COMMUNICATIONS IN COMPUTATIONAL PHYSICS
Vol. **1**, No. 1, pp. 53-81

Commun. Comput. Phys.
February 2006

# The Error-Minimization-Based Strategy for Moving Mesh Methods

Konstantin Lipnikov[1,*] and Mikhail Shashkov[1]

[1] *Los Alamos National Laboratory, MS B284, Los Alamos, NM, 87545, USA.*

Received 12 May 2005; Accepted (in revised version) 22 July 2005

**Abstract.** The typical elements in a numerical simulation of fluid flow using moving meshes are a time integration scheme, a rezone method in which a new mesh is defined, and a remapping (conservative interpolation) in which a solution is transferred to the new mesh. The objective of the rezone method is to move the computational mesh to improve the robustness, accuracy and eventually efficiency of the simulation. In this paper, we consider the one-dimensional viscous Burgers' equation and describe a new rezone strategy which minimizes the $L_2$ norm of error and maintains mesh smoothness. The efficiency of the proposed method is demonstrated with numerical examples.

**Key words**: Moving meshes; Burgers' equation; error estimates.

## 1 Introduction

In a numerical simulation of fluid flow, the relationship of the motion of computational mesh to the motion of the fluid is an important issue. There are two choices that are typically made. In Lagrangian methods the mesh moves with the local fluid velocity, while in Eulerian methods the fluid flows through the mesh that is fixed in space.

In general, the motion of the mesh can be chosen arbitrarily. The moving mesh method described in this paper exploits this freedom to improve the robustness, accuracy and eventually efficiency of the simulation. The main elements in the simulation with arbitrary moving meshes are an explicit time integration scheme, a rezone method in which a new mesh is defined, and a remapping (conservative interpolation) in which a solution is transferred to the new mesh [44].

Our ultimate goal is to develop a robust and efficient rezone strategy for the system of gasdynamics equations in 2D and 3D. Since this is the very challenging task, we commence with a simpler problem: the one-dimensional viscous Burgers' equation. This equation has many

*Correspondence to: Konstantin Lipnikov, Los Alamos National Laboratory, MS B284, Los Alamos, NM, 87545, USA. Email: lipnikov@lanl.gov

 

important features of gasdynamics equations. It expresses conservation law and its solution can develop shock-like structures.

In this paper, we consider both the Lagrangian and Eulerian forms of Burgers' equation. The new rezone method employs the *look ahead strategy*. It changes the mesh at time $t^n$ in such a way to minimize the $L_2$-norm of error at time $t^{n+1}$. The analysis shows that under reasonable assumptions about mesh smoothness, solution regularity, accuracy of the remapping and time step, the leading term in the error depends on accuracy with which the solution at time $t^n$ is represented by its exact mean values. The latter is the well-known interpolation problem of the best piecewise constant fit with adjustable nodes [6]. The remapping is based on the linearity preserving methods from [35] which are second-order accurate for viscous Burgers' equation.

The error analysis for viscous Burgers' equation assumes that the mesh is smooth. The mesh smoothness is absolutely critical for stability of the overall simulation. Therefore, we modified the standard algorithm for the best piecewise constant fit to guarantee smoothness of the resulting mesh and still reduce the error. We found that the modified algorithm gives just slightly larger error in comparison with the original algorithm.

The paper outline is as follows. In Section 2, we describe the Eulerian and the Lagrangian forms of viscous Burgers' equation. In Section 3, we describe the moving mesh method with the error-minimization-based (EMB) rezone strategy for both the Eulerian and the Lagrangian forms of Burger's equation. In Section 4, we analyze the moving mesh method. In Section 5, we present results of numerical experiments. In Section 6, we give a short overview of related methods, emphasize some common ideas and important distinctions with our method. In concluding Section 7, we discuss plans for future work.

## 2 Viscous Burgers' equation

In this section we consider different forms and some properties of Burgers' equation that will be useful for the purpose of this paper.

### 2.1 Burgers' equation in the Eulerian form

The standard Eulerian form of the one-dimensional Burgers' equation is

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \varepsilon\frac{\partial^2 u}{\partial x^2}, \qquad t \in (0,T), \tag{2.1}$$

where $\varepsilon << 1$ is a given constant. For a finite $\varepsilon$, solution of equation (2.1) is a smooth function which may have sharp gradients whose steepness depends on how small $\varepsilon$ is. Equation (2.1) is subject to the initial condition

$$u(x,0) = U(x), \qquad x \in (-\infty, +\infty). \tag{2.2}$$

For simplicity, in this section, we assume that

$$u(x,t), \quad \frac{\partial u}{\partial x}(x,t) \to 0 \quad \text{when} \quad x \to \pm\infty\,.$$

If we introduce flux

$$F(x,t) = \frac{u^2}{2} - \varepsilon \frac{\partial u}{\partial x},$$

then equation (2.1) can be written in a conservative form

$$\frac{\partial u}{\partial t} + \frac{\partial F}{\partial x} = 0$$

which implies conservation of integral of $u$:

$$\frac{\partial}{\partial t}\left(\int_{-\infty}^{+\infty} u(x,t)\,dx\right) = -\left(F(+\infty,t) - F(-\infty,t)\right) = 0\,. \tag{2.3}$$

To correctly model solutions with sharp gradients, it is very important to use conservative finite-difference methods which satisfy a discrete analog of (2.3).

## 2.2 Burgers' equation in the characteristics form

Following [37, p. 24], we introduce characteristics

$$x'(t) = u(x(t),t)\,. \tag{2.4}$$

Strictly speaking, equation (2.4) defines characteristics of inviscid Burgers' equation. In the case of viscous Burgers' equation, it can be considered as definition of a particle trajectory. Recall that the material time or substantive derivative $d/dt$ is defined as

$$\frac{d}{dt}u(x(t),t) = \frac{\partial}{\partial t}u(x(t),t) + x'(t)\,\frac{\partial}{\partial x}u(x(t),t)\,. \tag{2.5}$$

Using this definition, we can rewrite equation (2.1) in the following form:

$$\frac{du}{dt} = \varepsilon\frac{\partial^2 u}{\partial x^2}. \tag{2.6}$$

Now, computation of $u = u(x(t),t)$ requires integration of equation (2.4). Therefore, instead of one original equation, we have the system of two equations (2.4) and (2.6). We will call this system by the "characteristics" form of viscous Burgers' equation. Because of the finite value of $\varepsilon$, the "characteristics" defined by (2.4) do not cross [37].

## 2.3 Burgers' equation in the conservative Lagrangian form

Because the "characteristics" do not cross, we can introduce Lagrangian coordinates (see, e.g., [60, p.4]). It is convenient to choose the Lagrangian coordinates, $\xi$, as the initial position:

$$\xi = x(0).$$

In this context, equation (2.4) defines the trajectory of a point which identified initially by the Lagrangian coordinate $\xi$:

$$\frac{d}{dt}x(\xi,t) = u(x(\xi,t),t)\,, \quad x(\xi,0) = \xi.$$

Since the "characteristics" do not cross, the Jacobian $\partial x/\partial\xi$ of the transformation from Eulerian to Lagrangian coordinates is always positive.

In addition to the definition of material time derivative (2.5), we need the following formulas for transformation of spatial derivatives:

$$\frac{\partial u}{\partial \xi} = \frac{\partial u}{\partial x}\frac{\partial x}{\partial \xi} \qquad \text{and} \qquad \frac{\partial u}{\partial x} = \frac{\partial u}{\partial \xi}\Big/\frac{\partial x}{\partial \xi}.$$

This allows us to express $\partial^2 u/\partial x^2$ as follows

$$\frac{\partial^2 u}{\partial x^2} = \left[\frac{\partial}{\partial \xi}\left(\frac{\partial u}{\partial x}\right)\right]\Big/\left(\frac{\partial x}{\partial \xi}\right). \tag{2.7}$$

For future derivations, it is convenient to keep both Eulerian and Lagrangian spatial derivatives. Using (2.7), we can write equation (2.6) as follows

$$\left(\frac{\partial x}{\partial \xi}\right)\frac{du}{dt} = \varepsilon\,\frac{\partial}{\partial \xi}\left(\frac{\partial u}{\partial x}\right).$$

The left hand-side in this equation can be represented as

$$\left(\frac{\partial x}{\partial \xi}\right)\frac{du}{dt} = \frac{d}{dt}\left(\frac{\partial x}{\partial \xi}\,u\right) - \frac{\partial}{\partial \xi}\left(\frac{u^2}{2}\right).$$

Using the last equation, we can write Burgers' equation in the following form:

$$\frac{d}{dt}\left(\frac{\partial x}{\partial \xi}\,u\right) - \frac{\partial}{\partial \xi}\left(\frac{u^2}{2}\right) = \varepsilon\,\frac{\partial}{\partial \xi}\left(\frac{\partial u}{\partial x}\right) \tag{2.8}$$

which we shall refer to as the conservative Lagrangian form of Burgers' equation. For a general 1D conservation law, a similar equation is derived in [57, p.34-3].

## 2.4 Dynamics of Jacobian $\partial x/\partial\xi$

In this section we make a few remarks on dynamics of Jacobian $\partial x/\partial\xi$. As we mentioned before, viscous Burgers' equation has smooth solutions. Let us consider a particular solution given by

$$u(x,t) = \frac{1}{2}\left[1 - \tanh\left(\frac{x - \frac{1}{2}\,t - x_0}{4\,\varepsilon}\right)\right]. \tag{2.9}$$

It is easy to check that $\partial u/\partial x < 0$ and therefore $\partial u/\partial \xi < 0$. Taking derivative $\partial/\partial\xi$ of equation (2.4), we get

$$\frac{d}{dt}\left(\frac{\partial x}{\partial \xi}\right) = \frac{\partial u}{\partial \xi} < 0.$$

Thus, the Jacobian is always decreasing.

Now, if we consider a Lagrangian particle $[\xi\,, \xi+\Delta\xi]$, then its size in the Eulerian coordinates is

$$x(\xi+\Delta\xi, t) - x(\xi, t) \approx \frac{\partial x}{\partial \xi}\,\Delta\xi.$$

According to our previous considerations, it means that the size of the Lagrangian particle is going to zero. Let us give the rigorous proof of this fact. Introducing a new variable

$$y = \frac{1}{4\varepsilon}(x - \frac{1}{2}t - x_0),$$

we rewrite (2.9) as follows:

$$y' = \frac{1}{4\varepsilon}(x' - \frac{1}{2}) = -\frac{1}{8\varepsilon}\tanh y.$$

The solution of the last equation is

$$\sinh y(t) = \sinh y(0)\ \exp(-\frac{1}{8\varepsilon}t).$$

Differentiating the last formula with respect to $\xi$, we get

$$\frac{\partial y}{\partial \xi}(t) = \frac{\partial y}{\partial \xi}(0)\frac{\cosh y(0)}{\cosh y(t)}\exp(-\frac{1}{8\varepsilon}t) = \frac{\partial y}{\partial \xi}(0)\,\frac{\cosh y(0)}{\sqrt{1+\sinh^2 y(0)\exp(-\frac{1}{4\varepsilon}t)}}\exp(-\frac{1}{8\varepsilon}t).$$

Note that the denominator is bounded from below by 1. Therefore,

$$\frac{\partial x}{\partial \xi}(t) = 4\varepsilon\,\frac{\partial y}{\partial \xi}(t)\ \to\ 0 \quad \text{as} \quad t\to\infty. \tag{2.10}$$

The practical consequence of (2.10) will be seen in Section 5 where we present numerical results for the Lagrangian method.

