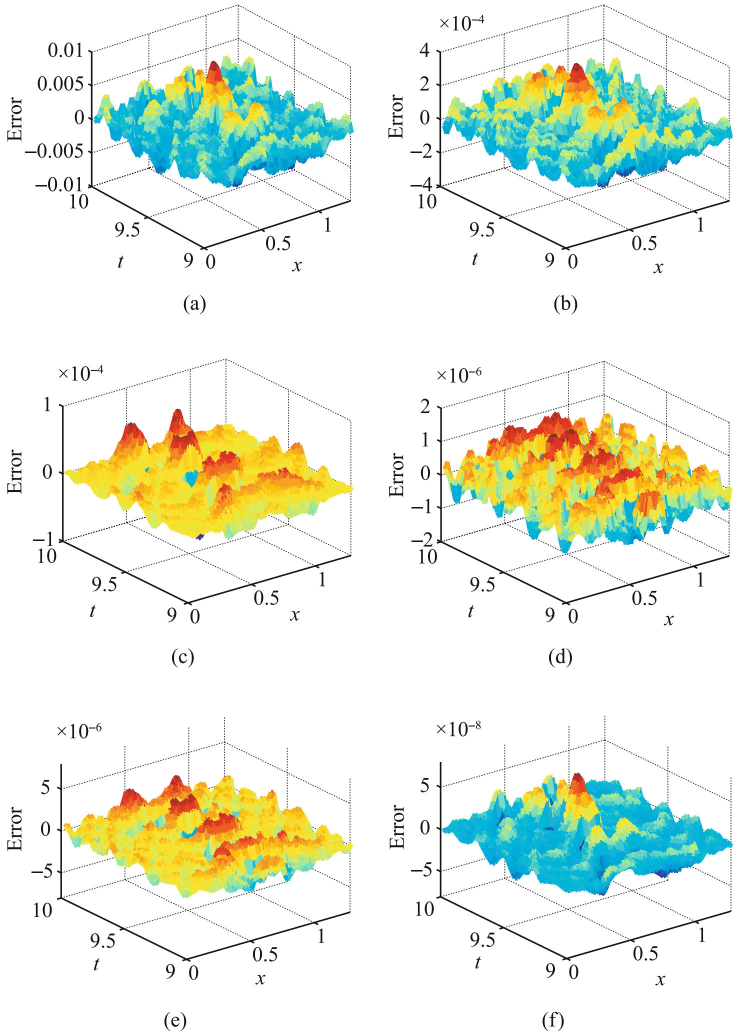

Fig. 13.4 Global errors (GE) of different methods on the region $(x, t) \in [0, 1.28] \times [9, 10]$. (**a**) GE of ERKN3s4 with $N = 2^7$. (**b**) GE of ERKN3s4b with $N = 2^8$. (**c**) GE of ERKN3s5 with $N = 2^8$. (**d**) GE of ERKN7s6 with $N = 2^8$. (**e**) GE of ERKN7s6b with $N = 2^8$. (**f**) GE of ERKN17s8 with $N = 2^9$

Problem 13.3 We then consider the two-dimensional sine-Gordon equation

$$u_{tt} - (u_{xx} + u_{yy}) = -\sin(u), \quad t > 0$$

with the homogeneous Neumann boundary condition

$$u_x(\pm 14, y, t) = 0, \quad u_y(x, \pm 14, t) = 0$$

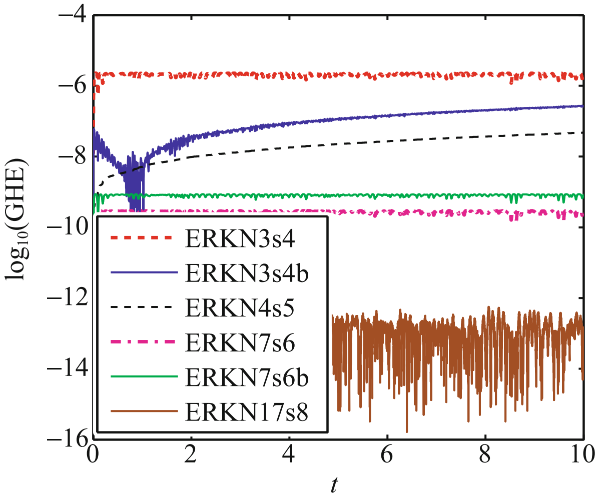

Fig. 13.5 Global energy errors (GHE) of different methods corresponding to Fig. 13.4

