Hindawi Publishing Corporation
Abstract and Applied Analysis
Volume 2012, Article ID 527467, 15 pages
doi:10.1155/2012/527467

Research Article

A Galerkin Solution for Burgers' Equation Using Cubic B-Spline Finite Elements

A. A. Soliman

Department of Mathematics, Faculty of Education, Suez Canal University, Al-Arish 45111, Egypt

Correspondence should be addressed to A. A. Soliman, asoliman_99@yahoo.com

Received 13 May 2012; Accepted 24 July 2012

Academic Editor: Benchawan Wiwatanapataphee

Numerical solutions for Burgers' equation based on the Galerkins' method using cubic B-splines as both weight and interpolation functions are set up. It is shown that this method is capable of solving Burgers' equation accurately for values of viscosity ranging from very small to large. Three standard problems are used to validate the proposed algorithm. A linear stability analysis shows that a numerical scheme based on a Cranck-Nicolson approximation in time is unconditionally stable.

1. Introduction

A study of Burgers' equation is important since it arises in the approximate theory of flow through a shock wave propagating in a viscous fluid [1] and in the modeling of turbulence [2]. Burgers' equation and the Navier-Stokes equation are similar in the form of their nonlinear terms and in the occurrence of higher order derivatives with small coefficients in both [3]. The applications have been found in field as diverse as number theory, gas dynamics, heat conduction, elasticity, and so forth [4]. The exact solutions of the one-dimensional Burgers' equation have been surveyed by Benton and Platzman [5]. However, difficulties arise in the numerical solution of Burgers' equation for small values of the viscosity coefficient, that is large Reynolds numbers, which correspond to steep wave fronts [6]. In these cases, numerical methods are likely to produce results which include large nonphysical oscillations unless the size of the elements in unrealistically small. Many studies have been done on the numerical solutions of Burgers' equation to deal with solutions for the large Reynolds number values. Some of the earlier numerical studies are documented as follows: a finite element method has been given by Caldwell et al. [7] to solve Burgers' equation by altering the size of the element at each stage using information from three previous steps. Galerkin and Petrov-Galerkin finite element methods involving a time dependent grid [8, 9], and product approximation methods [10], have been used successfully to obtain accurate numerical solutions even for small viscosity coefficients.

Rubin and Graves Jr. [11] have used the spline function technique and quasilinearization for the numerical solution of Burgers' equation in one space variable. A cubic spline collocation procedures were developed for Burgers' equation in the papers [12–15]. Soliman [16, 17] has solved Burgers' equation numerically using a new similarity technique, and analytically using the modified extended tanh-function method. Raslan [18] has solved Burgers' equation using collocation method based on quadratic B-spline finite element method. In this paper we set up the finite element method using Galerkins' method with cubic splines as shape functions to obtain a numerical solution for Burgers' equation. The resulting system will be a system of ordinary differential equations which can be solved using a Crank-Nicolson approximation in time.

2. The Governing Equation

Consider Burgers' equation in the form

$$u_t + \varepsilon u u_x - \nu u_{xx=0}, \quad a \leq x \leq b, \tag{2.1}$$

where ε and ν are positive parameters and the subscripts t and x denote temporal and spatial differentiation, respectively. Appropriate boundary conditions will be chosen from

$$\begin{gathered} u(a,t) = \alpha, \\ u(b,t) = \beta, \\ u_x(a,t) = u_x(b,t) = 0, \\ u_{xx}(a,t) = u_{xx}(b,t) = 0. \end{gathered} \tag{2.2}$$

Applying the Galerkin approach [19–21] to (2.1) with weight function v, integrating by parts, and choosing the boundary conditions from (2.2) lead to the weak form

$$\int_a^b v(u_t + \varepsilon u u_x)dx + \nu \int_a^b v_x u_x dx = \nu [v u_x]_a^b, \tag{2.3}$$

where the right-hand side of (2.3) is evaluated only at the boundaries. The conditions on the interpolation functions are now simply that only the functions and their first derivative need to be continuous throughout the region. However, we have chosen to use, as trial functions, the very adaptable cubic splines with their well-known advantages.

3. The B-Spline Finite Element Solution

The region $[a,b]$ is partitioned into N finite elements of equal length h by knots x_i such that $a = x_0 < x_1 \cdots < x_N = b$, and let $\phi_i(x)$ be the cubic B-splines with knots at the points x_i. The

Table 1

x	$x_{\ell-2}$	$x_{\ell-1}$	x_ℓ	$x_{\ell+1}$	$x_{\ell+2}$
ϕ_ℓ	0	1	4	1	0
ϕ'_ℓ	0	$3/h$	0	$-3/h$	0
ϕ''_ℓ	0	$6/h^2$	$-12/h^2$	$6/h^2$	0

set of splines $\{\phi_{-1},\phi_0,\dots,\phi_N,\phi_{N+1}\}$ form a basis for functions defined over $[a,b]$. We seek the approximation $u_N(x,t)$ to the solution $u(x,t)$, which uses these splines as trial functions

$$u_N(x,t)=\sum_{\ell=-1}^{N+1}\delta_\ell(t)\phi_\ell(x), \tag{3.1}$$

where the δ_ℓ are time-dependent quantities to be determined from (2.3).

