JOURNAL OF COMPUTATIONAL PHYSICS 93, 287–296 (1991)

Runge–Kutta Smoother for Suppression of Computational-Mode Instability of Leapfrog Scheme

AKIRA AOYAGI

Kyushu Sangyo University, Faculty of Engineering, 3-1, Matsugadai 2-chome, Higashi-ku, Fukuoka 813, Japan

AND

KANJI ABE

The University of Tokyo, College of Arts and Sciences, Komaba 3-8-1, Meguro-ku, Tokyo 153, Japan

Received June 7, 1989; revised November 14, 1989

The Runge–Kutta smoother is applied to suppress nonlinear numerical instabilities in the leapfrog scheme for time integration of the Korteweg–de Vries equation. The accuracy of integration is compared with that by the use of the second order smoother. The Runge–Kutta smoother enables us to make long-time integration of the Korteweg–de Vries equation for large amplitudes. © 1991 Academic Press, Inc.

1. INTRODUCTION

The leapfrog scheme has extensively been used for time integration of nonlinear partial differential equations of hyperbolic type. The scheme has the merit of being simple and free from dissipation errors. However, when we use the leapfrog scheme to integrate equations including no dissipative term, such as the nonlinear convective equation or the Korteweg–de Vries (K-dV) equation, the scheme often suffers from some numerical instabilities [1–4].

In the previous paper [4], we studied the nonlinear numerical instability in the leapfrog scheme applied to the K-dV equation. It was shown that the instability comes from parametric excitation of computational modes of wave. In order to carry out long-time integration of the K-dV equation, the previous paper proposed the Runge–Kutta smoother to suppress the instability.

This paper shows experimentally that the Runge–Kutta smoother combined with the leapfrog scheme by Zabusky and Kruskal [5] enables us to make long-time integration of the K-dV equation for large amplitudes.

0021-9991/91 $3.00

2. Observation of Numerical Instability

We consider the K-dV equation as in [4]:

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+\mu\frac{\partial^3 u}{\partial x^3}=0, \tag{1}$$

where μ is constant. We solve Eq. (1) under the periodic boundary condition

$$u(x, t)=u(x+2, t) \tag{2}$$

and the initial condition

$$u(x, 0)=A\cos(\pi x), \tag{3}$$

where A is the amplitude. We adopt the leapfrog scheme by Zabusky and Kruskal [5]:

$$u_j^{n+1}=u_j^{n-1}-\frac{\Delta t}{3\,\Delta x}(u_{j+1}^n+u_j^n+u_{j-1}^n)(u_{j+1}^n-u_{j-1}^n)$$

$$-\frac{\mu\,\Delta t}{(\Delta x)^3}(u_{j+2}^n-2u_{j+1}^n+2u_{j-1}^n-u_{j-2}^n), \tag{4}$$

where $u_j^n\equiv u(x=j\,\Delta x,\ t=n\,\Delta t)$, Δx is the spatial increment, and Δt is the temporal increment. The periodic boundary condition (2) becomes

$$u_j^n=u_{j+2J}^n,$$

where $2J\,\Delta x=2$, and the initial condition (3) becomes

$$u_j^0=A\cos(\pi j\,\Delta x).$$

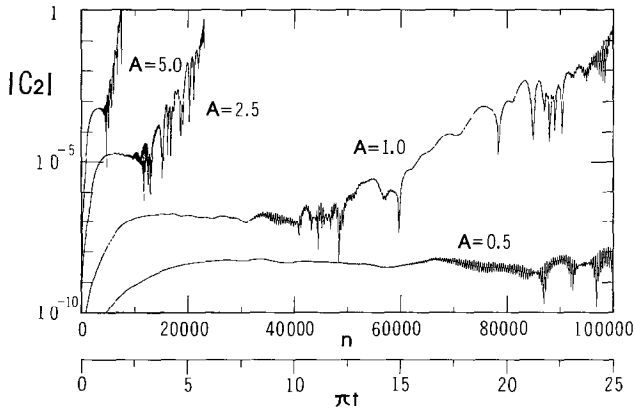

FIG. 1. Invariants $|C_2|$ as functions of time step n or time t for $A=0.5$, 1.0, 2.5, and 5.0; $\mu=0.022^2$; $\Delta x=2/400$; $\pi\,\Delta t=2.5\times10^{-4}$ in Eq. (4).