Table 13.5 CPU time (seconds) consumed by each method for different N with the time stepsize $h = 0.001$

	$N = 64$	$N = 128$	$N = 256$	$N = 512$
D3s4	3.180	8.807	39.573	241.768
A3s4	1.266	2.144	3.626	6.286
F3s4	0.935	1.706	3.106	5.569
D3s4b	2.534	10.096	38.341	242.901
A3s4b	1.241	2.147	3.569	6.337
D4s5	2.796	12.767	46.291	310.244
A4s5	1.487	2.792	4.344	7.432
D7s6	7.228	25.079	83.279	568.355
A7s6	2.374	4.347	6.665	11.645
F7s6	1.714	3.222	5.571	10.422
D7s6b	7.236	24.720	70.033	574.093
A7s6b	2.366	4.319	6.685	11.857
D17s8	30.047	102.216	319.589	2093.492
A17s8	6.663	10.357	16.204	29.152
F17s8	3.616	6.665	11.654	21.877

in the region $(x, y) \in [-14, 14] \times [-14, 14]$. The initial conditions are given by

$$u(x, y, 0) = 4\arctan\left(\exp\left(3 - \sqrt{x^2 + y^2}\right)\right), \quad u_t(x, y, 0) = 0,$$
$$(x, y) \in [-14, 14] \times [-14, 14].$$

The solutions of this problem are circular ring solitons (see, e.g. [18, 42, 43]).

For solving this problem, the eighth-order integrator ERKN17s8 implemented by **F17s8** is used to make a comparison with the method GLC4 in [18]. Note that

Table 13.6 Ratio of CPU time of direct calculation approach to that of the two algorithms

	$N = 64$	$N = 128$	$N = 256$	$N = 512$
D3s4/A3s4	2.5118	4.1077	10.9137	38.4613
D3s4/F3s4	3.4011	5.1624	12.7408	43.4132
D3s4b/A3s4b	2.0419	4.7024	10.7428	38.3306
D4s5/A4s5	1.8803	4.5727	10.6563	41.7443
D7s6/A7s6	3.0447	5.7693	12.4950	48.8068
D7s6/F7s6	4.2170	7.7837	14.9487	54.5342
D7s6b/F7s6b	3.0583	5.7235	10.761	48.4181
D17s8/A17s8	4.5085	9.8693	19.7228	71.8130
D17s8/F17s8	8.3095	15.3362	27.4231	95.6937

GLC4 is also of order eight. Here, we select the same time stepsize $h = 0.1$ and mesh region size 400×400 as those in [18]. Numerical results of $\sin(u/2)$ and the corresponding contour plots at the time points $t = 0,\ 4,\ 8,\ 11.5$, 13, and 15 are shown in Figs. 13.6 and 13.7, which are nearly the same as the corresponding figures in [18]. This shows the effectiveness of **F17s8** in solving this problem. In particular, for $t = 15$ the CPU time of **F17s8** is only 38.67 s, which is much less than that of GLC4, i.e. 668.05 s (see [18]). This fact again gives a further support for the high efficiency of the algorithms presented in this chapter. Finally, we display the good energy conservation of ERKN17s8 in Fig. 13.8.

With regard to the formulation and analysis of energy-preserving schemes for Klein–Gordon equations, see Chap. 9 (see also [44]). The framework of semi-analytical integrators for solving partial differential equations was initially proposed in [10], and this chapter focuses on the efficient implementation issue of the semi-analytical integrators.

13.7 Conclusions and Discussions

Although there has been far less numerical treatment of PDEs in the structure-preserving literature than that of ODEs, the recent growth of geometric integration for nonlinear Hamiltonian PDEs has led to the development of numerical schemes which systematically incorporate qualitative features of the underlying problem into their structure. In general, the qualitative characteristics of structure-preserving integrators are mainly concerned with the symmetry, the symplecticity, the multi-symplecticity, the conservation of energy or first integrals, the high oscillation or stiffness, and so on (see, e.g. [30, 45–47]). In this chapter, by presenting a class of semi-analytical ERKN integrators and their implementation approaches for solving nonlinear wave equations, we incorporated the spatial continuity into the structure-preserving property as well. These ERKN integrators can nearly preserve both the spatial continuity and the high oscillation of the original problem, in theory. In order to effectively realize the ERKN integrators on a computer, we presented