# 3 The moving mesh method

## 3.1 Discretization schemes

We shall use the superscript $n$ to specify the time step for all mesh-related quantities. At time $t = t^n$, the mesh $\mathbf{x}^n$ is defined as the ordered set of points:

$$x_0^n < x_1^n < \ldots < x_{M+1}^n.$$

Let $h^n_{i+1/2} = x^n_{i+1} - x^n_i$ be mesh steps and $x^n_{i+1/2} = (x^n_{i+1} + x^n_i)/2$ be middle points of mesh intervals.

The function $u(x(\xi,t),t)$ is represented by cell-centered values $\bar{u}^n_{i+1/2}$. For simplicity of notations, we shall use $\bar{\mathbf{u}}^n$ for either the vector of the cell-centered values or the piecewise constant function specified by these values. The actual meaning will be clear from context, so no confusion should arise. We define $\bar{u}^n_{i+1/2}$ as an approximation of the exact mean value of $u$ at time $t^n$:

$$\bar{u}^n_{i+1/2} \approx \bar{u}\left([x^n_i, x^n_{i+1}], t^n\right) \stackrel{\text{def}}{=} \frac{1}{h^n_{i+1/2}} \int_{x^n_i}^{x^n_{i+1}} u(x,t^n)\,dx.$$

As the simplest Eulerian method, we use the standard explicit donor-cell method:

$$\bar{u}^{n+1}_{i+1/2} = \bar{u}^n_{i+1/2} - \frac{\Delta t^n}{h_{i+1/2}}(f^n_{i+1} - f^n_i) + \frac{\varepsilon \Delta t^n}{h_{i+1/2}}\left(\left[\frac{\delta \bar{u}^n}{\delta x}\right]_{i+1} - \left[\frac{\delta \bar{u}^n}{\delta x}\right]_i\right). \tag{3.1}$$

Here, $\Delta t^n = t^{n+1} - t^n$ denotes the time step, $f^n_i$ denotes the convective flux at point $x_i$,

$$f^n_i = \begin{cases} (\bar{u}^n_{i-1/2})^2/2 & \text{if} \quad \bar{u}^n_{i+1/2} + \bar{u}^n_{i-1/2} \geq 0, \\ (\bar{u}^n_{i+1/2})^2/2 & \text{otherwise,} \end{cases} \tag{3.2}$$

and $\varepsilon[\delta u^n/\delta x]_i$ denotes the diffusive flux:

$$\varepsilon\left[\frac{\delta \bar{u}^n}{\delta x}\right]_i = \varepsilon \frac{\bar{u}^n_{i+1/2} - \bar{u}^n_{i-1/2}}{(h^n_{i+1/2} + h^n_{i-1/2})/2}$$

For future analysis, it is convenient to write equation (3.1) in the operator form

$$\bar{\mathbf{u}}^{n+1} = \mathcal{L}^n_{Eul}(\bar{\mathbf{u}}^n).$$

As the simplest Lagrangian method, we consider the following discretization of equations (2.4) and (2.8):

$$\begin{aligned} &\frac{h^{n+1}_{i+1/2}\bar{u}^{n+1}_{i+1/2} - h^n_{i+1/2}\bar{u}^n_{i+1/2}}{\Delta t^n} - \frac{1}{2}\left(\left(u^n_{i+1}\right)^2 - (u^n_i)^2\right) = \varepsilon\left(\left[\frac{\delta \bar{u}^n}{\delta x}\right]_{i+1} - \left[\frac{\delta \bar{u}^n}{\delta x}\right]_i\right) \\ &x^{n+1}_i = x^n_i + \Delta t^n\, u^n_i. \end{aligned} \tag{3.3}$$

Here, the nodal value $u^n_i$ is obtained by the linear interpolation of cell-centered values:

$$u^n_i = \frac{h^n_{i-1/2}\,\bar{u}^n_{i+1/2} + h^n_{i+1/2}\,\bar{u}^n_{i-1/2}}{h^n_{i-1/2} + h^n_{i+1/2}}. \tag{3.4}$$

Again, it is convenient to write equation (3.3) in the operator form:

$$\bar{\mathbf{u}}^{n+1} = \mathcal{L}^n_{Lag}(\bar{\mathbf{u}}^n).$$

In both discretization schemes, the time step $\Delta t^n$ is chosen according to the following practical stability condition:

$$\Delta t^n \le \min_i \left( \frac{|\bar{u}^n_{i+1/2}|}{h^n_{i+1/2}} + \frac{2\varepsilon}{(h^n_{i+1/2})^2} \right)^{-1}. \tag{3.5}$$

In [47,51] this condition is used for Eulerian methods. But here, we apply it to both discretization methods.

### 3.2 The EMB rezone strategy

We assume that the initial mesh $\mathbf{x}^0$, at time $t^0 = 0$, is chosen in such a way that all features of function $U(x)$ specifying the initial condition are well resolved. In order to do this, we use the concept of *the best piecewise constant fit* described, for example, in [6].

At time $t^n$, we only know the piecewise constant function $\bar{\mathbf{u}}^n$ on the mesh $\mathbf{x}^n$. We can start new time step using this data (no mesh movement) or generate a different mesh $\tilde{\mathbf{x}}^n$ and compute the corresponding piecewise constant function $\overline{\tilde{\mathbf{u}}}^n$. In this paper we follow the second approach, i.e. we use the time integration schemes (3.1) and (3.3) with $\overline{\tilde{\mathbf{u}}}^n$ instead of $\bar{\mathbf{u}}^n$ and $\tilde{\mathbf{x}}^n$ instead of $\mathbf{x}^n$. In order to compute $\overline{\tilde{\mathbf{u}}}^n$, we use the conservative linearity-and-bound preserving interpolation from [35].

In this paper, we develop a *error-minimization-based* (EMB) rezone method which employs the following look ahead strategy. Let consider the following functional:

$$\Phi_{ex}(\tilde{\mathbf{x}}^n) = \int_{x_0^{n+1}}^{x_{M+1}^{n+1}} \left| u(x,\, t^{n+1}) - \bar{\mathbf{u}}^{n+1} \right|^2 \mathrm{d}x$$

which is nothing else but the square of the $L_2$-norm of error at time $t^{n+1}$. The rezoned mesh $\tilde{\mathbf{x}}^n$ should *minimize* the above functional over a set of *smooth* meshes:

$$\min_{\text{smooth } \tilde{\mathbf{x}}^n} \Phi_{ex}(\tilde{\mathbf{x}}^n) = \min_{\text{smooth } \tilde{\mathbf{x}}^n} \int_{x_0^{n+1}}^{x_{M+1}^{n+1}} \left| u(x,\, t^{n+1}) - \mathcal{L}^n(\overline{\tilde{\mathbf{u}}}^n) \right|^2 \mathrm{d}x \tag{3.6}$$

where $\mathcal{L}^n$ is one of the linear operators $\mathcal{L}^n_{Eul}$ or $\mathcal{L}^n_{Lag}$.

The error at time $t^{n+1}$ is a superposition of three errors: (I) the remapping error in computation of $\overline{\tilde{\mathbf{u}}}^n$, (II) the error due to the time advancing method $\mathcal{L}^n$, and (III) the space discretization error. It is clear that the remapping error is zero when $\tilde{\mathbf{x}}^n = \mathbf{x}^n$; however, this mesh may not be the best one to capture dynamics of solution features. The goal of minimization problem (3.6) is to achieve balance (if possible) between these errors which will minimize the overall error at time $t^{n+1}$.

At first glance, problem (3.6) is very general and can not be recommended for practical applications. We shall prove in the next section, that under a few reasonable assumptions, the leading term in (3.6) *does not* depend on $\mathcal{L}^n$ and can be evaluated using only $\bar{\mathbf{u}}^n$ and $\mathbf{x}^n$.

### 3.3 The conservative remapping

In order to remap data, we use a conservative second-order accurate algorithm based on a piecewise linear reconstruction [35] on mesh $\mathbf{x}^n$:

$$\tilde{u}_R(x) = \bar{u}^n_{i+1/2} + s^n_{i+1/2}\,(x - x^n_{i+1/2}), \qquad x \in [x^n_i,\, x^n_{i+1}],$$

where $s^n_{i+1/2}$ is a limited slope. More precisely, we use the minmod limiter [36]. For a given mesh $\tilde{\mathbf{x}}^n$, the remapped function is computed by integrating $\tilde{u}_R(x)$:

$$\overline{\tilde{u}}^n_{i+1/2} = \frac{1}{\tilde{h}^n_{i+1/2}} \int_{\tilde{x}_i}^{\tilde{x}_{i+1}} \tilde{u}_R(x)\ \mathrm{d}x. \tag{3.7}$$

This remapping algorithm is exact for linear functions and conservative, i.e.

$$\sum_{i=0}^{M} \overline{\tilde{u}}^n_{i+1/2}\,\tilde{h}^n_{i+1/2} = \sum_{i=0}^{M} \bar{u}^n_{i+1/2}\,h^n_{i+1/2}\,.$$

Note that higher order reconstruction methods can be used for smooth solutions. The analysis presented in the next section does not rely on a particular method but rather assumes that the remapping is at least second-order accurate. In two dimensions, the second-order accurate remapping method has been proposed in [45].

## 4 Error analysis of the EMB method

Let us introduce a small parameter $h = (x^0_{M+1} - x^0_0)/(M+1)$. All accuracy estimates will be formulated with respect to this small parameter. We consider the asymptotic case, $h \to 0$, when we may ignore dependence on $\varepsilon$.

In order to simplify analysis, we first assume that the data associated with mesh $\mathbf{x}^n$ are exact. In reality, the mean values $\bar{u}^n_{i+1/2}$ accumulate errors from previous time integration steps. A rigorous analysis of this assumption will be the topic of future research.

We assume that the second derivatives of $u(x,\,t)$ are bounded. We also assume that there exist constants $c_h$, $C_h$ and $C_{\Delta h}$ which are independent of $h$ and satisfy the following conditions:

$$c_h\,h \le \tilde{h}^n_{i+1/2} \le C_h\,h \qquad \text{and} \qquad |\tilde{h}^n_{i+1/2} - \tilde{h}^n_{i-1/2}| \le C_{\Delta h}\,h^2. \tag{4.1}$$

These inequalities mean that mesh steps are of order $h$ and changes in mesh steps are of order $h^2$. The problem of generating meshes satisfying (4.1) will be addressed in Section 4.3.

Our final assumption is that the interpolation operator from mesh $\mathbf{x}^n$ to mesh $\tilde{\mathbf{x}}^n$ is second-order accurate, i.e.

$$\overline{\tilde{u}}^n_{i+1/2} = \overline{u}([\tilde{x}^n_i, \tilde{x}^n_{i+1}], t^n) + \mathcal{O}(h^2). \tag{4.2}$$

In this paper, we use the second-order accurate conservative remapping described in [35, 45].

