High-order Exponential Time Differencing Methods for Solving One-dimensional Burgers' Equation

Dingwen Deng*, and Jianqiang Xie

***Abstract*—This paper is devoted to the development and application of two high-order numerical methods for solving one-dimensional (1D) Burgers' equations, which are both fourth-order accurate in both time and space. One of them is based on the uses of Crank-Nicolson (CN) method combined with Richardson extrapolation method for temporal integration and fourth-order compact finite difference approximation for spatial discretization. Additionally, a combination of fourth-order time stepping method based on Padé approximation for temporal discretization with fourth-order compact finite difference method for spatial approximation yields the other fourth-order method. Using matrix analysis methods, we study their stability. Numerical experiments illustrate the accuracy and efficiency of new algorithms.**

***Index Terms*—Burgers equation; Hopf-Cole transformation; Compact finite difference scheme; Padé approximation; Stability;**

I. Introduction

In this article, we consider the numerical simulations of 1D Burgers' equation with nonhomogeneous boundary conditions as follows

$$\begin{cases} \dfrac{\partial u}{\partial t} + u\dfrac{\partial u}{\partial x} = \gamma \dfrac{\partial^2 u}{\partial x^2}, & (x,t) \in [a,b] \times [0,T], \\ u(a,t) = c, \quad u(b,t) = d, & t \in [0,T], \\ u(x,0) = \phi(x), & x \in [a,b], \end{cases} \tag{1}$$

where γ, a, b, c and d are all constants and γ is the coefficient of kinematic viscosity. This equation was first found by Bateman [1] and later applied to model free turbulence by Burgers [2], [3]. Since then, it is extensively referred to as Burgers' equation [1]–[6], which has became a very important nonlinear evolution equation because of its importance in various fields such as gas and fluid dynamics, traffic flow, heat conduction, wave propagation in acoustics and hydrodynamics, etc. Consequently, considerable attention has been paid on the study of Burgers equation. For example, using Hopf-Cole transformation, we can obtain exact solutions in terms of infinite Fourier series for given initial condition, which fail to converge for relatively small γ, such as $\gamma < 0.01$. Moreover, several classes of special analytic solutions can also be obtained by using homotopy perturbation method and differential transformation method [7] and tanh function expansion method [8], etc. However, it is very difficult to find the useful expression of exact solution to Burgers with arbitrary initial and boundary conditions. Thus, over the years, numerical studies for initial-boundary value problem (IBVP) (1) have attracted widespread attention. Many numerical methods including finite difference methods (FDMs) [9]–[15], finite element methods (FEMs) [16], [17], local discontinuous Galerkin methods [18], [19], spectral methods [20], [21], collocation method based on the Laplace transform [22], meshless approach [23] and differential quadrature methods [24], [25] have been developed and applied to solve IBVP (1).

Recently, high-order compact (HOC) FDMs, which have been widely used to deal with various computational problems arising from a wide range of applied fields due to their high accuracy, compactness and better resolution for high-frequency waves [26]–[29], have been proposed for solving Burgers equation. For example, a HOC FDM, which is second-order temporally accurate and fourth-order spatially accurate, has been developed in [9]. A sixth-order compact FDM combined with explicit third-order total variation diminishing (TVD) Runge-Kutta method (RKM) has also been developed in [13]. They were both directly constructed for IBVP (1) without the use of Hopf-Cole transformation. Whereas, another two HOC FDMs were established for linear diffusion equation obtained using Hopf-Cole transformation to Burgers equation in [11], [14], respectively. HOC methods stated above have been limited to the use of small time steps due to low-order accuracy in time, or strong stability restriction. Besides, low-order accuracy at boundary results in the increase of global error. For example, in the case of nonhomogeneous boundary, HOC FDM in [14] has only second-order accuracy at boundary, thus reducing resolution of numerical solutions.

Recently, a class of combined schemes, which consist of high-order time stepping methods based on Padé approximations and second-order centered difference for spatial variable, have been derived for linear diffusion equation obtained using Hopf-Cole transformation to Burger's equation (see [15]). Although they are high-order accurate in time and unconditionally stable, they have only second-order accuracy in space. Therefore, they may generate numerical solution of poor quality if the spatial mesh is not refined sufficiently. However, mesh refinement may lead to a large number of arithmetic operations.

More recently, a unconditionally stable fourth-order numerical method, which combines fourth-order time stepping

Manuscript received October 25, 2015; revised January 12, 2016. This work is partially supported by the National Natural Science Foundation of China (Grant Nos. 11401294, 11326046, 11261040), Youth Natural Science Foundation of Jiangxi Provincial Education Department (Grant No. GJJ14545) and Youth Natural Science Foundation of Jiangxi province (Grant No. 20142BAB211003).

Dingwen Deng is with the College of Mathematics and Information Science, Nanchang Hangkong University, Nanchang 330063, China (Corresponding Author, e-mail: dengdingwen2010@163.com)

Jianqiang Xie is with the College of Mathematics and Information Science, Nanchang Hangkong University, Nanchang 330063, China

methods based on Padé approximations with fourth-order compact FDM for spatial variable, have been proposed for convection-diffusion equation [30]. Numerical results testify high-performance and usefulness of that algorithm.

In this paper, enlightened by the work of [30], we derive two fourth-order numerical methods for Burgers equation. First, IBVP (1) is transformed into a linear diffusion equation with mixed boundary conditions by using the Hopf-Cole transformation. Secondly, a compact FDM which has fourth-order accuracy at both interior and boundary points has been derived for linearized equation, thus resulting in an initial value problem (IVP), which can be solved exactly using Duhamel's principle. Thirdly, using [1,1]-Padé to approximate matrix exponential function results in a second-order CN scheme, which can be improved to fourth-order accuracy in time by Richardson extrapolation method and the application of [2,2]-Padé approximation to matrix exponential function generates a fourth-order time-stepping scheme. Finally, the use of Simpson's integrable formula to Hopf-Cole transformation in subinterval $[x_{j-1}, x_{j+1}]$ (see Section 2) yields fourth-order approximate solution of IBVP (1) according to fourth-order numerical solution of linearized equation. The new methods overcome some deficiencies of those algorithms proposed in [9], [11], [13]–[15].

This paper is organized as follows. Construction and stability analysis of numerical algorithms are studied for IBVP (1) in Section 2 and Section 3, respectively. In Section 4, five examples are carried out to test the performance of our algorithms. Final section focuses on concluding remarks.

II. Fourth-order numerical methods

This section concentrates on the derivation of new numerical methods.

A. Notations and auxiliary Lemmas

$\Delta t = T/K$, $t_k = k\Delta t$, $0 \le k \le K$. Moreover, $h = (b-a)/n$ represents grid spacing. The spatial grid nodes $x_j = a + jh$, $j = 0, 1, \ldots, n$ form the following sets $\bar{\Omega}_h = \{x_j | 0 \le j \le n\}$. On $\bar{\Omega}_h$, we define grid function space $\mathbf{S}_h = \{\mathbf{w} | (w_0, w_1, \ldots, w_{n-1}, w_n)^T\}$ and introduce centered difference operator $\delta_x^2 w_j = (w_{j+1} - 2w_j + w_{j-1})/h^2$. To make this paper self-contained, we first give two lemmas used later.

Lemma 1 (cf. [30]) Assume that $w(x) \in \mathcal{C}^5[a,b]$, then

$$\begin{aligned} w^{'}(x_0) &= \frac{w(x_1) - w(x_0)}{h} - \frac{5h}{12} w^{''}(x_0) \\ &\quad - \frac{h}{12} w^{''}(x_1) - \frac{h^2}{12} w^{(3)}(x_0) + \mathcal{O}(h^4), \\ w^{'}(x_n) &= \frac{w(x_n) - w(x_{n-1})}{h} + \frac{5h}{12} w^{''}(x_n) \\ &\quad + \frac{h}{12} w^{''}(x_{n-1}) - \frac{h^2}{12} w^{(3)}(x_n) + \mathcal{O}(h^4). \end{aligned}$$

Lemma 2 (cf [30]) If $\forall\ z \in \mathbf{C}$ and has non-positive real part, then the following inequalities

$$\left|\frac{2+z}{2-z}\right| \le 1, \qquad \left|\frac{1 + z/2 + z^2/12}{1 - z/2 + z^2/12}\right| \le 1$$

hold.

