Some advantages of implementing an adaptive moving mesh for the solution to the Burgers equation

Sudi Mungkasi
Department of Mathematics, Faculty of Science and Technology,
Sanata Dharma University, Mriran, Tromol Pos 29, Yogyakarta 55002, Indonesia
E-mail: sudi@usd.ac.id

Abstract. The Burgers equation is considered. The equation is solved using finite difference methods. The standard finite difference method may lead to inaccurate solutions, unless a very fine mesh is used, which results in expensive computations. Therefore, we implement an adaptive finite difference moving mesh method as an alternative numerical method to solve the equation. The advantages of implementing the adaptive method are investigated.

1. Introduction
The Burgers equation has been widely used to model some problems in gas dynamics and traffic flows. Therefore, accurately solving the equation is useful. It is well-known that the behaviour of the solutions is influenced by the viscosity factor involved in the equation [1].

Based on the viscosity factor the Burgers equation is of two types, namely, viscous and inviscid [2]. When the viscosity factor is nonzero, the equation is viscous and so have a source term. Otherwise it is inviscid and does not have any source terms. This paper deals with the viscous type of the equation.

We solve the Burgers equation using finite difference methods and refer to the work of Huang and Russell [1] (see [3]-[5] for kinds of adaptive strategy). The standard finite difference method may lead to inaccurate solutions for coarse mesh. Therefore, an alternative finite difference method is considered. We implement an adaptive moving mesh finite difference method as the alternative. In this paper, this numerical method is called the adaptive moving mesh method or simply the adaptive method. Our contribution is identifying the advantages of this adaptive method.

This paper is structured as follows. We first write the problem and numerical methods that shall be used to solve the problem. Numerical results are then presented and discussed. Some concluding remarks are given at the final part.

2. The considered problem and solvers
In this section, we present the Burgers equation and the numerical methods to solve the equation.

2.1. The considered problem
The Burgers equation is well-known to be a simplification of the Navier-Stokes equation. The equation is named after Johannes Martinus Burgers, a Dutch scientist, for his fundamental work on the theory of turbulence [6]. The Burgers equation is

\[
\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2}
\]

where \( u \) is the velocity, \( \nu \) is the viscosity, and the equation is considered to be in a one-dimensional domain.
\[ u_t = \varepsilon u_{xx} - \left( \frac{u^2}{2} \right)_t \]  

where the space domain is all \( x \in (0, 1) \) and the time domain is all \( t > 0 \). The notation \( u_t \) means the first partial derivative of the quantity \( u(x,t) \) with respect to \( t \). The notations \( u_{xx} \) and \((u^2/2)_t\) are understood similarly. The parameter \( \varepsilon \) is positive, so that the Burgers equation (1) is viscous.

Following Huang and Russell [1], we take the boundary condition
\[ u(0,t) = u(1,t) = 0 \]  

and the initial condition
\[ u(x,0) = \sin(2\pi x) + \frac{1}{2}\sin(\pi x) . \]  

Equations (1)-(3) form an initial-boundary value problem.

2.2. The standard finite difference method
To solve the Burgers equation (1), we discretize the space domain pointwisely into \( N \) equidistant points, as
\[ x_j = (j - 1) h \]  

for \( j = 1,\ldots,N \). When we use central finite difference approximations for all spatial derivatives, we obtain the semi-discrete finite difference method
\[ \frac{du_j}{dt} = \frac{\varepsilon}{h^2}(u_{j+1} - 2u_j + u_{j-1}) - \frac{1}{4h}(u_{j+1}^2 - u_{j-1}^2) \]  

for \( j = 2,\ldots,N-1 \). This is a standard finite difference method. The numerical boundary conditions are \( u_1(t) = 0 \) and \( u_N(t) = 0 \) for \( t > 0 \). The numerical initial conditions are
\[ u_j(0) = \sin(2\pi x_j) + \frac{1}{2}\sin(\pi x_j) \]  

for \( j = 1,\ldots,N \).