### 4.1 The Eulerian scheme

Let us analyze the function

$$e_{i+1/2}(x,t^{n+1}) = u(x,t^{n+1}) - \overline{u}^{n+1}_{i+1/2}\,, \qquad x \in [x_i^{n+1}, x_{i+1}^{n+1}]\,, \tag{4.3}$$

representing the error between the exact solution at time $t^{n+1}$ and the numerical solution at $t^{n+1}$ determined by scheme (3.1). We begin with finding useful estimates for $u(x,t^{n+1})$. For our analysis it will be enough to use the Taylor expansion of the first order at space-time point $(x,\, t^n)$:

$$u(x,t^{n+1}) = u(x,t^n) + \Delta t^n \frac{\partial u}{\partial t}\,(x,t^n) + \mathcal{O}\left((\Delta t^n)^2\right).$$

Because all continuous functions are evaluated at the same space-time point $(x,\, t^n)$, this argument will be omitted in the sequel.

Note that the time derivative can be expressed via space derivatives using the Burgers equation. Therefore,

$$u(x,t^{n+1}) = u - \Delta t^n\, u\frac{\partial u}{\partial x} + \varepsilon\,\Delta t^n\,\frac{\partial^2 u}{\partial x^2} + \mathcal{O}\left((\Delta t^n)^2\right)\,.$$

The stability conditions (3.5) implies that $\varepsilon\,\Delta t^n < (\tilde{h}^n_{i+1/2})^2/2$. Thus, the $\varepsilon$-term is of order $h^2$ and

$$u(x,t^{n+1}) = u - \Delta t^n\, u\frac{\partial u}{\partial x} + \mathcal{O}\left(h^2 + (\Delta t^n)^2\right)\,. \tag{4.4}$$

On the rezoned mesh $\tilde{\mathbf{x}}^n$, the expression for $\overline{u}^{n+1}_{i+1/2}$ is given by (3.1). Let us show that the $\varepsilon$-term in this equation is also $\mathcal{O}(h^2)$. The fact that the exact mean value $\overline{u}([\tilde{x}^n_i, \tilde{x}^n_{i+1}], t^n)$ differs from the function value at middle point $\tilde{x}^n_{i+1/2}$ by $\mathcal{O}(h^2)$ and formula (4.2) allow us to prove that

$$\left[\frac{\delta\overline{\tilde{u}}^n}{\delta x}\right]_i = \frac{\partial u}{\partial x}(\tilde{x}_i, t^n)\, + \mathcal{O}(h)\,.$$

This equation and the previously used inequality $\varepsilon\Delta t^n < (\tilde{h}^n_{i+1/2})^2/2$ imply that

$$\frac{\varepsilon\Delta t^n}{\tilde{h}^n_{i+1/2}}\left(\left[\frac{\delta\overline{\tilde{u}}^n}{\delta x}\right]_{i+1} - \left[\frac{\delta\overline{\tilde{u}}^n}{\delta x}\right]_i\right) = \mathcal{O}(h^2). \tag{4.5}$$

Now, let us analyze the second term in (3.1) containing convective fluxes $\tilde{f}^n_{i+1}$ and $\tilde{f}^n_i$. Without loss of generality, we may assume that $\tilde{f}^n_i = (\overline{\tilde{u}}^n_{i-1/2})^2/2$. Then,

$$\frac{\Delta t^n}{\tilde{h}^n_{i+1/2}}(\tilde{f}^n_{i+1} - \tilde{f}^n_i) = \Delta t^n\,\frac{\overline{\tilde{u}}^n_{i+1/2} + \overline{\tilde{u}}^n_{i-1/2}}{2}\,\frac{\overline{\tilde{u}}^n_{i+1/2} - \overline{\tilde{u}}^n_{i-1/2}}{\tilde{h}^n_{i+1/2}}.$$

Using our assumptions about mesh smoothness and accuracy of the remapping, we can prove that

$$\frac{\Delta t^n}{\tilde{h}^n_{i+1/2}}(\tilde{f}^n_{i+1} - \tilde{f}^n_i) = \Delta t^n\, u\, \frac{\partial u}{\partial x} + \mathcal{O}(h\,\Delta t^n) \tag{4.6}$$

where all continuous functions are still evaluated at the same space-time point $(x,\, t^n)$.

Finally, definition (4.3) and formulas (4.4), (4.5) and (4.6) imply that the local error can be estimated as follows:

$$\begin{aligned} e_{i+1/2}(x, t^{n+1}) &= u(x, t^{n+1}) - \overline{u}^{n+1}_{i+1/2} \\ &= u(x, t^n) - \bar{u}([\tilde{x}^n_i,\, \tilde{x}^n_{i+1}],\, t^n) + \mathcal{O}((h + \Delta t^n)^2). \end{aligned} \tag{4.7}$$

Estimate (4.7) shows that the leading term in the error *does not* depend on $\mathcal{L}^n_{Eul}$. Summing the local errors, we get

$$\Phi_{ex}(\tilde{\mathbf{x}}^n) = \Phi_{ap}(\tilde{\mathbf{x}}^n) + \mathcal{O}((h + \Delta t^n)^3)$$

where

$$\Phi_{ap}(\tilde{\mathbf{x}}^n) = \sum_{i=0}^{M} \int_{\tilde{x}^n_i}^{\tilde{x}^n_{i+1}} \left(u(x, t^n) - \bar{u}([\tilde{x}^n_i,\, \tilde{x}^n_{i+1}],\, t^n)\right)^2 \mathrm{d}x. \tag{4.8}$$

To summarize, we have shown that under reasonable assumptions on mesh smoothness, solution regularity, accuracy of remapping and time stepping $\Delta t^n$, the leading term in the error at time $t^{n+1}$ depends on accuracy of representation of function $u(x, t^n)$ by its exact mean values. This is the standard approximation problem of the best piecewise constant fit with adjustable nodes [6].

The functional $\Phi_{ap}$ is suitable for the theoretical analysis but can not be used in numerical methods. Using once again our assumptions and some of the previous arguments, we may easily show that

$$\Phi_{ap}(\tilde{\mathbf{x}}^n) = \Phi_{num}(\tilde{\mathbf{x}}^n) + \mathcal{O}(h^3), \qquad \Phi_{num}(\tilde{\mathbf{x}}^n) = \frac{1}{12}\sum_{i=0}^{M} \left[\frac{\delta\overline{\tilde{u}}^n}{\delta x}\right]^2_{i+1/2} \tilde{h}^3_{i+1/2}, \tag{4.9}$$

where $[\delta\overline{\tilde{u}}^n/\delta x]_{i+1/2}$ is a first-order (at least) approximation of $\partial u/\partial x$ at point $\tilde{x}^n_{i+1/2}$. The functional $\Phi_{num}$ is suitable (can be easily computed) for numerical methods. First, we approximate $\partial u/\partial x$ on mesh $\mathbf{x}^n$ by a piecewise constant function with the cell-centered values given by

$$\left[\frac{\delta\bar{u}^n}{\delta x}\right]_{i+1/2} = \alpha_{i+1/2}\left[\frac{\delta\bar{u}^n}{\delta x}\right]_{i+1} + (1 - \alpha_{i+1/2})\left[\frac{\delta\bar{u}^n}{\delta x}\right]_i, \tag{4.10}$$

where

$$\alpha_{i+1/2} = \frac{h^n_{i-1/2} + h^n_{i+1/2}/2}{h^n_{i-1/2} + h^n_{i+1/2} + h^n_{i+3/2}}.$$

Second, we interpolate this discrete function onto the rezone mesh $\tilde{\mathbf{x}}^n$ with the algorithm described in Section 3.3 and use the result to compute functional $\Phi_{num}$. Since formula (4.10) is second-order accurate, the estimate for $\partial u/\partial x$ on the rezoned mesh is second-order accurate too.

### 4.2 The Lagrangian scheme

Using of the Lagrangian coordinate $\xi$, we rewrite (4.3) in the equivalent form:

$$e_{i+1/2}(x(\xi,t^{n+1}),t^{n+1}) = u(x(\xi,t^{n+1}),t^{n+1}) - \overline{u}^{n+1}_{i+1/2}, \qquad \xi \in [\xi_i,\xi_{i+1}].$$

We begin with finding useful estimates for $u(x,t^{n+1}) = u(x(\xi,t^{n+1}),t^{n+1})$ when $x \in [x_i^{n+1}, x_{i+1}^{n+1}]$, and correspondingly $\xi \in [\xi_i,\xi_{i+1}]$. For our analysis, it will be enough to use the Taylor expansion of the first-order at point $(\xi,\, t^n)$:

$$u(x(\xi,t^{n+1}),t^{n+1}) = u(x(\xi,t^n),t^n) + \Delta t^n \,\frac{du}{dt}\,(x(\xi,t^n),t^n) + \mathcal{O}\left((\Delta t^n)^2\right). \tag{4.11}$$

Because all continuous functions are computed at the same point $(\xi,\, t^n)$, these arguments will be omitted in the sequel. Using (2.6), equation (4.11) can be rewritten as follows:

$$u(x(\xi,t^{n+1}),t^{n+1}) = u + \varepsilon\,\Delta t^n \,\frac{\partial^2 u}{\partial x^2} + \mathcal{O}\left((\Delta t^n)^2\right). \tag{4.12}$$

Now, stability conditions (3.5) implies that the $\varepsilon$-term in (4.12) is of order $h^2$ and therefore

$$u(x(\xi,t^{n+1}),t^{n+1}) = u + \mathcal{O}\left(h^2 + (\Delta t^n)^2\right). \tag{4.13}$$

Now, we derive useful estimates for $\bar{u}^{n+1}_{i+1/2}$. The arguments used in the previous section, allow us to drop the $\varepsilon$-term in the expression for $\overline{u}^{n+1}_{i+1/2}$ resulting in

$$\bar{u}^{n+1}_{i+1/2} = \frac{\tilde{h}^n_{i+1/2}}{h^{n+1}_{i+1/2}}\,\overline{\tilde{u}}^n_{i+1/2} + \frac{\Delta t^n}{h^{n+1}_{i+1/2}}\,\frac{(\tilde{u}^n_{i+1})^2 - (\tilde{u}^n_i)^2}{2} + \mathcal{O}(h^2). \tag{4.14}$$

Note that the equation of mesh motion in (3.3) implies that

$$h^{n+1}_{i+1/2} = \tilde{h}^n_{i+1/2} + \Delta t^n\,(\tilde{u}^n_{i+1} - \tilde{u}^n_i)$$

and therefore

$$\frac{\tilde{h}^n_{i+1/2}}{h^{n+1}_{i+1/2}} = 1 - \Delta t^n\,\frac{\partial u}{\partial x}(\tilde{x}^n_{i+1/2},t^n) + \mathcal{O}(h\,\Delta t^n) = 1 + \mathcal{O}(\Delta t^n). \tag{4.15}$$

Using (4.15), the first term in the right hand side of (4.14) can be estimated as follows:

$$\frac{\tilde{h}^n_{i+1/2}}{h^{n+1}_{i+1/2}}\,\overline{\tilde{u}}^n_{i+1/2} = \overline{\tilde{u}}^n_{i+1/2} - \Delta t^n\,u(\tilde{x}^n_{i+1/2},t^n)\,\frac{\partial u}{\partial x}(\tilde{x}^n_{i+1/2},t^n) + \mathcal{O}(h\,\Delta t^n). \tag{4.16}$$

Let us analyze now the second term in the right hand side of (4.14). Using our assumptions about mesh smoothness and accuracy of remapping, one can prove that

$$\frac{\Delta t^n}{h^{n+1}_{i+1/2}}\,\frac{(\tilde{u}^n_{i+1})^2 - (\tilde{u}^n_i)^2}{2} = \Delta t^n\,u(\tilde{x}^n_{i+1/2},t^n)\,\frac{\partial u}{\partial x}(\tilde{x}^n_{i+1/2},t^n) + \mathcal{O}(h\,\Delta t^n). \tag{4.17}$$

From (4.16) and (4.17) we can conclude that

$$\bar{u}^{n+1}_{i+1/2} = \overline{\tilde{u}}^{n}_{i+1/2} + \mathcal{O}(h^2 + h\,\Delta t^n)\,. \tag{4.18}$$

Finally, definition (4.3) and equations (4.13), (4.18) imply that the local error equals to

$$\begin{aligned} e_{i+1/2}(x(\xi,t^{n+1}),t^{n+1}) &= u(x(\xi,t^{n+1}),t^{n+1}) - \overline{u}^{n+1}_{i+1/2} \\ &= u(x(\xi,t^{n}),t^{n}) - \bar{u}[\tilde{x}^n_i,\,\tilde{x}^n_{i+1}] + \mathcal{O}((h+\Delta t^n)^2). \end{aligned}$$

The above estimate is similar to estimate (4.7) derived for the Eulerian scheme. Thus, the same functional $\Phi_{num}$, as in the Eulerian method, has to be used to move the mesh in the Lagrangian method.

