Understanding Saul'yev-Type Unconditionally Stable Schemes from Exponential Splitting

Siu A. Chin
Department of Physics, Texas A&M University, College Station, Texas 77843

Received 23 October 2013; accepted 1 May 2014
Published online 24 May 2014 in Wiley Online Library (wileyonlinelibrary.com).
DOI 10.1002/num.21885

Saul'yev-type asymmetric schemes have been widely used in solving diffusion and advection equations. In this work, we show that Saul'yev-type schemes can be derived from the exponential splitting of the semidiscretized equation which fundamentally explains their unconditional stability. Furthermore, we show that optimal schemes are obtained by forcing each scheme's amplification factor to match that of the exact amplification factor. A new second-order explicit scheme is found for solving the advection equation with the identical amplification factor as the implicit Crank–Nicolson algorithm. Other new schemes for solving the advection–diffusion equation are also derived. © 2014 Wiley Periodicals, Inc. Numer Methods Partial Differential Eq 30: 1961–1983, 2014

Keywords: exponential splitting; Saul'yev schemes; unconditional stable schemes

I. INTRODUCTION

The one-dimensional (1D) diffusion equation

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2}, \tag{1.1}$$

can be solved numerically by applying the forward-time and central-difference approximations to yield the explicit algorithm

$$u'_j = u_j + r[(u_{j+1} - u_j) - (u_j - u_{j-1})], \tag{1.2}$$

where $x_j = j\Delta x$, $u_j = u(x_j, t)$, $u'_j = u(x_j, t + \Delta t)$ and

$$r = \frac{\Delta t D}{\Delta x^2}. \tag{1.3}$$

Correspondence to: Siu A. Chin, Department of Physics, Texas A&M University, College Station, TX 77843 (e-mail: chin@physics.tamu.edu)

Contract grant sponsor: The Qatar National Research Fund (a member of Qatar Foundation); contract grant number: NPRP #5-674-1-114

Under this (Euler) algorithm, each Fourier component $\tilde{u}_k = e^{ikx}$ with wave number k is amplified by a factor of

$$g = 1 - 4r\sin^2(k\Delta x/2), \tag{1.4}$$

restricting stability ($|g| \leq 1$) to the Courant–Friedrichs–Lewy [1] limit,

$$r \leq \frac{1}{2}. \tag{1.5}$$

As explicit finite-difference methods approximate the exact amplification factor by power-series such as (1.4), it seems inevitable that they will eventually blow-up and be limited in stability. However, Saul'yev [2, 3] showed in the 50's that, by simply replacing in (1.2), either

$$(u_j - u_{j-1}) \rightarrow (u'_j - u'_{j-1}) \quad \text{or} \quad (u_{j+1} - u_j) \rightarrow (u'_{j+1} - u'_j), \tag{1.6}$$

one would have unconditionally stable algorithms:

$$u'_j = \beta_S u'_{j-1} + \gamma_S u_j + \beta_S u_{j+1}, \tag{1.7}$$

or

$$u'_j = \beta_S u_{j-1} + \gamma_S u_j + \beta_S u'_{j+1}, \tag{1.8}$$

where γ_S and β_S are Saul'yev's coefficients given by

$$\gamma_S = \frac{1-r}{1+r} \quad \text{and} \quad \beta_S = \frac{r}{1+r}. \tag{1.9}$$

Algorithm (1.7) is explicit if it is evaluated in ascending order in j from left-to-right (LR) and if the left-most u_1 is a boundary value fixed in time. Similarly, algorithm (1.8) is explicit if it is evaluated in descending order in j from right-to-left (RL) and if the right-most u_N is a boundary value fixed in time. Saul'yev also realized that both algorithms have large errors, but if they are applied alternately, the error would be greatly reduced. This then gives rise to alternating direction explicit algorithms for solving the diffusion equation advocated by Larkin [4] and by Barakat and Clark [5]. Similar asymmetric schemes have been derived for solving the advection equation and hence the advection–diffusion equation by Robert and Weiss [6], Towler and Yang [7], Campbell and Yin [8], Xie et al.,[9], and generalized to alternating group explicit methods by Evans and Abdullah [10] and Evans [11]. Because of its explicit nature and unconditional stability, Saul'yev-type algorithms are used in many applications and are included in textbook [12] discussions of numerical methods for solving partial differential equations.

However, there remain unanswered questions about Saul'yev-type schemes: (1) although it is easy to show that algorithm (1.7) and (1.8) are unconditionally stable, there is no deeper understanding of this stability. (2) The algorithms are not explicit in the case of periodic boundary conditions. What would be the algorithm if there are no fixed boundary values? (3) The alternating application of (1.7) and (1.8) greatly reduces the resulting error. How can one characterize this improvement precisely? (4) Is it useful to generalize Saul'yev-type algorithms to higher order time discretization?

This work presents a new way of deriving finite-difference schemes based on exponential-splittings rather than Taylor expansions. The basic idea is to first solve the semidiscrete form of the finite-difference equation to obtain an updating matrix of the form $e^{\Delta t \mathbf{A}}$, then approximate this exponential matrix by splitting

$$\mathbf{A} = \sum_{i}^{N} \mathbf{A}_i, \tag{1.10}$$

where the exponential of each component matrix $\mathbf{A}_i$ can be exactly evaluated. This is the same type of splitting that produces symplectic integrators [13–15]. Here, we show that such splitting also naturally produces Saul'yev-type schemes. For later reference, we summarize some basic results on splittings below.

Splitting methods are based on approximating $e^{\epsilon(\mathbf{A}+\mathbf{B})}$ to any order in ϵ via a single product decomposition

$$e^{\epsilon(\mathbf{A}+\mathbf{B})} = \prod_i e^{a_i \epsilon \mathbf{A}} e^{b_i \epsilon \mathbf{B}}, \tag{1.11}$$

where **A** and **B** are noncommuting operators or matrices. The key idea is to preserve the exponential form of the matrix. The two first-order Trotter [16] approximations are

$$T_{1A}(\epsilon) = e^{\epsilon \mathbf{A}} e^{\epsilon \mathbf{B}}, \quad T_{1B}(\epsilon) = e^{\epsilon \mathbf{B}} e^{\epsilon \mathbf{A}}, \tag{1.12}$$

and the two second-order Strang [17] product approximations are

$$\begin{aligned} T_{2A}(\epsilon) &= T_{1A}(\epsilon/2) T_{1B}(\epsilon/2) = e^{\frac{1}{2}\epsilon \mathbf{A}} e^{\epsilon \mathbf{B}} e^{\frac{1}{2}\epsilon \mathbf{A}}, \\ T_{2B}(\epsilon) &= T_{1B}(\epsilon/2) T_{1A}(\epsilon/2) = e^{\frac{1}{2}\epsilon \mathbf{B}} e^{\epsilon \mathbf{A}} e^{\frac{1}{2}\epsilon \mathbf{B}}. \end{aligned} \tag{1.13}$$

The average approximation

$$T_{2C}(\epsilon) = \frac{1}{2}[T_{1A}(\epsilon) + T_{1B}(\epsilon)], \tag{1.14}$$

is also second order, but it is no longer a single product of exponentials. This is undesirable in other splitting contexts as being nonunitary or no longer symplectic. However, in the present application, as we will see in Section IV, while it is less accurate than the product approximations (because its time-step is twice as large), it converges better.

In the next two sections, we will give new derivations of Saul'yev-type diffusion and advection algorithms, followed by detailed discussions of their convergences, schemes for solving the diffusion–advection equation and finally some conclusions.

II. SPLITTING DIFFUSION ALGORITHMS

Consider solving the diffusion equation (1.1) with periodic boundary condition $u_{N+1} = u_1$ in the semidiscretized form,

$$\frac{du_j}{dt} = \frac{D}{\Delta x^2}(u_{j+1} - 2u_j + u_{j-1}). \tag{2.1}$$

Regarding u_j as a vector, this is

$$\frac{d\mathbf{u}}{dt} = \mathbf{A}\mathbf{u}, \tag{2.2}$$

with

$$\mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{pmatrix}, \quad \mathbf{A} = \frac{D}{\Delta x^2}\begin{pmatrix} -2 & 1 & & & 1 \\ 1 & -2 & 1 & & \\ & & \ddots & & \\ & & 1 & -2 & 1 \\ 1 & & & 1 & -2 \end{pmatrix}, \tag{2.3}$$

and exact solution

$$\mathbf{u}(t + \Delta t) = \mathrm{e}^{\Delta t \mathbf{A}}\mathbf{u}(t). \tag{2.4}$$

The Euler algorithm corresponds to expanding out the exponential to first order in Δt

$$\mathbf{u}(t + \Delta t) = (1 + \Delta t\mathbf{A})\mathbf{u}(t), \tag{2.5}$$

resulting in a power-series amplification factor (1.4), with limited stability.

