Direct discontinuous Galerkin method for the generalized Burgers–Fisher equation*

Zhang Rong-Pei(张荣培)[a)†] and Zhang Li-Wei(张立伟)[b)c)]

[a)] School of Sciences, Liaoning Shihua University, Fushun 113001, China

[b)] Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China

[c)] The Chinese University of Hong Kong, Hong Kong, China

(Received 5 January 2012; revised manuscript received 14 February 2012)

In this study, we use the direct discontinuous Galerkin method to solve the generalized Burgers–Fisher equation. The method is based on the direct weak formulation of the Burgers–Fisher equation. The two adjacent cells are jointed by a numerical flux that includes the convection numerical flux and the diffusion numerical flux. We solve the ordinary differential equations arising in the direct Galerkin method by using the strong stability preserving Runge–Kutta method. Numerical results are compared with the exact solution and the other results to show the accuracy and reliability of the method.

Keywords: direct discontinuous Galerkin method, Burgers–Fisher equation, strong stability preserving Runge–Kutta method

PACS: 02.70.Dh, 52.35.–g **DOI:** 10.1088/1674-1056/21/9/090206

1. Introduction

The nonlinear partial differential equation plays an important role in physical science and engineering. Recently, the nonlinear equations have attracted much attention of researchers. There are various powerful mathematical methods, including the first integral method,[1] the variational iteration method,[2,3] the homotopic mapping method,[4,5] the tanh method,[6] and the other methods,[7,8] been proposed to obtain exact or approximate analytic solutions for the nonlinear equations.

In this paper, we consider the generalized Burgers–Fisher equation

$$\frac{\partial u}{\partial t} + \alpha u^{\delta}\frac{\partial u}{\partial x} - \mu\frac{\partial^2 u}{\partial x^2} = \beta u(1-u^{\delta}), \quad a \le x \le b, t \ge 0. \tag{1}$$

where μ is the diffusion coefficient, and α, β, and δ are the parameters. The exact solution is

$$u(x,t) = \left(\frac{1}{2} + \frac{1}{2}\tanh(\theta_1(x-\theta_2 t))\right)^{1/\delta}, \quad a \le x \le b, t \ge 0, \tag{2}$$

where

$$\theta_1 = \frac{-\alpha\delta}{2\mu(1+\delta)},$$

$$\theta_2 = \frac{\alpha}{1+\delta} + \frac{\mu\beta(1+\delta)}{\alpha}.$$

The initial and the boundary conditions are determined by Eq. (2). The generalized Burgers–Fisher equation has many applications in fields including gas dynamics, number theory, heat conduction, elasticity, etc.[9]

In order to solve Eq. (1) numerically, many researchers have proposed various numerical methods. Zhu *et al.*[9] investigated the Burgers–Fisher equation by employing the B-spline quasi-interpolation method. Ismail *et al.*[10] studied the Adomian decomposition methods. Golbabai and Javidi proposed a spectral domain decomposition method.[11] Khattak[12] presented a computational meshless method for the Burgers–Huxley and the Burgers–Fisher equations. Zhang and Yan[13] developed a lattice Boltzmann model for the one- and the two-dimensional Burgers–Fisher equations. Sari *et al.*[14] proposed a compact finite difference method for the generalized Burgers–Fisher equation.

The discontinuous Galerkin (DG) method has emerged as an attractive tool for simulating the convection-diffusion problem.[15] The main advantage

*Project supported by the National Natural Science Foundation of China (Grant Nos. 61105130 and 61175124).

†Corresponding author. E-mail: rongpeizhang@163.com

© 2012 Chinese Physical Society and IOP Publishing Ltd

of the DG method lies in their accuracy and flexibility thanks to their high degree of locality. The accuracy of the numerical scheme depends on the degree of the piecewise polynomials used, which can be easily increased. Liu and Yan extended the traditional DG method and proposed the direct discontinuous Galerkin (DDG) method.[16,17] Unlike the local discontinuous Galerkin method (LDG),[18,19] the DDG method is directly based on the direct weak formulation of Eq. (1) and the construction of the suitable numerical flux on the cell edges without introducing any auxiliary variables.

In this paper, equation (1) is discretized spatially by using the DDG technique to obtain an ordinary differential equation (ODE) system. We then use the third-order strong stability-preserving (SSP) Runge–Kutta method[20] to solve the ODEs. The paper is organized as follows. In Section 2, we construct the DDG scheme for Eq. (1). In Section 3, we show some numerical results to demonstrate the high accuracy of the DDG scheme. Section 4 gives the conclusion.