## 4.3 Achieving mesh regularity

The direct minimization of functional $\Phi_{num}$ produces meshes that are optimal from the error minimization viewpoint. However, these meshes may not be appropriate for solving time dependent problems due to strong disproportionality of neighboring mesh cells, which may affect stability of the overall method. Moreover, the error analysis performed in previous sections assumes that the mesh is smooth which is also consistent with the smoothness constraint in the original minimization problem (3.6). Therefore, we need a algorithm which allows us to produce meshes which are simultaneously smooth and adapted to a solution. It is also important to enforce mesh smoothness in regions where the solution is constant, i.e. the local error is zero for any mesh.

### 4.3.1 The guaranteed spatial smoothing

In this paper, we use the idea of the guaranteed smoothing proposed in [25]. Let $\alpha$ be a positive number. We define a smooth mesh as the mesh whose mesh steps satisfy the following condition:

$$\frac{\alpha}{\alpha+1} \le \frac{\tilde{h}^n_{i-1/2}}{\tilde{h}^n_{i+1/2}} \le \frac{\alpha+1}{\alpha}, \qquad i = 1,\ldots,M. \tag{4.19}$$

One way to build a smooth adapted mesh is to apply algorithms from [25] to the minimizer of functional $\Phi_{num}$. Another approach is to modify this functional such that its minimizer will be a smooth mesh. Our numerical experiments show that the second approach requires less computational resources than the first one. This is due to better properties of the modified functional and therefore faster convergence of optimization algorithms.

We begin with deriving necessary conditions for extrema of functional $\Phi_{ap}$ which is more suitable for analysis. Then, the obtained results will be applied to functional $\Phi_{num}$. The straightforward differentiation with respect to $\tilde{x}^n_i$ gives

$$\frac{\partial \Phi_{ap}}{\partial \tilde{x}^n_i} = 2\,u(\tilde{x}^n_i,\,t^n) - \bar{u}([\tilde{x}^n_{i-1},\,\tilde{x}^n_i],\,t^n) - \bar{u}([\tilde{x}^n_i,\,\tilde{x}^n_{i+1}],\,t^n) = 0. \tag{4.20}$$

Using the Taylor expansions at point $\tilde{x}_i^n$ and neglecting terms of the third-order and higher, we get

$$\left.\frac{\partial u}{\partial x}\right|_{\tilde{x}_i^n}\left(\tilde{h}_{i+1/2}^n-\tilde{h}_{i-1/2}^n\right)+\frac{1}{3}\left.\frac{\partial^2 u}{\partial x^2}\right|_{\tilde{x}_i^n}\left((\tilde{h}_{i+1/2}^n)^2+(\tilde{h}_{i-1/2}^n)^2\right)=0. \tag{4.21}$$

Now, we assume that the mesh satisfying this equation is described by a smooth function $\tilde{h}(x)$ such that $\tilde{h}_{i+1/2}^n=\tilde{h}(x_{i+1/2}^n)$. Then

$$\frac{\tilde{h}_{i+1/2}^n-\tilde{h}_{i-1/2}^n}{(\tilde{h}_{i+1/2}^n+\tilde{h}_{i-1/2}^n)/2}\approx\left.\frac{\partial\tilde{h}}{\partial x}\right|_{\tilde{x}_i}\qquad\text{and}\qquad(\tilde{h}_{i+1/2}^n)^2+(\tilde{h}_{i-1/2}^n)^2\approx 2\left(\tilde{h}(\tilde{x}_i^n)\right)^2.$$

Substituting these estimates into (4.21), we get the following equation at point $\tilde{x}_i^n$:

$$\frac{2}{3}\frac{\partial}{\partial x}\left(\ln\frac{\partial u}{\partial x}\right)=-\frac{\partial}{\partial x}\left(\ln\tilde{h}(x)\right).$$

This equation means that the mesh $\tilde{\mathbf{x}}^n$, minimizing functional $\Phi_{ap}$ satisfies approximately the following condition:

$$\left(\left.\frac{\partial u}{\partial x}\right|_{\tilde{x}_{i+1/2}^n}\right)^{2/3}\tilde{h}_{i+1/2}^n=\text{constant}. \tag{4.22}$$

In other words, in order to generate a mesh satisfying (4.19), we apply algorithms of guaranteed smoothing to coefficients $[\delta\overline{\tilde{u}}^n/\delta x]_{i+1/2}^{2/3}$ of functional $\Phi_{num}$ which we denote by $\omega_{i+1/2}^n$ to simplify notation. Let $\tilde{\boldsymbol{\omega}}^n$ be a smooth piecewise constant function preserving main features of function $\boldsymbol{\omega}^n$. Then, the modified functional reads:

$$\Phi_{sm}(\tilde{\mathbf{x}}^n)=\sum_{i=0}^{M}\left(\tilde{\omega}_{i+1/2}^n\,\tilde{h}_{i+1/2}^n\right)^3=\sum_{i=0}^{M}\left(\tilde{\omega}_{i+1/2}^n\,(\tilde{x}_{i+1}^n-\tilde{x}_i^n)\right)^3. \tag{4.23}$$

**Proposition 4.1.** Let $\omega_{i+1/2}^n$, $i=0,\dots,M$, be given positive numbers. The numbers $\tilde{\omega}_{i+1/2}^n$, $i=0,\dots,M$, satisfying

$$\frac{\alpha}{\alpha+1}\le\frac{\tilde{\omega}_{i+1/2}^n}{\tilde{\omega}_{i-1/2}^n}\le\frac{\alpha+1}{\alpha},\qquad i=1,\dots,M,$$

can be computed by solving the system of $M+1$ linear equations:

$$\tilde{\omega}_{i+1/2}^n-\alpha(\alpha+1)(\tilde{\omega}_{i+3/2}^n-2\tilde{\omega}_{i+1/2}^n+\tilde{\omega}_{i-1/2}^n)=\omega_{i+1/2}^n \tag{4.24}$$

where $\tilde{\omega}_{-1/2}^n=\tilde{\omega}_{1/2}^n$ and $\tilde{\omega}_{M+3/2}^n=\tilde{\omega}_{M+1/2}^n$.

The proof of Proposition 4.1 can be found in [41, 63]. Note that equation (4.24) can be considered as the discretization of the following continuous problem:

$$\tilde{\omega} - (\Delta\xi)^2 \alpha (\alpha + 1) \frac{\partial^2 \tilde{\omega}}{\partial \xi^2} = \omega.$$

Since function $\tilde{\omega}(\xi)$ is smoother than $\omega(\xi)$, its second derivative is bounded and the second term in the left-hand side is $\mathcal{O}\left((\Delta\xi)^2\right)$. Therefore, the whole operator acting on $\tilde{\omega}$ is close to the identity operator. This gives us a simple explanation of why the solution of (4.24) is smoother than the right hand side but still preserves its main features.

Let us show that the mesh minimizing functional $\Phi_{sm}$ does not have very small space intervals. Let us consider the extreme case when the mesh is geometrically refined to point $x^n_{M+1}$:

$$\tilde{h}^n_{i+1/2} = \tilde{h}^n_{1/2} \left(\frac{\alpha}{\alpha+1}\right)^i, \qquad i = 0, \ldots, M.$$

Thus, the minimal mesh step is

$$\tilde{h}^n_{M+1/2} = (x^n_{M+1} - x^n_0) \frac{\alpha^M}{(1+\alpha)^{M+1} - \alpha^{M+1}}$$

which guarantees that the time step $\Delta t^n$ is bounded from below. It is pertinent to note that the error analysis assumes that the second derivatives of $u(x,\, t)$ are bounded, i.e. only a coarse mesh can be geometrically refined to point $x^n_{M+1}$.

### 4.3.2 The effect of mesh smoothing

Let us consider a function $U(x) = u(x, 0)$ where $u(x, t)$ is the test function from [48] satisfying viscous Burgers' equation:

$$\begin{aligned} &u(x,t) = 1 - \frac{9\, r_1 + 5\, r_2}{10\, (r_1 + r_2 + r_3)}, \qquad t \in (0,\, T), \\ &r_1 = \exp\left(\frac{1/2 - x}{20\, \varepsilon} - \frac{99\, t}{400\, \varepsilon}\right), \\ &r_2 = \exp\left(\frac{1/2 - x}{4\, \varepsilon} - \frac{3\, t}{16\, \varepsilon}\right), \quad r_3 = \exp\left(\frac{3/8 - x}{2\, \varepsilon}\right). \end{aligned} \tag{4.25}$$

We study the effect of smoothing for function $U(x)$. The results of numerical experiments are shown in Table 1. The minimizers of functionals $\Phi_{ap}$, $\Phi_{num}$ and $\Phi_{sm}$ are denoted by $\mathbf{x}^{ap}$, $\mathbf{x}^{num}$ and $\mathbf{x}^{sm}$, respectively. The uniform mesh is denoted by $\mathbf{x}^{uni}$. The loss in accuracy due to smoothing is only 7.4% on the mesh with 64 mesh intervals. Also note that the optimal distribution of mesh points gives us approximately 3 times smaller errors.

In Fig. 1, we compare minimizers of functionals $\Phi_{num}$ and $\Phi_{sm}$ for $M = 32$. As expected, the ratio of neighboring mesh steps satisfy inequality (4.19) with $\alpha = 1$. Observe that coefficients of the smoothed functional preserve the main features of the original functional.