If the exponential in (2.4) can be solved exactly, the semidiscretize amplification factor would be

$$g_{\mathrm{sd}} = \mathrm{e}^{-h_{\mathrm{sd}}}, \tag{2.6}$$

where

$$h_{sd} = r4\sin^2(\theta/2) \quad \text{and} \quad \theta \equiv k\Delta x. \tag{2.7}$$

The resulting exact semidiscrete algorithm will be unconditionally stable for all $r > 0$. In the limit of $\Delta x \to 0$,

$$h_{sd} \to h_{ex} = r\theta^2 = \Delta t D k^2, \tag{2.8}$$

each k-Fourier components will be damped by the exact amplification factor

$$g_{ex} = \mathrm{e}^{-h_{ex}} = \mathrm{e}^{-\Delta t D k^2}, \tag{2.9}$$

which is the exact solution to (1.1). (Thus solving the semidiscretized equation exactly may still be far from solving the equation exactly.)

To preserve the unconditional stability of the exact semidiscrete solution, one must seek alternative ways of approximating of $\mathrm{e}^{\Delta t\mathbf{A}}$ without doing any Taylor expansion. The structure of $\mathbf{A}$ immediately suggests that it should decompose as

$$\mathbf{A} = \sum_{j=1}^{N} \mathbf{A}_j, \tag{2.10}$$

where each $\mathbf{A}_j$ has only a single, nonvanishing 2×2 matrix along the diagonal connecting the j and the $j + 1$ elements:

$$\mathbf{A}_j = \frac{D}{\Delta x^2}\begin{pmatrix} \ddots & & & \\ & -1 & 1 & \\ & 1 & -1 & \\ & & & \ddots \end{pmatrix} \quad \text{and} \quad \mathbf{A}_N = \frac{D}{\Delta x^2}\begin{pmatrix} -1 & & & 1 \\ & \ddots & & \\ & & \ddots & \\ 1 & & & -1 \end{pmatrix}. \tag{2.11}$$

The exponential of each $\mathbf{A}_j$ can now be evaluated exactly:

$$\mathrm{e}^{\Delta t \mathbf{A}_j} = \begin{pmatrix} 1 & & & \\ & \alpha & \beta & \\ & \beta & \alpha & \\ & & & 1 \end{pmatrix}, \quad \mathrm{e}^{\Delta t \mathbf{A}_N} = \begin{pmatrix} \alpha & & & \beta \\ & 1 & & \\ & & 1 & \\ \beta & & & \alpha \end{pmatrix}, \tag{2.12}$$

where

$$\alpha = \frac{1}{2}(1 + \gamma), \quad \beta = \frac{1}{2}(1 - \gamma), \quad \text{and} \quad \gamma = \mathrm{e}^{-2r}. \tag{2.13}$$

Each $\mathrm{e}^{\Delta t \mathbf{A}_j}$ updates only u_j and u_{j+1} as

$$\begin{aligned} u'_j &= \alpha u_j + \beta u_{j+1}, \\ u'_{j+1} &= \beta u_j + \alpha u_{j+1}. \end{aligned} \tag{2.14}$$

The eigenvalues of this updating matrix are $\alpha \pm \beta = 1, \gamma$, with $\det = \gamma$. This means that the updating is dissipative for $r > 0$ and unstable for $r < 0$. As α and β are given in terms of γ, the resulting algorithm depends only on a single parameter γ.

As noted by proponents of non-standard finite-difference methods [18], there is arbitrariness, or freedom, in choosing the form of the derivative approximation, for example,

$$\frac{u_{j+1} - 2u_j + u_{j-1}}{h(\Delta x)}, \tag{2.15}$$

as long as $h(\Delta x) \to \Delta x^2$ as $\Delta x \to 0$. Therefore, instead of just r as defined by (1.3), one is also free to chose

$$r' = \frac{Dg(\Delta t)}{h(\Delta x)}, \tag{2.16}$$

so long as $g(\Delta t) \to \Delta t$ as $\Delta t \to 0$. Hence, one should regard γ, the single parameter of the algorithm, as a more general function of r', as $\gamma = \exp(-2r')$. As we shall see later, it is sufficient to regard

$$\gamma(r) = e^{-2f(r)} > 0 \quad \text{such that} \quad \lim_{\Delta t \to 0, \Delta x \to 0} f(r) \to r. \tag{2.17}$$

(Note that this requires r to be small, but does not require $r \to 0$.) This additional degree of freedom will allow us to further optimize our algorithms.

One can now decompose $\exp(\Delta t\mathbf{A})$ to first order in Δt (apply (1.12) repeatedly) via either

$$T_{1A}(\Delta t) = e^{\Delta t\mathbf{A}_N} \cdots e^{\Delta t\mathbf{A}_2} e^{\Delta t\mathbf{A}_1}, \tag{2.18}$$

or

$$T_{1B}(\Delta t) = e^{\Delta t\mathbf{A}_1} \cdots e^{\Delta t\mathbf{A}_{N-1}} e^{\Delta t\mathbf{A}_N}. \tag{2.19}$$

These algorithms update the grid points sequentially, two by two at a time according to (2.14), but each grid point is updated twice in successions. This is crucial for dealing with the periodic boundary condition. Let u_j^* denotes the first time when u_j is updated and u'_j the second (and final) time it is updated. One then has for Algorithm 1A:

$$\begin{aligned}
u_1^* &= \alpha u_1 + \beta u_2, \qquad (2.20)\\
u_2^* &= \beta u_1 + \alpha u_2,\\
u_2' &= \alpha u_2^* + \beta u_3,\\
u_3^* &= \beta u_2^* + \alpha u_3,\\
&\cdots\\
u_j' &= \alpha u_j^* + \beta u_{j+1},\\
u_{j+1}^* &= \beta u_j^* + \alpha u_{j+1},\\
&\cdots\\
u_N' &= \alpha u_N^* + \beta u_1^*,\\
u_1' &= \beta u_N^* + \alpha u_1^*. \qquad (2.21)
\end{aligned}$$

As $\alpha + \beta = 1$, summing up both sides from (2.20) to (2.21) gives,

$$\sum_{j=1}^{N} u_j' = \sum_{j=1}^{N} u_j. \tag{2.22}$$

The algorithm is, therefore, norm-preserving. The same is true of Algorithm 1B below. For $2 < j < N$ one has

$$\begin{aligned}
u_j' &= \alpha u_j^* + \beta u_{j+1},\\
&= \alpha(\beta u_{j-1}^* + \alpha u_j) + \beta u_{j+1},\\
&= \beta(u_{j-1}' - \beta u_j) + \alpha^2 u_j + \beta u_{j+1},\\
&= \beta u_{j-1}' + \gamma u_j + \beta u_{j+1},
\end{aligned} \tag{2.23}$$

and for $j = 2, N$,

$$\begin{aligned}
u_2' &= \beta u_1^* + \gamma u_2 + \beta u_3,\\
u_N' &= \beta u_{N-1}' + \gamma u_N + \beta u_1^*.
\end{aligned} \tag{2.24}$$

Finally, when the snake bits its tail, one has

$$u_1' = \frac{\beta}{\alpha}u_N' + \gamma u_1 + \frac{\gamma\beta}{\alpha}u_2. \tag{2.25}$$

Similarly, 1B is given by

$$u_1^* = \beta u_N + \alpha u_1, \tag{2.26}$$

$$u_N' = \beta u_{N-1} + \gamma u_N + \beta u_1^*,$$

$$u_j' = \beta u_{j-1} + \gamma u_j + \beta u_{j+1}', \tag{2.27}$$

$$u_2' = \beta u_1^* + \gamma u_2 + \beta u'_3,$$

$$u_1' = \frac{\gamma\beta}{\alpha}u_N + \gamma u_1 + \frac{\beta}{\alpha}u_2'. \tag{2.28}$$

Algorithms 1A and 1B are essentially given by (2.23) and (2.27), respectively, except for three values of u_1', u_2', and u_N'. They correspond to the LR and RL form of Saul'yev's schemes (1.7) and (1.8), but with different coefficients. Saul'yev's original coefficient $\gamma_S = (1-r)/(1+r)$ corresponds to the choice of

$$f(r) = \tanh^{-1}(r) = r + \frac{r^3}{3} + \frac{r^5}{5} + \cdots. \tag{2.29}$$

However, in our derivation, the positivity requirement on γ limits Saul'yev's scheme to $r < 1$. There is no such requirement in Saul'yev's original derivation. Note that his $\beta_S = r/(1+r)$ is also given by $\beta_S = (1-\gamma_S)/2$. In contrast to Saul'yev's original algorithm, which cannot be started for periodic boundary conditions, Algorithms 1A and 1B are truly explicit because they are fundamentally given by the sequential updating of (2.14). Each algorithm can get started by first updating u_1 to u_1^*, then updating it again at the end to u'_1. These two schemes have amplification factors

$$g_{1A} = \frac{\gamma + \beta e^{i\theta}}{1 - \beta e^{-i\theta}},$$

$$g_{1B} = \frac{\gamma + \beta e^{-i\theta}}{1 - \beta e^{+i\theta}}, \tag{2.30}$$

with opposite phase errors. As the semidiscrete amplification factor (2.6) is purely real with no phase error, one is immediately alerted to the fact that these first-order schemes do not preserve this qualitative feature of the exact solution. (This is also true of first-order symplectic integrators in general; they are not time-reversible as the exact solution.) As we will see later, the convergence behavior of these first-order schemes are also rather poor.