III. Establishment of numerical method

Applying Hopf-Cole transformation to the Burgers' Eq. (1):

$$u(x,t) = -2\gamma \frac{v_x(x,t)}{v(x,t)}, \tag{2}$$

Eq. (1) can be rewritten equivalently as

$$\begin{cases} \dfrac{\partial v}{\partial t} = \gamma \dfrac{\partial^2 v}{\partial x^2}, \quad (x,t) \in (a,b) \times [0,T], \\ 2\gamma v_x(a,t) + c v(a,t) = 0, \\ 2\gamma v_x(b,t) + d v(b,t) = 0, t \in [0,T], \\ v(x,0) = \exp\Big\{ -\displaystyle\int_a^x \frac{\phi(s)}{2\gamma} ds \Big\}, \quad a \le x \le b, \end{cases} \tag{3}$$

In this paper, $V_j(t)$ and V_j^k denote the approximations to $v(x_j,t)$ and $v(x_j,t_k)$, respectively, whereas the approximation to $u(x_j,t_k)$ is represented by U_j^k. Clearly, corresponding vectors $\mathbf{V}^k(t)$, $\mathbf{V}^k$ and $\mathbf{U}^k$ belong to $\mathbf{S}_h$.

First of all, we develop fourth-order spatial discretization at boundary nodes. Using Lemma 1, it is not difficult to find that

$$\begin{aligned} \frac{\partial v(x_0,t)}{\partial x} &= \frac{v(x_1,t) - v(x_0,t)}{h} - \frac{5h}{12} \frac{\partial^2 v(x_0,t)}{\partial x^2} \\ &\quad - \frac{h}{12} \frac{\partial^2 v(x_1,t)}{\partial x^2} - \frac{h^2}{12} \frac{\partial^3 v(x_0,t)}{\partial x^3} + \mathcal{O}(h^4), \end{aligned} \tag{4}$$

$$\begin{aligned} \frac{\partial v(x_n,t)}{\partial x} &= \frac{v(x_n,t) - v(x_{n-1},t)}{h} + \frac{5h}{12} \frac{\partial^2 v(x_n,t)}{\partial x^2} \\ &\quad + \frac{h}{12} \frac{\partial^2 v(x_{n-1},t)}{\partial x^2} - \frac{h^2}{12} \frac{\partial^3 v(x_n,t)}{\partial x^3} + \mathcal{O}(h^4). \end{aligned} \tag{5}$$

By Eq. (3), we have that

$$\begin{aligned} &v_{xx}(x_0,t) = \frac{1}{\gamma} v_t(x_0,t), \qquad v_{xx}(x_1,t) = \frac{1}{\gamma} v_t(x_1,t), \\ &v_{xxx}(x_0,t) = \frac{1}{\gamma} v_{tx}(x_0,t) = -\frac{c}{2\gamma^2} v_t(x_0,t). \end{aligned} \tag{6}$$

Inserting above equations (6) into Eq. (4) yields

$$\begin{aligned} &\Big(\frac{5}{12} - \frac{ch}{24\gamma}\Big) v_t(x_0,t) + \frac{1}{12} v_t(x_1,t) \\ &= \Big(\frac{c}{2h} - \frac{\gamma}{h^2}\Big) v(x_0,t) + \frac{\gamma}{h^2} v(x_1,t). \end{aligned} \tag{7}$$

Using the technique similar to that used in the derivation of (7), it holds that

$$\begin{aligned} &\Big(\frac{5}{12} + \frac{dh}{24\gamma}\Big) v_t(x_n,t) + \frac{1}{12} v_t(x_{n-1},t) \\ &= \frac{\gamma}{h^2} v(x_{n-1},t) + \Big(-\frac{d}{2h} - \frac{\gamma}{h^2}\Big) v(x_n,t). \end{aligned} \tag{8}$$

Secondly, the application of fourth-order compact finite difference method to discretize second-order spatial derivative for Eq. (3) at interior nodes gives that

$$\begin{aligned} &v_t(x_j,t) = \gamma \frac{\delta_x^2}{1 + (h^2\delta_x^2)/12} v(x_j,t) + \mathcal{O}(h^4) \\ &j = 1, 2, \ldots, n-1, \end{aligned} \tag{9}$$

which can be equivalently written as

$$\begin{aligned} &\Big(1 + \frac{h^2\delta_x^2}{12}\Big) v_t(x_j,t) = \gamma \delta_x^2 v(x_j,t) + \mathcal{O}(h^4) \\ &j = 1, 2, \ldots, n-1. \end{aligned} \tag{10}$$

Finally, we introduce two tri-diagonal matrices of order $n+1$ as follows,

$$A = \begin{pmatrix} \frac{5}{12}-\frac{ch}{24\gamma} & \frac{1}{12} & & & \\ \frac{1}{12} & \frac{5}{6} & \frac{1}{12} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{1}{12} & \frac{5}{6} & \frac{1}{12} \\ & & & \frac{1}{12} & \frac{5}{12}+\frac{dh}{24\gamma} \end{pmatrix}_{(n+1)\times(n+1)},$$

$$B = \begin{pmatrix} \eta_1 & \frac{\gamma}{h^2} & & & \\ \frac{\gamma}{h^2} & -\frac{2\gamma}{h^2} & \frac{\gamma}{h^2} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{\gamma}{h^2} & -\frac{2\gamma}{h^2} & \frac{\gamma}{h^2} \\ & & & \frac{\gamma}{h^2} & \eta_2 \end{pmatrix}_{(n+1)\times(n+1)}.$$

, where $\eta_1 = c/(2h) - \gamma/h^2$ and $\eta_2 = -\gamma/h^2 - d/(2h)$.

Therefore,it follows from (7), (8) and (10) that we obtain a semi-discretization scheme of IBVP (3) as follows

$$\begin{cases} \dfrac{d\mathbf{V}(t)}{dt} = (A^{-1}B)\mathbf{V}(t), \qquad 0 \le t \le T, \\ \mathbf{V}(0) = \mathbf{V}^0, \end{cases} \tag{11}$$

whose exact solution is $\mathbf{V}(t) = \exp(A^{-1}Bt)\mathbf{V}(0)$, which implies that

$$\mathbf{V}(t_{k+1}) = \exp(A^{-1}B\Delta t)\mathbf{V}(t_k).$$

Here, we should suitably choose meshsize h as follows. (1) As $c \le 0$ and $d \ge 0$, one can choose arbitrary meshsize h. (2) As $c \le 0$ and $d < 0$, meshsize h should satisfy $h \le (-8\gamma)/d$. (3) As $c > 0$ and $d \ge 0$, meshsize h should admits $h \le (-8\gamma)/c$. (4) As $c > 0$ and $d < 0$, meshsize h should satisfy $h \le \min\{8\gamma/c, (-8\gamma)/d\}$. The selections of h stated above can make matrix A strictly diagonally dominant, and thus ensure the invertibility of the matrix A.

In what follows, we consider time integration. Clearly, applying $[1,1]$-Padé approximation to $\exp(x)$ yields that

$$[2I - (A^{-1}B)\Delta t]\mathbf{V}^{k+1} = [2I + (A^{-1}B)\Delta t]\mathbf{V}^k, \tag{12}$$

which has a truncation error in the form of $\mathcal{O}(\Delta t^2 + \Delta t^4 + h^4)$. This is CN scheme. Denote the numerical solution of $v(x_j, t_k)$ obtained using CN scheme (12) with meshsizes Δt and h by $V_j^k(\Delta t, h)$. So, the local Richardson extrapolation method defined as follow

$$V_j^{k+1} = \frac{4V_j^{2(k+1)}(0.5\Delta t, h) - V_j^{k+1}(\Delta t, h)}{3} \tag{13}$$

can be used to eliminate the term $\mathcal{O}(\Delta t^2)$, thus obtain numerical solution of order 4 in both time and space.