Defining a local coordinate system for the finite element $[x_\ell,x_{\ell+1}]$ with nodes at x_ℓ and $x_{\ell+1}$, each cubic B-spline covers four elements [22]; consequently each element $[x_\ell,x_{\ell+1}]$ is covered by four splines $(\phi_{\ell-1},\phi_\ell,\phi_{\ell+1},\phi_{\ell+2})$ which are given in terms of a local coordinate system η, where $\eta=x-x_\ell$ and $0\le\eta\le h$, by

$$\left\{\begin{matrix}\phi_{\ell-1}\\ \phi_\ell\\ \phi_{\ell+1}\\ \phi_{\ell+2}\end{matrix}\right. = \frac{1}{h^3}\begin{cases}(h-\eta)^3,\\ h^3+3h^2(h-\eta)+3h(h-\eta)^2-3(h-\eta)^3,\\ h^3+3h^2\eta+3h\eta^2-3\eta^3,\\ \eta^3, \quad 0\le\eta\le h.\end{cases} \tag{3.2}$$

It is the representation of cubic B-spline over a single element which is most appropriate for the finite element approach where all other splines are zero over this element.

The spline $\phi_\ell(x)$ and its two principle derivatives vanish outside the interval $[x_{\ell-2},x_{\ell+2}]$. In Table 1 we list for convenience the values of $\phi_\ell(x)$ and its derivatives $\phi'_\ell(x)$, $\phi''_\ell(x)$ at the relevant knots.

We now identify the finite elements for the problem with the intervals $[x_\ell,x_{\ell+1}]$, and the element nodes with the knots $x_\ell, x_{\ell+1}$.

Using (3.1) and Table 1, we see that the nodal parameters u_ℓ are given in terms of the parameters δ_ℓ by

$$\begin{aligned}u_\ell&=u(x_\ell)=\delta_{\ell-1}+4\delta_\ell+\delta_{\ell+1},\\ u_{\ell+1}&=u(x_{\ell+1})=\delta_\ell+4\delta_{\ell+1}+\delta_{\ell+2},\end{aligned} \tag{3.3}$$

and the variation of a function u over the element $[x_\ell,x_{\ell+1}]$ is given by

$$u=\sum_{j=\ell-1}^{\ell+2}\delta_j\phi_j. \tag{3.4}$$

In addition, we have the valuable property that $\delta_{\ell-1} - \delta_{\ell+2}$ determine also the first and second derivatives at the nodes (element boundaries), and these are also continuous

$$u' = \frac{3}{h}(-\delta_{\ell-1} + \delta_{\ell+1}), \tag{3.5}$$

$$u'' = \frac{6}{h^2}(\delta_{\ell-1} - 2\delta_{\ell} + \delta_{\ell+1}). \tag{3.6}$$

The finite element equations we will set up will not be expressed in terms of the nodal parameters u_ℓ, u'_ℓ but in terms of element parameters δ_ℓ, so we shall not directly determine the nodal values as is the case with the usual finite element formulations, but these can always be recovered using (3.3) and (3.5).

We will now set up the element matrices relevant to (2.3). For a typical element $[x_\ell, x_{\ell+1}]$, we have the contribution

$$\int_{x_\ell}^{x_{\ell+1}} v(u_t + \varepsilon u u_x)dx + \nu \int_{x_\ell}^{x_{\ell+1}} v_x u_x dx = 0. \tag{3.7}$$

Using (3.4) in (3.7) and identifying the weight function $v(\eta)$ with the cubic splines we obtain

$$\sum_{j=\ell-1}^{\ell+2}\left[\int_0^h \phi_i\phi_j d\eta\right]\frac{d\delta_j^e}{dt} + \varepsilon \sum_{j=\ell-1}^{\ell+2}\sum_{k=\ell-1}^{\ell+2}\left[\int_0^h \phi_i\phi'_j\phi_k d\eta\right]\delta_k^e\delta_j^e + \nu \sum_{j=\ell-1}^{\ell+2}\left[\int_0^h \phi'_i\phi'_j d\eta\right]\delta_j^e = 0, \tag{3.8}$$

which can be written in matrix form as

$$A^e\frac{d\boldsymbol{\delta}^e}{dt} + \varepsilon\boldsymbol{\delta}^{e^T}L^e\boldsymbol{\delta}^e + \nu C^e\boldsymbol{\delta}^e = 0, \tag{3.9}$$

where $\boldsymbol{\delta}^e = (\delta_{\ell-1}^e, \delta_\ell^e, \delta_{\ell+1}^e, \delta_{\ell+2}^e)^T$.

The element matrices are given by the integrals

$$\begin{aligned} A_{ij}^e &= \int_0^h \phi_i\phi_j d\eta, \\ C_{ij}^e &= \int_0^h \phi'_i\phi'_j d\eta, \\ L_{ijk}^e &= \int_0^h \phi_i\phi'_j\phi_k d\eta, \end{aligned} \tag{3.10}$$

where i, j, k take only the values $\ell-1, \ell, \ell+1, \ell+2$, for this typical element $[x_\ell, x_{\ell+1}]$. The matrices A^e, C^e are therefore 4×4 and the matrix L^e is $4 \times 4 \times 4$. An associated 4×4 matrix B^e is given by

$$B_{ij}^e = \sum_{k=\ell-1}^{\ell+2} L_{ijk}^e \delta_k^e, \tag{3.11}$$

which also depends on the parameters δ_k^e and will be used in the following theoretical discussions.