By virtue of (1.13), one can now immediately generate a second-order time-marching algorithm via the symmetric product,

$$T_2(\Delta t) = T_{1B}\left(\frac{\Delta t}{2}\right)T_{1A}\left(\frac{\Delta t}{2}\right),$$

$$= e^{\frac{1}{2}\Delta t\mathbf{A}_1}e^{\frac{1}{2}\Delta t\mathbf{A}_2}\cdots e^{\frac{1}{2}\Delta t\mathbf{A}_N}e^{\frac{1}{2}\Delta t\mathbf{A}_N}\cdots e^{\frac{1}{2}\Delta t\mathbf{A}_2}e^{\frac{1}{2}\Delta t\mathbf{A}_1}. \tag{2.31}$$

Note that this algorithm starts with $\mathbf{A}_1$, ascends to $\mathbf{A}_N$ then descends back to $\mathbf{A}_1$, thus naturally producing an alternating-direction LR then RL algorithm. In the original discussion of Saulyev [3] and Larkin [4], this alternating use of the LR and RL algorithms is an ad hoc procedure. Here, this procedure is an automatic consequence of the second-order algorithm.

If the boundary effects of u_1', u_2', and u_N' are ignored, and 1A and 1B are considered as given by (2.23) and (2.27), then the alternative product $T_{1A}(\Delta t/2)T_{1B}(\Delta t/2)$ yields the same second-order algorithm with amplification factors

$$g_2 = g_{1B}\left(\frac{\Delta t}{2}\right)g_{1A}\left(\frac{\Delta t}{2}\right),$$

$$= \frac{\tilde{\gamma}^2 + \tilde{\beta}^2 + 2\tilde{\beta}\tilde{\gamma}\cos\theta}{1 + \tilde{\beta}^2 - 2\tilde{\beta}\cos\theta}, \tag{2.32}$$

$$= \frac{1 - \left(4\tilde{\beta}\tilde{\gamma}/\tilde{\alpha}^2\right)\sin^2\theta/2}{1 + \left(4\tilde{\beta}/\tilde{\alpha}^2\right)\sin^2\theta/2} = e^{-h_2}, \tag{2.33}$$

having no phase error and where

$$\tilde{\alpha} = \frac{1}{2}(1+\tilde{\gamma}), \quad \tilde{\beta} = \frac{1}{2}(1-\tilde{\gamma}) \quad \text{and} \quad \tilde{\gamma} = \gamma(r/2). \tag{2.34}$$

Thus, only the second-order scheme is qualitatively similar to the exact solution. Equation (2.33) makes it clear that this algorithm is unconditionally stable as $0 \le \tilde{\gamma} \le 1$. Algorithms 1A and 1B are also unconditionally stable since $|g_{1A,1B}| = \sqrt{g_2}$ with $\tilde{\gamma} \to \gamma$. (Note that this also proves the unconditional stability of Saul'yev's original scheme even if his original γ_S can turn negative, since it only approaches -1 as $r \to \infty$.) Conventional explicit methods, like that of the Euler algorithm, are limited in stability because they have power-series amplification factors. By contrast, splitting finite-difference scheme derived here are unconditionally stable because they produce Saul'yev-type schemes with rational-function amplification factors. This type of stability is usually associated only with implicit methods.

In our derivation, we have the freedom in $f(r)$ to optimize the algorithm. Expanding h_2 of (2.33) in powers of θ gives,

$$h_2 = \frac{2(1-\tilde{\gamma})}{1+\tilde{\gamma}}\theta^2 - \frac{(13 - 35\tilde{\gamma} + 35\tilde{\gamma}^2 - 13\tilde{\gamma}^3)}{6(1+\tilde{\gamma})^3}\theta^4 + \cdots. \tag{2.35}$$

Comparing this to the exact exponent,

$$h_{ex} = r\theta^2, \tag{2.36}$$

one sees that this term can be matched by requiring

$$\frac{1-\tilde{\gamma}(r)}{1+\tilde{\gamma}(r)} = \frac{r}{2} \to \tilde{\gamma}(r) = \gamma(r/2) = \frac{1 - r/2}{1 + r/2}, \tag{2.37}$$

which is precisely Saul'yev's original coefficient. We shall refer to this algorithm as DS2. With this choice for $\tilde{\gamma}(r)$, (2.33) reads

$$g_2 = \frac{1 - 2r(1 - r/2)\sin^2(\theta/2)}{1 + 2r(1 + r/2)\sin^2(\theta/2)}, \tag{2.38}$$

with exponent

$$h_2 = r\theta^2 - \left(\frac{r}{12} + \frac{r^3}{4}\right)\theta^4 + \left(\frac{r}{360} + \frac{r^3}{8} + \frac{r^5}{16}\right)\theta^6 + \cdots . \tag{2.39}$$

All terms except the first are error terms of the algorithm. Unfortunately, we have no more degrees of freedom to eliminate these higher order error terms. Comparing this to that of the implicit Crank–Nicolson (CN) scheme [which is without the $\pm r/2$ terms in (2.38)]:

$$h_{\mathrm{CN}} = r\theta^2 - \frac{r}{12}\theta^4 + \left(\frac{r}{360} + \frac{r^3}{12}\right)\theta^6 + \cdots , \tag{2.40}$$

one sees that both have comparable $O(\theta^4)$ errors but CN's error is smaller.

In this section, we have shown that by splitting the matrix form of the semidiscrete equation, one naturally produces Saul'yev-type unconditional algorithms. Although first-order algorithms do not preserve the qualitative feature of the exact solution, second-order schemes do and can be fine-tuned to match the exact amplification exponent to the leading order. We will examine the convergence behavior of these diffusion schemes in Section IV.

III. SPLITTING ADVECTION ALGORITHMS

For the advection equation

$$\frac{\partial u}{\partial t} = -v\frac{\partial u}{\partial x}, \tag{3.1}$$

its usual semidiscrete form is

$$\frac{\partial u_j}{\partial t} = -\frac{v}{2\Delta x}(u_{j+1} - u_{j-1}), \tag{3.2}$$

with discretization matrix

$$\mathbf{B} = \frac{v}{2\Delta x}\begin{pmatrix} 0 & -1 & & & 1 \\ 1 & 0 & -1 & & \\ & & \ddots & & \\ & & 1 & 0 & -1 \\ -1 & & & 1 & 0 \end{pmatrix}, \tag{3.3}$$

and solution

$$\mathbf{u}(t + \Delta t) = \mathrm{e}^{\Delta t\mathbf{B}}\mathbf{u}(t). \tag{3.4}$$

The semidiscrete amplification factor ($\theta = k\Delta x$)

$$g_{sd} = \mathrm{e}^{-i\eta\sin\theta}, \quad \text{with} \quad \eta = \frac{v\Delta t}{\Delta x}, \tag{3.5}$$

is unitary (or dissipationless) and causes a phase-shift of each Fourier component. In the limit of $\Delta x \to 0$,

$$g_{sd} \to g_{ex} = \mathrm{e}^{-i\eta\theta} = \mathrm{e}^{-ikv\Delta t}, \tag{3.6}$$

the phase-shift becomes uniform for all Fourier modes $\mathrm{e}^{ikx} \to \mathrm{e}^{ik(x-v\Delta t)}$, resulting in a uniform shift of the entire function $u(x) \to u(x - v\Delta t)$, which is the exact solution to (3.1). Any Taylor expansion of (3.4) will produce algorithms with a nonunitary g, resulting in unwanted dissipations or instability. The situation here is much more delicate than in the diffusion case. The natural decomposition is similarly,

$$\mathbf{B} = \sum_{i=1}^{N} \mathbf{B}_j, \tag{3.7}$$

where

$$\mathbf{B}_j = \frac{v}{2\Delta x}\begin{pmatrix} \ddots & & & \\ & 0 & -1 & \\ & 1 & 0 & \\ & & & \ddots \end{pmatrix} \quad \text{with} \quad \mathbf{B}_N = \frac{v}{2\Delta x}\begin{pmatrix} 0 & & & 1 \\ & \ddots & & \\ & & \ddots & \\ -1 & & & 0 \end{pmatrix}. \tag{3.8}$$