TABLE I

Blowup and Near-Recurrence Times

Amplitude A	5.0	2.5	1.0	0.5
Blowup time (πt_b)	1.90	5.88	25.3	75.4
Near-recurrence time (πt_r)	13.8	19.3	30.2	42.5

We make the numerical integrations based on scheme (4) fixing $\mu = 0.022^2$, $\Delta x = 2/400$, and $\pi\,\Delta t = 2.5 \times 10^{-4}$ for $A = 0.5$, 1.0, 2.5, and 5.0. The temporal increment given above satisfies the linear stability condition (9) given later. The accuracy of integrations is checked by observing the conservation law of the K-dV equation (1):

$$C_2 \equiv \int_0^2 u(x, t)^2\,dx - A^2$$
$$= \Delta x \sum_{j=0}^{2J-1} (u_j^n)^2 - A^2 = 0. \tag{5}$$

All of the computations are made by using numbers of 16 figures (double precision).

Figure 1 gives the values of $|C_2|$ as functions of time step n or time t. We see from the figure that after some time step $|C_2|$ exponentially grows without bound. The growth rate becomes larger with the amplitude A. We define the blowup time t_b at which $|C_2|$ exceeds A^2. The numerical solution obtained by the leapfrog scheme (4) for $t > t_b$ has no meaning. Table I gives the blowup time t_b and the near-recurrence time t_r whose definition will be given later.

Figure 2 gives the magnified curve of C_2 in Fig. 1 for $A = 1.0$ and $n = 59{,}900 \sim 60{,}100$. The value of C_2 changes its sign at each time step.

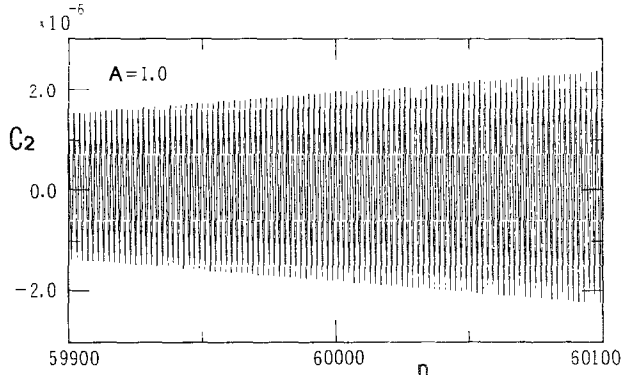

FIG. 2. Saw-toothed oscillation of C_2 corresponding to Fig. 1 for $A = 1.0$, and $n = 59{,}900 \sim 60{,}100$.

3. Physical and Computational Modes

In order to explain the saw-toothed oscillation of C_2, let us start from the linear stability analysis of scheme (4). For rough estimation of the linear stability we replace $(u_{j+1}^n + u_j^n + u_{j-1}^n)/3$ in the second term of the right-hand side of Eq. (4) by a constant c and expand u_j^n in terms of the discrete Fourier coefficients U_k^n:

$$u_j^n = \sum_{k=-J}^{J-1} U_k^n \exp\left(\frac{i\pi kj}{J}\right). \tag{6}$$

Thus we obtain from the linear version of scheme (4)

$$U_k^{n+1} - 2i\,\Delta t\left[\omega_k - \frac{c}{\Delta x}\sin(\pi k\,\Delta x)\right] U_k^n - U_k^{n-1} = 0, \tag{7}$$

where

$$\omega_k \equiv \frac{2\mu}{(\Delta x)^3}\sin(\pi k\,\Delta x)[1-\cos(\pi k\,\Delta x)].$$

The solution of Eq. (7) has the form of $U_k^n \propto \exp(in\theta_k)$, where

$$\sin\theta_k = \Delta t\left[\omega_k - \frac{c}{\Delta x}\sin(\pi k\,\Delta x)\right]. \tag{8}$$

The condition that θ_k is real or $|\sin\theta_k| \leqslant 1$ leads to the linear stability condition for scheme (4). The condition $|\sin\theta_k| \leqslant 1$ is satisfied if

$$\Delta t\left[\max|\omega_k| + \max\left|\frac{c}{\Delta x}\sin(\pi k\,\Delta x)\right|\right] \leqslant 1$$

or

$$\frac{\Delta t}{\Delta x}\left[|c| + \frac{4\mu}{(\Delta x)^2}\right] \leqslant 1. \tag{9}$$