The element matrices A^e, C^e, L^e, and B^e can be determined algebraically from (3.2). Then, we obtain

$$\begin{gathered}
A^e = \frac{h}{140}\begin{pmatrix} 20 & 129 & 60 & 1 \\ 129 & 1188 & 933 & 60 \\ 60 & 933 & 1188 & 129 \\ 1 & 60 & 129 & 20 \end{pmatrix}, \\
C^e = \frac{1}{10h}\begin{pmatrix} 18 & 21 & -36 & -3 \\ 21 & 102 & -87 & -36 \\ -36 & -87 & 102 & 21 \\ -3 & -36 & 21 & 18 \end{pmatrix}, \\
B_{11}^e = -\frac{1}{840}(280, 1605, 630, 5)\boldsymbol{\delta}^e, \\
B_{12}^e = -\frac{1}{840}(150, 1305, 792, 21)\boldsymbol{\delta}^e, \\
B_{13}^e = \frac{1}{840}(420, 2781, 1314, 21)\boldsymbol{\delta}^e, \\
B_{14}^e = \frac{1}{840}(10, 129, 108, 5)\boldsymbol{\delta}^e, \\
B_{21}^e = -\frac{1}{840}(1605, 10830, 5349, 108)\boldsymbol{\delta}^e, \\
B_{22}^e = -\frac{1}{840}(1305, 17640, 17541, 1314)\boldsymbol{\delta}^e, \\
B_{23}^e = \frac{1}{840}(2781, 25002, 17541, 792)\boldsymbol{\delta}^e, \\
B_{24}^e = \frac{1}{840}(129, 3468, 5349, 630)\boldsymbol{\delta}^e, \\
B_{31}^e = -\frac{1}{840}(630, 5349, 3468, 129)\boldsymbol{\delta}^e, \\
B_{32}^e = -\frac{1}{840}(792, 17541, 25002, 2781)\boldsymbol{\delta}^e,
\end{gathered} \tag{3.12}$$

$$\begin{aligned}
B_{33}^{e} &= \frac{1}{840}(1314, 17541, 17640, 1305)\boldsymbol{\delta}^{e},\\
B_{34}^{e} &= \frac{1}{840}(108, 5349, 10830, 1605)\boldsymbol{\delta}^{e},\\
B_{41}^{e} &= -\frac{1}{840}(5, 108, 129, 10)\boldsymbol{\delta}^{e},\\
B_{42}^{e} &= -\frac{1}{840}(21, 1314, 2781, 420)\boldsymbol{\delta}^{e},\\
B_{43}^{e} &= \frac{1}{840}(21, 792, 1305, 150)\boldsymbol{\delta}^{e},\\
B_{44}^{e} &= \frac{1}{840}(5, 630, 1605, 280)\boldsymbol{\delta}^{e}.
\end{aligned} \tag{3.13}$$

Combining contributions from all elements leads to the matrix equation

$$A\frac{d\boldsymbol{\delta}}{dt} + B\boldsymbol{\delta} + \nu C\boldsymbol{\delta} = 0, \tag{3.14}$$

where

$$\boldsymbol{\delta} = (\delta_{-1}, \delta_{0}, \delta_{1}, \ldots, \delta_{N+1})^{T}, \tag{3.15}$$

and A, B, C assembled from the element matrices A^{e}, B^{e}, C^{e} in the usual way, are septadiagonal in form. We have assumed homogeneous boundary conditions on u or u' and so the term on the right-hand side of (2.3) is zero. The general row for each matrix has the following form:

$$\begin{aligned}
&A\colon \frac{h}{140}(1, 120, 1191, 2416, 1191, 120, 1),\\
&C\colon \frac{-1}{10h}(3, 72, 45, -240, 45, 72, 3),\\
&B\colon \frac{1}{840}(-(5, 108, 129, 10, 0, 0, 0)\boldsymbol{\delta}, -(21, 1944, 8130, 3888, 129, 0, 0)\boldsymbol{\delta},\\
&\qquad -(-21, 0, 17841, 35682, 8130, 108, 0)\boldsymbol{\delta}, (5, 1944, 17841, 0, -17841, -1944, -5)\boldsymbol{\delta},\\
&\qquad (0, 108, 8130, 35682, 17841, 0, -21)\boldsymbol{\delta}, (0, 0, 129, 3888, 8130, 1944, 21)\boldsymbol{\delta},\\
&\qquad (0, 0, 0, 10, 129, 108, 5)\boldsymbol{\delta}),
\end{aligned} \tag{3.16}$$

where $\boldsymbol{\delta} = (\delta_{\ell-3}, \delta_{\ell-2}, \delta_{\ell-1}, \delta_{\ell}, \delta_{\ell+1}, \delta_{\ell+2}, \delta_{\ell+3})^{T}$, for row ℓ. The matrices A and C are symmetric while the form of the matrix B has a somewhat more complex structure.

We will time centre on $t = (n + 1/2)\Delta t$ and write

$$\boldsymbol{\delta} = \frac{1}{2}\left(\boldsymbol{\delta}^n + \boldsymbol{\delta}^{n+1}\right), \qquad \frac{d\boldsymbol{\delta}}{dt} = \frac{1}{\Delta t}\left(\boldsymbol{\delta}^{n+1} - \boldsymbol{\delta}^n\right), \tag{3.17}$$

where the $\boldsymbol{\delta}^n$ are the parameters at time $n\Delta t$, and Δt is the timestep. Then (3.14), can be written as

$$\left[A + \frac{\varepsilon\Delta t}{2}B + \frac{\nu\Delta t}{2}C\right]\boldsymbol{\delta}^{n+1} = \left[A - \frac{\varepsilon\Delta t}{2}B - \frac{\nu\Delta t}{2}C\right]\boldsymbol{\delta}^n. \tag{3.18}$$

The matrices A, C are independent of the time; so, they will remain constant throughout the calculations. While the matrix B is dependent on the time, it must therefore be recalculated at each step. Apply the boundary conditions and eliminate $\delta_{-1}, \delta_{N+1}$ from (3.18) which then becomes $(N + 1) \times (N + 1)$ septadiagonal system and can be solved using septadiagonal algorithm.