It follows that

$$\mathrm{e}^{\Delta t \mathbf{B}_j} = \begin{pmatrix} 1 & & & \\ & c & -s & \\ & s & c & \\ & & & 1 \end{pmatrix}, \quad \mathrm{e}^{\Delta t \mathbf{B}_N} = \begin{pmatrix} c & & & s \\ & 1 & & \\ & & 1 & \\ -s & & & c \end{pmatrix}, \tag{3.9}$$

where now

$$c = \cos(\eta/2) \quad \text{and} \quad s = \sin(\eta/2). \tag{3.10}$$

Each $\mathrm{e}^{\Delta t \mathbf{B}_j}$ only updates u_j and u_{j+1} as

$$\begin{aligned} u'_j &= cu_j - su_{j+1}, \\ u'_{j+1} &= su_j + cu_{j+1}. \end{aligned} \tag{3.11}$$

As in the diffusion case, the above updating can be recasted into the following forms for Algorithms 1A and 1B, with 1A given by

$$\begin{aligned} u_1^* &= cu_1 - su_2, \\ u'_2 &= su_1^* + u_2 - su_3, \\ u'_j &= su'_{j-1} + u_j - su_{j+1}, \quad (2 < j < N) \\ u'_N &= su'_{N-1} + u_N - su_1^*, \\ cu'_1 &= su'_N + cu_1 - su_2. \end{aligned} \tag{3.12}$$

and 1B given by

$$\begin{aligned}
u_1^* &= su_N + cu_1,\\
u'_N &= su_{N-1} + u_N - su_1^*,\\
u'_j &= su_{j-1} + u_j - su'_{j+1}, \quad (2 < j < N)\\
u'_2 &= su_1^* + u_2 - su'_3,\\
cu'_1 &= su_N + cu_1 - su'_2.
\end{aligned} \tag{3.13}$$

Again, Algorithms 1A and 1B correspond to the LR and RL form of Saulyev's schemes. In contrast to the diffusion case, these algorithms are not exactly norm-preserving for periodic boundary condition. By adding up both sides of the above algorithms, one finds that what is preserved by 1A is not the usual norm $N = \sum_{j=1}^{N} u_j$, but a modified norm given by

$$\tilde{N}_{1A} = N + \left(\frac{c}{1-s} - 1\right) u_1. \tag{3.14}$$

Similarly, what is preserved by 1B is

$$\tilde{N}_{1B} = N + \left(\frac{c}{1+s} - 1\right) u_1. \tag{3.15}$$

If initially $u_1 = 0$, then $\tilde{N}_{1A} = \tilde{N}_{1B} = N_0$, where N_0 is the initial norm. As the system evolves, each algorithm's actual norm will evolve as

$$\begin{aligned}
N_{1A} &= N_0 - \left(\frac{c}{1-s} - 1\right) u_1,\\
N_{1B} &= N_0 - \left(\frac{c}{1+s} - 1\right) u_1.
\end{aligned} \tag{3.16}$$

The error is due to a single point u_1, where it is the only point not updated twice immediately. As the wave form travels around the periodic box, u_1 will trace out the shape of the wave and imprint that as the error of the norm in time. For a sharp pulse, the norm error will return to zero after the pulse peak has passed through u_1. Thus, norm-preservation will be periodic. We emphasized that this only apply to the periodic boundary case. For the fixed boundary case of $u_1 = 0$, the norm is conserved.

If the boundary values u'_1, u'_2, and u'_N are ignored for now, then again the resulting second-order algorithm is unique, independent of the order of applying 1A or 1B. The amplification factors are all unitary:

$$g_{1A} = \frac{1 - se^{i\theta}}{1 - se^{-i\theta}} = \exp(-i\phi_{1A}), \tag{3.17}$$

$$g_{1B} = \frac{1 + se^{-i\theta}}{1 + se^{i\theta}} = \exp(-i\phi_{1B}), \tag{3.18}$$

$$g_2 = g_{1B}(\Delta t/2) g_{1A}(\Delta t/2)$$

$$= \frac{1 - i2(\tilde{s}/\tilde{c}^2)\sin\theta}{1 + i2(\tilde{s}/\tilde{c}^2)\sin\theta} = \exp(-i\phi_2), \tag{3.19}$$

with phase angles

$$\phi_{1A} = 2\tan^{-1}\left(\frac{s\sin\theta}{1 - s\cos\theta}\right),$$

$$\phi_{1B} = 2\tan^{-1}\left(\frac{s\sin\theta}{1 + s\cos\theta}\right),$$

$$\phi_2 = \phi_{1A}(\Delta t/2) + \phi_{1B}(\Delta t/2),$$

$$= 2\tan^{-1}\left(\frac{2\tilde{s}}{1 - \tilde{s}^2}\sin\theta\right), \tag{3.20}$$

where here

$$\tilde{s} = \sin(\eta/4) \quad \text{and} \quad \tilde{c} = \cos(\eta/4). \tag{3.21}$$

As g_{1A} and g_{1B} are not complex conjugate of each other, their phase errors do not exactly cancel. Their residual difference is the error of the second-order algorithm.

Algorithms (3.12) and (3.13) are our derivations of Saul'yev-type schemes for solving the advection equation. The coefficient here, $s = \sin(\eta/2)$, is again different from the coefficient of $s = \eta/2$ in Saul'yev's schemes by simply modifying the equation. Our derivation showed why it makes no sense to apply Saul'yev's schemes at $s = \eta/2 > 1$, since such a scheme cannot be derived from the fundamental updating matrix (3.11) with a real $c = \sqrt{1 - s^2}$. At $s > 1$, Saul'yev's schemes are in fact unstable, suffering from spatial amplification [8], despite the unimodulus appearance of (3.17) and (3.18). This is easy to see in the case of Algorithm 1A. If initially $u_j = 0$ for $j \geq J$, but $u_{J-1} \neq 0$, then according to (3.12), $u'_{J+n} = s^{n+1}u'_{J-1}$ increases without bound as a function of n. Even the case of $s = 1$ is pathological. For Saul'yev's coefficient $s = \eta/2 = 1$, one has

$$g_{1A} = -e^{ik\Delta x} = -e^{ik(v/2)\Delta t} \quad \text{and} \quad g_{1B} = e^{-ik\Delta x} = e^{-ik(v/2)\Delta t}. \tag{3.22}$$

Under Algorithm 1A, Fourier mode e^{ikx} will flip its sign and propagate with velocity $-v/2$. Under 1B, it will propagate with velocity $v/2$. The resulting second-order algorithm by concatenation would then leave the Fourier mode stationary with only a sign flip. This is completely contrary to the behavior of the exact solution and is a source of great error for Saul'yev-type schemes in solving the advection equation.

By appealing to the freedom in choosing the form of the derivative approximation, one can generalize to

$$s(\eta) = \sin(f(\eta/2)), \tag{3.23}$$

which implies that a general $s(\eta)$ must still obey

$$-1 \leq s(\eta) \leq 1. \tag{3.24}$$

As we will show later, alternative choices for s other than $s = \eta/2$ will eliminate much of the above unphysical behaviors.

Although the derived choice of $s = \sin(\eta/2)$ is unconditionally stable for all η, the resulting Algorithms 1A and 1B have huge phase errors, and are no better than Saul'yev's choice of $s = \eta/2$. This is because in comparison with the exact phase for the advection equation:

$$\phi_{ex} = \eta\theta = v\Delta tk, \tag{3.25}$$

Algorithms 1A and 1B have expansions

$$\begin{aligned}\phi_{1\mathrm{A}} &= \frac{2s}{1-s}\theta - \frac{s(1+s)}{3(1-s)^3}\theta^3 + \cdots,\\ \phi_{1\mathrm{B}} &= \frac{2s}{1+s}\theta - \frac{s(1-s)}{3(1+s)^3}\theta^3 + \cdots,\end{aligned} \tag{3.26}$$

and neither $s = \eta/2$ nor $s = \sin(\eta/2)$ can result in a first-order coefficient of θ matching that of ϕ_{ex} exactly. As any coefficient $(1+a)$ multiplying $\eta\theta$ will displace the Fourier mode to $e^{ik(x-(1+a)v\Delta t)}$, with a displacement error of $\delta x = av\Delta t$, forcing a to zero is the most basic requirement of any advection scheme.