Furthermore, if $[2,2]$-Padé approximation to $\exp(x)$ is used, we obtain

$$\begin{aligned} &[12I - 6(A^{-1}B)\Delta t + (A^{-1}B)^2\Delta t^2]\mathbf{V}^{k+1} \\ &= [12I + 6(A^{-1}B)\Delta t + (A^{-1}B)^2\Delta t^2]\mathbf{V}^k, \end{aligned} \tag{14}$$

which has convergence order of $\mathcal{O}(\Delta t^4 + h^4)$.

Integrating (2) with respect to variable x on the interval $[x_{j-1}, x_{j+1}]$ for $j = 1, 2, \ldots, n-1$ gives that

$$\begin{aligned} \int_{x_{j-1}}^{x_{j+1}} u(x, t_{k+1})dx &= -2\gamma \int_{x_{j-1}}^{x_{j+1}} \frac{v_x(x, t_{k+1})}{v(x, t_{k+1})}dx \\ &= -2\gamma \ln\left|\frac{v(x_{j+1}, t_{k+1})}{v(x_{j-1}, t_{k+1})}\right| \end{aligned}$$

Applying Simpson's rule for the integration to above equation deduces that

$$\begin{aligned} &u(x_{j-1}, t_{k+1}) + 4u(x_j, t_{k+1}) + u(x_{j+1}, t_{k+1}) \\ &= -\frac{6\gamma}{h}\ln\left|\frac{v(x_{j+1}, t_{k+1})}{v(x_{j-1}, t_{k+1})}\right| + \mathcal{O}(h^4). \end{aligned}$$

Omitting the truncation error and replacing $v(x_j, t_{k+1})$ by V_j^{k+1} in above equation gives that

$$\begin{cases} 4U_1^{k+1} + U_2^{k+1} = F_1^{k+1} - u(x_0, t_{k+1}), \\ U_{j-1}^{k+1} + 4U_j^{k+1} + U_{j+1}^{k+1} = F_j^{k+1}, \\ (j = 2, \ldots, n-2), \\ U_{n-2}^{k+1} + 4U_{n-1}^{k+1} = F_{n-1}^{k+1} - u(x_n, t_{k+1}), \end{cases} \tag{15}$$

where $u(x_0, t_{k+1}) = c$, $u(x_n, t_{k+1}) = d$ and

$$F_j^{k+1} = -\frac{6\gamma}{h}\ln\left|\frac{V_{j+1}^{k+1}}{V_{j-1}^{k+1}}\right|, \quad j = 1, 2, \ldots, n-1.$$

As algebraic equations (15), whose coefficient matrix is strictly diagonally dominant, is a tridiagonal linear system, it owns unique solution and can be easily solved by Thomas algorithm.

Finally, for clearness and convenience, it is worthwhile concluding our algorithms as follows. Suppose that $\mathbf{V}^k$ is known.

Algorithm 1: $\mathbf{V}^{k+1}$ is firstly computed using CN scheme (12), then $\mathbf{U}^{k+1}$ is obtained by the use of the Thomas algorithm to (15).

Algorithm 2: $\mathbf{V}^{k+1}(\Delta t, h)$ and $\mathbf{V}^{2(k+1)}(0.5\Delta t, h)$ are obtained using CN scheme (12) with $(\Delta t, h)$ and $(0.5\Delta t, h)$, respectively. Then from extrapolation method (13) we have $\mathbf{V}^{k+1}$. Finally, we obtain $\mathbf{U}^{k+1}$ by applying the Thomas algorithm to (15).

Algorithm 3: Firstly compute $\mathbf{V}^{k+1}$ by solving equation (14), then calculate $\mathbf{U}^{k+1}$ by the application of the Thomas algorithm to (15).

Obviously, Algorithm 1 has a convergence rate of $\mathcal{O}(\Delta t^2 + h^4)$, while Algorithm 2 and Algorithm 3 are both of order four in both time and space. This conclusion is testified numerically in section 4.

IV. Stability analysis

In this section, we only study the stability of the present methods applied Burgers' equation with homogeneous boundary conditions, (i.e. $c = d = 0$).

Theorem 1 Let $c = d = 0$. Then the eigenvalues of matrix $A^{-1}B$ are all real and non-positive.

Proof. As $c = d = 0$, we easily find that the matrix A is a strictly diagonally dominant, symmetric and real matrix, which infers that A^{-1} exists and its eigenvalues are all real, and the matrix B is also a symmetric and real matrix, which implies that the eigenvalues of matrix B are all real, too. Whereas, the eigenvalues of $A^{-1}B$ are all real.

Let λ be an arbitrary eigenvalue of $A^{-1}B$, and $\mathbf{x} \in \mathbf{R^{n+1}}$ be corresponding eigenvector. Then we have that $(A^{-1}B)\mathbf{x} = \lambda\mathbf{x}$, which further implies that $\lambda\mathbf{x}^T A\mathbf{x} = \mathbf{x}^T B\mathbf{x}$. By simple computation, we easily find that $\mathbf{x}^T B\mathbf{x} = -\dfrac{\gamma}{h^2}\sum_{j=2}^{n}(x_{j-1} - x_j)^2 \le 0$, and $\mathbf{x}^T A\mathbf{x} == \dfrac{1}{3}x_1^2 + \dfrac{2}{3}\sum_{j=2}^{n-1}x_j^2 +$

$\frac{1}{3}x_n^2 + \frac{1}{12}\sum_{j=1}^{n-1}(x_j + x_{j+1})^2 \geq 0$, which imply that λ must be less than zero to make $\lambda \mathbf{x}^T A\mathbf{x} = \mathbf{x}^T B\mathbf{x}$ hold.

Theorem 2 The present methods applied to Burgers' equation with homogeneous boundary conditions are unconditionally stable.

Proof. Let λ_j $(j = 1, 2, \ldots, n+1)$ be eigenvalues of matrix $A^{-1}B$. Write $P = [2I - (A^{-1}B)\Delta t]^{-1}[2I + (A^{-1}B)\Delta t]$, Then, we have that the eigenvalues of matrix P

$$(\lambda(P))_j = \frac{2 + \lambda_j \Delta t}{2 - \lambda_j \Delta t}, \qquad j = 1, 2, \ldots, n+1.$$

As $\lambda_j \leq 0$, according to lemma 2, it is easy to find that $\rho(P) = \max_j |(\lambda(P))_j| \leq 1$, where $\rho(P)$ represents the spectral radius of P. So CN scheme (12) (i.e. Algorithm 1) is unconditionally stable.

As extrapolation solution is a linear combination of two numerical solutions obtained using CN scheme (12) with meshsize $(\Delta t, h)$ and $(0.5\Delta t, h)$, respectively, extrapolation solution is also stable.

Denote $Q = [12I - 6(A^{-1}B)\Delta t + (A^{-1}B)^2\Delta t^2]^{-1}[12I + 6(A^{-1}B)\Delta t + (A^{-1}B)^2\Delta t^2]$. We easily find that the eigenvalues of the matrix Q

$$(\lambda(Q))_j = \frac{12 + 6\Delta t\lambda_j + (\lambda_j \Delta t)^2}{12 - 6\Delta t\lambda_j + (\lambda_j \Delta t)^2}, \qquad j = 1, 2, \ldots, n+1.$$

Likewise, by Lemma 2, $\lambda_j \leq 0$ implies that $\rho(Q) = \max_j |(\lambda(Q))_j| \leq 1$, which show that difference scheme (14) (i.e. Algorithm 3) is also unconditionally stable.

V. Numerical Examples

In this section, five IBVPs are solved to illustrate the performance of our algorithms. L^2- and L^∞-norm errors at $t = k\Delta t$ between exact and numerical solutions are defined by

$$err2 = \left[\sum_{j=1}^{n-1}(U_j^k - u_j^k)^2 h\right]^{\frac{1}{2}}, \qquad err = \max_{1\leq j\leq n-1} |U_j^k - u_j^k|,$$

respectively, and CPU time are applied to measure the accuracy and efficiency of the new algorithms. Convergence rates in L^∞- and L^2-norms are defined as follows: rate=$\log_2\left[\frac{err(2h)}{err(h)}\right]$ and rate2=$\log_2\left[\frac{err2(2h)}{err2(h)}\right]$, respectively, as $\Delta t = h$, (see Tables I, III and IV).