2. The DDG method

In this section, we illustrate the formulation of the DDG method for the generalized Burgers–Fisher equation (1) in detail. The computational domain $[a,b]$ is divided into N cells as follows:

$$a = x_{1/2} < x_{3/2} < \cdots < x_{N+(1/2)} = b. \tag{3}$$

We denote the j-th cell by $I_j = \left[x_{j-(1/2)}, x_{j+(1/2)}\right]$ and the center of the cell by $x_j = \left(x_{j-(1/2)} + x_{j+(1/2)}\right)/2$. The size of each cell is denoted by $\Delta x = (b-a)/N$. The approximation solution and the test function space are defined as

$$V_h^k = \{v_h : v|_{I_j} \in P^k(I_j), \quad j = 1, \ldots, N\}, \tag{4}$$

where $P^k(I_j)$ denotes the set composed of all polynomials of degree at most k on I_j. The basis functions in $P^k(I_j)$ are defined by scaled Legendre polynomials, $v_0(x) = 1$, $v_1(x) = (x - x_j)/\Delta x/2 = \xi_j$, $v_2(x) = (3\xi_j^2 - 1)/2$, $v_3(x) = (5\xi_j^3 - 3\xi_j)/2, \ldots$

Then the numerical solution for $x \in I_j$ in space V_h^k can be written as

$$u(x,t) = \sum_{l=0}^{k} u_j^l(t) v_l^j(x) = \boldsymbol{V}^{\mathrm{T}}(x)\boldsymbol{U}_j(t), \tag{5}$$

where $\boldsymbol{V}(x) = (v_0^j(x), v_1^j(x), \ldots)^{\mathrm{T}}$ and $\boldsymbol{U}_j(t) = (u_j^0(t), u_j^1(t), \ldots)^{\mathrm{T}}$. We use $u_{j+(1/2)}^+$ and $u_{j+(1/2)}^-$ to denote the values of u at $x_{j+(1/2)}$ from the right cell I_{j+1} and from the left cell I_j, respectively,

$$u_{j+(1/2)}^+ = \lim_{\epsilon \to 0+} u(x_{j+(1/2)} + \epsilon),$$

$$u_{j+(1/2)}^- = \lim_{\epsilon \to 0+} u(x_{j+(1/2)} - \epsilon).$$

We use the usual notations $\{u\} = (u^+ + u^-)/2$ and $[u] = u^+ - u^-$ to denote the mean and the jump of function u at each element boundary point, respectively.

The semi-discrete DDG method is applied to Eq. (1). We then find $u_h \in V_h^k$ such that for all test functions $v \in V_h^k$, we have

$$\begin{aligned} &\int_{I_j} \frac{\partial u_h}{\partial t} v \,\mathrm{d}x - \int_{I_j} \left(f(u_h) - \mu \frac{\partial u_h}{\partial x} \right) \frac{\partial v}{\partial x} \mathrm{d}x \\ &+ (\hat{f} - \hat{h})_{j+(1/2)} v_{j+(1/2)}^- - (\hat{f} - \hat{h})_{j-(1/2)} v_{j-(1/2)}^+ \\ &= \int_{I_j} \beta u_h (1 - u_h^\delta) v \,\mathrm{d}x, \end{aligned} \tag{6}$$

for all cells $1 \le j \le N$, where $f(u_h) = [\alpha/(\delta+1)]u_h^{\delta+1}$. The hat terms in Eq. (6) are numerical fluxes, which are single-valued functions defined at the cell interfaces and in general depend on the values of numerical solution u_h from both sides of the interface. There are two types of numerical fluxes, one is the diffusion numerical flux $\hat{h}$, the other is the convection numerical flux $\hat{f}$. The convection flux $\hat{f}$ is chosen to be the local Lax–Friedrichs flux[21]

$$\begin{aligned} \hat{f} &= \frac{1}{2}(f(u^-) + f(u^+) - \alpha(u^+ - u^-)), \\ \alpha &= \max_u |f'(u)|, \end{aligned} \tag{7}$$

where the maximum is taken over a relevant range of u. The diffusion flux $\hat{h}$ is defined as[16]

$$\hat{h}_{j+\frac{1}{2}} = \frac{\beta_0 \mu}{\Delta x}[u_h] + \mu \left\{ \frac{\partial u_h}{\partial x} \right\} + \beta_1 \mu \Delta x \left[\frac{\partial^2 u_h}{\partial x^2} \right]. \tag{8}$$

The integrations in Eq. (6) can be approximated by using suitable quadrature rules, such as the five-point Gaussian quadrature.