The choice of s that can do this is, for 1A,

$$\frac{2s}{1-s} = \eta \rightarrow s = \frac{\eta}{2+\eta}, \tag{3.27}$$

and for 1B,

$$\frac{2s}{1+s} = \eta \rightarrow s = \frac{\eta}{2-\eta}. \tag{3.28}$$

This then reproduces the Robert and Weiss [6, 8] forms of the Saul'yev-type algorithm and will be denoted as RW1A and RW1B. For $\eta > 0$, only RW1A is unconditionally stable and RW1B is limited by spatial amplification to $\eta < 1$. The pathological behavior of 1A at $s = 1$ can no longer occur at any finite η. For the above choices of s, the corresponding phase angles are

$$\begin{aligned}\phi_{1\mathrm{A}} &= \eta\theta - \left(\frac{\eta}{6} + \frac{\eta^2}{4} + \frac{\eta^3}{12}\right)\theta^3 + \cdots,\\ \phi_{1\mathrm{B}} &= \eta\theta - \left(\frac{\eta}{6} - \frac{\eta^2}{4} + \frac{\eta^3}{12}\right)\theta^3 + \cdots.\end{aligned} \tag{3.29}$$

The third and higher order terms in θ are now dispersion errors of the schemes.

The second-order algorithm (2.31) now corresponds to applying RW1A then RW1B in succession, each at $\Delta t/2$, yielding

$$\begin{aligned}\phi_2 &= \phi_{1\mathrm{A}}(\eta/2) + \phi_{1\mathrm{B}}(\eta/2),\\ &= \eta\theta - \left(\frac{\eta}{6} + \frac{\eta^3}{48}\right)\theta^3 + \left(\frac{\eta}{120} + \frac{5\eta^3}{192} + \frac{\eta^5}{1280}\right)\theta^5 + \cdots.\end{aligned} \tag{3.30}$$

This second-order advection algorithm will be denoted as RW2. Because RW1B is limited by spatial amplification to $\eta < 1$, RW2 is limited in stability to $\eta < 2$.

TABLE I. Properties of various advection algorithms discussed in the text. 1A and 1B are the first-order left-to-right and right-to-left schemes, respectively, derived in this work. S1A, S1B and RWIA, RW1B are Saulyev's and Robert and Weiss' forms [8] of the same algorithms, respectively. The second-order advection algorithms 2, S2 and RW2 are alternating-direction algorithms by applying their respective 1A and 1B algorithms in succession at half the time-step size as mandated by (2.31). Algorithm C2 is the unique second-order algorithm with the same s function for both 1A and 1B but without any first-order phase error. As defined in the text, $\eta = v\Delta t/\Delta x$.

	1A	1B	S1A	S1B	RW1A	RW1B	2	S2	RW2	C2
$s =$	$\sin(\frac{\eta}{2})$	$\sin(\frac{\eta}{2})$	$\frac{\eta}{2}$	$\frac{\eta}{2}$	$\frac{\eta/2}{1+\eta/2}$	$\frac{\eta/2}{1-\eta/2}$				$\frac{\sqrt{1+\eta^2}-1}{\eta}$
First-order phase error?	Yes	Yes	Yes	Yes	No	No	Yes	Yes	No	No
Stable against spatial amplification for	$\eta < \pi$	$\eta < \pi$	$\eta < 2$	$\eta < 2$	All η	$\eta < 1$	$\eta < 2\pi$	$\eta < 4$	$\eta < 2$	All η

To generate a stable second-order algorithm for all η, one can apply 1A then 1B with the same $\tilde{s}$. To match the first-order term θ to ϕ_{ex} then requires

$$\frac{2\tilde{s}}{1-\tilde{s}} + \frac{2\tilde{s}}{1-\tilde{s}} = \eta \rightarrow \frac{2\tilde{s}}{1-\tilde{s}^2} = \frac{\eta}{2} \rightarrow \tilde{s} = \frac{2}{\eta}\left(\sqrt{1+\frac{\eta^2}{4}} - 1\right). \tag{3.31}$$

The resulting amplification factor is, according to (3.19),

$$g_2 = \frac{1 - i(\eta/2)\sin\theta}{1 + i(\eta/2)\sin\theta}, \tag{3.32}$$

which is precisely the implicit CN amplification factor. Since by (3.31), $\tilde{s} \le 1$, and $\tilde{c} = \sqrt{1-\tilde{s}^2}$ is well-defined for all η, the algorithm is unconditionally stable and can be applied to periodic boundary problems via the fundamental updating (3.11). Corresponding to (3.32), the phase-angle has the expansion,

$$\phi_2 = \eta\theta - \left(\frac{\eta}{6} + \frac{\eta^3}{12}\right)\theta^3 + \left(\frac{\eta}{120} + \frac{\eta^3}{24} + \frac{3\eta^5}{240}\right)\theta^5 + \cdots. \tag{3.33}$$

We shall designate this second-order algorithm, with $\tilde{s}$ given by (3.31), as C2. The second-order algorithm corresponding to Saul'yev's choice of $\tilde{s} = \eta/4$ will be denoted as S2, and the initially derived result of $\tilde{s} = \sin(\eta/4)$ as just Algorithm 2. Comparing (3.33) and (3.30) to the phase angle of the Lax–Wendroff (LW) scheme,

$$\phi_{\rm LW} = \eta\theta - \frac{1}{6}(\eta - \eta^3)\theta^3 - i\frac{1}{8}(\eta^2 - \eta^4)\theta^4 + \frac{1}{120}(\eta + 5\eta^3 - 6\eta^5)\theta^5 + \cdots, \tag{3.34}$$

one sees that LW has smaller dispersion errors. Its imaginary part, signifying damping, also helps to smooth out much of these dispersive oscillations.

The distinctive features of all Saulyev-type advection algorithms derived here are summarized in Table I.

In Fig. 1, we show the working of these algorithms in propagating an initial profile

$$u(x,0) = \exp\left[-\left(\frac{x}{2}\right)^6\right]. \tag{3.35}$$

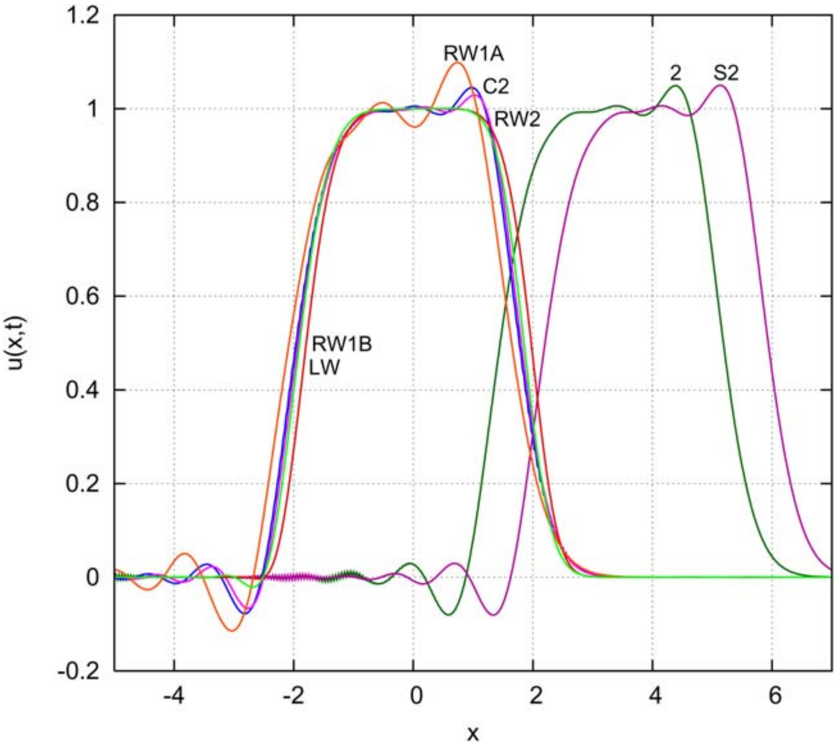

FIG. 1. The propagation of initial profile (3.35) five times around a periodic box of $[-10,10]$ with $\Delta x = 0.025$, $\Delta t = 0.02$, $v = 1$, and $\eta = 0.8$, corresponding to 5000 iterations of each algorithm. If there were no first-order phase error, the profile would remain centered on $x = 0$. Algorithms 2 and S2 have large positive phase errors. All second-order schemes suffer from dispersive, oscillating errors. The LW scheme's oscillations are damped because it is dissipative (the bright green line). From (3.29), scheme RW1B's third-order dispersive error coefficient vanishes for $\eta = 1$ and is only 0.016 at $\eta = 0.8$. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

The power of 6 was chosen to provide a steep, but continuous profile so that both the first-order phase error (due to $a \neq 0$) and the dispersion error (due to the $O(\theta^3)$ and higher order terms) are visible. If the profile were too steep, like that of a square wave, the dispersion errors would have overwhelmed the calculation and masked the first-order phase error. Our CN-like scheme C2 is more dispersive than LW because it has no damping.