As we know, TVD RKMs, which own the property of preserving strong stability [19], [31], have been proven to be very useful in the simulations of discontinuous problems. For comparing computational efficiency between them and [2,2]-Padé, a famous third-order TVD RKM (3-TVD-RKM) (cf. [13], [19], [31]) has been applied to solve the corresponding IVPs (11) in Example 2 and Example 3. Moreover, All computer programs were coded in Matlab 7.0.

Example 1 To test the accuracy of our algorithms, we consider Burgers' equation (1) with initial and boundary conditions [11]

$$u(x, 0) = 2\gamma\frac{\pi \sin(\pi x)}{\sigma + \cos(\pi x)}, \qquad x \in (0, 1),$$
$$u(0, t) = u(1, t) = 0, \qquad t \in (0, T],$$

where $\sigma > 1$ is a parameter.

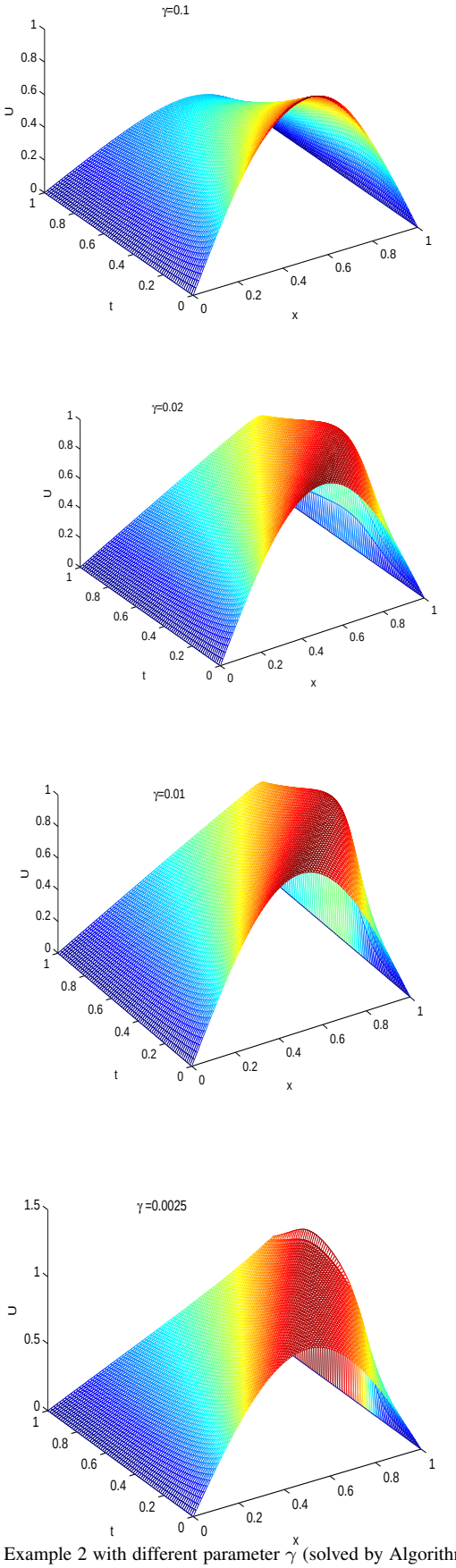

Fig. 1. Example 2 with different parameter γ (solved by Algorithm 3 with $\Delta t = h = 0.0125$): Time evolution graphs of numerical solution at $t = 1$.

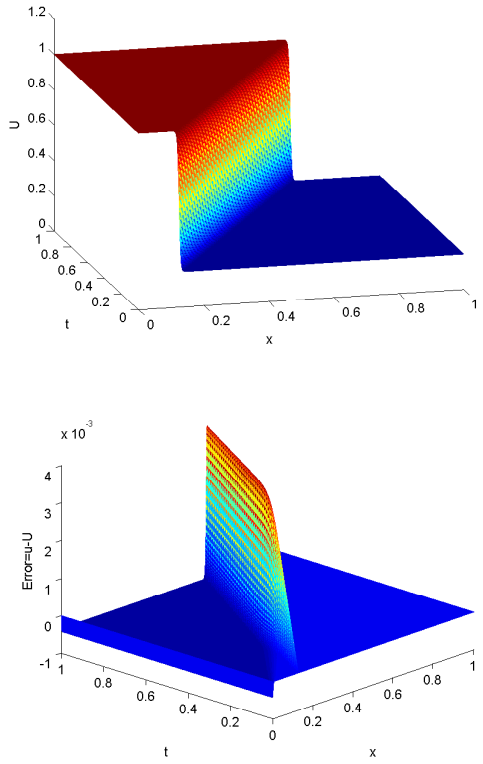

Fig. 2. Example 3 with $\gamma = 0.001$ (solved by Algorithm 3 with $\Delta t = h = 0.001$): Numerical solution and corresponding errors

Here $v(x,0)$ and corresponding exact solution can be respectively derived as follows:

$$v(x,0) = \frac{\sigma + \cos(\pi x)}{\sigma + 1} \qquad u(x,t) = \frac{2\gamma\pi e^{-\pi^2\gamma t}\sin(\pi x)}{\sigma + e^{-\pi^2\gamma t}\cos(\pi x)}.$$

where $(x,t) \in [0,1]\times[0,T]$. For this problem, we give $\sigma = 2$ and $\gamma = 0.1$, and display the numerical results in Tables I-IV. As computational cost is very low for running one time, to accurately assess the performance of three methods, for a fixed grid, we run 1000 times, then take the average computational time as time cost (CPU time) in Tables I-IV. From these data, we can give several conclusions as follows:

(1) Table I shows that Algorithm 1 has a convergence order of $\mathcal{O}(h^2)$ in L^2- and L^∞-norms as $\Delta t = h$. Meanwhile, from Table II, we also can see that as Δt and h are decreased by a factor of $1/2$ and $1/4$ each time, respectively, L^2- and L^∞-errors are approximately reduced by a factor of $1/16$. These results exactly show that Algorithm 1 has a convergence rate of $\mathcal{O}(\Delta t^2 + h^4)$ in L^2- and L^∞-norms.

(2) From Table III and Table IV, we can find that both Algorithm 2 and Algorithm 3 are fourth-order in both time and space with respect to L^2- and L^∞-norms, however, Algorithm 3 costs less time than Algorithm 2 in the case of attaining almost same error tolerance. Furthermore, in comparison with data in Table II, it is clear that Algorithm 3 is the most efficient, and Algorithm 2 is more efficient than Algorithm 1. For example, $err = 8.2508e-10$ obtained using Algorithm 1 with $h = 1/128$ and $\Delta t = 1/5120$ costs $7.156e-02$ s; $err = 5.3096e-10$ provided using Algorithm 2 with $h = \Delta t = 1/128$ costs $1.475e-02$ s; $err = 5.3029e-10$ generated using Algorithm 3 with $h = \Delta t = 1/128$ costs the least CPU time, namely $4.828e-03$ s.

Example 2 For comparing with other existing numerical methods, we consider the following Burgers' equation (1) with initial and boundary conditions [11]

$$u(x,0) = \sin(\pi x), \quad x \in (0,1),$$
$$u(0,t) = u(1,t) = 0, \quad t \in (0,T].$$

For this problem, using the Hopf-Cole transformation yields the following exact solution

$$u(x,t) = 2\pi\gamma \frac{\sum_{n=1}^{\infty} c_n \exp(-n^2\pi^2\gamma t) n\sin(n\pi x)}{c_0 + \sum_{n=1}^{\infty} c_n \exp(-n^2\pi^2\gamma t)\cos(n\pi x)}, \quad (16)$$

in which coefficients are defined as follows

$$c_0 = \int_0^1 \exp(-\frac{1-\cos(\pi x)}{2\pi\gamma})dx, \qquad c_n =$$
$$2\int_0^1 \exp(-\frac{1-\cos(\pi x)}{2\pi\gamma})\cos(n\pi x)dx, \ \ (n = 1,2,3,\ldots).$$

Meanwhile, it is easy to find that

$$v(x,0) = \exp(-\frac{1-\cos(\pi x)}{2\pi\gamma}), \qquad x \in [0,1].$$

For testing the accuracy, Fourier series (16) should be evaluated in this example, where a number N is taken such that the coefficient c_N is less than $1.0e-15$.

