LA-UR-03-8241

Arbitrary Lagrangian-Eulerian Method with Error-Minimization-Based Rezone Strategy for Burgers' Equation

Konstantin Lipnikov[*]   Mikhail Shashkov[†]

Abstract

The philosophy of the Arbitrary Lagrangian-Eulerian (ALE) methodology for solving multidimensional fluid flow problems is to move the computational grid, using the flow as a guide, to improve the robustness, accuracy and efficiency of the simulation. The main elements in the ALE simulation are an explicit Lagrangian phase, a rezone phase in which a new grid is defined, and a remapping (conservative interpolation) phase in which a Lagrangian solution is transferred to the new grid. In most multidimensional ALE codes, the main goal of rezone phase is to maintain geometrical quality of the mesh, that is, to prevent it from tangling. In this paper, we describe a new rezone strategy for one-dimensional viscous Burgers' equation which minimizes the L_2 norm of error and maintains smoothness of the grid. We present comparison of numerical results obtained by ALE method with the new error-minimization-based (EMB) rezone strategy, ALE method with the Reference Jacobian (RJ) rezone strategy, and the pure Lagrangian method. Numerical results demonstrate superiority of our method.

1 Introduction

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

In general, the motion of the grid can be chosen arbitrarily in flow simulations. The philosophy of the Arbitrary Lagrangian-Eulerian methodology (ALE; cf [39], [14], [61], [45], [46], [68]) is to exploit this degree of freedom to improve the robustness, accuracy and efficiency of the simulation. The main elements in an ALE simulation are an explicit

[*]Los Alamos National Laboratory, MS B284, Los Alamos, NM, 87545. Email: `lipnikov@lanl.gov`, `shashkov@lanl.gov`.

Lagrangian phase in which mesh moves with fluid; a rezone phase in which a new mesh is defined (new mesh has the same connectivity as the Lagrangian mesh); and a remapping phase (conservative interpolation) in which the solution obtained by the Lagrangian step is transferred to the new grid [61].

The Lagrangian phase of ALE methodology is well developed and described in the following papers: [22], [24], [23]. The Lagrangian phase is conservative, i.e mass, momentum, and total energy are conserved. The conservation property is very important for modeling high-speed compressible flow with shocks, which is our primary interest. For a given pure Lagrangian method (that is when the mesh is moving with the fluid) the robustness, accuracy, and efficiency strongly depend on a choice of initial mesh, because it is the only moment when we have control of the mesh. It means that the user experienced with a particular class of problems has to create an initial grid containing some knowledge of how the flow will develop in time. In many situations, it allows successful completion of calculations. However, if the flow develops shear deformations, or other similar features that lead to mesh tangling, it is impossible to use pure Lagrangian methods, and some change in the mesh (rezoning) is required.

There are two principally different strategies *when* to determine when rezoning is appropriate.

The first strategy suggests running the calculations in a pure Lagrangian regime, perform some rezoning when the mesh became tangled, remap solution, run again in a Lagrangian regime and so on. The first argument in support of such a strategy is that in pure Lagrangian regime mesh moves with fluid and therefore "naturally adapts" itself to flow features. The second argument is that the remap is performed just several times during simulation; therefore, possible errors due to remap are negligible. The authors of this paper believe that it is not a good strategy because errors due to mesh distortions accumulate with time and destroy possible advantages of "natural adaptivity" of Lagrangian grids. Moreover, this strategy usually requires global remeshing because small changes in the grid cannot produce an untangled grid or a grid with good geometrical quality such that it will not be tangled again in a couple of time steps. Global remeshing (which usually requires user intervention) can produce a mesh which is very far from the original mesh, which may lead to a significant error due to remap.

The second rezone strategy (continuous rezone) recommends not waiting until the mesh becomes tangled but to perform rezoning regularly, say at each k-th time step where $k \approx 10$ (in principle, rezone can be performed each time step). An example of a sophisticated rezoning strategy suitable for this approach is the Reference Jacobian Matrix (RJ) method which is aimed at maintaining geometrical quality of the mesh while keeping it as close as possible to the Lagrangian mesh [47]. This strategy does not require any user intervention, which is very important because rezoning is performed many times during the simulation. Authors of [47] believe that their strategy is superior to the one described in the previous paragraph because it constantly maintains geometrical quality of mesh reducing discretization error due to mesh distortion, and at the same time preserves some "natural adaptivity" of the Lagrangian mesh by keeping the rezoned mesh as close as possible to the Lagrangian mesh. It is clear that an additional error can be introduced by frequent remappings. However, because the rezoned mesh remains very close to the Lagrangian one and high-quality remapping methods are used (see for example, ([62], [49])), all changes are made only where they are needed and errors due to remap are small. Effectiveness of the RJ rezone strategy has been demonstrated by numerical examples in [47].

The RJ rezone strategy does not directly address the question of improving accuracy and relies on assumption that if the mesh has good geometrical quality and is close to the Lagrangian mesh then it improves accuracy. This is clearly true in comparison with pure Lagrangian methods and some rezone strategies which were available to authors of [47], but the question still remains: Can we improve accuracy even more by using a strategy based directly on reducing some measure of error?