Table 1: The effect of smoothing for $\varepsilon = 0.005$, $M = 32$ and $\alpha = 1$.

| $M$ | $\sqrt{\Phi_{ap}(\mathbf{x}^{uni})}$ | $\sqrt{\Phi_{ap}(\mathbf{x}^{ap})}$ | $\sqrt{\Phi_{ap}(\mathbf{x}^{num})}$ | $\sqrt{\Phi_{ap}(\mathbf{x}^{sm})}$ |
|---|---|---|---|---|
| 16 | 2.99e-2 | 1.01e-2 | 1.19e-2 | 1.75e-2 |
| 32 | 1.59e-2 | 4.99e-3 | 5.18e-3 | 6.28e-3 |
| 64 | 7.99e-3 | 2.48e-3 | 2.50e-3 | 2.70e-3 |
| 128 | 4.00e-3 | 1.24e-3 | 1.24e-3 | 1.28e-3 |

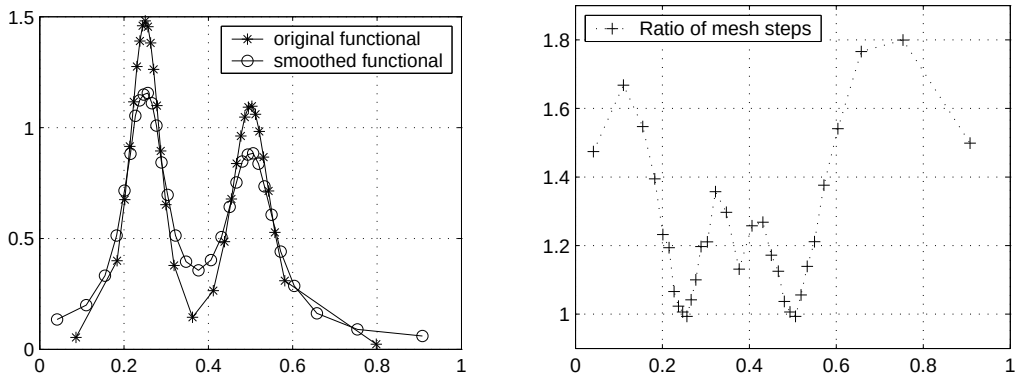


Figure 1: Left pictures shows coefficients $\omega_{i+1/2}$ and $\tilde{\omega}_{i+1/2}$ of functionals $\Phi_{num}$ and $\Phi_{sm}$, respectively. The right picture shows the ratio of neighboring mesh steps in the smooth mesh.

# 5 Numerical experiments

## 5.1 Implementation issues

In this section, we consider some numerical issues related to implementation of one step of the moving mesh method. For this reason, the time superscript 'n' is omitted here.

We use the non-linear conjugate gradient method for minimizing functional $\Phi_{sm}$:

$$\begin{aligned} x_i^{(k+1)} &= x_i^{(k)} + \lambda^k d_i^{(k)}, \\ d_i^{(k)} &= -\frac{\delta\Phi_{sm}}{\delta\tilde{x}_i}(\mathbf{x}^{(k)}) + \beta^k d_i^{(k-1)}, \quad k = 1, 2, \ldots, \end{aligned} \tag{5.1}$$

where $\mathbf{x}^{(0)}$ is the given mesh, $x_i^{(0)} = x_i$, $\mathbf{d}^{(k)}$ is the descent direction, and $\delta\Phi_{sm}/\delta\tilde{x}_i$ is a finite difference approximation of the continuous derivative $\partial\Phi_{sm}/\partial\tilde{x}_i$. In numerical experiments we use Polak-Ribière's choice for parameter $\beta^k$ [49]. The parameter $\lambda^k$ is specified by a line search algorithm:

$$\lambda^k = \arg\min_{\lambda} \Phi_{sm}(\mathbf{x}^{(k)} + \lambda\,\mathbf{d}^{(k)}). \tag{5.2}$$

The quality of the line search algorithm is crucial to preserve the mutual conjugacy property

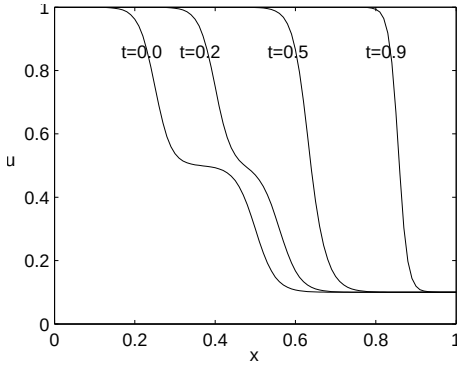


Figure 2: Viscous Burgers' equation with $\varepsilon = 0.005$. Exact solution for different $t$.

of the descent directions. The details of efficient numerical methods for solving (5.2) can be found in [49]. Some modifications of these methods are required to enforce the mesh validity constraints, $\tilde{x}_{i+1}^{(k)} > \tilde{x}_i^{(k)}$.

We terminate the non-linear iterations (5.1) when either $\mathbf{d}^{(k)}$ is no longer a descent direction or the relative change in the position of mesh nodes is less than the user prescribed tolerance $TOL$, i.e.

$$\max_{1\le i\le M} \frac{|x_i^{(k)} - x_i^{(k+1)}|}{x_{i+1}^{(k)} - x_{i-1}^{(k)}} \le TOL \qquad \text{for some } k.$$

Note that the smoothing is the most time-consuming step in evaluation of $\Phi_{sm}$. Therefore, we perform it only once per each conjugate gradient iteration. The smoothed values then remapped on all intermediate meshes using the method from Section 3.3.

## 5.2 Adaptation for Burgers' equation in the Eulerian form

In the first set of experiments we consider viscous Burgers' equation with the exact solution given by (4.25). Let the final time be $T = 0.9$ and $\varepsilon = 0.005$. At time moment $t = 0$, this function consists of two shock-like structures, which then move with different speeds and finally merge at $t \approx 0.5$ (see Fig. 2).

We set $k_{max} = 50$ (the maximal number of CG iterations) and $TOL = 10^{-3}$ for all simulations. The time step was chosen to be twice less than that recommended by the stability condition (3.5).

The left panel in Fig. 3 shows the $L_2$-norm of error with respect to the number $M$ of mesh steps. We observe the linear convergence rate on both adaptive and uniform meshes. However, for the given accuracy, e.g. $2 \cdot 10^{-3}$, we need about 10 times less mesh points in the adaptive mesh than in the uniform one. This difference grows when $\varepsilon$ tends to zero, i.e., when the solution profile becomes more sharp.

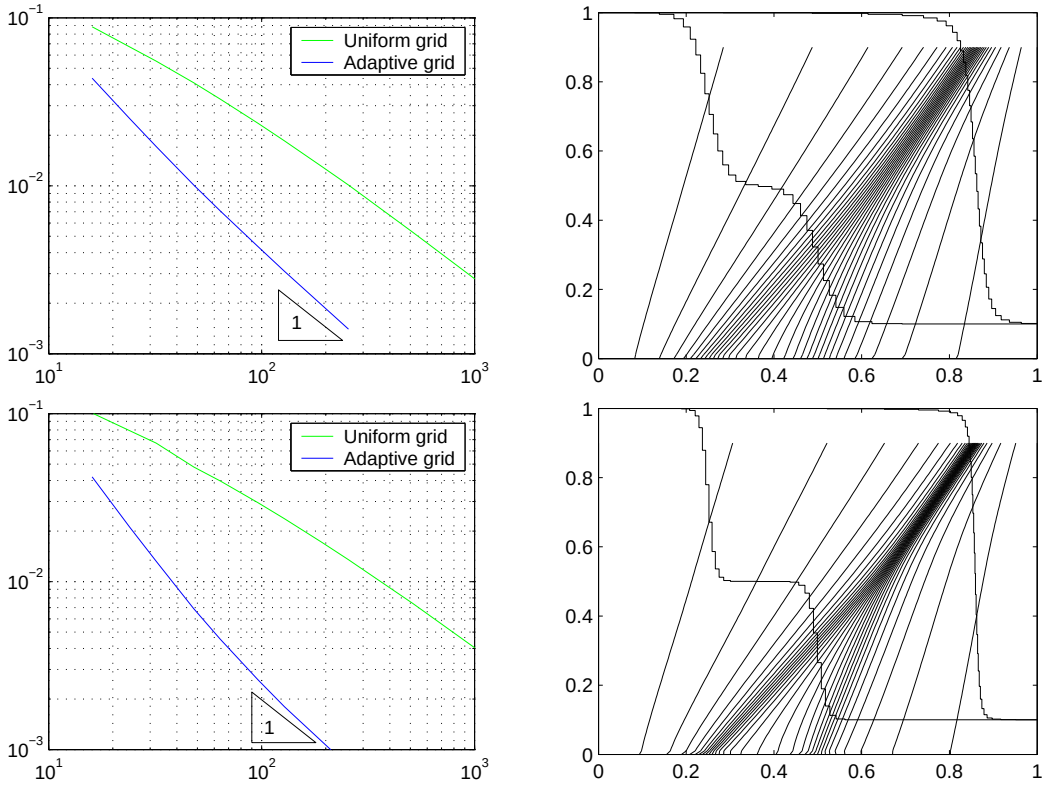


Figure 3: Viscous Burgers' equation with $\varepsilon = 0.005$ (top row) and $\varepsilon = 0.002$ (bottom row). The left panel shows convergence rates. The right panel shows trajectories of mesh points for $M = 32$, $\alpha = 1$ and discrete solutions at $t = 0$ and $t = T$.

In the right panel in Fig. 3, we present trajectories of mesh points for $M = 32$. Horizontal slices represent the mesh at the corresponding time moments. The trajectories follow the solution features. The stronger gradient of the solution, the more mesh points are required to resolve it. Nevertheless, the smoothing procedure guarantees that the adapted mesh will always satisfy condition (4.19).

It is typical for one-dimensional hyperbolic problems that the arithmetical complexity for a fixed accuracy is less for uniform meshes than for adaptive ones. In two and three dimensions it is highly possible that the solution of a hyperbolic problem is anisotropic and the directions of the anisotropy are not aligned with axes of the Cartesian coordinate system. In this case, we expect the computational efficiency of adaptive meshes.

In the second set of experiments we consider inviscid Burgers's equation ($\varepsilon = 0$) whose solution may develop shock-like structures. Note that our theory is not valid for such solutions; however, the presented results are very promising. Let $T = 0.5$ and the initial condition be the

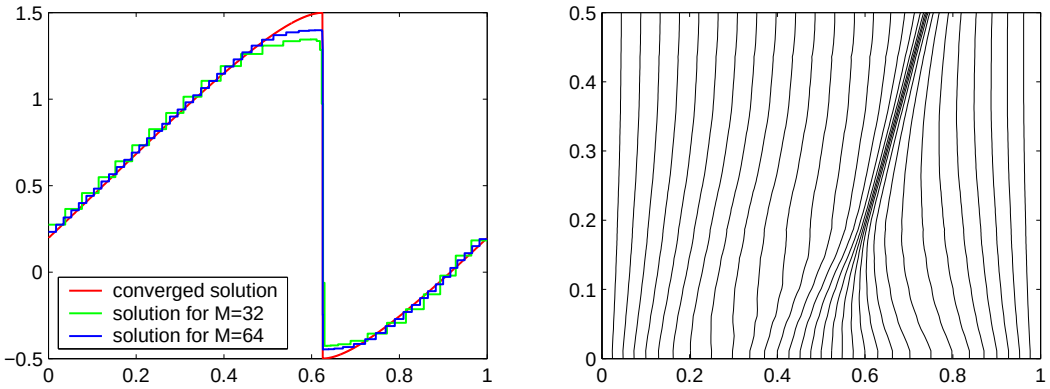


Figure 4: Inviscid Burgers' equation: the discrete and exact solutions at $t = 0.25$ (left picture), trajectories of mesh points for the case $M = 32$ and $\alpha = 1$ (right picture).

periodic function

$$U(x) = 0.5 + \sin(2\pi x), \qquad x \in (0, 1).$$

This example was also studied in [56]. A shock wave is developed with time and reaches its maximal strength at $T_0 = 0.25$. The computed solution is compared with the exact one in Fig. 4. Note that the donor scheme (3.1) smears the discrete solution before and after the shock. A better resolution of the shock may be obtained with higher order time integration schemes. The trajectories of mesh points shown in Fig. 4 confirm the theoretically known result that the shock has a constant velocity 0.5.