Inequality (9) is the severe stability condition for Δt. When inequality (9) is satisfied, Eq. (8) has two solutions of θ_k. If we write one of them as θ_k, the other is $\pi - \theta_k$. Then the solution of Eq. (7) is written as

$$\begin{aligned} U_k^n &= V_k \exp(in\theta_k) + W_k \exp[in(\pi-\theta_k)] \\ &= V_k \exp(in\theta_k) + (-1)^n W_k \exp(-in\theta_k), \end{aligned} \tag{10}$$

where the coefficients V_k and W_k are constants independent of n. By substituting Eq. (10) into Eq. (6), we can express u_j^n as

$$u_j^n = v_j^n + (-1)^n w_j^n, \tag{11}$$

where

$$\begin{bmatrix} v_j^n \\ w_j^n \end{bmatrix} = \sum_{k=-J}^{J-1} \begin{bmatrix} V_k^n \\ W_k^n \end{bmatrix} \exp\left(\frac{i\pi kj}{J}\right),$$

$$V_k^n = V_k \exp(in\theta_k), \qquad W_k^n = W_k \exp(-in\theta_k).$$

We call v_j^n as physical modes and $(-1)^n w_j^n$ as computational modes. When Δt approaches zero, physical modes converge to the physical solution and computational modes vanish. Note that computational modes change their signs at each time step n.

In view of the linear solution (11), we may decompose also solutions u_j^n of the nonlinear equation (4) into the physical modes v_j^n and the computational modes $(-1)^n w_j^n$ as in Eq. (11). The saw-toothed oscillation in Fig. 2 comes from the computational modes $(-1)^n w_j^n$ in Eq. (11).

4. Runge–Kutta Smoother

The simplest way [6] to exclude the computational modes is to evalutate $u_j^{n+1} + 2u_j^n + u_j^{n-1}$ from Eq. (11) as

$$u_j^{n+1} + 2u_j^n + u_j^{n-1} = v_j^{n+1} + 2v_j^n + v_j^{n-1} - (-1)^n(w_j^{n+1} - 2w_j^n + w_j^{n-1})$$

and approximate $v_j^{n+1} + 2v_j^n + v_j^{n-1} \simeq 4v_j^n$ and $w_j^{n+1} - 2w_j^n + w_j^{n-1} \simeq 0$. Then we obtain

$$v_j^n = (u_j^{n+1} + 2u_j^n + u_j^{n-1})/4. \tag{12}$$

Replacing n by $n-1$ in the above equation, we obtain

$$v_j^{n-1} = (u_j^n + 2u_j^{n-1} + u_j^{n-2})/4.$$

We can restart the leapfrog integration based on Eq. (4) using v_j^n and v_j^{n-1} in place of u_j^n and u_j^{n-1} at some time step. This replacement operation, however, violates the conversation law of C_2. We call the smoother (12) as the second-order smoother.

A more accurate way to exclude the computational modes is to apply the following Runge–Kutta smoother. By replacing n in Eq. (11) by $n-1$, we obtain

$$u_j^{n-1} = v_j^{n-1} - (-1)^n w_j^{n-1}. \tag{13}$$

In order to obtain v_j^n, w_j^n, v_j^{n-1}, and w_j^{n-1} from u_j^n and u_j^{n-1}, we need two more equations as well as Eqs. (11) and (13). For this purpose, we use the differential equation recovered from Eq. (4):

$$\frac{\partial u_j}{\partial t} = -\frac{1}{6\,\Delta x}(u_{j+1} + u_j + u_{j-1})(u_{j+1} - u_{j-1})$$

$$-\frac{\mu}{2(\Delta x)^3}(u_{j+2} - 2u_{j+1} + 2u_{j-1} - u_{j-2}), \tag{14}$$

where $u_j(t)$ are regarded as functions of continuous time t. We integrate Eq. (14) using the Runge–Kutta scheme to obtain $u_j(t=n\,\Delta t)$ from u_j^{n-1} which are the leapfrog solutions of Eq. (4). The step-size used in the Runge–Kutta scheme is Δt, same as in the leapfrog scheme. Since the Runge–Kutta scheme is free from the saw-toothed oscillation, all of u_j^{n-1} in Eq. (13) are advanced in time with no such oscillations as the computational modes. Therefore $u_j(t=n\,\Delta t)$ integrated from u_j^{n-1} may be expressed as

$$u_j(t=n\,\Delta t)=v_j^n-(-1)^n w_j^n. \tag{15}$$

We have ignored the errors incurred by the use of the Runge–Kutta scheme instead of the leapfrog scheme. The validity of Eq. (15) will be ascertained experimentally. Then from Eqs. (11) and (15), we obtain

$$v_j^n=[u_j^n+u_j(t=n\,\Delta t)]/2,$$
$$(-1)^n w_j^n=[u_j^n-u_j(t=n\,\Delta t)]/2.$$