Now we obtain the ordinary differential equation (ODE) system from the direct discontinuous Galerkin space discretization

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{U}_h = L(\boldsymbol{U}_h, t). \tag{9}$$

In the current work, the ODE system (9) is integrated in time by using the following SSP-RK3 scheme[20]

$$\boldsymbol{U}_h^{(1)} = \boldsymbol{U}_h^n + \Delta t L(\boldsymbol{U}_h^n, t^n), \tag{10}$$

$$\boldsymbol{U}_h^{(2)} = \frac{3}{4}\boldsymbol{U}_h^n + \frac{1}{4}\boldsymbol{U}_h^{(1)} + \frac{1}{4}\Delta t L\left(\boldsymbol{U}_h^{(1)}, t^n + \Delta t\right), \quad (11)$$

$$\boldsymbol{U}_h^{(3)} = \frac{1}{3}\boldsymbol{U}_h^n + \frac{2}{3}\boldsymbol{U}_h^{(2)} + \frac{2}{3}\Delta t L\left(\boldsymbol{U}_h^{(2)}, t^n + \frac{1}{2}\Delta t\right). \quad (12)$$

3. Results and discussion

In this section, we obtain DDG solutions of Eq. (1). To show the accuracy of the DDG method for the test problems, we define the maximum error

$$L_\infty = \max_{1\le j\le N}\{|u(x_j,t) - u_h(x_j,t)|\},$$

where $u(x_j,t)$ is the exact solution, and $u_h(x_j,t)$ is the solution obtained by the DDG method at the center of cell I_j. Here, $u_h(x_j,t) = u_j^0(t) - u_j^2(t)/2$ for P^2 (which means $k=2$ in Eq. (4)) and P^3 elements, and $u_h(x_j,t) = u_j^0(t) - u_j^2(t)/2 + 3u_j^4(t)/8$ for P^4 element. In Examples 1–3, we choose the computation domain $[0,1]$, which is divided into 5 cells, and the diffusion coefficient is $\mu = 1$.

Example 1 Equation (1) is reduced to the Burgers–Fisher equation when $\delta = 1$. In Table 1, we show the maximum errors of the DDG method and the spectral domain decomposition (SDD) method in Ref. [11] at $t = 0.5$ and $t = 1.0$ for various values of α and β. We choose the P^4 element and set $\beta_0 = 12$, $\beta_1 = 0$ in Eq. (8). The comparison between the results of the current study and that of Ref. [11] shows that the obtained results are very reliable and accurate.

Example 2 In Table 2, we present the numerical results for Eq. (1) obtained in the DDG scheme at $t = 0.3$ and $t = 0.9$ with $\alpha = 0.1$, $\beta = -0.25$ and different δ. In this table, we show the maximum errors for the P^4 element compared with those for the SDD[11] and the B-spline quasi-interpolation (BSQI)[9] methods. The comparison of the current results with the previous ones shows that the present results are very accurate.

Table 1. Maximum errors in Example 1 with $\delta = 1$ and $\Delta t = 10^{-4}$.

t	β	$\alpha = 10^{-2}$		$\alpha = 10^{-3}$		$\alpha = 10^{-4}$	
		SDD/10^{-12}	DDG/10^{-14}	SDD/10^{-12}	DDG/10^{-14}	SDD/10^{-12}	DDG/10^{-14}
0.5	1	4.6763	34.313	4.5374	34.438	4.7526	34.468
	10	6.2529	33.536	6.0540	33.480	6.1933	33.471
	100	8.0269	1.9085	0.81424	1.8439	8.1912	1.9085
1.0	1	10.996	53.323	10.740	53.394	11.011	53.353
	10	10.952	3.6016	10.683	3.5975	10.762	3.5616
	100	1.9129	1.9085	1.9245	1.8439	1.9293	1.9085

Table 2. Maximum errors in Example 2 with $\alpha = 0.1$, $\beta = -0.25$, and $\Delta t = 10^{-5}$.

t	δ	SDD/10^{-12}	BSQI/10^{-8}	DDG/10^{-14}
0.3	1	2.1143	0.036959	2.4155
	4	3.8276	1.5630	20.487
	8	3.9059	4.8731	45.458
0.9	1	7.3929	0.091896	2.5543
	4	12.857	1.0155	9.3530
	8	12.907	1.2017	8.6455

Example 3 This example demonstrates the DDG approximate solution of the Burgers–Fisher equation (3) with $\alpha = \beta = 0.01$ and $\delta = 1$. We compute the problem by using the P^4 element at $t = 1,\ 10,\ 50$ respectively. In Table 3, we present the absolute errors in the DDG method compared with the variational iteration (VIM) method, the Adomian decomposition (ADM) method, and the lattice Boltzmann (LBM) method results from Ref. [13]. The table shows that our method can provide smaller errors at long time.

Example 4 We compute the Burgers–Fisher equation (3) in $x \in [-0.5, 0.5]$ by using the P^2 element. In the numerical simulation, the region of time evolution is set in $[0,1]$. The parameters in Eq. (1) used are $\alpha = \delta = 1$, $\beta = 0.01$, and $\mu = 0.01$. The numerical and the analytical solutions are plotted in Figs. 1(a) and 1(b), respectively. As shown in Fig. 1, the numerical solution from the DDG method is in good agreement with the exact one.