Table V and Table VI illustrate that with the same meshsize, Algorithm 3 is almost as accurate as Algorithm 2, but faster than Algorithm 2, and has an evident advantage in terms of accuracy comparing with FEM [17], FDM [10] and HOC FDM [14].

As we know, because 3-TVD-RKM is conditionally stable, a very small time step compared with spatial increment should be used, thus consuming expensive time. From these two tables, we can find that 3-TVD-RKM is slower than Algorithm 3 under condition of achieving almost the same accurate solutions, and 3-TVD-RKM with $h = 1/160$ and $\Delta t = 1.0e-03$ is invalid for this problem. These results illustrate that although 3-TVD-RKM is very useful for discontinuous problems, it may be less efficient than [2,2]-Padé in the solutions of continuous problems. Similar numerical results can be found in the Table VIII of the next example.

Time evolution of numerical solutions for different γ are shown in Figure 1, from which we can observe the interesting physical phenomenon of this problem.

Example 3 We consider solution of IBVP (1) with boundary condition $u(0,t) = 1$, $u(1,t) = 0.2$ and initial condition $u(x,0) = \{[\alpha+\mu+(\mu-\alpha)\exp[\alpha(x-\beta)/\gamma]\}/\{1+\exp[\alpha(x-\beta)/\gamma]\}$, whose exact solution, i.e. traveling wave is

$$u(x,t) = \frac{\alpha+\mu+(\mu-\alpha)\exp(\eta)}{1+\exp(\eta)}$$

where $\eta = \alpha(x-\mu t-\beta)/\gamma$, and α, β and μ are constants. Corresponding initial condition of IBVP (3) is

$$v(x,0) = \exp[\frac{(\alpha-\mu)(x-a)}{2\gamma}]\frac{1+\exp\{[\alpha(\beta-x)]/\gamma\}}{1+\exp\{[\alpha(\beta-a)]/\gamma\}}.$$

Like literatures [14], we take parameters $\alpha = 0.4$, $\mu = 0.6$, and $\beta = 0.125$. Numerical results are listed in Tables VII–IX. From these data, we can deduce the following Conclusions: (1) For a fixed meshsize, solution obtained

TABLE I
COMPUTATIONAL RESULTS AT $t = 1$ FOR EXAMPLE 1, OBTAINED USING ALGORITHM 1 WITH $\Delta t = h$.

h	$1/4$	$1/8$	$1/16$	$1/32$	$1/64$	$1/128$
err	$1.2734e-03$	$1.7144e-04$	$3.9949e-05$	$9.8571e-06$	$2.4570e-06$	$6.1391e-07$
$rate$	–	2.8929	2.1015	2.0189	2.0043	2.0008
$err2$	$6.6575e-04$	$1.1596e-04$	$2.7471e-05$	$6.7935e-06$	$1.6940e-06$	$4.2323e-07$
$rate2$	–	2.5214	2.0776	2.0157	2.0037	2.0009
CPU	$6.312e-05$	$7.900e-05$	$1.260e-04$	$2.810e-04$	$9.070e-04$	$4.907e-03$

TABLE II
NUMERICAL RESULTS AT $t = 1$ FOR EXAMPLE 1, OBTAINED USING ALGORITHM 1.

h	Δt	err	$\frac{err(h,\Delta t)}{err(0.5h,0.25\Delta t)}$	$err2$	$\frac{err2(h,\Delta t)}{err2(0.5h,0.25\Delta t)}$	CPU
$\frac{1}{4}$	$\frac{1}{5}$	$1.0774e-03$	–	$5.4417e-04$	–	$6.450e-05$
$\frac{1}{8}$	$\frac{1}{20}$	$5.5860e-05$	19.2874	$2.9620e-05$	18.3721	$9.400e-05$
$\frac{1}{16}$	$\frac{1}{80}$	$3.3830e-06$	16.5120	$1.7829e-06$	16.6129	$2.340e-04$
$\frac{1}{32}$	$\frac{1}{320}$	$2.1206e-07$	15.9532	$1.1044e-07$	16.1443	$9.530e-04$
$\frac{1}{64}$	$\frac{1}{1280}$	$1.3212e-08$	16.0501	$6.8870e-09$	16.0356	$6.718e-03$
$\frac{1}{128}$	$\frac{1}{5120}$	$8.2508e-10$	16.0134	$4.3022e-10$	16.0082	$7.156e-02$

TABLE III
COMPUTATIONAL RESULTS AT $t = 1$ FOR EXAMPLE 1 USING ALGORITHM 2 WITH $\Delta t = h$.

h	$1/4$	$1/8$	$1/16$	$1/32$	$1/64$	$1/128$
err	$7.3018e-04$	$3.5360e-05$	$2.2402e-06$	$1.3687e-07$	$8.5058e-09$	$5.3096e-10$
$rate$	–	4.3680	3.9804	4.0328	4.0081	4.0018
$err2$	$3.9475e-04$	$1.9135e-05$	$1.1063e-06$	$6.7827e-08$	$4.2190e-09$	$2.6341e-10$
$rate2$	–	4.3667	4.1124	4.0277	4.0069	4.0015
CPU	$1.982e-04$	$2.442e-04$	$3.880e-04$	$8.450e-04$	$2.821e-013$	$1.475e-02$

TABLE IV
COMPUTATIONAL RESULTS AT $t = 1$ FOR EXAMPLE 1, USING ALGORITHM 3 WITH $\Delta t = h$.

h	$1/4$	$1/8$	$1/16$	$1/32$	$1/64$	$1/128$
err	$7.3044e-04$	$3.5368e-05$	$2.2410e-06$	$1.3691e-07$	$8.5087e-09$	$5.3029e-10$
$rate$	–	4.3682	3.9803	4.0328	4.0082	4.0041
$err2$	$3.9479e-04$	$1.9137e-05$	$1.1064e-06$	$6.7835e-08$	$4.2195e-09$	$2.6330e-10$
$rate2$	–	4.3666	4.1124	4.0277	4.0069	4.0023
CPU	$6.25e-05$	$7.801e-05$	$1.250e-04$	$2.650e-04$	$9.060e-04$	$4.828e-03$

using Algorithm 1 is more accurate than the one provided using HOC FDM [14], and needs almost the same CPU time as HOC FDM [14]. This meets our anticipation because HOC FDM [14] has only second-order accuracy at boundary. (2) Using the same meshsize and relatively small γ, Algorithm 3 has approximately as high accuracy as Algorithm 2, and is much more accurate than Algorithm 1 and HOC FDM [14]. Meanwhile, computational time of Algorithm 3 is much lower than the one of Algorithm 2, and is roughly as much as Algorithm 1 and HOC FDM [14]. (3) For very small γ, with the same meshsize, solution computed using Algorithm 3 is more accurate than the one provided than Algorithm 2. In a word, Algorithm 3 is the most efficient in the aspects of accuracy and computational cost.

Finally, the use of Algorithm 3 with $\Delta t = h = 0.001$ to this problem with $\gamma = 0.001$ is carried out. Evolution graph of numerical solution and the propagation of errors are plotted in the left and right columns of Figure 2, respectively, from which we can observe that the errors become large when the values of the solution vary violently. This shows that Algorithm 3 has a good capacity of simulation for a wide range of γ.