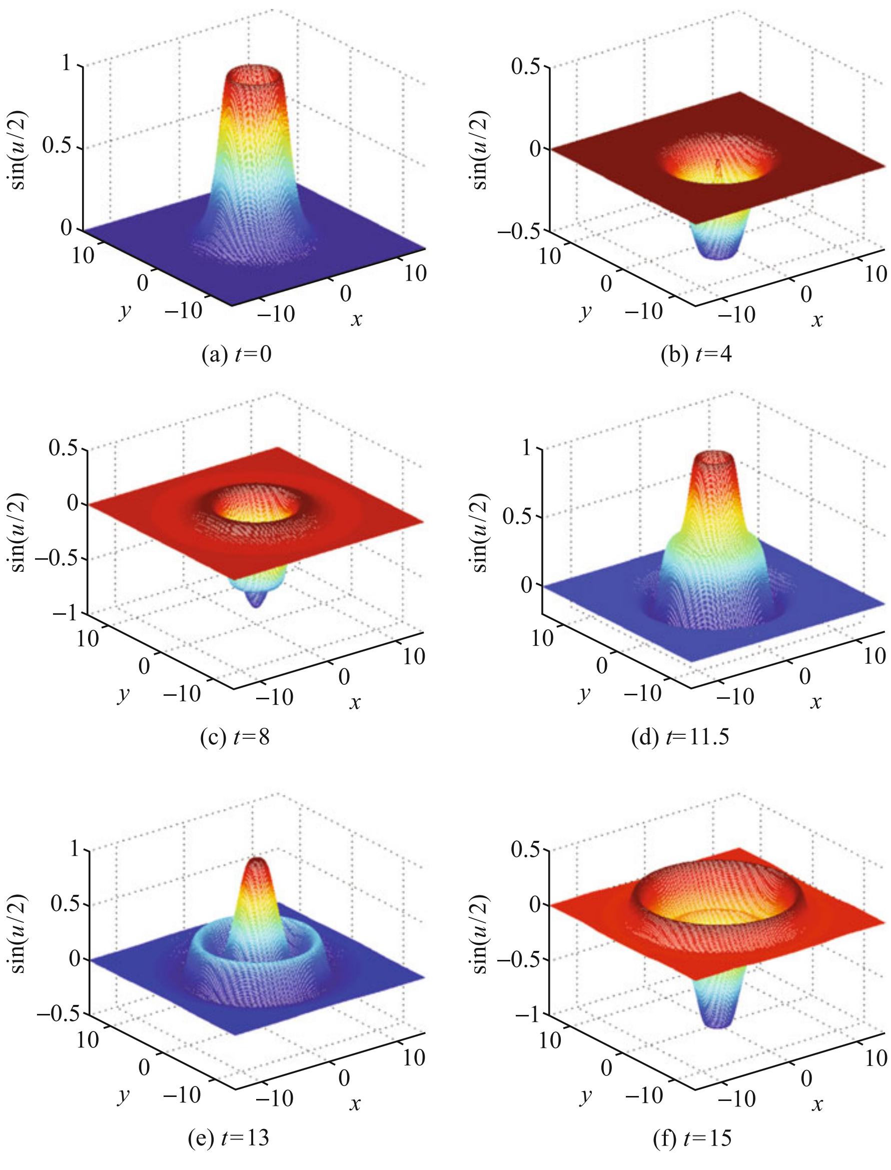

Fig. 13.6 Circular ring solitons: the function of $\sin(u/2)$ at the time $t = 0, 4, 8, 11.5, 13$ and 15

two algorithms accompanied with FFT technique, besides the direct calculation procedure. It follows from the detailed analysis of the computational cost and the memory storage that the algorithms presented in this chapter possess the superiority of smaller computational cost and lower memory storage over the direct calculation procedure. In particular, for nonlinear wave equations of high dimension, the direct