The time evolution of the approximate solution $u_N(x, t)$ is determined by the time evolution of the vector $\boldsymbol{\delta}^n$. This is found by repeatedly solving the system (3.18), once the initial vector $\boldsymbol{\delta}^0$ has been computed from the initial conditions. The system (3.18) is septadiagonal and so can be solved using the septadiagonal algorithm, but an inner iteration is needed at each timestep to scope with the nonlinear terms in which the matrix B depends on $\boldsymbol{\delta} = (1/2)(\boldsymbol{\delta}^n + \boldsymbol{\delta}^{n+1})$. The following solution procedure is followed.

(i) At time $t = 0$, for the initial step of the inner iteration, we approximate B by B^* calculated from $\boldsymbol{\delta}^0$ only and obtain a first approximation to $\boldsymbol{\delta}^1$ from (3.18). We then iterate, using (3.18) with matrix B calculated from $\boldsymbol{\delta} = (1/2)(\boldsymbol{\delta}^0 + \boldsymbol{\delta}^1)$, for up to 10-times to refine the approximation to $\boldsymbol{\delta}^1$.

(ii) At the other timesteps we use for the matrix B, at the first step of inner iteration, B^* derived from $\boldsymbol{\delta}^* = \boldsymbol{\delta}^n + (1/2)(\boldsymbol{\delta}^n - \boldsymbol{\delta}^{n-1})$, to obtain a first approximation to $\boldsymbol{\delta}^{n+1}$, by solving (3.18). We then iterate, using (3.18) with matrix B calculated from $\boldsymbol{\delta} = (1/2)(\boldsymbol{\delta}^n + \boldsymbol{\delta}^{n+1})$, two- or three-times to refine the approximation to $\boldsymbol{\delta}^{n+1}$.

4. The Stability Analysis

An investigation into the stability of the numerical scheme (3.18) is based on the Von Neumann theory in which the growth factor of a typical Fourier mode defined as

$$\delta_j^n = \widehat{\delta}^n e^{ijkh}, \tag{4.1}$$

where k is the mode number, and h is the element size, is determined for a linearization of the numerical scheme (3.18).

In this linearization, we assume that the quantity u in the nonlinear term uu_x is locally constant. This is equivalent to assuming that the corresponding values of δ_j are also constant and equal to d.

A linearized recurrence relationship corresponding to (3.18) is then given by

$$\begin{aligned}&\alpha_1\delta_{\ell-3}^{n+1}+\alpha_2\delta_{\ell-2}^{n+1}+\alpha_3\delta_{\ell-1}^{n+1}+\alpha_4\delta_{\ell}^{n+1}+\alpha_5\delta_{\ell+1}^{n+1}+\alpha_6\delta_{\ell+2}^{n+1}+\alpha_7\delta_{\ell+3}^{n+1}\\&\quad=\alpha_7\delta_{\ell-3}^{n}+\alpha_6\delta_{\ell-2}^{n}+\alpha_5\delta_{\ell-1}^{n}+\alpha_8\delta_{\ell}^{n}+\alpha_3\delta_{\ell+1}^{n}+\alpha_2\delta_{\ell+2}^{n}+\alpha_1\delta_{\ell+3}^{n},\end{aligned}\tag{4.2}$$

where

$$\begin{gathered}\alpha_1=r_1-3r_2-3r_3,\qquad \alpha_2=120r_1-168r_2-72r_3,\\ \alpha_3=1191r_1-735r_2-45r_3,\qquad \alpha_4=2416r_1+240r_3,\\ \alpha_5=1191r_1+735r_2+45r_3,\qquad \alpha_6=120r_1+168r_2+72r_3,\\ \alpha_7=r_1+3r_2+3r_3,\qquad \alpha_8=2416r_1-240r_3,\\ r_1=\frac{h}{140},\qquad r_2=\frac{\varepsilon d\Delta t}{20},\qquad r_3=\frac{\nu\Delta t}{20}.\end{gathered}\tag{4.3}$$

Substituting (4.1) into (4.2) we obtain

$$(a+120+ib)\widehat{\delta}^{n+1}=(a-120-ib)\widehat{\delta}^{n},\tag{4.4}$$

where

$$\begin{aligned}&i=\sqrt{-1},\\&a=r_1\cos 3kh+120r_1\cos 2kh+1191r_1\cos kh+1208r_1,\\&b=(3r_2+3r_3)\sin 3kh+(168r_2+72r_3)\sin 2kh+(1470r_2+45r_3)\sin kh.\end{aligned}\tag{4.5}$$

Let $\widehat{\delta}^{n+1}=g\widehat{\delta}^{n}$ and substitute in (4.4) to give

$$g=\frac{a-120-ib}{a+120+ib},\tag{4.6}$$

where g is the growth factor for the mode. Then, the modulus of the growth factor is

$$|g|=\sqrt{g\overline{g}}=\sqrt{\frac{(a-120)^2+b^2}{(a+120)^2+b^2}}<1.\tag{4.7}$$

Therefore the linearized scheme is unconditionally stable.

5. The Initial State

From the initial condition $u(x,0)$ on the function $u(x,t)$, we must determine the initial vector $\boldsymbol{\delta}^0$ so that the time evolution of $\boldsymbol{\delta}$, using (3.18), can be started.