Example 4 In what follows, the IBVP (1) which has a shock-wave solution [14]

$$u(x,t) = \frac{\lambda}{2}\{1 + \tanh[\frac{\lambda}{8\gamma}(-2x + \lambda t)]\}$$

is solved using our methods and HOC FDM [14]. Obviously, corresponding initial condition of IBVP (3) is

$$v(x,0) = \exp[\alpha(x-a)]\frac{\cosh(\alpha x)}{\cosh(\alpha a)},$$

in which $\alpha = (-\lambda)/(4\gamma)$. In computation process, we choose $\lambda = 1.6$ and boundary conditions $u(-5,t) = 1.6$ and $u(10,t) = 0$.

Numerical results listed in Table X and Table XI further show the evident superiority of Algorithm 3 over Algorithm 2, Algorithm 1, and HOC FDM [14] in terms of accuracy and computational cost.

Figure 3 shows the graphs of numerical solutions to example 4 with $\gamma = 0.1$ at $t = 0.25,\ 1.5,\ 2.75$ and the propagation of the corresponding errors on mesh points. From this figure, we can see that the error becomes much large in the vicinity of the place, where the gradient of the solution varies very quickly.

Example 5 Finally, we consider numerical simulation of Burgers' equation with the initial and boundary conditions

$$u(x,0) = \begin{cases} 1, & 0 \le x < 5, \\ 6 - x, & 5 \le x < 6, \\ 0, & 6 \le x < 12. \end{cases}$$

and $u(0,t) = 1$, $u(12,t) = 0$. By some simple computations,

TABLE V
COMPARISONS OF L^∞-ERRORS AND CPU AT $t = 0.2$ FOR EXAMPLE 2 WITH $\gamma = 0.05$, ($\Delta t = h$ WITH EXCEPTION OF $\Delta t = 1.0e-03$ FOR 3-TVD-RKM).

Methods	$h = 1/10$ (CPU)	$h = 1/40$ (CPU)	$h = 1/160$ (CPU)
Algorithm 2	$1.6837e-03$ $(3.750e-02)$	$6.0494e-06$ (0.589)	$2.5095e-08$ (0.813)
Algorithm 3	$1.7097e-03$ $(7.750e-05)$	$6.0084e-06$ $(2.350e-04)$	$2.3522e-08$ $(5.742e-03)$
FEM [17]	$3.5903e-02)$ $(1.870e-05)$	$2.3661e-03$ $(4.530e-05)$	$1.4858e-04$ $(6.516e-04)$
FDM [10]	$1.7956e-02$ $(2.030e-05)$	$1.1415e-03$ $(4.680e-05)$	$7.1605e-05$ $(6.531e-04)$
HOC FDM [14]	$5.8288e-03$ $(1.500e-04)$	$5.2591e-04$ $(3.100e-04)$	$3.3486e-05$ $(6.400e-03)$
3-TVD-RKM	$1.920e-03$ (0.452)	$7.0122e-06$ (0.608)	NaN

TABLE VI
COMPARISONS OF L^2-ERRORS AND CPU AT DIFFERENT TIME LEVELS FOR EXAMPLE 2 WITH $\gamma = 0.05$, ($\Delta t = h = 0.01$ WITH EXCEPTION OF $\Delta t = 5.0e-04, h = 0.01$ FOR 3-TVD-RKM).

Methods	$t = 0.2$ (CPU)	$t = 0.6$ (CPU)	$t = 1.0$ (CPU)
Algorithm 2	$8.8773e-08$ (0.515)	$2.2125e-07$ (1.217)	$9.8134e-08$ (1.716)
Algorithm 3	$8.5563e-08$ (0.171)	$2.2233e-07$ (0.515)	$9.7987e-08$ (0.717)
FEM [17]	$7.5569e-04$ (0.005)	$3.5996e-04$ (0.006)	$1.2426e-04$ (0.008)
FDM [10]	$5.7455e-04$ (0.005)	$4.8370e-05$ (0.006)	$3.1252e-05$ (0.008)
HOC FDM [14]	$4.1443e-05$ (0.014)	$1.3622e-05$ (0.015)	$7.2409e-06$ (0.016)
3-TVD-RKM	$1.807e-07$ (0.562)	$9.106e-007$ (0.624)	$3.5517e-07$ (0.780)

TABLE VII
COMPARISON BETWEEN EXACT AND NUMERICAL SOLUTIONS OF EXAMPLE 3 WITH $\gamma = 0.005$ AT $t = 1$, ($\Delta t = h = 0.002$).

x	Algorithm 1	Algorithm 2	Algorithm 3	HOC FDM [14]	exact solution
0.2	1.00000884705024	1.00000884705023	1.00000884705024	0.99345055174620	1.00000000000000
0.4	1.00000884704643	1.00000884704627	1.00000884704628	0.99345055174583	0.99999999999591
0.6	0.99997401773015	0.99997254089226	0.99997253343592	0.99340571924913	0.99996368170504
0.8	0.20206202643915	0.20197749501259	0.20197794728201	0.20178887544250	0.20197809852531
err	$8.3402e-03$	$2.2455e-05$	$1.1218e-05$	$3.1874e-02$	
$err2$	$1.5245e-03$	$8.8047e-06$	$7.4719e-06$	$8.1970e-03$	

TABLE VIII
COMPARISON OF NUMERICAL RESULTS AT DIFFERENT TIME LEVELS FOR EXAMPLE 3 WITH $\gamma = 0.003$, ($h = 0.0025$).

		$t = 0.2$	$t = 0.4$	$t = 0.6$	$t = 0.8$	$t = 1.0$
Algorithm 1	err	$1.1734e-02$	$2.3730e-02$	$3.5815e-02$	$4.7847e-02$	$5.9792e-02$
$\Delta t = h$	CPU	0.094	0.109	0.110	0.121	0.125
Algorithm 2	err	$4.1822e-04$	$6.3672e-04$	$7.9259e-04$	$1.0129e-03$	$1.3136e-03$
$\Delta t = h$	CPU	0.293	0.325	0.356	0.387	0.434
Algorithm 3	err	$4.0566e-04$	$5.4574e-04$	$5.5112e-04$	$5.3979e-04$	$5.2724e-04$
$\Delta t = h$	CPU	0.091	0.105	0.112	0.122	0.123
HOC FDM [14]	err	$3.4087e-02$	$3.4087e-02$	$8.4274e-02$	$1.4105e-01$	$1.9799e-01$
$\Delta t = h$	CPU	0.093	0.094	0.109	0.110	0.121
3-TVD-RKM	err	$4.0663e-04$	$5.4763e-04$	$5.5396e-04$	$5.4358e-04$	$5.3197e-04$
$\Delta t = 5.0e-04$	CPU	3.781	7.389	10.767	14.181	17.740

we have that

$$v(x,0) = \begin{cases} \exp(\frac{-x}{2\gamma}), & 0 \le x < 5, \\ \exp(\frac{x^2}{4\gamma} - \frac{3x}{\gamma} + \frac{25}{4\gamma}), & 5 \le x < 6, \\ \exp(\frac{-11}{4\gamma}), & 6 \le x < 12. \end{cases}$$

for this case.

Algorithm 3 is applied to solve this problem. Profiles of numerical solution at $t = 1, 2, 3, 4$ are displayed in Figure 4, which is similar to the patterns of Figure 1 and Figure 2 in [21]. This exactly confirms that numerical solution obtained using Algorithm 3 can exhibit correct physical behavior.

VI. CONCLUSIONS

In this article, three numerical solvers (i.e. Algorithm 1, Algorithm 2, Algorithm 3) for solving Burgers' equation in one dimension based on Hopf-Cole transformation have been developed. Numerical results show the superiority of Algorithm 3 over Algorithm 1 and Algorithm 2 in term of computational efficiency, though Algorithm 2 has the same accuracy as Algorithm 3. Also, numerical results reveal that Algorithm 3 outperforms numerical solvers devised in [10], [14], [17]. However, as exact solution is not smooth enough or discontinuous, the presented methods can not attain the claimed accuracy, even invalid. As we know, the initial or/and boundary condition has jumps, the exact solution to Burgers' equation is discontinuous, i.e. shock wave. The proposed methods, which do not satisfy total variation diminishing property [31], thus yielding seriously spurious oscillations near discontinuities, are not suitable for numerical simulations of discontinuous problems. This is a regretful deficiency. In the future work, it is possible to pay much more attention to the study of exponential integrators [32], [33] with TVD property.