Thus we can decompose u_j^n into the physical modes v_j^n and the computational modes $(-1)^n w_j^n$. In a quite similar manner, we make time-reversing Runge–Kutta integration of Eq. (14) to obtain $u_j(t=(n-1)\,\Delta t)$ from u_j^n. As in the case of Eq. (15), $u_j(t=(n-1)\,\Delta t)$ obtained thus may be expressed as

$$u_j(t=(n-1)\,\Delta t)=v_j^{n-1}+(-1)^n w_j^{n-1}.$$

Equation (13) and the last equation give

$$v_j^{n-1}=[u_j^{n-1}+u_j(t=(n-1)\,\Delta t)]/2,$$
$$(-1)^{n-1} w_j^{n-1}=[u_j^{n-1}-u_j(t=(n-1)\,\Delta t)]/2.$$

The schematic graph for the Runge–Kutta smoother is given in Fig. 3.

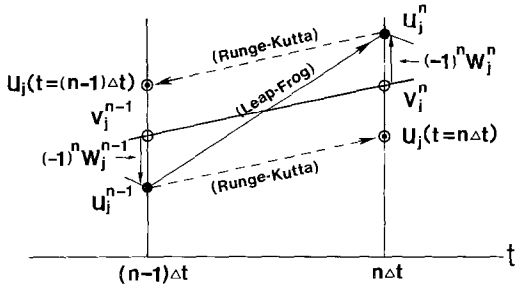

FIG. 3. Schematic graph for the Runge–Kutta smoother: u_j^n, u_j^{n-1}, leapfrog solutions; $u_j(t=n\,\Delta t)$, $u_j(t=(n-1)\,\Delta t)$, Runge–Kutta solutions.

5. Effects of Smoothing

We substitute Eq. (11) into the right-hand side of Eq. (5) to obtain

$$C_2 = C_{2p} + C_{2c},$$

where

$$C_{2p} \equiv \Delta x \sum_{j=0}^{2J-1} (v_j^n)^2 - A^2,$$

$$C_{2c} \equiv \Delta x \sum_{j=0}^{2J-1} [2(-1)^n w_j^n v_j^n + (w_j^n)^2].$$

The C_{2p} denotes the error of physical modes and C_{2c} comes from computational modes. Since in earlier time stage $|w_j^n| \ll |v_j^n|$, the last equation becomes

$$C_{2c} \simeq 2\,\Delta x(-1)^n \sum_{j=0}^{2J-1} w_j^n v_j^n.$$

Thus C_{2c} changes its sign at each time step n. Figure 4 gives $|C_{2p}|$ and $|C_{2c}|$ obtained thus for $A = 1.0$. The C_{2c} in Fig. 4 changes its sign at each time step as shown in Fig. 2. Note that the saw-toothed oscillation of C_{2c} does not appear in $|C_{2c}|$. Also is noted in Fig. 1 that the saw-toothed oscillation of C_2 does not appear in $|C_2|$ when $|C_{2c}| \gg |C_{2p}|$. The saw-toothed oscillation of $|C_2|$ appears only when $|C_{2c}| \approx |C_{2p}|$.

Now we apply the smoother to scheme (4); that is to say, we replace u_j^n and u_j^{n-1} by the physical modes v_j^n and v_j^{n-1}, respectively, at some time step. Figures 5 and 6 give C_{2p} and C_{2c} for $A = 1.0$ when the second-order smoother is applied at the time step $n = 60{,}000$. We see from the figures that the second-order smoother cannot exclude the computational modes completely, and the remaining computational modes grow rapidly to blow up the integration. Figures 7 and 8 give C_{2p} and C_{2c} for $A = 1.0$ when the Runge–Kutta smoother is applied at the time step $n = 60{,}000$. The Runge–Kutta smoother can exclude the computational modes almost completely and enables us to make the long-time leapfrog integration if we apply the Runge–Kutta smoother periodically.

We also applied the second-order and Runge–Kutta smoothers to integrations for $A = 0.5$, 2.5, and 5.0 and found that the second-order smoother cannot exclude the computational modes completely, while the Runge–Kutta smoother can exclude them almost completely. This fact may assure the validity of Eqs. (11) and (15) even for $A = 2.5$ and 5.0.