First rewrite (3.1) for the initial condition as

$$u_N(x,0) = \sum_{j=-1}^{N+1} \delta_j^0 \phi_j(x), \tag{5.1}$$

where $\boldsymbol{\delta}_j^0$ are unknown parameters to be determined. To do this we require $u_N(x,0)$ to satisfy the following constraints:

(a) it must agree with the initial condition $u(x,0)$ at the knots; (3.3) leads to $N+1$ conditions, that is, $u_N(x_\ell,0) = u(x_\ell,0), \ell = 0,\ldots,N$,

(b) the first or second derivatives of the approximate initial condition shall agree with those of the exact initial condition at both ends of the range; (3.5) or (3.6) produces two further equations, that is, $u'(x_0,0) = u'(x_N,0) = 0$.

The initial vector $\boldsymbol{\delta}^0$ is then determined as the solution of a matrix equation derived from (3.3)–(3.6)

$$M\boldsymbol{\delta}^0 = \mathbf{b}. \tag{5.2}$$

This system can be solved using the Thomas algorithm.

6. The Test Problems

Numerical solutions of Burgers' equation for three test problems will now be considered. To measure the accuracy of the numerical algorithm, we compute the difference between the analytic and numerical solutions at each mesh point after specified timesteps and use these to calculate the discrete L_2- and L_∞-error norms.

(a) Burgers' equation has the analytic solution [23]

$$u(x,t) = \frac{x/t}{1+\sqrt{(t/t_0)}\exp(x^2/4\nu t)}, \quad 0 \le x \le 1,\ t \ge 1, \tag{6.1}$$

where $t_0 = \exp(1/8\nu)$. With (6.1), at time $t = 1$ as initial condition, numerical solutions are determined up to a time of $t = 4$, and use as boundary conditions $u(0,t) = u(1,t) = 0$. We discuss the three cases.

(i) Figures 1 and 2 shows us the behavior of the numerical solutions for viscosity coefficients $\nu = 0.005$ and 0.0005 at times from $t = 1$ to 3.1 and from $t = 1$ to 3.25, respectively. It is seen that for the smaller viscosity value the propagation front is steeper. These graphs agree with those reported by Nguyen and Reynen [23] to within plotting accuracy. In both cases, if the exact solutions are plotted on the same diagram, the curves are indistinguishable. The L_2- and L_∞-error norms are compared with results presented in the papers [9, 14], which are given in Tables 1, 2, and 3. From these tables, we find that the errors obtained by the present algorithm are smaller than the errors calculated by [9, 14].

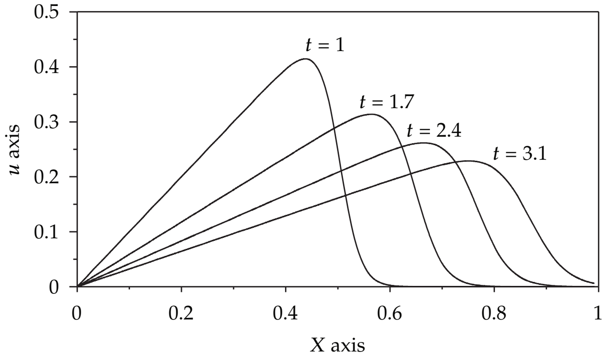

Figure 1: Example (a): $v = 0.005, h = 0.02, \Delta t = 0.1$.

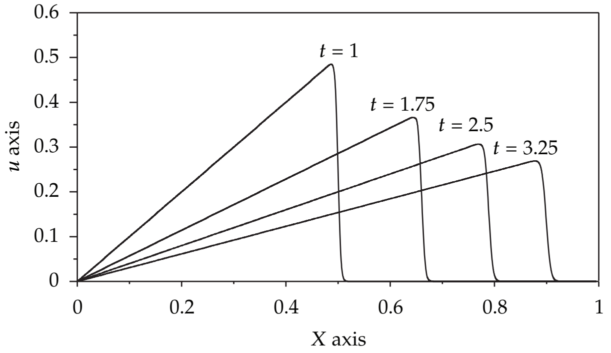

Figure 2: Example (a): $v = 0.0005, h = 0.005, \Delta t = 0.01$.

(ii) The case with $\nu = 0.0005$, $h = 0.005$, $\Delta t = 0.01$, at different times $t = 1.7, 2.5$, and 3.25, respectively. The numerical and the analytical solutions computed by the present approach are given in Table 4, with corresponding previous results [15]. From Table 4, we deduce that the numerical solutions computed by the present algorithm are more accurate than those evaluated by [15], at different times.

(iii) In this case, we take $h = 0.01$, $\Delta t = 0.01$, and viscosity coefficients $\nu = 0.005$, $\nu = 0.01$. The results given in Tables 5 and 6 show us that the computed errors using the present algorithm are smaller than the corresponding errors obtained by Raslan [18], even, when we chose large timestep.