In addition, over the past several decades, stabilized explicit Runge-Kutta methods (SERKMs) [34], [35] have attracted much more attention. They possess large stability domains and have been proven to be very useful in the solutions of IBVPs. Also, it is very interesting and challenging to solve multi-dimensional IBVPs by the combinations

TABLE IX
COMPARISON OF NUMERICAL RESULTS AT $t = 1$ FOR EXAMPLE 3 WITH DIFFERENT γ, ($\Delta t = h = 0.002$).

		$\gamma = 0.01$	$\gamma = 0.008$	$\gamma = 0.006$	$\gamma = 0.004$	$\gamma = 0.002$
Algorithm 1	err	$3.2528e-03$	$2.6712e-03$	$4.8987e-03$	$1.6257e-02$	$1.2838e-01$
	$err2$	$8.4138e-04$	$6.1754e-04$	$9.8071e-04$	$2.6583e-03$	$1.4968e-02$
Algorithm 2	err	$2.2112e-03$	$6.3690e-04$	$7.5088e-05$	$1.0588e-04$	$5.3862e-03$
	$err2$	$5.7237e-04$	$1.4739e-04$	$1.5553e-05$	$2.4448e-05$	$6.6394e-04$
Algorithm 3	err	$2.2112e-03$	$6.3699e-04$	$7.4626e-05$	$4.6404e-05$	$1.7570e-03$
	$err2$	$5.7238e-04$	$1.4743e-04$	$1.5562e-05$	$1.9288e-05$	$3.4468e-04$
HOC FDM [14]	err	$2.6558e-03$	$8.0053e-03$	$1.9081e-02$	$5.9661e-02$	$3.7600e-01$
	$err2$	$1.5423e-03$	$2.9058e-03$	$5.5360e-03$	$1.3245e-02$	$5.6780e-02$

TABLE X
COMPARISON BETWEEN EXACT AND NUMERICAL SOLUTIONS OF EXAMPLE 4 WITH $\gamma = 0.25$ AT $t = 1.5$, ($\Delta t = h = 0.025$).

x	Algorithm 1	Algorithm 2	Algorithm 3	HOC FDM [14]	exact solution
0	1.56637702742831	1.56633382966077	1.56633382656578	1.56637363394596	1.56633384476725
1.25	0.73665723258733	0.73613585176191	0.73613590308168	0.73665722340424	0.73613618471109
2.5	0.02462008147105	0.02458829874397	0.02458830650549	0.02462008147077	0.02458832890444
err	$5.2435e-04$	$4.6538e-07$	$4.6537e-07$	$2.1576e-03$	
$err2$	$1.2359e-04$	$1.6947e-07$	$1.6482e-07$	$6.4710e-04$	
CPU	0.378	1.128	0.391	0.375	

TABLE XI
COMPARISON BETWEEN EXACT AND NUMERICAL SOLUTIONS OF EXAMPLE 4 WITH $\gamma = 0.05$ AT $t = 1.5$, ($\Delta t = h = 0.025$).

x	Algorithm 1	Algorithm 2	Algorithm 3	HOC FDM [14]	exact solution
0	1.59999999377673	1.59999999266475	1.59999999264451	1.59999999322371	1.59999999266051
1.25	0.55398528835588	0.49472699352290	0.49527918783689	0.55398528835588	0.49604083019580
1.5	0.01536819730726	0.01297644607361	0.01302853320031	0.01536819730727	0.01306011384506
err	$6.3569e-02$	$1.3138e-03$	$9.2627e-04$	$6.5569e-02$	
$err2$	$6.9271e-03$	$1.5405e-04$	$1.3003e-04$	$1.6828e-02$	
CPU	0.390	1.185	0.391	0.375	

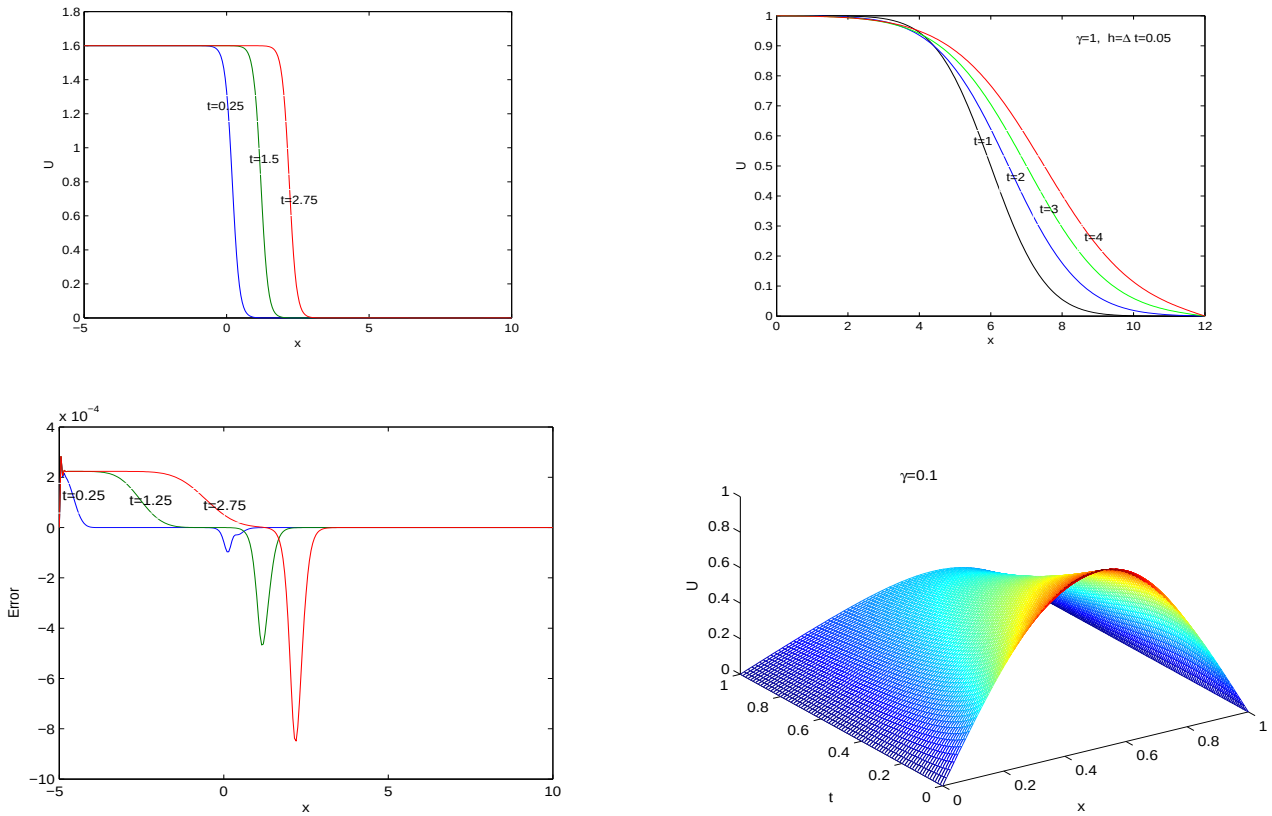

Fig. 3. Example4 with $\gamma = 0.1$ (solved by Algorithm 3 with $\Delta t = h = 0.05$): Numerical solution and corresponding errors.

Fig. 4. Example 5 (solved by Algorithm 3): Numerical solutions at different time levels.

of SERKMs with HOC FDMs. It is possible that some techniques developed in this paper are useful for the research of SERKMs combined with HOC FDMs in future.

Finally, the extensions of the proposed methods to some high-dimensional Burgers equations [36], [37] are impossible if they are not transformed into heat equation with mixed boundary by Hopf-Cole transformation. Of course, similar techniques may be generalized to the following system of two-dimensional Burgers equations:

$$\begin{cases} u_t + uu_x + vu_y = \Delta u \\ v_t + uv_x + vv_y = \Delta v \end{cases}$$

with potential symmetry condition $u_y = v_x$ because it can be transformed into heat equation by Hopf-cole transformation. However, for higher complicated matrix, it is also challenging to investigate the good approximation of matrix exponentials.