Figure 9 gives $|C_{2p}|$ at the near-recurrence time t_r as a function of the smoothing period. The Runge–Kutta smoother is applied at the end of each smoothing period. We evaluate the fundamental Fourier coefficient U_1^n in Eq. (6) from the leapfrog solutions u_j^n. The near-recurrence time t_r, which has been given in Table I, is the

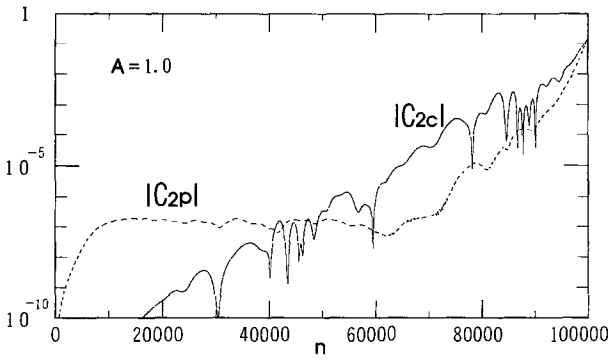

FIG. 4. $|C_{2p}|$ and $|C_{2c}|$ as functions of time step n for $A = 1.0$ in Fig. 1. Note that C_{2p} is not the physical part of the constant but the error of the constant, $C_2 = C_{2p} + C_{2c}$.

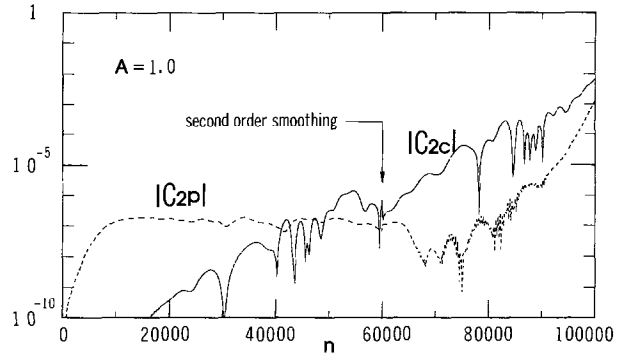

FIG. 5. Time developments of $|C_{2p}|$ and $|C_{2c}|$ for $A = 1.0$ when the second-order smoother is applied at the time step $n = 60{,}000$.

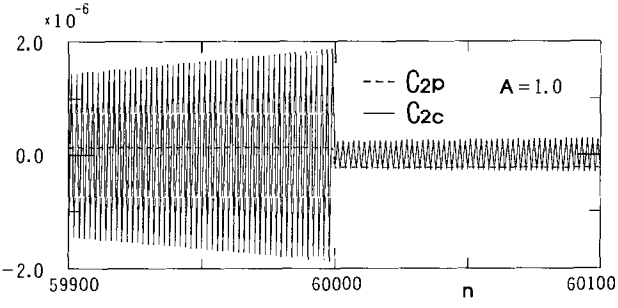

FIG. 6. Magnified curves of C_{2p} and C_{2c} at the smoothing time step $n = 60{,}000$ in Fig. 5.

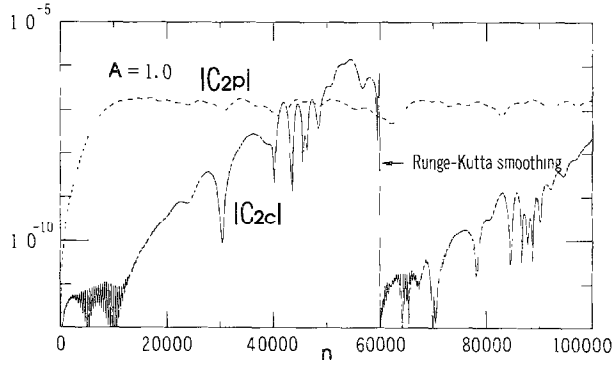

FIG. 7. Time developments of $|C_{2p}|$ and $|C_{2c}|$ for $A = 1.0$ when the Runge–Kutta smoother is applied at the time step $n = 60{,}000$.