With the initial conditions (6), we can solve the semi-discrete equation (5) using an ODE solver. One can use ODE solvers available in some computer programming language, such as MATLAB, Python, or any others. In this paper, for simplicity, we use the Euler method to solve equation (5). We present the numerical results later in Section 3.

2.3. An adaptive moving mesh finite difference method
Alternatively, we use an adaptive moving mesh method (see mainly [1] and further [3-5]). We discretize the space domain pointwisely into \( N \) points as follows. We consider the transformation of the quantity
\[ \hat{u}(\xi,t) = u(x(\xi,t),t) , \]  

with the transformation of the space
\[ x_j(t) = x(\xi_j,t) \]  

for \( j = 1,\ldots,N \). Here
\[ \xi_j = \frac{j - 1}{N - 1} \]  

for \( j = 1,\ldots,N \). We then have
\[ \hat{u}_\xi = u_x x_\xi \quad \text{and} \quad \hat{u}_t = u_t + u_x x_t \]  

Note that \( x_t = \partial x/\partial t \), where \( x = x(\xi,t) \). Substitution of the transformed variables into the Burgers equation leads to
When we use central difference approximations for all spatial derivatives, we obtain the adaptive finite difference moving mesh method

\[
\begin{align*}
\frac{du_j}{dt} & - \frac{(u_{j+1} - u_{j-1})}{x_{j+1} - x_{j-1}} \frac{dx_j}{dt} = 2\varepsilon \left[ \frac{(u_{j+1} - u_j)}{(x_{j+1} - x_j)} - \frac{(u_j - u_{j-1})}{(x_j - x_{j-1})} \right] - \frac{1}{2} \frac{(u_{j+1}^2 - u_{j-1}^2)}{x_{j+1} - x_{j-1}} \\
& \quad \text{for } j = 2, \ldots, N-1, \text{ where } u_j(t) = \hat{u}(\xi_j, t) := u(x_j(t), t). \text{ The time derivative in the computational domain is defined by}
\end{align*}
\]

\[
x_j = \frac{1}{\rho \tau} (\rho x_j)_{\xi}
\]

with \(x(0, t) = 0\) and \(x(1, t) = 1\). In addition, we take

\[
\rho = \left( 1 + \frac{1}{\alpha} |u_{xx}|^2 \right)^{\frac{1}{3}}.
\]

Here

\[
\alpha = \max \left\{ 1, \left[ \int_0^1 |u_{xx}|^{2/3} \, dx \right]^{-\frac{3}{2}} \right\}.
\]

Therefore, we obtain

\[
\frac{dx_j}{dt} = \frac{1}{\rho_j \tau \Delta \xi^2} \left[ \rho_{j+1} + \rho_j \frac{(x_{j+1} - x_j)}{2} - \rho_j + \rho_{j-1} \frac{(x_j - x_{j-1})}{2} \right]
\]

for \( j = 2, \ldots, N-1 \), and also, \( \frac{dx_i}{dt} = 0 \) as well as \( \frac{dx_N}{dt} = 0 \). Note that \( \Delta \xi = 1/(N-1) \). In the computations, we use the discretization of \( \rho \) as

\[
\rho_j = \left( 1 + \frac{1}{\alpha_h} |u_{xx,j}|^2 \right)^{\frac{1}{3}}
\]

for \( j = 1, \ldots, N \), where

\[
\alpha_h = \max \left\{ 1, \left[ \sum_{j=2}^{N} \frac{(x_j - x_{j-1})}{2} \left( |u_{xx,j}|^{2/3} + |u_{xx,j-1}|^{2/3} \right) \right]^{-\frac{3}{2}} \right\}.
\]