REFERENCES

[1] H. Bateman, "Some recent researchs on the motion of fluids," Monthly Weather Review, vol. 43, pp. 163–170, 1915.

[2] J.M. Burgers, "Mathematical examples illustrating relations occurring in the theory of turbulent fluid motion," Trans. Roy. Neth. Acad. Sci. Amsterdam, vol. 17 pp. 1–53, 1939.

[3] J.M. Burgers, "A mathematical model illustrating the theory of turbulence," Advances in Applied Mechanics, vol. 1 pp. 171–199, 1948.

[4] E. Benton, and G.W. Platzman, "A table of solutions of the one-dimensional Burgersequations," Quarterly of Applied Mathematics, vol. 30 pp. 195–212, 1972.

[5] J.D. Cole, "On a quasi-linear parabolic equations occuring in aerodynamics," Quarterly of Applied Mathematics, vol. 9 pp. 225–236, 1951.

[6] E. Hopf, "The partial differential equqation $u_t + uu_x = \mu u_{xx}$," Communications on Pure and Applied Mathematics, vol. 3 pp. 201–230, 1950.

[7] N. Taghizadeh, M. Akbari, and A. Ghelichzadeh, "Exact solution of Burgers squations by Homotopy perturbation method and reduced differential transformation method," Australian Journal of Basic and Applied Sciences, vol. 5 pp. 580–589, 2011.

[8] Y. Jin, M. Jia, and S. Lou, "B*ä*klund transformations and interaction solutions of the Burgers equation," Chinese Physics letter, vol. 30 020203, 2013.

[9] I.A. Hassanien, A.A. Salama, and H.A. Hosham, "Fourth-order finite difference method for solving Burgers' equation," Applied Mathematics and Computation, vol. 170 pp. 781–800, 2005.

[10] Mohan. K. Kadalbajoo, and A. Awasthi, "A numerical method based on Crank-Nicolson scheme for Burgers equation," Applied Mathematics and Computation, vol. 182 pp. 1430–1442, 2006.

[11] W. Liao, and J. Zhu, "An implicit fourth-order compact finite difference scheme for one-dimensional Burgers' equation," Applied Mathematics and Computation, vol. 206 pp. 755–764, 2008.

[12] D.K. Salkuyeh, and F.S. Sharafeh, "On the numerical solution of the Burgers's equation," International Journal of Computer Mathematics, vol. 86 pp. 1334–1344, 2009.

[13] M. Sari, and G. G*ü*rarslan, "A sixth-order compact finite difference scheme to the numerical solutions of Burgers equation," Applied Mathematics and Computation, vol. 208 pp. 475–483, 2009.

[14] S. Xie, G. Li, and S. Heo, "A compact finite difference method for solving Burgers' equation," International Journal for Numerical Methods in Fluids, vol. 62 pp. 747–764, 2010.

[15] M. Yousuf, "On the class of high order time stepping schemes baded on Pad*é* approxiamtions for the numerical solution of Burgers' equation," Applied Mathematics and Computation, vol. 205 pp. 442–453, 2008.

[16] S. Kutluay, A. Esen, and I. Dagb, "Numerical solutions of the Burgers equation by the least-squares quadratic B-spline finite element method," Journal of Computational and Applied Mathematics, vol. 167 pp. 21–33, 2004.

[17] T. Özis, E.N. Aksan, and A. Özdes, "A finite element approach for solution of Burgers' equation," Applied Mathematics and Computation, vol. 139 pp. 417–428, 2003.

[18] L. Shao, X. Feng, and Y. He, "The local discontinuous Galerkin finite element method for Burgers equation," Mathematical and Computer Modelling, vol. 54 pp. 2943–2954, 2011.

[19] R. Zhang, X. Yu, and G. Zhao, "Local discontinuous Galerkin method for solving Burgers and coupled Burgers equations," Chinese Physics Letter, vol. 20 110205, 2011.

[20] C. Basdevant, M. Devilie, and P. Haldenwang, et. al., "Spectral and finite difference solutions of the Burger equation," Computer & Fluids, vol. 14 pp. 23–41, 1986.

[21] R.C. Mittal, and P. Singhal, "Numerical solution of Burger's equation," Communications in Numerical Methods in Engineering, vol. 9 pp. 397–406, 1993.

[22] S. Chen, X. Wu, Y. Wang, and W. Kong, "The Laplace transform method for Burgers' equation," International Journal for Numerical Methods in Fluids, vol. 63, 1060–1076, 2010.

[23] A. Hashemian, and H.M. Shodja, "A meshless approach for solution of Burgers equation," Journal of Computational and Applied Mathematics, vol. 220 pp. 226–239, 2008.

[24] A. Korkmaz, and *İ*, "Da*ğ*, Polynomial based differential quadrature method for numerical solution of nonlinear Burgers equation," Journal of the Franklin Institute, vol. 348 pp. 2863–2875, 2011.

[25] R. Mokhtari, A.S. Toodar, and N.G. Chegini, "Application of the generalized differential quadrature method in solving Burgers'equations, Communications in Theoretical Physics vol. 56 pp. 1009–1015, 2011.

[26] R.K. Lele, "Compact finite difference schemes with spectral-like resolution," Journal of Computational Physics, vol. 103 pp. 16–42, 1992.

[27] D. Deng, and T. Pan, "A Fourth-order Singly Diagonally Implicit Runge-Kutta Method for Solving One-dimensional Burgers' Equation," IAENG International Journal of Applied Mathematics, vol. 45, no. 4, pp. 327–333, 2015.

[28] B. Wongsaijai, K. Poochinapan, and T. Disyadej, "A compact finite difference methodfor solving the general Rosenau-RLW Equation," IAENG International Journal of Applied Mathematics, vol. 44, no. 4, pp. 192-199, 2014.

[29] J.C. Chen and W. Chen, "Two-dimensional nonlinear wave dynamics in blasius boundary layer flow using combined compact difference methods," IAENG International Journal of Applied Mathematics, vol. 41, no. 2, pp. 162-171, 2011.

[30] H. Cao, L. Liu, Y. Zhang, and S. Fu, "A fourth-order method of the convection-diffusion equations with Neumann boundary conditions," Applied Mathematics and Computation, vol. 217 pp. 9133–9141, 2011.

[31] Y. Hadjimichael, C. B. Macdonald, D. I. Ketcheson, and J. H. Verner, "Strong stability preserving explicit Runge-Kutta methods of maximal effective order," SIAM Journal on Numerical Analysis, vol. 51 pp. 2149–2165, 2013.

[32] M. Hochbruck, Ch. Lubich, and H. Selhofer. "Exponential integrators for large systems of differential equations," SIAM Journal on Scientific Computing, vol. 19 pp. 1552–1574, 1998.

[33] A.Q.M. Khaliq, J. Martn-Vaquero, B.A.Wade, and M. Yousuf. "Smoothing schemes for reaction-diffusion systems with nonsmooth data," Journal of Computational and Applied Mathematics, vol. 223 pp. 374–386, 2009.

[34] B. Kleefeld, and J. Martín-Vaquero, "Serk2v2: A new second-order stabilized explicit Runge-Kutta method for stiff problems," Numerical Methods for Partial Differential Equations, vol. 29 pp. 170–185, 2013.

[35] J. Martín-Vaquero, A.Q.M. Khaliq, and B. Kleefeld, "Stabilized explicit Runge-Kutta methods for multi-asset American options," Computers & Mathematics with Applications, vol. 67 pp. 1293–1308, 2014.

[36] C.h. Dai, and F.B. Yu, "Special solitonic localized structures for the (3+1)- dimensional burgers equation in water waves," Wave Motion, vol. 51 pp. 52–59, 2014.

[37] A. Wazwaz, Multiple kink solutions for M-component Burgers equations in (1+1)-dimensions and (2+1)-dimensions, Applied Mathematics and Computation, 217 pp. 3564–3570, 2010.