## 5.3 Adaptation for Burgers' equation in the Lagrangian form

In the context of Lagrangian methods, it is more standard to refer to moving mesh methods as the arbitrary Lagrangian-Eulerian (ALE) methods. In most ALE simulations, the main objective of a rezone method is to maintain high quality of the mesh while keeping it as close as possible to the Lagrangian mesh. One example of such a method is the reference Jacobian matrix (RJM) method. In this section, we compare the RJM and EMB rezone methods.

### 5.3.1 Reference Jacobian matrix method

The essential idea of the RJM method [33] is the recognition that the Lagrangian solution contains sufficient information about the flow to constrain our measure of mesh smoothness. More specifically, the Lagrangian mesh reflects both the physical motion of the fluid and non-physical distortion. It is assumed that the non-physical distortion of a computational mesh has a much shorter wavelength, and so it can be separated from the physical motion by averaging over a small neighborhood of a cell. This assumption naturally leads to a desire to have a rezoned

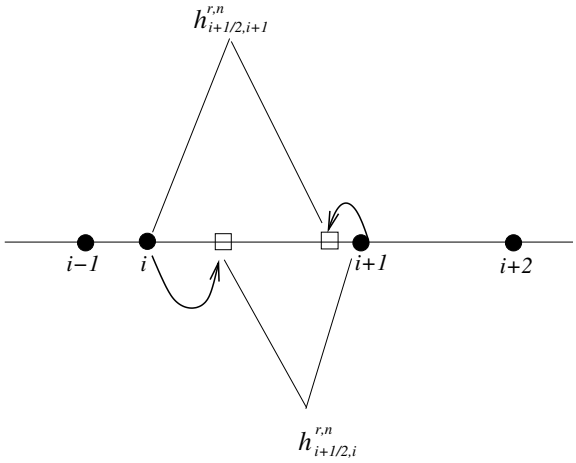


Figure 5: Construction of reference Jacobians. Solid circles are positions of nodes after the Lagrangian step. The reference positions of nodes $i$ and $i+1$ marked by squares. Their virtual movement is shown by arrows.

mesh which is close to the Lagrangian mesh but is "smoother" (i.e., it has better geometrical quality).

The most fundamental object describing a map is its Jacobian matrix. One can expect that if two maps have similar Jacobians, then the maps themselves must be similar and must produce meshes which are close to each other. This leads to the following strategy. We first construct reference Jacobians that are based on the geometry of the nearest neighbors of a Lagrangian cell and effectively smooth the short-wavelength mesh deformation. Then, we construct a global functional that measures the difference between the reference Jacobians and the Jacobians of the rezoned mesh ($\tilde{h}^n_{i+1/2}$ in 1D) over all mesh cells. By minimizing this functional, we get the rezoned mesh.

Following [33], we consider the patch containing cells $i+1/2$ and $i-1/2$. We fix nodes $i-1$, $i+1$ in their Lagrangian positions and find locally optimal virtual position for node $i$ (optimal from viewpoint of mesh smoothness). In 1D, it is the middle point between nodes $i-1$ and $i+1$ which makes virtual cells $i+1/2$ and $i-1/2$ equal (see the square shaded with diagonal lines in Fig. 5). It gives one reference Jacobian, $h^{r,n}_{i+1/2,i}$ for cell $i+1/2$,

$$h^{r,n}_{i+1/2,i} = (h^n_{i+1/2} + h^n_{i-1/2})/2\,. \tag{5.3}$$

The other reference Jacobian for cell $i+1/2$ is obtained by a similar virtual movement of node $i+1$ inside the patch consisting of cells $i+1/2$ and $i+3/2$:

$$h^{r,n}_{i+1/2,i+1} = (h^n_{i+1/2} + h^n_{i+3/2})/2\,. \tag{5.4}$$

Finally, the global functional is

$$F(\tilde{\mathbf{x}}^n) = \sum_{i=1}^{M} \frac{\left[(\tilde{x}^n_{i+1} - \tilde{x}^n_i) - h^{r,n}_{i+1/2,i}\right]^2}{(\tilde{x}^n_{i+1} - \tilde{x}^n_i)\, h^{r,n}_{i+1/2,i}} + \sum_{i=0}^{M-1} \frac{\left[(\tilde{x}^n_{i+1} - \tilde{x}^n_i) - h^{r,n}_{i+1/2,i+1}\right]^2}{(\tilde{x}^n_{i+1} - \tilde{x}^n_i)\, h^{r,n}_{i+1/2,i+1}}. \tag{5.5}$$

The denominators in (5.5) make each term in the functional dimensionless and introduce a barrier preventing mesh from folding.

### 5.3.2 Comparison of two rezone strategies

In this section we consider numerical simulations with the Lagrangian form of Burgers' equation (see (2.8) and (3.3)). We use the same test function (4.25) as in the Eulerian case. The end mesh points $x_0$ and $x_{M+1}$ are moved with the prescribed velocities 1 and 0.1, respectively.

The trajectories of mesh nodes for the pure Lagrangian method are presented in the top-left picture in Fig. 6. As one can see from this picture, the nodes get progressively closer to each other, that is, the size of each cell diminishes. It diminishes faster in regions where the solution has bigger negative gradient. Such behavior is in agreement with the considerations of Section 2.4 about the dynamics of the Jacobian of the transformation from Eulerian to Lagrangian coordinates. It also confirms the statement from [57] (Section 37.2.2) that the method of characteristics is not well-suited for general hyperbolic partial differential equations (PDEs) and in particular for Burgers' equation. From a practical point of view, constantly diminishing size of the Lagrangian cells means that the time step goes to zero and becomes so small that the simulation stalls.

In the top-right picture in Fig. 6, we show trajectories of mesh nodes and the numerical solution corresponding to the moving mesh method with the RJM rezone strategy. It is clear that trajectories roughly follow the Lagrangian trajectories. The mesh is smooth and does not have problems of the pure Lagrangian mesh; however, main features of the solution are under-resolved.

In the bottom picture in Fig. 6, we show trajectories of mesh nodes and the numerical solution corresponding to the moving mesh method with the EMB rezone strategy. At each time moment, the mesh is smooth and resolves features of the solution. The superiority of the EMB rezone strategy will become more obvious, if we plot the error as the function of time (see Fig. 7). The pure Lagrangian method is accurate up to about $t = 0.1$. After this moment, dynamics of Lagrangian cells do not correspond to dynamics of solution features and accuracy starts to degrade. For the moving mesh method with the RJM rezone strategy, the error first starts to grow because the RJM method makes the mesh smoother compared to the original one and by doing this, it reduces resolution of the features. Then, the error stays approximately constant. After $t = 0.3$, the accuracy of the method is significantly better then in the pure Lagrangian method. For the moving mesh method with the EMB rezone strategy, the error is about the same as for the pure Lagrangian method for $t \leq 0.15$, but after this moment, it is significantly less than in the other two methods.

The right picture in Fig. 7 demonstrates the $L_2$-norm of error versus the number of mesh steps. We observe the linear convergence rate in both moving mesh methods. However, for the

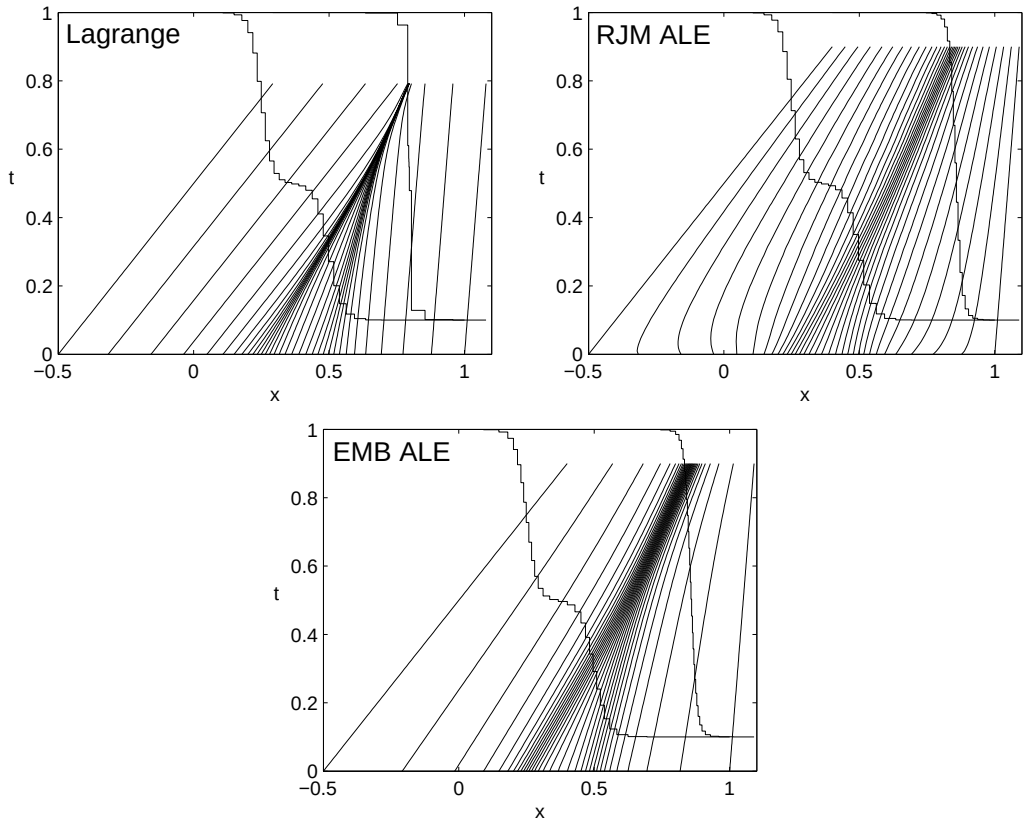


Figure 6: Viscous Burgers' equation with $\varepsilon = 0.005$: trajectories of 34 mesh points and discrete solutions at initial and final time moments.

given accuracy, e.g. $2 \cdot 10^{-3}$, we need about 2 times less mesh points in the method employing the EMB rezone strategy. This difference grows when $\varepsilon$ decreases and the solution profile becomes more sharp. For example, for $\varepsilon = 0.002$ and the same accuracy $2 \cdot 10^{-3}$, we need about 3 times less mesh points.

# 6 Related Methods

In this section, we review the fundamental ideas used in mesh $r$-adaptation methods ("$r$" stands for relocation, redistribution, or repositioning) and position our method with respect to other methods.

According to [43, Ch. 14, p. 261], any adaptive scheme is composed of three main ingredients: (I) an optimal-mesh criterion, (II) an error indicator, and (III) an algorithm or a strategy for mesh improvement. These ingredients answer to the following three questions. What is the optimal mesh? Where is the mesh change required? How is the improved mesh constructed?