In this section, we have shown that Saul'yev-type algorithm can also be systematically derived for solving the advection equation. More importantly we have shown that it is the *amplification exponent* (and not the amplification factor itself) that is crucial in deriving the first-order Robert and Weiss [6, 8] schemes and the new CN-like second-order algorithm C2.

IV. CONVERGENCE

As Saul'yev-type schemes are unconditionally stable, in accordance with Lax's equivalence theorem [19], their convergence depends on whether they are consistent. Truncation errors in these schemes have been extensively studied in Refs. [4–8]. Here, we will also pinpoint their peculiar origin. Consider first the advection case. For Algorithm 1A (3.12), making a Taylor expansion in t gives,

$$\begin{aligned} u'_j &= u_j - s(u_{j+1} - u'_{j-1}), \\ &= u_j - s(u_{j+1} - u_{j-1}) + s\Delta t \partial_t u_{j-1} + s\frac{\Delta t^2}{2}\partial_t^2 u_{j-1} + \cdots . \end{aligned} \tag{4.1}$$

For the original Saul'yev case, $s = \eta/2 \propto \Delta t/\Delta x$. In all other cases, this is the leading order term as $\Delta t \to 0$. When s is multiplying a paired term,

$$(u_{j+1} - u_{j-1}) = 2\Delta x \partial_x u_j + O(\Delta x^3), \tag{4.2}$$

or when s is multiplying a spatially expanded term in powers of Δx, its singular $1/\Delta x$ dependence would be safely removed. So, the only "dangerous" terms that contribute to the truncation errors in (4.1) (with one power of Δt divided out) are the spatially unexpanded terms

$$s\partial_t u_j + s\frac{\Delta t}{2}\partial_t^2 u_j + \cdots = \frac{v\Delta t}{2\Delta x}\partial_t u_j + \frac{v\Delta t^2}{4\Delta x}\partial_t^2 u_j + \cdots . \tag{4.3}$$

Similarly for Algorithm 1B,

$$\begin{aligned} u'_j &= u_j - s(u'_{j+1} - u_{j-1}) \\ &= u_j - s(u_{j+1} - u_{j-1}) - s\Delta t \partial_t u_{j+1} - s\frac{\Delta t^2}{2}\partial_t^2 u_{j+1} + \cdots , \end{aligned} \tag{4.4}$$

but now the same unexpanded terms are all negative. As the leading truncation error terms are $\propto \Delta t/\Delta x$, both algorithms are not consistent unless, as $\Delta t \to 0$ and $\Delta x \to 0$,

$$\frac{\Delta t}{\Delta x} \to 0. \tag{4.5}$$

This is a severe limitation on the size of Δt that can be used. This is a well-known property of Saul'yev-type algorithms [4–8]. However, we already knew that splitting first-order algorithms are not representative of the exact solution and should not be used in isolation.

The second-order scheme by averaging 1A and 1B would have all such "dangerous" terms canceled,

$$\begin{aligned} u'_j &= u_j - \frac{s}{2}(u_{j+1} - u_{j-1}) - \frac{s}{2}(u'_{j+1} - u'_{j-1}), \\ \partial_t u_j + \frac{\Delta t}{2}\partial_t^2 u_j + \cdots &= -\frac{s}{2\Delta t}(2\Delta x \partial_x u_j + 2\Delta x \partial_x u'_j + O(\Delta x^3)), \\ &= -\frac{v}{4\Delta x}(4\Delta x \partial_x u_j + 2\Delta x \Delta t \partial_x \partial_t u_j + O(\Delta x \Delta t^2) + O(\Delta x^3)), \\ \partial_t u_j + v\partial_x u_j &= -\frac{1}{2}\Delta t \partial_t(\partial_t u_j + v\partial_x u_j) + O(\Delta t^2) + O(\Delta x^2). \end{aligned} \tag{4.6}$$

resulting in a bona fide second-order algorithm.

For the case of the product second-order algoritm in which one uses Scheme 1A for $\Delta t/2$ then Scheme 1B for $\Delta t/2$, one has

$$u_j^* = \tilde{s}u_{j-1}^* + u_j - \tilde{s}u_{j+1}, \tag{4.7}$$

$$u_j' = \tilde{s}u_{j-1}^* + u_j^* - \tilde{s}u_{j+1}', \tag{4.8}$$

where $u'_j = u(t+\Delta t, j\Delta x)$, $u_j^* = u(t+\Delta t/2, j\Delta x)$, $u_j = u(t, j\Delta x)$, and $\tilde{s} = s(\eta/2)$. Substituting (4.7) into (4.8) gives,

$$\begin{aligned} u_j' &= u_j - \tilde{s}\left(u_{j+1}' - 2u_{j-1}^* + u_{j+1}\right), \\ &= u_j - 2\tilde{s}(u_{j+1} - u_{j-1}) - \tilde{s}\Delta t\partial_t(u_{j+1} - u_{j-1}) \\ &\quad - \tilde{s}\frac{1}{2}\Delta t^2\partial_t^2(u_{j+1} - \frac{1}{2}u_{j-1}) + \cdots . \end{aligned} \tag{4.9}$$

One immediately sees that the last pairing is incomplete and there will be a residual unexpanded term. In the limit of $\tilde{s} \to \eta/4 = v\Delta t/(4\Delta x)$, the truncation errors are

$$\begin{aligned} \partial_t u_j + \frac{1}{2}\Delta t\partial_t^2 u_j &= -\frac{2\tilde{s}}{\Delta t}(2\Delta x\partial_x u_j) - \tilde{s}\partial_t(2\Delta x\partial_x u_j) \\ &\quad - \tilde{s}\frac{1}{2}\Delta t\partial_t^2\left(\frac{1}{2}u_j\right) + O(\Delta t^2) + O(\Delta x^2), \\ \partial_t u_j + v\partial_x u_j &= -\frac{v\Delta t^2}{16\Delta x}\partial_t^2 u_j - \frac{\Delta t}{2}\partial_t(\partial_t u_j + v\partial_x u_j) + O(\Delta t^2) + O(\Delta x^2). \end{aligned} \tag{4.10}$$

The leading error is now $\propto \Delta t^2/\Delta x$, which vanishes as $\Delta t \to 0$ as long as $\Delta t/\Delta x \propto$ constant. This is no worse than the conventional LW scheme which required $\Delta t/\Delta x \le 1/v =$ constant for stability and hence, for convergence.

Consider now the diffusion case. The first-order Saul'yev schemes also have truncation error $\propto \Delta t/\Delta x$, as clearly shown by Larkin [4]. For the second-order scheme, he only gave result for the average of Algorithms 1A and 1B. Here, we give the truncation error for the product form of the second-order algorithm. The result is very similar to the advection case. For Scheme 1A (2.23), we have

$$\begin{aligned} u_j^* - \tilde{\beta}&\left(u_j^* - \Delta x\partial_x u_j^* + \frac{1}{2}\Delta x^2\partial_x^2 u_j^* + \cdots\right) \\ &= \tilde{\gamma}u_j + \tilde{\beta}\left(u_j + \Delta x\partial_x u_j + \frac{1}{2}\Delta x^2\partial_x^2 u_j + \cdots\right). \end{aligned} \tag{4.11}$$

Since $\tilde{\gamma} + \tilde{\beta} = 1 - \tilde{\beta}$, it follows that

$$u_j^* = u_j + \frac{\tilde{\beta}}{1-\tilde{\beta}}\left(-\Delta x\partial_x(u_j^* - u_j) + \frac{1}{2}\Delta x^2\partial_x^2(u_j^* + u_j) + O(\Delta x^3)\right). \tag{4.12}$$

Similarly for Algorithm 1B,

$$u_j' = u_j^* + \frac{\tilde{\beta}}{1-\tilde{\beta}}\left(\Delta x\partial_x(u_j' - u_j^*) + \frac{1}{2}\Delta x^2\partial_x^2(u_j' + u_j^*) + O(\Delta x^3)\right). \tag{4.13}$$

Substituting (4.12) into (4.13) then yields,

$$u'_j = u_j + \frac{\tilde{\beta}}{1-\tilde{\beta}}\left(\Delta x\partial_x(u'_j - 2u^*_j + u_j) + \frac{1}{2}\Delta x^2\partial_x^2(u'_j + 2u^*_j + u_j) + O(\Delta x^3)\right). \quad (4.14)$$

For Saul'yev original algorithm, one has exactly

$$\frac{\tilde{\beta}}{1-\tilde{\beta}} = \frac{r}{2}. \quad (4.15)$$

For the more general case of (2.17), this is also the leading order term in r as $\Delta t \to 0$. Hence,

$$\begin{aligned}\partial_t u_j + \frac{1}{2}\Delta t\partial_t^2 u_j + O(\Delta t^2) &= \frac{r}{2\Delta t}\left(\frac{1}{4}\Delta x\Delta t^2\partial_x\partial_t^2 u_j + \frac{1}{2}\Delta x^2\partial_x^2(4u_j + 2\Delta t\partial_t u_j)\right.\\ &\quad \left. + O(\Delta x\Delta t^3) + O(\Delta x^2\Delta t^2) + O(\Delta x^3)\right),\\ \partial_t u_j - D\partial_x^2 u_j &= \frac{r}{8}\Delta t\Delta x\partial_x\partial_t^2 u_j - \frac{\Delta t}{2}\partial_t(\partial_t - D\partial_x^2)u_j\\ &\quad + O(\Delta t^2) + O(\Delta t^3/\Delta x). \end{aligned} \quad (4.16)$$

If $\Delta t \to 0$ and $\Delta x \to 0$ such that r is a constant, then the above algorithm is second order in $\Delta t\Delta x$ but only of time order $\Delta t^{3/2}$. In comparison, conventional first- and second-order schemes for solving the diffusion equation require three and five grid points, respectively. Saul'yev-type algorithms achieve order $\Delta t^{3/2}$ with only three grid points. The average algorithm, as in the advection case, would be strictly second order.