(b) A second analytic solution is [11]

$$u(x,t) = \frac{1}{2}\left[1 - \tanh\left\{\frac{1}{4\nu}\left(x - \frac{1}{2}t - 15\right)\right\}\right]. \tag{6.2}$$

We take as initial condition (6.2), at $t = 0$, over the range $0 \le x \le 30$, with boundary conditions

$$u(0,t) = 1, \quad u(30,t) = 0, \quad t > 0. \tag{6.3}$$

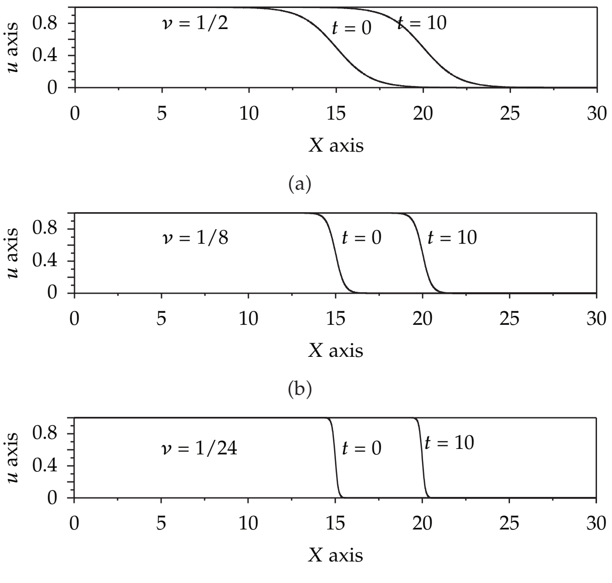

Figure 3: The behavior of the numerical solution with different viscosity at different times.

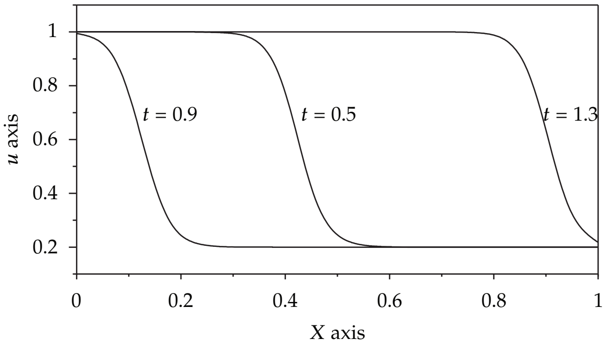

Figure 4: Example (c): the numerical solution with $v = 0.01$ at different times.

Table 2: Error norm $L_2 \times 10^3$ for problem (a).

		$\nu = 0.005$			$\nu = 0.0005$	
	$t = 1.7$	$t = 2.4$	$t = 3.1$	$t = 1.75$	$t = 2.5$	$t = 3.25$
Present	1.206	0.653	0.434	0.581	0.252	0.179
[14]	1.003	0.598	0.522	0.567	0.308	0.291
[9]	2.281	1.175	0.808	2.023	2.094	0.933

Table 3: Error norm $L_\infty \times 10^3$ for problem (a).

		$\nu = 0.005$			$\nu = 0.0005$	
	$t = 1.7$	$t = 2.4$	$t = 3.1$	$t = 1.75$	$t = 2.5$	$t = 3.25$
Present	3.868	1.679	1.295	5.371	1.619	0.821
[14]	3.730	2.029	1.537	5.880	2.705	2.291
[9]	7.150	2.921	2.297	14.033	12.615	8.394

Table 4: Comparison of results at different times for $\nu = 0.0005$ and $0 \leq x \leq 1$, with $h = 0.005$ and $\Delta t = 0.01$, problem (a).

	$t = 1.7$	$t = 1.7$	$t = 1.7$	$t = 2.5$	$t = 2.5$	$t = 2.5$	$t = 3.25$	$t = 3.25$	$t = 3.25$
x	Exact	Numer	Numer	Exact	Numer	Numer	Exact	Numer	Numer
		Present	[15]		Present	[15]		Present	[15]
0.1	0.05882	0.05882	0.05883	0.04000	0.04003	0.04000	0.03077	0.03082	0.03077
0.2	0.11765	0.11765	0.11765	0.08000	0.08000	0.08000	0.06154	0.06154	0.06154
0.3	0.17647	0.17647	0.17648	0.12000	0.12000	0.12001	0.09231	0.09231	0.09231
0.4	0.23529	0.23529	0.23531	0.16000	0.16000	0.16001	0.12308	0.12308	0.12308
0.5	0.29412	0.29412	0.29414	0.20000	0.20000	0.20001	0.15385	0.15385	0.15385
0.6	0.35294	0.35287	0.35296	0.24000	0.24000	0.24001	0.18462	0.18462	0.18462
0.7	0.00000	0.00000	0.00000	0.28000	0.28000	0.28001	0.21538	0.21538	0.21539
0.8	0.00000	0.00000	0.00000	0.00977	0.00994	0.00811	0.24615	0.24615	0.24616
0.9	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.12433	0.12401	0.12358

Table 5: Comparison of errors at different times for $\nu = 0.005$ and $0 \leq x \leq 1$, with $h = 0.01$, problem (a).

	L_2-error	L_2-error	L_∞-error	L_∞-error
Time	Present	[18]	Present	[18]
	$\Delta t = 0.01$	$\Delta t = 0.001$	$\Delta t = 0.01$	$\Delta t = 0.001$
1.5	0.00019	0.00770	0.00132	0.04210
2.0	0.00018	0.01154	0.00102	0.05166
2.5	0.00017	0.01431	0.00082	0.05623
3.0	0.00016	0.01654	0.00069	0.05882
3.5	0.00019	0.01872	0.00109	0.06046
4.0	0.00035	0.02255	0.00268	0.07691

Table 6: Comparison of errors at different times for $\nu = 0.01$ and $0 \leq x \leq 1$, with $h = 0.01$, problem (a).