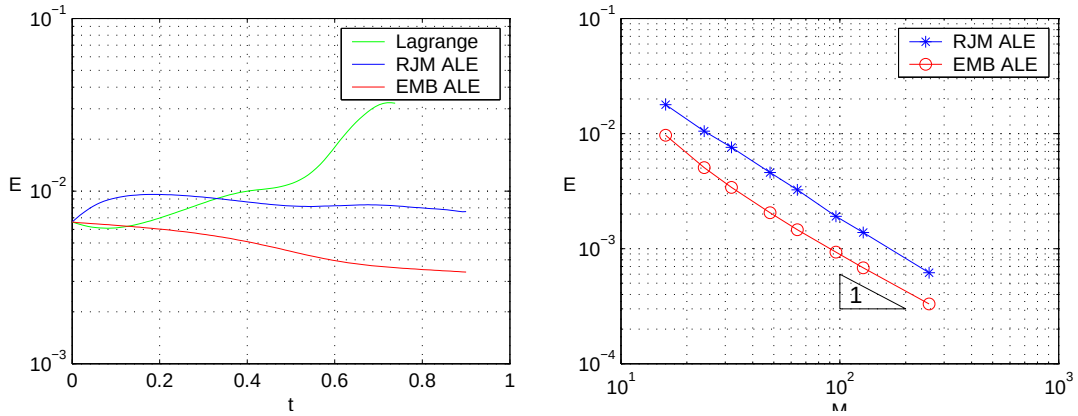


Figure 7: The left picture shows accuracy versus time for the case $\varepsilon = 0.005$ and $M = 32$. The right picture shows convergence of the moving mesh methods.

## 6.1 Optimal-mesh criteria

It is clear that the optimal mesh can be defined as the mesh for which efficiency of the overall algorithm is improved, that is, given accuracy is achieved with the least amount of work. Work in this context should be understood not only as the CPU time but also as memory and man-hour resources. However, to design an adaptive method, one needs a quantitative assessment of optimality of the adaptive mesh procedure.

One of the most widely spread concepts in mesh r-adaptation is the error equidistribution principle (see, e.g., [22, 50]). The goal is to find a mesh on which error is uniformly distributed in space. The list of methods which use some form of this concept includes the moving mesh partial differential equations (MMPDE) [30], the method based on the geometric conservation law (GCL) [18], the deformation method [12, 40, 53] and the methods based on harmonic maps [3, 4, 15, 38].

Another important concept in mesh r-adaptation was introduced by the moving finite element (MFE) methods [46]. In these methods, movement of mesh points is driven by the residual of a finite element approximation. A simplified adaptive mesh technique based on the MFE method is presented in [26].

An interesting idea is introduced in [5–7] and is termed as the moving best fit (MBF) method in [30]. The goal of the MBF method is to find a mesh which minimizes the $L_2$-norm of error of representation of a continuous function by either a piecewise constant or a piecewise linear function.

The new in our method is that the optimal-mesh criterion at time $t^n$ is the minimization of the $L_2$-norm of error of piecewise constant representation of the exact solution at time $t^{n+1}$. In the case of viscous Burgers' equation, our analysis shows connections with other methods. We have proved that the leading term in the error does depend on accuracy of a piecewise constant

representation at time $t^n$. Formula (4.22) is the link to the error equidistribution principle.

## 6.2 Error indicators and estimators

Error indicators and estimators is the basis on which one can decide where mesh changes are required. According to [43], the most popular error indicators used presently in production codes may be grouped into the following categories: jumps in a monitored physical variable (such as density or entropy); difference in approximations of different order; difference in significant derivatives computed with schemes of different order; the residual of PDEs over adjacent cells; a physics-based quantity (such as the energy norm); etc.

A well-developed theory of *a posteriori* error estimates does exist for elliptic equations [1,59]; however, this is not the case for hyperbolic equations. In most moving mesh methods, the frequently used error indicators are either *monitor functions* or *weight functions*. In general, a monitor function can be a matrix. Monitor functions are used as coefficients in equations defining the mesh movement. One of the most traditional monitor functions uses the gradient of a solution $u(x,t)$:

$$M(x,t) = \sqrt{1 + \alpha\,|\nabla\, u(x,t)|^p}. \tag{6.1}$$

Here $\nabla\, u$ plays the role of error indicator, $\alpha$ controls the level of adaptivity, $p$ is responsible for some smoothing and 1 is added to prevent possible mesh degeneracy in regions where function $u$ is constant. Any reasonable error indicator can be put in place of $\nabla u$.

In most methods based on some form of equidistribution, the equidistribution principle is applied to the monitor function. Thus, mesh points are concentrated in regions where the monitor function is large. Extensive discussion and numerical experiments with different error indicators and monitor functions can be found in [16,17,31]. Some discussion on how to choose a monitor function sensitive to both shocks and contact discontinuities is presented in [55].

In our method, the error estimator is the leading term in the $L_2$-norm of error. The error indicator is the square root of the smoothed functional $\Phi_{sm}$. It is important to note that the error indicator is optimal for smooth solutions, i.e. it converges to the real error when the spatial and temporal meshes are refined. This is not the case of methods based on monitor function (6.1).

## 6.3 Coupled and decoupled algorithms

Moving mesh methods can be divided in two groups with respect to how mesh movement is incorporated into the overall solution algorithm. In the first group, a mesh movement equation is coupled with governing physical PDEs. This coupling leads to a system of nonlinear equations even if the original PDEs are linear. Moreover, it introduces new terms in the governing PDEs which require development of new discretizations and new analysis (e.g., stability) of the coupled nonlinear system. On the other hand, the methods from this group do not require any explicit data interpolation between meshes. In the second group, the mesh movement equation is decoupled from the governing PDEs. It means that spatial and temporal discretizations of

the PDEs can be done with the standard methods. On the other hand, after each time step, the solution has to be interpolated to a rezoned mesh.

This classification of moving mesh methods follows roughly the one presented at T. Tang's web-page (`http://lsec.cc.ac.cn/~ttang/MMref`) where he proposes to distinguish interpolation-free and interpolation-based methods.

Examples of interpolation-free methods are the classical 1D method by Dorfi and Drury [25], the previously mentioned MMPDE family of methods which includes also the methods described in [9,10,55], the deformation method, the MFE methods, and the arbitrary Lagrangian-Eulerian (ALE) methods [24, 32]. For all these methods, the governing PDEs are written in a moving coordinate frame that introduces additional terms in the original equations depending on the mesh velocity. The variety of methods comes from a choice of a mesh movement equation and a solution algorithm.

Examples of the interpolation-based methods (PDEs and a mesh movement equation are decoupled) can be found in [3,4,14,28,34,38,44,52,56,61]. Because the space-time discretization of governing PDEs is standard, the methods differ by a choice of the mesh movement equation and the interpolation algorithm. It is clear that the interpolation procedure has to be conservative to maintain conservation property of the overall method. This is the obvious requirement for methods solving the hyperbolic conservation laws; however, in context of general moving mesh methods it is still considered as a new idea [56]. Indeed, the conservation property may be also important for other types of governing equations, since it preserves some norm of a discrete solution.

Our method is the example of an interpolation-based method.

### 6.4 Strategies for mesh movement

One can distinguish the moving mesh methods by a strategy they employ to find new positions of mesh nodes. In the first group of methods, these positions are computed directly. In the second group, nodal velocities are defined to update the positions. This classification of rezone strategies is used in [33] for ALE methods and in [19] for moving mesh methods. The velocity-based approach is usually used in algorithms where the mesh movement equation is coupled with the governing PDEs while the location-based approach is mainly used in decoupled algorithms.

Typical velocity-based approaches are the classical Lagrangian method [54, Ch. 5], [20], the MFE method [6, 21, 46], the methods based on the geometric conservation law [18], the deformation methods [12, 40, 53], the method presented in the seminal paper [25] and called the moving finite-difference (MFD) method in [57, Ch. 37.3.1], and the MMPDE family of methods [30].

In Lagrangian methods, the velocities of mesh nodes equal to fluid velocity. In the MFE method, the equation for mesh movement is based on minimization of the residual of the governing PDEs written in a moving coordinate frame. The distinctive feature of this method is that it does not use any monitor function. The deformation method generates a time-dependent nodal mapping between the computational and physical domains. The distinctive feature of this method is the existence of a valid mesh, because the Jacobian of the mapping is strictly

positive. The GCL is an extension of the deformation method where a more general equation is solved for mesh velocity. In particular, it allows generation of an irrotational mesh velocity by a special choice of parameters. The MFD method and its extension, the MMPDE method, use a parabolic PDE (with coefficients depending on a monitor function) for mesh movement. Their distinctive feature is mesh smoothness in time and space.

Typical location-based approaches are the equidistribution methods in 1D [42, Sec. 7.2], the methods minimizing the Winslow variable diffusion functional [61, Sec. 7.10.6], the harmonic mapping methods [3, 15, 27, 29], the methods using the Brackbill and Saltzman functionals [14] and the directional control functional [13], the Jacobian-weighted elliptic mesh generation methods [34], the Thompson method [58], the variational approach suggested in [19], the MBF method [5, 7], and the method described in one of the first papers on moving mesh methods [62]. Other examples of location-based methods are various methods using physical analogies with spring systems and modified elasticity equations [2, 8, 11, 23] where stiffness of springs and properties of artificial elastic media depend on a monitor function.

The location-based methods can be interpreted as variational methods. In particular, the functional corresponding to a harmonic mapping has the determinant of Jacobian of transformation in the denominator which plays the role of a *barrier* and produces unfolded meshes.

In our method, the strategy for the mesh movement is the direct minimization of functional $\Phi_{sm}$ which is based on *a posteriori* error estimates. The new in our method is that the guaranteed mesh smoothing is embedded into $\Phi_{sm}$.

### 6.5 Mesh smoothness

One of the most important questions in moving mesh methods is how to maintain smoothness of the mesh in space and time. Almost any paper on moving mesh methods addresses this problem. In 1D, many methods use the original idea of a guaranteed smoothing introduced in the seminal paper by Dorfi and Drury [25]. Detailed exposition of this idea is presented in dissertation [63]. In 2D, spatial smoothing usually achieved by using smoothed monitor functions and time dependent equations with relaxation factors for mesh movement.

Smoothness of the mesh is closely related to the overall stability of moving mesh methods [39, 50]. A guaranteed mesh smoothing in multidimensions is a very active research area.

The distinctive feature of our method is that the guaranteed smoothing is applied directly to *a posteriori* error estimates.

## 7 Conclusion

We have developed the error-minimization-based rezone strategy for the moving mesh method solving 1D viscous Burgers' equation in both the Eulerian and the Lagrangian forms. The goal of this strategy is to change the mesh at time $t^n$ to reduce the $L_2$-norm of error at time $t^{n+1}$. For the Eulerian form of Burgers' equation, we have demonstrated the superiority of our method over the same method on uniform meshes. For the Lagrangian form of Burgers' equation, we

have demonstrated superiority of our method over the pure Lagrangian method and the moving mesh method with the RJM rezone strategy.

In future we plan to extend our rezone strategy to the full system of gasdynamics equations in 2D and 3D.