Developing a rezone strategy for reducing error is a challenging task for full gas dynamics equations in 2D and 3D. Therefore we have decided to start with a simpler problem and to develop an error-minimization-based (EMB) rezone strategy for 1D viscous Burgers' equation. This equation has many important features of gasdynamics equations: it expresses conservation law, it can develop shock-like structures, and its Lagrangian form resembles one for a full system of gas dynamics equations.

In the Lagrangian stage of our ALE method, we use the simplest first-order discretization (in time and space) of the conservative Lagrangian form of Burgers' equation. Primary unknowns in the discretization are mean values over mesh cells. Nodes of the mesh are advanced using a simple explicit time integration.

On the remapping stage, we use linearity preserving methods from [49]. Under assumption of solution smoothness (which is the case of viscous Burgers' equation) the methods from [49] are second-order accurate.

The goal of our rezoning strategy is to reduce L_2 norm of error at a new time level t^{n+1}. Analysis shows that under reasonable assumptions about grid smoothness, accuracy of interpolation and time step, the leading term in the error depends on accuracy with which the solution is represented at time level t^n by its exact mean values.

This is the well-known interpolation problem of the best piecewise constant fit with adjustable nodes (see, e.g., [8]). We consider this interpolation problem in detail and develop a direct minimization technique for its solution, which will be used in implementation of our rezone strategy.

Error analysis of the ALE method for Burgers' equation is done under assumption that the mesh is smooth. Therefore, we describe modification of the algorithm for the best piecewise constant fit which guarantees smoothness of the resulting mesh and still reduces the error. We have found that the modified algorithm gives just slightly larger error in comparison with the original method. The smoothness of the mesh (in space and time) is absolutely critical for stability of the overall ALE algorithm.

The outline of our paper is as follows. In the next Section, we describe viscous Burgers' equation. In particular, we derive the conservative Lagrangian form of Burgers' equation which is the basis for our ALE method. In Section 3 we describe the Lagrangian stage of our ALE method, that is, we describe discretization of Burgers' equation in conservative Lagrangian form and discretization of the equation for trajectories of mesh points. Rezone strategies are described in Section 4. We describe both the 1D version of the Reference Jacobian rezone strategy and the Error-Minimization-Based rezone strategy. Analysis of numerical results obtained with pure Lagrangian method, ALE method with the RJ rezone strategy and ALE method with the EMB rezone strategy is presented in Section 5. These results clearly demonstrate superiority of our new EMB ALE method. In Section 6 we present short review of related methods using philosophy of moving grids, emphasize some common ideas and important distinctions with our new method, and finally 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 purpose of this paper. For simplicity, we consider initial value problem.

2.1 Burgers' Equation in Eulerian Form

The standard Eulerian form of 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. 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

$$u(x,t), \quad \frac{\partial u}{\partial x}(x,t) \to 0 \quad \text{when} \quad x \to \pm\infty\,. \tag{2.3}$$

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 that

$$\frac{\partial}{\partial t}\left(\int_{-\infty}^{+\infty} u(x,t)\,dx\right) = -\left(F(+\infty,t) - F(-\infty,t)\right) = 0\,. \tag{2.4}$$

Here we have used the fact that under our assumptions $F(+\infty,t) = F(-\infty,t) = 0$. Thus, $\int_{-\infty}^{\infty} u(x,t)\,dx$ is conserved.

For a finite ε, solution of equation (2.1) is a smooth function which may have sharp gradients whose steepness depends on how small ε is.

To correctly model solutions with sharp gradients, it is very important to use conservative finite-difference methods, which satisfy the discrete analog of equation (2.4).

2.2 Burgers' Equation in Characteristics Form

Following [51, p. 24], we introduce characteristics

$$x'(t) = u(x(t),t)\,. \tag{2.5}$$

Strictly speaking, equation (2.5) defines characteristics of inviscid Burgers' equation. In case of viscous Burgers' equation, (2.5) can be considered as a definition of particle trajectory. The material time or substantive derivative $\frac{d}{dt}$ (the derivative following motion) is defined as

$$\frac{d}{dt}u(x(t),t) = \frac{\partial}{\partial t}u(x(t),t) + x'(t)\cdot\frac{\partial}{\partial x}u(x(t),t)\,. \tag{2.6}$$

Using definition of the material time derivative, we can rewrite equation (2.1) in the following form

$$\frac{du}{dt} = \varepsilon \frac{\partial^2 u}{\partial x^2}\,. \tag{2.7}$$

Now $u = u(x(t), t)$, that is, its computation requires integration of (2.5). Therefore, instead of one equation (2.1) we really have system of equations (2.5) and (2.7). We will call these system by "characteristics" form of Burgers' equation. Because of finite value of ε, "characteristics" defined by (2.5) do not cross [51].