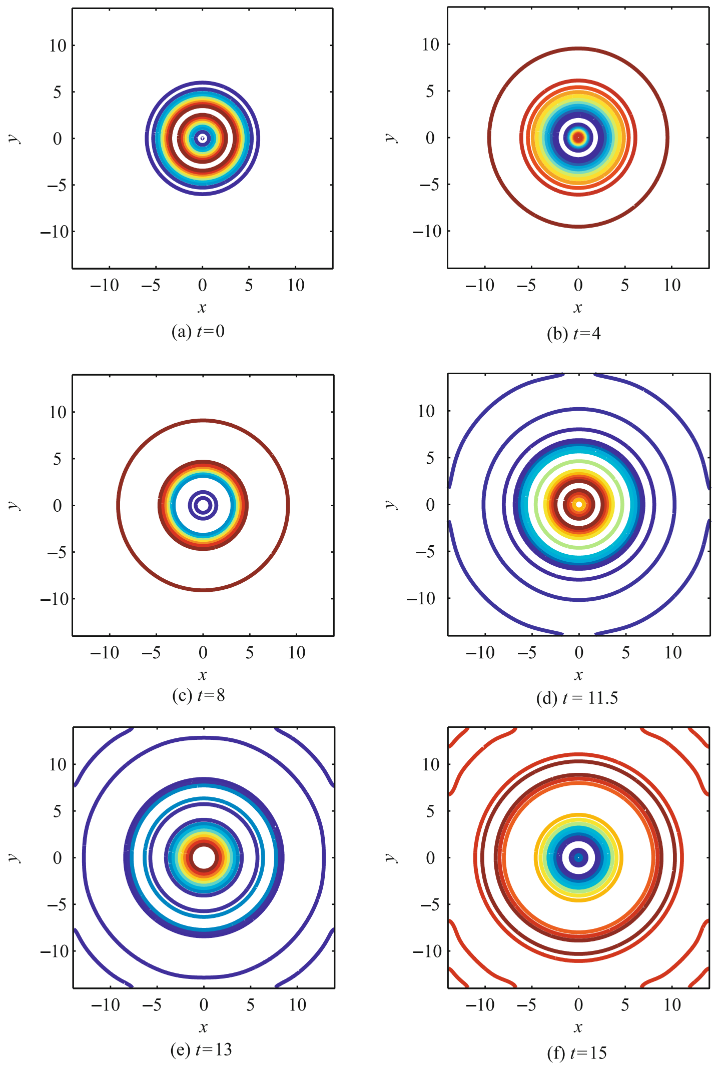

Fig. 13.7 Circular ring solitons: contours of $\sin(u/2)$ at the time $t = 0, 4, 8, 11.5, 13$ and 15

calculation procedure could hardly be used to preserve the spatial continuity, due to the crucial restriction on the dimension of the spatial grid points, while the two algorithms, Algorithms 1 and 2, do not suffer from this trouble, due to the larger maximum number of admissible spatial grid point. Moreover, Algorithm 2, which is suitable for important symplectic ERKN integrators for Hamiltonian systems, is a bit more efficient than Algorithm 1. Finally, we conducted numerical experiments including one-dimensional and two-dimensional wave equations in Sect. 13.6, and the numerical results show strong support for the theoretical analysis

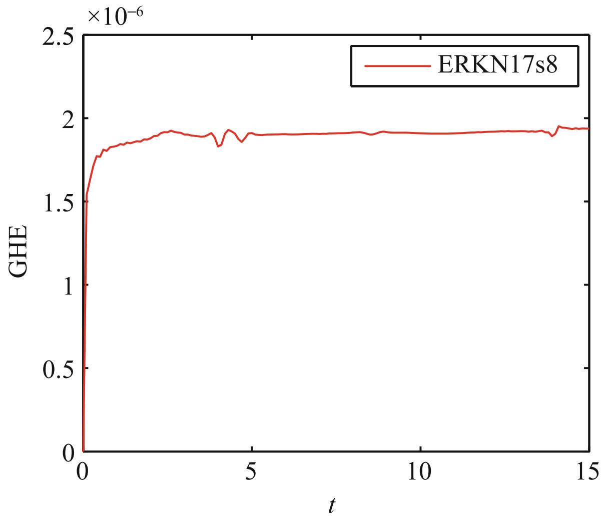

Fig. 13.8 Global energy errors (GHE) of ERKN17s8

presented in this chapter. The numerical results also demonstrate an important fact that, in comparison with the finite difference method, the algorithms presented in this chapter, which can nearly preserve both the spatial continuity and the high oscillation, are robust and efficient when applied to nonlinear wave equations.

It is noted that for nonlinear wave equations equipped with other boundary conditions, such as the homogeneous Dirichlet or Neumann boundary conditions, we can also design such kind of efficient algorithms by just replacing the FFT with the Discrete Fast Cosine/Sine Transform, once the Fourier spectral discretisation is replaced by the cosine/sine scheme [18, 42, 43]. Furthermore, for general boundary conditions, the Chebyshev spectral discretisation accompanied with the FFT technique is also possible, and a further research in this area is needed.

The material in this chapter is based on the work by Mei et al. [48].