	L_2-error	L_2-error	L_∞-error	L_∞-error
Time	Present	[18]	Present	[18]
	$\Delta t = 0.01$	$\Delta t = 0.001$	$\Delta t = 0.01$	$\Delta t = 0.001$
1.07	0.00019	0.00207	0.00168	0.01660
1.14	0.00021	0.00333	0.00166	0.02413
1.35	0.00023	0.00613	0.00147	0.03667
1.70	0.00022	0.00938	0.00120	0.04677
2.40	0.00020	0.01376	0.00086	0.05537
3.10	0.00021	0.01689	0.00074	0.05906
3.24	0.00022	0.01746	0.00089	0.05955
3.31	0.00023	0.01775	0.00096	0.05977

Table 7: Comparison of $L_2 \times 10^3$ at different times for $0 \leq x \leq 30$, with $h = 0.1$, and $\Delta t = 0.025$, problem (b).

	$t = 2$	$t = 4$	$t = 6$	$t = 8$	$t = 10$	ν
Present	0.002	0.003	0.003	0.004	0.004	1/2
[9]	0.006	0.010	0.014	0.022	0.043	1/2
Present	0.032	0.036	0.038	0.038	0.038	1/8
[9]	0.228	0.227	0.227	0.227	0.226	1/8
Present	0.431	0.433	0.433	0.433	0.433	1/24
[9]	3.431	3.426	3.426	3.426	3.429	1/24

Table 8: Comparison of $L_\infty \times 10^3$ at different times for $0 \leq x \leq 30$, with $h = 0.1$, and $\Delta t = 0.025$, problem (b).

	$t = 2$	$t = 4$	$t = 6$	$t = 8$	$t = 10$	ν
Present	0.001	0.002	0.002	0.003	0.003	1/2
[9]	0.004	0.006	0.007	0.017	0.045	1/2
Present	0.044	0.049	0.050	0.051	0.051	1/8
[9]	0.346	0.339	0.340	0.347	0.356	1/8
Present	0.937	0.939	0.939	0.939	0.939	1/24
[9]	4.340	4.336	4.363	4.402	4.432	1/24

Table 9: Comparison of results for various numerical schemes with exact at time $t = 0.5$, for $\nu = 0.01$.

	SGA [10]	PAG [10]	ML [9]	CSCM [14]	CSCM [15]	Present	
x	$h = 1/18$	$h = 1/18$	$h = 1/36$	$h = 1/36$	$h = 1/36$	$h = 1/36$	Exact
	$\Delta t = .001$	$\Delta t = .001$	$\Delta t = .025$	$\Delta t = .025$	$\Delta t = .025$	$\Delta t = .025$	
.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
.056	1.000	1.000	1.000	1.000	1.000	1.000	1.000
.111	1.000	1.000	1.000	1.000	1.000	1.000	1.000
.167	1.000	1.000	1.000	1.000	1.000	1.000	1.000
.222	0.998	0.999	1.000	1.000	1.000	1.000	1.000
.278	0.991	0.997	0.998	0.999	0.999	0.999	0.998
.333	0.970	0.982	0.985	0.985	0.986	0.986	0.980
.389	0.862	0.850	0.851	0.847	0.850	0.850	0.847
.444	0.461	0.444	0.447	0.452	0.448	0.448	0.452
.500	0.159	0.171	0.238	0.238	0.236	0.238	0.238
.556	0.300	0.286	0.205	0.204	0.204	0.204	0.204
.611	0.194	0.197	0.201	0.200	0.200	0.200	0.200
.667	0.213	0.211	0.200	0.200	0.200	0.200	0.200
.722	0.211	0.210	0.200	0.200	0.200	0.200	0.200
.778	0.188	0.190	0.200	0.200	0.200	0.200	0.200
.833	0.201	0.207	0.200	0.200	0.200	0.200	0.200
.889	0.191	0.193	0.200	0.200	0.200	0.200	0.200
.944	0.203	0.202	0.200	0.200	0.200	0.200	0.200
1.00	0.200	0.200	0.200	0.200	0.200	0.200	0.200

For $\nu = 1/2$, a very weak shock wave develops, when $\nu = 1/8$, we obtain a moderate shock wave and when $\nu = 1/24$, a strong shock wave is produced. As the value of ν is decreased the propagation front becomes steeper. The L_2- and L_∞-error norms for these simulations with the corresponding errors obtained by [9], are given in Tables 7 and 8. For large values of ν the errors are small and as the value of ν is decreased the errors tend to increase, but for the values of ν used here, the errors are still acceptable. From Tables 7 and 8, we deduce that the errors computed by the present algorithm are more accurate than those evaluated by [9] at different times. Figure 3 shows us the behavior of the numerical solutions for viscosity coefficients $\nu = 1/24$, at different times from $t = 0$ to 10.

(c) A third test example, consider the particular analytic solution of Burgers' equation [8, 10]

$$u(x,t) = \frac{[\alpha + \mu + (\alpha - \mu)\exp(\eta)]}{1 + \exp(\eta)}, \quad 0 \le x \le 1,\ t \ge 0, \tag{6.4}$$

where $\eta = \alpha(x - \mu t - \beta)/\nu, \alpha, \mu$, and β are constants.

The comparison can be made with [8–10, 14, 15], these constants are chosen to have the values $\alpha = 0.4$, $\beta = 0.125$, $\mu = 0.6$, and $\nu = 0.01$. The initial condition is obtained from (6.4), when $t = 0$. The boundary conditions are taken as

$$u(0,t) = 1, \quad (1,t) = 0.2, \quad t \ge 0. \tag{6.5}$$

From Figure 4, it can be seen that the solution represents a travelling wave, initially situated at $x = \beta$, moving to the right with speed μ. The present method behaves very well until the wave encounters the right-hand boundary. The numerical results at $t = 0.5$ are compared with the analytic solution, and the results presented in papers [8–10, 14, 15], known as methods of method of lines (ML), standard Galerkin approach (SGA), product approximation Galerkin method (PAG), cubic B-spline collocation method (CSCM), and a numerical solution of Burgers' equation using cubic B-spline (CSCM), are given in Table 9. It is seen from Table 9 that the agreement between the numerical and the exact solution appears very satisfactorily.