V. SOLVING THE ADVECTION–DIFFUSION EQUATION

The advection–diffusion equation,

$$\frac{\partial u}{\partial t} = -v\frac{\partial u}{\partial x} + D\frac{\partial^2 u}{\partial x^2}, \quad (5.1)$$

has the exact operator solution

$$u(x,\Delta t) = \mathrm{e}^{-v\Delta t\frac{\partial}{\partial x} + D\Delta t\frac{\partial^2}{\partial x^2}}u(x,0). \quad (5.2)$$

If v and D are just constants, then as $[\frac{\partial}{\partial x}, \frac{\partial^2}{\partial x^2}] = 0$, one has

$$\begin{aligned} u(x,\Delta t) &= \mathrm{e}^{-v\Delta t\frac{\partial}{\partial x}}\mathrm{e}^{D\Delta t\frac{\partial^2}{\partial x^2}}u(x,0),\\ &= \mathrm{e}^{-v\Delta t\frac{\partial}{\partial x}}\tilde{u}(x,\Delta t),\\ &= \tilde{u}(x - v\Delta t,\Delta t), \end{aligned} \quad (5.3)$$

where $\tilde{u}(x,\Delta t)$ is the diffused solution. The complete solution is, therefore, the exact diffused solution $\tilde{u}(x,\Delta t)$ displaced by $v\Delta t$.

For periodic boundary condition, our matrices also commute, $[\mathbf{A}, \mathbf{B}] = 0$, so that the discretized version also holds,

$$\mathbf{u}(t + \Delta t) = \mathrm{e}^{\Delta t \mathbf{A}} \mathrm{e}^{\Delta t \mathbf{B}} \mathbf{u}(t). \tag{5.4}$$

Thus, second-order algorithms can be obtained by applying second-order advection and diffusion algorithms in turns from the previous sections. However, because the advection algorithm has a one-point norm failure at u_1, all Saul'yev-type advection–diffusion algorithms described in this section will not be norm-conserving for periodic boundary condition. Of course, for more practical applications with fixed boundary value of $u_1 = 0$, there is no such norm-conserving problem.

As we have learned in the advection case, any initial algorithm from splitting the semidiscretize matrix may not be optimal. Therefore, to solve the advection–diffusion equation (5.1), one may as well directly start with an assumed updating matrix,

$$\begin{aligned} u'_j &= \alpha u_j + \lambda u_{j+1}, \\ u'_{j+1} &= \beta u_j + \alpha u_{j+1}, \end{aligned} \tag{5.5}$$

and determine its elements by enforcing the norm-conserving condition and by matching its amplification exponent to that of the exact amplification exponent. The sequential applications of the above updating matrix yields Saul'yev-type Algorithms 1A and 1B,

$$\begin{aligned} u'_j &= \beta u'_{j-1} + \gamma u_j + \lambda u_{j+1}, \\ u'_j &= \beta u_{j-1} + \gamma u_j + \lambda u'_{j+1}. \end{aligned} \tag{5.6}$$

These schemes will be norm-preserving, if one imposes

$$\beta + \gamma + \lambda = 1. \tag{5.7}$$

The determinant $\gamma = \alpha^2 - \beta\lambda$ is to be regarded as fixing α as a function of γ and β via $\alpha = \sqrt{\gamma + \beta\lambda}$. The resulting amplification factors are then

$$\begin{aligned} g_{1A} &= \frac{\gamma + \lambda \mathrm{e}^{i\theta}}{1 - \beta \mathrm{e}^{-i\theta}} = \mathrm{e}^{-h_{1A}}, \\ g_{1B} &= \frac{\gamma + \beta \mathrm{e}^{-i\theta}}{1 - \lambda \mathrm{e}^{i\theta}} = \mathrm{e}^{-h_{1B}}. \end{aligned} \tag{5.8}$$

The norm condition (5.7) fixes λ in terms of β and γ. In terms of γ and β Algorithms 1A and 1B have expansions,

$$\begin{aligned} h_{1A} &= \frac{2\beta - (1-\gamma)}{1-\beta} i\theta + \frac{(1-\gamma)(\beta+\gamma)}{2(1-\beta)^2}\theta^2 + O(\theta^3), \\ h_{1B} &= \frac{2\beta - (1-\gamma)}{\gamma+\beta} i\theta + \frac{(1-\gamma)(1-\beta)}{2(\gamma+\beta)^2}\theta^2 + O(\theta^3). \end{aligned} \tag{5.9}$$

Matching the first- and second-order coefficients of the exact exponent

$$h_{ex} = i\eta\theta + r\theta^2, \tag{5.10}$$

then completely determines, for 1A and 1B, respectively,

$$\beta = \frac{1-\gamma+\eta}{2+\eta} \quad \gamma = \frac{1-wr}{1+wr} \quad w = \frac{2}{2+\eta(3+\eta)}, \tag{5.11}$$

$$\beta = \frac{1-\gamma+\gamma\eta}{2-\eta} \quad \gamma = \frac{1-wr}{1+wr} \quad w = \frac{2}{2-\eta(3-\eta)}. \tag{5.12}$$

These are the generalized Robert–Weiss algorithms for solving the advection–diffusion equation, which will be designated as ADRW1A and ADRW1B, respectively. Second-order algorithm can again be derived by applying ADRW1A then ADRW1B according to (2.31).

One can also determine a second-order algorithm directly by matching its amplification exponent:

$$g_2 = \left(\frac{\tilde{\gamma}+\tilde{\lambda}\mathrm{e}^{i\theta}}{1-\tilde{\beta}\mathrm{e}^{-i\theta}}\right)\left(\frac{\tilde{\gamma}+\tilde{\beta}\mathrm{e}^{-i\theta}}{1-\tilde{\lambda}\mathrm{e}^{+i\theta}}\right) = \mathrm{e}^{-h_2}, \tag{5.13}$$

where $\tilde{\gamma} = \gamma(\Delta t/2)$, and so forth. In terms of $\tilde{\gamma}$ and $\tilde{\beta}$, h_2 has the expansion,

$$h_2 = i\theta\left(\frac{(1+\tilde{\gamma})(\tilde{\gamma}-1+2\tilde{\beta})}{(1-\tilde{\beta})(\tilde{\gamma}+\tilde{\beta})}\right) + O(\theta^2). \tag{5.14}$$

Matching this to the first-order coefficient of the exact exponent (5.10) determines

$$\tilde{\beta} = \frac{1}{2}(1-\tilde{\gamma}) + \frac{1}{2}(1+\tilde{\gamma})\tilde{s}, \tag{5.15}$$

and

$$\tilde{\lambda} = \frac{1}{2}(1-\tilde{\gamma}) - \frac{1}{2}(1+\tilde{\gamma})\tilde{s}. \tag{5.16}$$

where $\tilde{s}$ has been previously defined by (3.31). In terms of only $\tilde{\gamma}$,

$$h_2 = i\eta\theta + 2\frac{(1-\tilde{\gamma})}{(1+\tilde{\gamma})}\frac{(1+3\tilde{s}^2)}{(1-\tilde{s}^2)^2}\theta^2 + O(\theta^3), \tag{5.17}$$

and matching the second-order coefficient in (5.10) determines

$$\tilde{\gamma} = \frac{1-wr/2}{1+wr/2} \quad \text{with} \quad w = (1-\tilde{s}^2)^2/(1+3\tilde{s}^2). \tag{5.18}$$

This is then the generalization of advection algorithm C2 to the advection–diffusion case. We shall refer to this algorithm as ADC2. If $\eta = 0$, $\tilde{s} = 0$, one recovers Saulyev's form (2.37) of the diffusion algorithm DS2. If $r = 0$, then $\gamma = 1$ and one recovers the advection algorithm C2 with $\tilde{\lambda} = -\tilde{\beta} = -\tilde{s}$.