The spatial derivatives are taken as

\[
u_{xx,j} = \frac{2}{(x_{j+1} - x_{j-1})} \left[ \frac{(u_{j+1} - u_j)}{(x_{j+1} - x_j)} - \frac{(u_j - u_{j-1})}{(x_j - x_{j-1})} \right]
\]

for \( j = 2, \ldots, N-1 \), and also,

\[
u_{xx,1} = \frac{2}{(x_2 - x_1)} \frac{[(x_2 - x_1)(u_3 - u_2) - (x_3 - x_1)(u_2 - u_1)]}{(x_2 - x_1)(x_3 - x_2)(x_3 - x_1)}
\]

as well as

\[
u_{xx,N} = \frac{2}{(x_{N+1} - x_N)} \frac{[(x_{N+1} - x_N)(u_{N-2} - u_{N-1}) - (x_{N-2} - x_N)(u_{N-1} - u_N)]}{(x_{N+1} - x_N)(x_{N-1} - x_N)(x_{N-2} - x_{N-1})}
\]

Furthermore,

\[
\rho_j := \frac{1}{4} \rho_{j-1} + \frac{1}{2} \rho_j + \frac{1}{4} \rho_{j+1}
\]
for $j = 2, \ldots, N - 1$, and also,
\[
\rho_j := \frac{1}{2} \rho_{j-1} + \frac{1}{2} \rho_j \tag{23}
\]
as well as
\[
\rho_N := \frac{1}{2} \rho_{N-1} + \frac{1}{2} \rho_N. \tag{24}
\]

We solve equation (12) using the ode15i MATLAB function, which is an implicit method [1]. The numerical results are presented in the next section.

3. Numerical results
We shall compare the results of the adaptive moving mesh method with the standard finite difference method in this section.

![Figure 1](image1.png) \hspace{2cm} ![Figure 2](image2.png)

**Figure 1.** Results of the standard method with 21 points and viscosity $10^{-2}$ for time instants $t = 0, 0.07, 0.15, 0.3, 0.64, 1$.

**Figure 2.** Results of the standard method with 101 points and viscosity $10^{-2}$ for time instants $t = 0, 0.07, 0.15, 0.3, 0.64, 1$.

![Figure 3](image3.png) \hspace{2cm} ![Figure 4](image4.png)

**Figure 3.** Results of the standard method with 1001 points and viscosity $10^{-4}$ for time instants $t = 0, 0.07, 0.15, 0.3, 0.64, 1$.

**Figure 4.** Results of the standard method with 5001 points and viscosity $10^{-4}$ for time instants $t = 0, 0.07, 0.15, 0.3, 0.64, 1$.

![Figure 5](image5.png) \hspace{2cm} ![Figure 6](image6.png)

**Figure 5.** Results of the adaptive method with 21 points and viscosity $10^{-4}$ for time instants $t = 0, 0.07, 0.15, 0.3, 0.64, 1$.

**Figure 6.** Mesh evolution of the adaptive method with 21 points and viscosity $10^{-4}$ for time instants $t = 0, 0.07, 0.15, 0.3, 0.64, 1$. 
Table 1. Results for the largest magnitude of the solutions produced by the adaptive method with 21 points and the standard method with 5001 points. Here the results of the standard method is considered as the reference solution, so the absolute and relative errors can be computed.

| The considered time (second) | The maximum magnitude of the solution produced | Absolute error | Relative error |
|------------------------------|-----------------------------------------------|----------------|----------------|
|                              | by the adaptive method with 21 points          | by the standard method with 5001 points |                |                |
| 0.00                         | 1.3555650                                     | 1.3679074      | 0.0123424      | 0.0090228      |
| 0.07                         | 1.3648793                                     | 1.3678883      | 0.0030090      | 0.0021997      |
| 0.15                         | 1.3657246                                     | 1.3678663      | 0.0021417      | 0.0015657      |
| 0.30                         | 1.3228471                                     | 1.3350550      | 0.0122078      | 0.0091440      |
| 0.64                         | 0.9630831                                     | 0.9497996      | 0.0132834      | 0.0139855      |
| 1.00                         | 0.7684366                                     | 0.7553905      | 0.0130460      | 0.0172706      |

The average absolute error is 0.0093384
The average relative error is 0.0088647

We consider two cases. The first is the Burgers equation with the viscosity factor $\varepsilon = 10^{-2}$. The second is with the viscosity factor $\varepsilon = 10^{-4}$. A smaller viscosity factor results in a solution containing a sharper "discontinuity-like" as time evolves. We call a discontinuity-like, because it resembles a shock discontinuity, but in fact the exact solution is continuous. If the viscosity factor is zero, then the exact solution contains a shock discontinuity, which is not a discontinuity-like solution.