7. Conclusions

We have seen that the algorithm proposed here using Galerkins' method with cubic spline shape functions compares well in the accuracy with the other published methods, such as methods of lines, standard Galerkin approach, product approximation Galerkin method, and cubic B-spline collocation method. The L_2- and L_∞-error norms are satisfactorily small during the simulations. A linear stability analysis based on the Von Neumann theory shows that the numerical scheme is unconditionally stable. We conclude that a finite element approach based on Galerkins' method with cubic spline shape functions is eminently suitable for the computation of solutions to Burgers' equation. We believe that the approach should be suitable for other applications where the continuity of derivatives is essential.

References

[1] J. D. Cole, "On a quasi-linear parabolic equation occurring in aerodynamics," *Quarterly of Applied Mathematics*, vol. 9, pp. 225–236, 1951.

[2] J. M. Burgers, "A mathematical model illustrating the theory of turbulence," in *Advances in Applied Mechanics*, pp. 171–199, Academic Press, New York, NY, USA, 1948.

[3] E. Varoğlu and W. D. L. Finn, "Space-time finite elements incorporating characteristics for the Burgers equation," *International Journal for Numerical Methods in Engineering*, vol. 16, pp. 171–184, 1980.

[4] E. Hopf, "The partial differential equation $u_t + uu_{xx} = \nu u_{xx}$," *Communications on Pure and Applied Mathematics*, vol. 3, pp. 201–230, 1950.

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

[6] D. U. Rosenberg, *Methods for Solution of Partial Differntial Equations*, vol. 113, American Elsevier, New York, NY, USA, 1969.

[7] J. Caldwell, P. Wanless, and A. E. Cook, "A finite element approach to Burgers' equation," *Applied Mathematical Modelling*, vol. 5, no. 3, pp. 189–193, 1981.

[8] B. M. Herbst, S. W. Schoombie, and A. R. Mitchell, "A moving Petrov-Galerkin method for transport equations," *International Journal for Numerical Methods in Engineering*, vol. 18, no. 9, pp. 1321–1336, 1982.

[9] L. R. T. Gardner, G. A. Gardner, and A. H. A. Ali, "A method of lines solutions for Burgers' equation," in *Computer Mech*, Y. K. Cheung et al., Ed., pp. 1555–1561, Balkema, Rotterdam, 1991.

[10] I. Christie, D. F. Griffiths, A. R. Mitchell, and J. M. Sanz-Serna, "Product approximation for nonlinear problems in the finite element method," *Journal of Numerical Analysis*, vol. 1, no. 3, pp. 253–266, 1981.

[11] S. G. Rubin and R. A. Graves, Jr., "Viscous flow solutions with a cubic spline approximation," *Computers & Fluids*, vol. 3, pp. 1–36, 1975.

[12] S. G. Rubin and P. K. Khosla, "Higher-order numerical solutions using cubic splines," *American Institute of Aeronautics and Astronautics*, vol. 14, no. 7, pp. 851–858, 1976.

[13] J. Caldwell, "Application of cubic splines to the nonlinear Burgers' equation," in *Numerical Methods for Nonlinear Problems*, vol. 3, pp. 253–261, Pineridge Press, 1986.

[14] A. H. A. Ali, G. A. Gardner, and L. R. T. Gardner, "A collocation solution for Burgers' equation using cubic Cubic-spline finite elements," *Computer Methods in Applied Mechanics and Engineering*, vol. 100, no. 3, pp. 325–337, 1992.

[15] İ. Dağ, D. Irk, and B. Saka, "A numerical solution of the Burgers' equation using cubic B-splines," *Applied Mathematics and Computation*, vol. 163, no. 1, pp. 199–211, 2005.

[16] A. A. Soliman, "Numerical technique for Burgers' equation based on similarity reductions," in *International Conference on Applied Computational Fluid Dynamics*, pp. 559–566., Bejing, China, October 2000.

[17] A. A. Soliman, "The modified extended tanh-function method for solving Burgers-type equations," *Physica A*, vol. 361, no. 2, pp. 394–404, 2006.

[18] K. R. Raslan, "A collocation solution for Burgers equation using quadratic B-spline finite elements," *International Journal of Computer Mathematics*, vol. 80, no. 7, pp. 931–938, 2003.

[19] G. A. Gardner, A. H. A. Ali, and L. R. T. Gardner, "A finite element solution for the korteweg-de vries equation using cubic B-spline shape function," in *Proceedings of the Information Security Policy Made Easy (ISNME '89)*, R. Gruber, J. Periaux, and R. P. Shaw, Eds., vol. 2, pp. 565–570, Springer, 1989.

[20] L. R. T. Gardner and G. A. Gardner, "Solitary waves of the regularised long-wave equation," *Journal of Computational Physics*, vol. 91, no. 2, pp. 441–459, 1990.

[21] O. C. Zienkiewicz, *The Finite Elment Method*, McGraw Hill, London, UK, 3th edition, 1977.

[22] P. M. Prenter, *Splines and Variational Methods*, John Wiley & Sons, New York, NY, USA, 1989.

[23] H. Nguyen and J. Reynen, "A space-time finite element approach to Burgers' equation," in *Numerical Methods for Nonlinear Problems*, E. Hinton, Ed., vol. 2, pp. 718–728, Pineridge Press, 1984.