References

1. Wu, X., Mei, L., Liu, C.: An analytical expression of solutions to nonlinear wave equations in higher dimensions with Robin boundary conditions. J. Math. Anal. Appl. **426**, 1164–1173 (2015)
2. Liu, C., Wu, X.: The boundness of the operator-valued functions for multidimensional nonlinear wave equations with applications. Appl. Math. Lett. **74**, 60–67 (2017)
3. Liu, C., Wu, X.: An energy-preserving and symmetric scheme for nonlinear Hamiltonian wave equations. J. Math. Anal. Appl. **440**, 167–182 (2016)
4. Mei, L., Wu, X.: The construction of arbitrary order ERKN methods based on group theory for solving oscillatory Hamiltonian systems with applications. J. Comput. Phys. **323**, 171–190 (2016)

5. Wang, B., Iserles, A., Wu, X.: Arbitrary-order trigonometric Fourier collocation methods for multi-frequency oscillatory systems. Found. Comput. Math. **16**, 151–181 (2016)
6. Wu, X., Liu, K., Shi, W.: Structure-Preserving Algorithms for Oscillatory Differential Equations II. Springer, Heidelberg (2015)
7. Wu, X., Wang, B.: Recent Developments in Structure-Preserving Algorithms for Oscillatory Differential Equations. Springer Nature Singapore Pte Ltd., Singapore (2018)
8. Wu, X., You, X., Wang, B.: Structure-Preserving Algorithms for Oscillatory Differential Equations. Springer, Berlin (2013)
9. Jung, C., Nguyen, T.B.: Semi-analytical time differencing methods for stiff problems. J. Sci. Comput. **63**, 355–373 (2015)
10. Wu, X., Liu, C., Mei, L.: A new framework for solving partial differential equations using semi-analytical explicit RK(N)-type integrators. J. Comput. Appl. Math. **301**, 74–90 (2016)
11. Boyd, J.P.: Chebyshev and Fourier Spectral Methods. 2nd edn. Dover Publications, New York (2013)
12. Trefethen, L.N.: Spectral Methods in MATLAB. SIAM, Philadelphia (2000)
13. Mei, L., Liu, C., Wu, X.: An essential extension of the finite-energy condition for extended Runge-Kutta-Nyström integrators when applied to nonlinear wave equations. Commun. Comput. Phys. **22**, 742–764 (2017)
14. Cooley, J., Tukey, J.: An algorithm for the machine calculation of complex Fourier series. Math. Comput. **19**, 297–301 (1965)
15. Bao, W., Dong, X.: Analysis and comparison of numerical methods for the Klein-Gordon equation in the nonrelativistic limit regime. Numer. Math. **120**, 189–229 (2012)
16. Lax Peter, D.: Hyperbolic Partial Differential Equations. American Mathematical Society/Courant Institute of Mathematical Sciences, New York (2006)
17. Morton, K.W., Mayers, D.F.: Numerical Solution of Partial Differential Equations, 2nd edn. Cambridge University, Cambridge (2005)
18. Liu, C., Wu, X.: Arbitrarily high-order time-stepping schemes based on the operator spectrum theory for high-dimensional nonlinear Klein-Gordon equations. J. Comput. Phys. **340**, 243–275 (2017)
19. Bao, W., Cai, Y., Zhao, X.: A uniformly accurate multiscale time integrator pseudospectral method for the Klein-Gordon equation in the nonrelativistic limit regime. SIAM J. Numer. Anal. **52**, 2488–2511 ((2014))
20. Caliari, M., Zuccher, S.: Reliability of the time splitting Fourier method for singular solutions in quantum fluids. Comput. Phys. Commun. **222**, 46–58 (2018)
21. Cao, W., Guo, B.: Fourier collocation method for solving nonlinear Klein-Gordon equation. J. Comput. Phys. **108**, 296–305 (1993)
22. Cheng, K., Feng, W., Gottlieb, S., et al.: A Fourier pseudospectral method for the "good" boussinesq equation with second-order temporal accuracy. Numer. Methods Partial Differ. Equ. **31**, 202–224 (2015)
23. Shen, J., Tang, T., Wang, L.: Spectral Methods: Algorithms, Analysis and Applications. Springer, Berlin (2011)
24. Dehghan, M., Mohebbi, A., Asgari, Z.: Fourth-order compact solution of the nonlinear Klein-Gordon equation. Numer. Algor. **52**, 523–540 (2009)
25. Dehghan, M., Shokri, A.: A numerical method for solution of the two-dimensional sine-Gordon equation using the radial basis functions. Math. Comput. Simul. **79**, 700–715 (2008)
26. Dehghan, M., Shokri, A.: Numerical solution of the nonlinear Klein-Gordon equation using radial basis functions. J. Comput. Appl. Math. **230**, 400–410 (2009)
27. Mirzaei, D., Dehghan, M.: Boundary element solution of the two-dimensional sine-Gordon equation using continuous linear elements. Eng. Anal. Bound. Elem. **33**, 12–24 (2009)
28. Moghaderi, H., Dehghan, M.: A multigrid compact finite difference method for solving the one-dimensional nonlinear sine-Gordon equation. Math. Method. Appl. Sci. **38**, 3901–3922 (2015)
29. Taleei, A., Dehghan, M.: A pseudo-spectral method that uses an overlapping multidomain technique for the numerical solution of sine-Gordon equation in one and two spatial dimensions. Math. Method. Appl. Sci. **37**, 1909–1923 (2014)