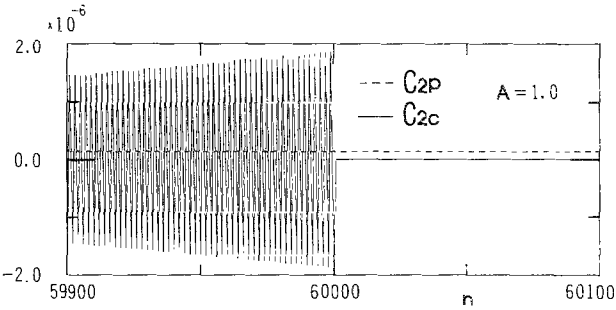

FIG. 8. Magnified curves of C_{2p} and C_{2c} at the smoothing time step $n = 60{,}000$ in Fig. 7.

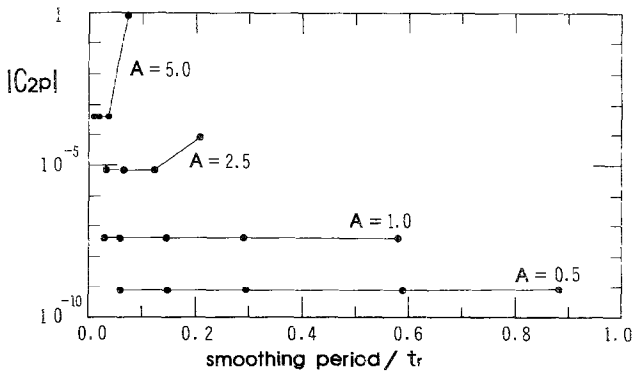

FIG. 9. $|C_{2p}|$ at the near-recurrence time t_r as a function of smoothing period for $A = 0.5$, 1.0, 2.5, and 5.0. The smoothing period is normalized by the near-recurrence time t_r.

TABLE II

Values of $|C_1|$, $|C_2|$, and $|C_3|$ at Near-Recurrence Time t_r by Using Runge–Kutta Smoother

Amplitude A	5.0	2.5	1.0	0.5
smoothing period/t_r (step n)	0.0453 (2,500)	0.0648 (5,000)	0.331 (40,000)	0.471 (80,000)
$\lvert C_1\rvert$	1.55×10^{-12}	1.98×10^{-12}	1.32×10^{-12}	9.32×10^{-14}
$\lvert C_2\rvert$	3.60×10^{-4}	7.06×10^{-6}	4.05×10^{-8}	7.62×10^{-10}
$\lvert C_3\rvert$	3.27×10^{-2}	7.12×10^{-3}	5.41×10^{-5}	3.25×10^{-6}

time when $|U_1^n|$ returns to take the maximal value for the first time. The values of t_r in Table I agree with those evaluated by Abe and Satofuka [7] within the accuracy of 5 percent. From Fig. 9 we are required to apply the Runge–Kutta smoother more frequently as the amplitude increases. The cpu time required for smoothing by the Runge–Kutta smoother was 3.6 ms per smoothing which was about four times as large as that by the second-order smoother in which we used the leapfrog scheme twice to obtain u_j^{n+1} and $u_j^{n\ 2}$ from u_j^n and u_j^{n-1}. However, the cpu time for smoothing forms a negligible percentage of the total running time.

The K-dV equation has conservation laws

$$C_1 \equiv \int_0^2 u(x, t)\, dx = 0,$$

$$C_3 \equiv \int_0^2 \left[\frac{1}{3}u^3 - \mu\left(\frac{\partial u}{\partial x}\right)^2\right] dx + \mu(\pi A)^2 = 0$$

as well as C_2. The values of $|C_1|$, $|C_2|$, and $|C_3|$ at the near-recurrence times are given in Table II, where the Runge–Kutta smoother is applied at the end of every smoothing period. We see that the Runge–Kutta smoother enables us to make long-time leapfrog integration of the K-dV equation for large amplitudes as well as small amplitudes.

REFERENCES

1. B. FORNBERG, *Math. Comput.* **27**, 45 (1973).
2. D. M. SLOAN AND A. R. MITCHELL, *J. Comput. Phys.* **67**, 372 (1986).
3. W. L. BRIGGS, A. C. NEWELL, AND T. SARIE, *J. Comput. Phys.* **51**, 83 (1983).
4. A. AOYAGI AND K. ABE, *J. Comput. Phys.* **83**, 447 (1989).
5. N. J. ZABUSKY AND M. D. KRUSKAL, *Phys. Rev. Lett.* **15**, 240 (1965).
6. N. J. ZABUSKY, private communication.
7. K. ABE AND N. SATOFUKA, *Phys. Fluids* **24**, 1045 (1981).