## Acknowledgments

The work was performed at Los Alamos National Laboratory operated by the University of California for the US Department of Energy under contract W-7405-ENG-36. The authors acknowledge the partial support of the DOE/ASCR Program in the Applied Mathematical Sciences, the Laboratory Directed Research and Development program (LDRD), and DOE's Accelerated Strategic Computing Initiative (ASCI). The authors thank B. Wendroff, L. Margolin, M. Hyman, S. Li, R. Garimella (LANL) and reviewers for valuable comments.

### References

[1] M. Ainsworth and J. T. Oden, Posteriori Error Estimation in Finite Element Analysis. A Wiley-Interscience Publication, John Wiley & Sons, Inc., New York, 2000.
[2] D. Ait-Ali-Yahia, G. Baruzzi, W. G. Habashi, M. Fortin, J. Dompierre, and M.-G. Vallet, Anisotropic mesh adaptation: Toward user-independent, mesh-independent and solver-independent CFD. part i: Structured grids, Int. J. Numer. Meth. Fluids, 39:657–673, 2002.
[3] B. N. Azarenok, Variational barrier method of adaptive grid generation in hyperbolic problems of gas dynamics, SIAM J. Numer. Anal., 40:651–682, 2002.
[4] B. N. Azarenok and S. A. Ivanenko, Application of adaptive grids in numerical analysis of time-dependent problems in gas dynamics, Comput. Math. Math. Phys., 40(9):1330–1349, 2000.
[5] M. J. Baines, Algorithms for optimal discontinuous piecewise linear and constant $L_2$ fits to continuous functions with adjustable nodes in one and two dimensions, Math. Comput., 62(206):645–669, 1994.
[6] M. J. Baines, Moving Finite Elements. Oxford University Press, New York, 1994.
[7] M. J. Baines, Grid adaptation via node movement, Appl. Numer. Math., 26:77–96, 1998.
[8] T. J. Baker, Mesh adaptation strategies for problems in fluid dynamics, Finite Elem. Anal. Des., 25:243–273, 1997.
[9] G. Beckett, J. A. Mackenzie, A. Ramage, and D. M. Sloan, On the numerical solution of one-dimensional PDEs using adaptive methods based on equidistribution, J. Comput. Phys., 167:372–392, 2001.
[10] G. Beckett, J. A. Mackenzie, A. Ramage, and D. M. Sloan, Computational solution of two-dimensional unsteady PDEs using moving mesh methods, J. Comput. Phys., 182:478–495, 2002.
[11] T. Belytschko, W. K. Liu, and B. Moran, Nonlinear Finite Elements for Continua and Structures. John Wiley & Sons, Ltd., Chichester, 2000.
[12] P. Bochev, G. Liao, and G. Pena, Analysis and computation of adaptive moving grids by deformation. Numer. Meth. PDEs, 12:489–506, 1996.
[13] J. U. Brackbill, An adaptive grid with directional control, J. Comput. Phys., 108:38–50, 1993.
[14] J. U. Brackbill and J. S. Saltzman, Adaptive zoning for singular problems in two dimensions. J. Comput. Phys., 46:342–368, 1982.

[15] W. Cao, R. Carretero-Gonzalez, W. Huang, and R. Russell, Variational mesh adaptation methods for axisymmetrical problems, SIAM J. Numer. Anal., 41(1):235–257, 2003.
[16] W. Cao, W. Huang, and R. D. Russell, Comparison of two-dimensional r-adaptive finite element methods using various error indicators, Math. Comp. Simul., 56:127–143, 2001.
[17] W. Cao, W. Huang, and R. D. Russell, An error indicator monitor function for r-adaptive finite-element method, J. Comput. Phys., 170:871–892, 2001.
[18] W. Cao, W. Huang, and R. D. Russell, A moving mesh method based on the geometric conservation law, SIAM J. Sci. Comput., 24:118–142, 2002.
[19] W. Cao, W. Huang, and R. D. Russell, Approaches for generating moving adaptive meshes: Location versus velocity, Appl. Numer. Math., 2003.
[20] E. J. Caramana, D. E. Burton, M. J. Shashkov, and P. P. Whalen, The construction of compatible hydrodynamics algorithms utilizing conservation of total energy, J. Comput. Phys., 146:227–262, 1998.
[21] N. N. Carlson and K. Miller, Design and application of gradient-weighted moving finite element code. I. In one dimension, SIAM J. Sci. Comp., 19:728–765, 1998.
[22] C. de Boor, Good Approximation by Splines with Variable Knots, II. Springer Lecture Series 363, Springer-Verlag, NY, 1973.
[23] C. Degand and C. Farhat, A three-dimensional torsional spring analogy method for unstructured dynamic meshes, Comput. & Structures, 80:305–316, 2002.
[24] J. Donea, S. Guliani, and J. P. Halleux, An arbitrary Lagrangian-Eulerian finite-element method for transient dynamic fluid structure interaction, Comput. Methods Appl. Mech. Engrg., 33:689–723, 1982.
[25] E. Dorfi and L. Drury, Simple adaptive grids for 1-D initial value problems, J. Comput. Phys., 69:175–195, 1987.
[26] J. K. Dukowicz, A simplified adaptive mesh technique derived from the moving finite element method, J. Comput. Phys., 56:324–342, 2002.
[27] A. S. Dvinsky, Adaptive grid generation from harmonic maps on Riemannian manifolds, J. Comput. Phys., 95:450–476, 1991.
[28] C. W. Hirt, A. Amsden, and J. Cook, An arbitrary Lagrangian-Eulerian computing method for all flow speeds, J. Comput. Phys., 14:227–253, 1974.
[29] W. Huang, Variational mesh adaptation: isotropy and equidistribution, J. Comput. Phys., 174:903–924, 2001.
[30] W. Huang and R. D. Russell, Adaptive mesh movement – the MMPDE approach and its applications, J. Comput. Appl. Math., 128:383–398, 2001.
[31] W. Huang and W. Sun, Variational mesh adaptation ii: Error estimates and monitor functions, J. Comput. Phys., 184:619–648, 2003.
[32] D. S. Kershaw, M. K. Prasad, M. J. Shaw, and J. L. Milovich, 3D unstructured mesh ALE hydrodynamics with the upwind discontinuous finite element method, Comput. Methods Appl. Mech. Engrg., 158:81–116, 1998.
[33] P. Knupp, L. G. Margolin, and M. Shashkov, Reference Jacobian optimization-based rezone strategies for arbitrary Lagrangian Eulerian methods, J. Comput. Phys., 176:93–128, 2002.
[34] P. M. Knupp, Jacobian-weighted elliptic grid generation, SIAM J. Sci. Comput., 17:1475–1490, 1996.
[35] M. Kucharik, M. Shashkov, and B. Wendroff, An efficient linearity-and-bound-preserving remapping methods, J. Comput. Phys., 188:462–471, 2003.
[36] C. B. Laney, Computational Gasdynamics. Cambridge University Press, 1998.
[37] R. J. LeVeque, Numerical Methods for Conservation Laws. Birkhauser Verlag, Basel, Boston, Berlin, 1992.

[38] R. Li, T. Tang, and P. Zhang, Moving mesh methods in multiple dimensions based on harmonic maps, J. Comput. Phys., 170:562–588, 2001.
[39] S. Li, L. Petzold, and Y. Ren, Stability of moving mesh systems of partial differential equations, SIAM J. Sci. Comput., 20:719–738, 1998.
[40] G. Liao, F. Liu, G. C. de la Pena, D. Peng, and S. Osher, Level-set-based deformation methods for adaptive girds, J. Comput. Phys., 159:103–122, 2000.
[41] K. Lipnikov and M. Shashkov, Moving meshes for the Burgers equation, Report LA-UR-03-7605, Los Alamos National Laboratory, 2003.
[42] V. Liseikin, Grid Generation Methods, Springer, Berlin, Heidelberg, New York, 1999.
[43] R. Löhner, Applied CFD Techniques. An Introduction Based on Finite Element Methods. John Wiley & Sons, Ltd., Chichester, 2001.
[44] L. G. Margolin, Introduction to "an arbitrary Lagrangian-Eulerian computing method for all flow speeds", J. Comput. Phys., 135:198–202, 1997.
[45] L. G. Margolin and M. Shashkov, Second-order sign-preserving conservative interpolation (remapping) on general grids, J. Comput. Phys., 184:226–298, 2003.
[46] K. Miller and R. N. Miller, Moving finite elements I, SIAM J. Numer. Anal., 18:1019–1032, 1981.
[47] K. Morton, Numerical Solution of Convection-Diffusion Problems. Chapmann & Hall, London, 1996.
[48] L. S. Mulholland, Y. Qui, and D. M. Sloan, Solution of evolutionary partial differential equations using adaptive finite differences with pseudospectral post-processing, J. Comput. Phys., 131:280–298, 1997.
[49] J. Nocedal and S. Wright, Numerical Optimization. Springer, Berlin, Heidelberg, New York, 1999.
[50] Y. Ren and R. D. Russell, Moving mesh techniques based upon equidistribution, and their stability, SIAM J. Sci. Comput., 13:1265–1286, 1992.
[51] A. A. Samarskii, The Theory of Finite Difference Schemes. Marcel Dekker, New York, 2001.
[52] P. Saucez, A. V. Wouwer, and W. E. Schiesser, Some observations on a static spatial remeshing method based on equidistribution principles, J. Comput. Phys., 128:274–288, 1996.
[53] B. Semper and G. Liao, A moving grid finite-element method using grid deformation, Numer. Meth. Partial Differential Equations, 11:603–615, 1995.
[54] M. Shashkov, Conservative Finite-Difference Methods on General Grids. CRC Press, Boca Raton, 1996.
[55] J. M. Stockie, J. A. Mackenzie, and R. D. Russell, A moving mesh method for one-dimensional hyperbolic conservation laws, SIAM J. Sci. Comput., 22:1791–1813, 2001.
[56] H. Tang and T. Tang, Adaptive mesh methods for one- and two-dimensional hyperbolic conservation laws, SIAM J. Numer. Anal., 41:487–515, 2003.
[57] J. F. Thompson, B. K. Soni, and N. P. Weatherill, editors, Handbook of Grid Generation. CRC Press, Boca Raton, 1999.
[58] J. F. Thompson, Z. A. Warsi, and C. W. Mastin, editors, Numerical Grid Generation: Foundations and Applications. North-Holland, New York, 1985.
[59] R. Verfürth, A Review of a Posteriori Error Estimation and Adaptive Mesh Refinement Techniques. Teubner Verlag and J. Wiley, Stuttgart, 1996.
[60] Z. U. A. Warsi, Fluid Dynamics: Theoretical and Computational Approaches. CRC Press, Boca Raton, 1993.
[61] A. Winslow, Adaptive Mesh Zoning by Equipotential Method, Report UCID-19062, Lawrence Livermore Laboratory, 1981.
[62] N. N. Yanenko, E. A. Kroshko, V. V. Liseikin, V. M. Fomin, V. P. Shapeev, and Y. A. Shitov, Methods for the Construction of Moving Grids for Problems of Fluid Dynamics with Big Deformations, in Proceedings of the Fifth International Conference on Numerical Methods in Fluid Mechanics,

Lecture Notes in Physics, A. I. van de Vooren and P. J. Zandbergen (eds.), 59:454–459, 1976.
[63] P. A. Zegeling, Moving-Grid Methods for Time-Dependent Partial Differential Equations. Centrum voor Wiskunde en Informatica, Amsterdam, 1992.