30. Hairer, E., Lubich, C., Wanner, G.: Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations, 2nd edn. Springer, Berlin (2006)
31. Hairer, E., Nørsett, S.P., Wanner, G.: Solving Ordinary Differential Equations I: Nonstiff Problems, 2nd edn. Springer, Berlin (1993)
32. Gauckler, L.: Error analysis of trigonometric integrators for semilinear wave equations. SIAM J. Numer. Anal. **53**, 1082–1106 (2015)
33. Liu, C., Iserles, A., Wu, X.: Symmetric and arbitrarily high-order Birkhoff-Hermite time integrators and their long-time behavior for solving nonlinear Klein-Gordon equations. J. Comput. Phys. 356, 1–30 (2018)
34. Hochbruck, M., Ostermann, A.: Exponential integrators. Acta Numer. **19**, 209–286 (2010)
35. Stein, E., Shakarchi, R.: Fourier Analysis: An Introduction, Princeton Lectures in Analysis, vol 1. Princeton University, Princeton, NJ (2003)
36. Zygmund, A.: Trigonometric Series, 3rd edn. Cambridge University, Cambridge (2002)
37. Blanes, S.: Explicit symplectic RKN methods for perturbed non-autonomous oscillators: splitting, extended and exponentially fitting methods. Comput. Phys. Commun. **21**, 10–18 (2015)
38. Liu, K., Wu, X.: High-order symplectic and symmetric composition methods for multifrequency and multi-dimensional oscillatory Hamiltonian systems. J. Comput. Math. **33**, 356–378 (2015)
39. Lele, S.K.: Compact finite difference schemes with spectral-like resolution. J. Comput. Phys. **103**, 16–42 (1992)
40. Jiménez, S., Vázquez, L.: Analysis of four numerical schemes for a nonlinear Klein-Gordon equation. Appl. Math. Comput. **35**, 61–94 (1990)
41. Wang, Y., Wang, B.: High-order multi-symplectic schemes for the nonlinear Klein-Gordon equation. Appl. Math. Comput. **166**, 608–632 (2005)
42. Bratsos, A.G.: The solution of the two-dimensional sine-Gordon equation using the method of lines. J. Comput. Appl. Math. 206, 251–277 (2007)
43. Sheng, Q., Khaliq, A.Q.M., Voss, D.A.: Numerical simulation of two-dimensional sine-Gordon solitons via a split cosine scheme. Math. Comput. Simul. **68**, 355–373 (2005)
44. Wang, B., Wu, X.: The formulation and analysis of energy-preserving schemes for solving high-dimensional nonlinear Klein-Gordon equations. IMA J. Numer. Anal. **39**, 2016–2044 (2019)
45. Mei, L., Wu, X.: Symplectic exponential Runge-Kutta methods for solving nonlinear Hamiltonian systems. J. Comput. Phys. **338**, 567–584 (2017)
46. Okunbor, D.I., Skeel, R.D.: Canonical Runge-Kutta-Nyström methods of orders five and six. J. Comp. Appl. Math. **51**, 375–382 (1994)
47. Sanz-Serna, J.M., Calvo, M.P.: Numerical Hamiltonian Problems. Chapman and Hall, London (1994)
48. Mei, L., Li, H., Wu, X., et al.: Semi-analytical exponential RKN integrators for efficiently solving high-dimensional nonlinear wave equations based on FFT techniques. Comput. Phys. Commun. **243**, 68–80 (2019)