In Fig. 2, we illustrate the working of these new algorithms and the effect of this norm-loss error at a large value of $r = 1.33$. For clarity, only results from three representative algorithms ADRW1A, A/D, and ADC2 are shown. The exact solutions are given as black dotted lines. All three algorithms track the exact solution closely prior to the peak passing through the right side

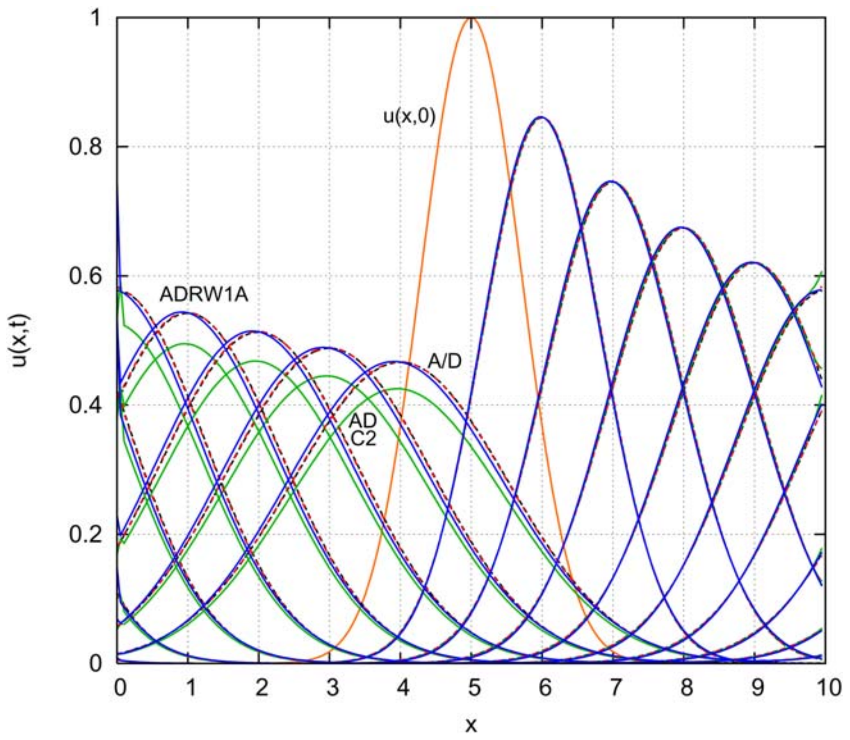

FIG. 2. The propagation of a Gaussian profile in a periodic box of [0,10] with $\Delta x = 0.05$, $\Delta t = 0.033$, $v = 1$, $D = 0.1$, $\eta = 0.66$, and $r = 1.33$. (The conventional first-order diffusion scheme (1.2) is unstable for $r > 0.5$.) The profile is initially centered at $x = 5$. The exact solutions are given as black dotted lines, and are coincided by algorithm A/D's result of red dotted lines. The blue and green lines are results of ADRW1A and ADC2, respectively. The slight asymmetry in ADRW1A's results reflects the residual asymmetry of its underlying first-order diffusion algorithm. All three profiles produced by ADRW1A, ADC2, and A/D are in essential agreement prior to the pulse peak hitting the right periodic edge. As the profiles reappear from the left, the norm of ADC2 is noticeably lower. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

of the periodic box. As the Gaussian peak emerges from the left, algorithm ADC2 suffers an irreversible norm-loss and its peak is noticeably lower. Algorithm A/D basically coincided with the exact solution. Again, ADC2's norm-loss error only occurs for periodic boundary conditions. For nonperiodic applications, ADC2 should be more efficient as it only requires half the computational effort as A/D.

VI. CONCLUSIONS AND FUTURE PROSPECTS

In this work, we have shown that Saul'yev-type schemes can be fundamentally derived from the exponential form of the semidiscrete equation. The unconditional stability of these schemes follows directly from splitting of the exponential matrix. The resulting schemes can be further optimized by matching the scheme's amplification exponent to that of the exact exponent. From this, new second-order Saul'yev-type schemes for solving the advection and the advection–diffusion equations can be derived.

Conventionally, the amplification factor has been used mostly to decide the stability of the algorithm. This work showed that the amplification exponent is of even greater importance, and can be used systematically to improve or even directly derive, as in Section V, Saul'yev-type (or conventional) finite-difference schemes.

Saul'yev-type schemes considered in this work are at most second-order because they only use three grid points. Also, their amplification exponents can only match the exact one to leading order. To eliminate higher order error terms in the exponent, more grid points would be needed. Higher-order Saul'yev-type scheme can then be derived by directly matching the exact amplification exponent, as illustrated in Section V.

The generalization to higher dimensions can be done by dimensional splitting, resulting in unconditionally stable, alternating-direction-explicit methods. For example, in two dimensions, (2.4) generalizes to

$$\mathbf{u}(t+\Delta t) = \mathrm{e}^{\Delta t \mathbf{A}_x}\mathrm{e}^{\Delta t \mathbf{A}_y}\mathbf{u}(t), \tag{6.1}$$

where now $\mathbf{A}_x$ and $\mathbf{A}_y$ are the diffusion matrices in the x and y directions, respectively. One can then decompose each matrix as in the 1D case.

The generalization to nonconstant coefficients is equally easy. For $D(x)$, one can take

$$\mathbf{A}_j = \frac{\tilde{D}_j}{\Delta x^2}\begin{pmatrix} \ddots & & & \\ & -1 & 1 & \\ & 1 & -1 & \\ & & & \ddots \end{pmatrix}, \tag{6.2}$$

with $\tilde{D}_j = D((x_j + x_{j+1})/2)$. The resulting updating matrix elements α_j and β_j will then be spatially dependent and the resulting algorithm can no longer be combined into Saul'yev's form.

Future and more practical applications of these new algorithms will determine their ultimate usefulness.

References

1. R. Courant, K. O. Friedrichs, and H. Lewy, Über die partiellen Differenzengleichungen der mathematischen Physik, Math Anal 100 (1928), 32–74.
2. V. K. Saul'yev, On a method of numerical integration of a diffusion equation, Dokl Akad Nauk SSSR (in Russian) 115 (1957), 1077–1079.
3. V. K. Saul'yev, Integration of equation of earabolic type by the method of nets, Pergamon Press, New York, 1964.
4. B. K. Larkin, Some stable explicit difference approximations to the diffusion equation, Math Comput 18 (1964), 196–201.
5. H. Z. Barakat and J. A. Clark, On the solution of the diffusion equations by numerical methods, Transaction of the ASME, J Heat Transfer 88 (1966), 421–427.
6. K. V. Robert and N. O. Weiss, Convective difference schemes, Math Comput 20 (1966), 272–299.
7. B. F. Towler and R. Y. K. Yang, Numerical stability of the classical and modified Saul'yev finite difference methods, Comput Chem Eng 2 (1978), 45–51.
8. L. J. Campbell and B. Yin, On the stability of alternating-direction explicit methods for advection-diffusion equations, Numer Methods Partial Differential Equations 23 (2007), 1429–1444.

9. Z. Xie, J. Lin, and J. Zhou, A new unconditionally stable explicit scheme for the convection-diffusion equation with Robin boundary conditions, Int J Comput Math 85 (2008), 1833–1847.

10. D. J. Evans and A. R. B. Abdullah, Group explicit methods for parabolic equations, Int J Comput Math 14 (1983), 73–105.

11. D. J. Evans, Alternating group explicit methods for the diffusion equations, Appl Math Model 9 (1985), 201–206.

12. L. Lapidus and G. F. Pinder, Numerical solution of partial differential equations in science and engineering, Wiley, New York, 1982.

13. H. Yoshida, Recent progress in the theory and application of symplectic integrators, Celest Mech Dyn Astron 56 (1993), 27–43.

14. E. Hairer, C. Lubich, and G. Wanner, Geometric numerical integration, Springer-Verlag, Berlin, New York, 2002.

15. R. I. McLachlan and G. R. W. Quispel, Splitting methods, Acta Numer 11 (2002), 241–434.

16. H. F. Trotter, Approximation of semi-groups of operators, Pac J Math 8 (1958), 887–919.

17. G. Strang, On the construction and comparison of difference schemes, SIAM J Numer Anal 5 (1968), 506–517.

18. R. E. Mickens, Nonstandard finite difference models of differential equations, World Scientific, Singapore, 1994.

19. P. D. Lax and R. D. Richtmyer, Survey of the stability of linear finite-difference equations, Commun Pure Appl Math 9 (1956), 267–293.