When the first case with $\varepsilon = 10^{-2}$ is solved using the standard finite difference method, there is no obvious a discontinuity-like in the solution. The results for the first case is shown in Figure 1 and Figure 2. Figure 1 is produced using 21 uniform spatial points. We see in Figure 1 that the solution is oscillatory, because the number of spatial points does not give enough resolution for this case. However if we refine the space discretization by taking more number of points we have the solution with the correct behaviour. That is, no oscillation appears in the solution, as shown in Figure 2. Here Figure 2 is produced using 101 uniform spatial points.

The problem becomes more difficult to solve when the viscosity factor is smaller, such as where $\varepsilon = 10^{-4}$ which we consider as the second case. Even when we discretize the spatial domain into 1001 points, the resolution is not enough. This is illustrated in Figure 3, where oscillation appears around the position of the discontinuity-like. This artificial oscillation does not occur in the solution if we discretize the space into much finer mesh, such as 5001 uniform points. With this excessive number of points, the solutions are very accurate. This is shown in Figure 4. However, this fine mesh makes the computation is very expensive. The average computational time with 5001 points using the standard method is 10 seconds.

In contrast, fine mesh is not necessary in the adaptive moving mesh method. For example we solve the second case, which has the viscosity factor $\varepsilon = 10^{-4}$. We use only 21 computational points to obtain relatively the same accurate solutions. The numerical solutions are shown in Figure 5. The corresponding moving meshes are shown in Figure 6. The average of relative $L^\infty$ errors between the reference solutions in Figure 4 and the numerical solutions in Figure 5 is 0.89 % (see Table 1 for detailed results). Therefore, the adaptive moving mesh method results in very accurate solutions. Furthermore we note that the average computational time with 21 points using the adaptive method is about 3 seconds. This means that the adaptive method is not only accurate, but also efficient in terms
of computer memory as well as the computational time. These give great advantages for us, as accurate results and fast computation are always desired in practice.

4. Conclusion
The adaptive moving mesh method is very accurate and efficient, and in addition, it requires small computer memory. We have demonstrated these advantages by solving the Burgers equation using the adaptive method. That is, the adaptive moving mesh method has better performance than the standard finite difference method. Extending the adaptive moving mesh method to solve higher dimensional problems could be a future direction of this research.

Acknowledgment
The author thanks Professor Weizhang Huang at The University of Kansas and Professor Robert Russell at Simon Fraser University for sharing some of their computer codes. An early version of this paper was titled “On the advantage of implementing an adaptive moving mesh to solve the Burgers’ equation” presented in the 9th Curtin University of Technology Science and Engineering International Conference (CUTSE 2014) at Curtin University Sarawak Malaysia on 3-4 December 2014.

References
[1] Huang W and Russell R D 2011 Adaptive Moving Mesh Methods (New York: Springer)
[2] LeVeque R J 2002 Finite Volume Methods for Hyperbolic Problems (Cambridge: Cambridge University Press)
[3] Budd C J, Huang W and Russell R D 2009 Adaptivity with moving grids Acta Numer. 18 1–131
[4] Remski J, Zhang J and Du Q 2014 On balanced moving mesh methods J. Comput. Appl. Math. 265 255–63
[5] Tang H and Tang T 2003 Adaptive mesh methods for one- and two-dimensional hyperbolic conservation laws SIAM J. Numer. Anal. 41 487–515
[6] Burgers J M 1948 A mathematical model illustrating the theory of turbulence Advances in Applied Mechanics 1 171–99