Table 3. Maximum errors in Example 3 with $\alpha = 0.01$, $\beta = 0.01$, $\delta = 1$, and $\Delta t = 2 \times 10^{-4}$.

t	x	VIM	ADM	LBM/10^{-4}	DDG/10^{-14}
1	0.1	1.78×10^{-8}	1.78×10^{-8}	1.08	1.14
	0.5	5.29×10^{-9}	5.29×10^{-9}	0.325	3.31
	0.9	7.28×10^{-9}	7.28×10^{-9}	1.73	1.08
10	0.1	2.07×10^{-5}	2.07×10^{-5}	1.08	1.28
	0.5	1.94×10^{-5}	1.94×10^{-5}	0.319	3.28
	0.9	1.81×10^{-5}	1.81×10^{-5}	1.72	1.32
50	0.1	0.002552	0.002552	1.01	1.64
	0.5	0.002552	0.002552	0.306	4.39
	0.9	0.002492	0.002492	1.62	1.44

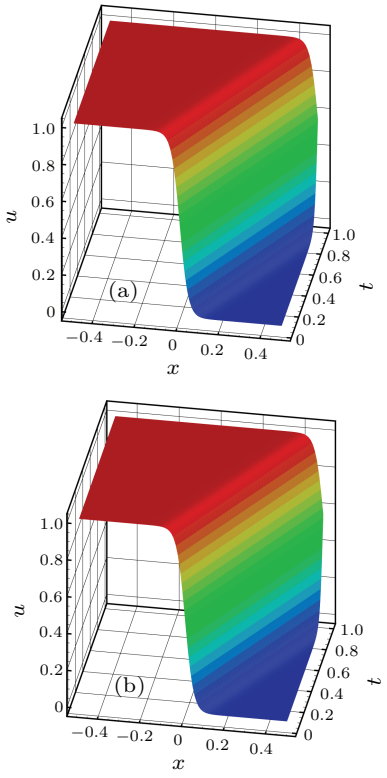


Fig. 1. (colour online) Space–time patterns for Example 4 with $\alpha = 1$, $\beta = 0.01$, $\delta = 1$, and $\mu = 0.01$. Panels (a) and (b) show the exact and the numerical solutions, respectively.

4. Conclusion

In this paper, the DDG method is proposed to solve the Burgers–Fisher equation and the generalized Burgers–Fisher equation. We compute Eq. (1) for various parameters by using the DDG method. The test examples show that the DDG scheme can give more accurate results than the other numerical schemes. Moreover, the DDG scheme is proved to be very accurate for some large values of time.

References

[1] Jiang L, Guo Y C and Xu S J 2007 *Chin. Phys.* **16** 2514
[2] Abdou M A and Soliman A A 2005 *J. Comput. Appl. Math.* **181** 245
[3] Zhao G Z, Yu X J, Xu Y and Zhu J 2010 *Chin. Phys.* B **19** 070203
[4] Shi L F and Zhou X C 2010 *Acta Phys. Sin.* **59** 2915 (in Chinese)
[5] Mo J Q and Chen X F 2010 *Chin. Phys.* B **19** 100203
[6] Wazwaz A M 2005 *Appl. Math. Comput.* **169** 321
[7] Deng X J, Yan Z Z and Han L B 2009 *Chin. Phys.* B **18** 3169
[8] Taogetusang 2011 *Acta Phys. Sin.* **60** 010202 (in Chinese)
[9] Zhu C G and Kang W S 2010 *Appl. Math. Comput.* **216** 2679
[10] Ismail H N A, Raslan K and Rabboh A A A 2004 *Appl. Math. Comput.* **159** 291
[11] Golbabai A and Javidi M 2009 *Chaos Soliton. Fract.* **39** 385
[12] Khattak A J 2009 *Appl. Math. Model* **33** 3718
[13] Zhang J and Yan G 2010 *Chaos* **20** 023129
[14] Sari M, Gürarslan and Dağ İ 2010 *Numer. Methods Part. Diff. Eq.* **26** 125
[15] Cockburn B and Shu C W 2001 *J. Sci. Comp.* **16** 173
[16] Liu H L and Yan J 2009 *SIAM J. Numer. Anal.* **47** 675
[17] Liu H L and Yan J 2010 *Commun. Comput. Phys.* **3** 541
[18] Cockburn B and Shu C W 1998 *SIAM J. Numer. Anal.* **35** 2440
[19] Zhang R P, Yu X J and Zhao G Z 2011 *Chin. Phys.* B **20** 110205
[20] Gottlieb S, Shu C W and Tadmor E. 2001 *SIAM Review* **43** 89
[21] Li B Q 2006 *Discontinuus Finite Elements in Fluid Dynamics and Heat Transfer* (Berlin: Springer)