2.3 Burgers' Equation in Conservative Lagrangian Form

Because characteristics do not cross, we can introduce Lagrangian coordinates (see, e.g., [82, p. 4]). It will be convenient for us to choose Lagrangian coordinates, ξ, as initial position, i.e.

$$\xi = x(0)\,. \tag{2.8}$$

In this context, equation (2.5) defines the trajectory of a point which initially identified by Lagrangian coordinate ξ:

$$\frac{d}{dt} x(\xi, t) = u(x(\xi, t), t)\,, \quad x(\xi, 0) = \xi\,. \tag{2.9}$$

In addition to the definition of material time derivative (2.6), we need the following formulas for transformation of the spatial derivative

$$\frac{\partial u}{\partial \xi} = \frac{\partial u}{\partial x}\frac{\partial x}{\partial \xi}, \qquad \frac{\partial u}{\partial x} = \frac{\partial u}{\partial \xi} \Big/ \frac{\partial x}{\partial \xi}\,, \tag{2.10}$$

where $\partial x/\partial \xi$ is the Jacobian of the transformation from Eulerian to Lagrangian coordinates. Since trajectories of particles never cross each other, this Jacobian is always positive.

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

$$\frac{\partial^2 u}{\partial x^2} = \left[\frac{\partial}{\partial \xi}\left(\frac{\partial u}{\partial x}\right)\right] \Big/ \left(\frac{\partial x}{\partial \xi}\right)\,. \tag{2.11}$$

For future derivations, it is convenient to keep both Eulerian and Lagrangian spatial derivatives. Using (2.11), we can write equation (2.7) as follows

$$\left(\frac{\partial x}{\partial \xi}\right)\frac{du}{dt} = \varepsilon \frac{\partial}{\partial \xi}\left(\frac{\partial u}{\partial x}\right)\,. \tag{2.12}$$

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

$$\left(\frac{\partial x}{\partial \xi}\right)\frac{du}{dt} = \frac{d}{dt}\left(\frac{\partial x}{\partial \xi}u\right) - \frac{\partial}{\partial \xi}\left(\frac{u^2}{2}\right)\,. \tag{2.13}$$

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

$$\frac{d}{dt}\left(\frac{\partial x}{\partial \xi}u\right) - \frac{\partial}{\partial \xi}\left(\frac{u^2}{2}\right) = \varepsilon \frac{\partial}{\partial \xi}\left(\frac{\partial u}{\partial x}\right) \tag{2.14}$$

which we shall refer to as the conservative Lagrangian form of Burgers' equation (for general 1D conservation law, a similar equation is derived in [79, p.34-3]).

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

In this section we investigate behavior of Jacobian of transformation from Lagrangian to Eulerian coordinates, $\partial x/\partial\xi$, as function of time for a given ξ.

As we have already mentioned, viscous Burgers' equation has smooth solutions and trajectories do not cross, i.e. $\partial x/\partial\xi > 0$. In particular it has the following solution:

$$u(x,t) = \frac{1}{2}\left[1 - \tanh\left(\frac{x - \frac{1}{2}t - x_0}{4\,\varepsilon}\right)\right]. \tag{2.15}$$

It is easy to check that $\partial u/\partial x < 0$ and therefore $\partial u/\partial\xi < 0$.

Now taking derivative $\partial/\partial\xi$ of equation (2.5) we get

$$\frac{d}{dt}\left(\frac{\partial x}{\partial\xi}\right) = \frac{\partial u}{\partial\xi}\,. \tag{2.16}$$

Therefore, for the solution given by (2.15), we have

$$\frac{d}{dt}\left(\frac{\partial x}{\partial\xi}\right) < 0\,, \tag{2.17}$$

that is, Jacobian is always decreasing.

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

$$x(\xi + \Delta\xi, t) - x(\xi, t) \approx \frac{\partial x}{\partial\xi}\,\Delta\xi\,. \tag{2.18}$$

According to our previous considerations, it means that the size of Lagrangian particle is going to zero. The rigorous proof of this fact is very short and is given below. Introducing a new variable

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

we rewrite (2.15) 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 ξ, 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,

$$\frac{\partial x}{\partial\xi}(t) = 4\varepsilon\,\frac{\partial y}{\partial\xi}(t) \;\to\; 0 \quad \text{as} \quad t \to \infty. \tag{2.19}$$

The practical consequence of (2.19) will be seen in Section 5 where we present numerical results for the pure Lagrangian method.

3 Lagrangian Stage of ALE Method

In our numerical simulations, we solve Burgers' equation on a finite interval with boundary points moving with prescribed velocities.

In the Lagrangian stage of ALE method, the mesh moves according to some approximation of equation (2.5). Therefore, we introduce the superscript n which specifies the time level for all grid-related quantities. At time $t = t^n$, a grid $\{x_i^n\}$ is given by:

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

Let $h_{i+1/2}^n = x_{i+1}^n - x_i^n$ be mesh steps and $x_{i+1/2}^n = (x_{i+1}^n + x_i^n)/2$ be middle points of mesh intervals. On this grid, the function $u(x(\xi,t),t)$ will be represented by cell-centered values $\bar{u}_{i+1/2}^n$. We shall consider $\bar{u}_{i+1/2}^n$ as an approximation of the exact mean value at $t = t^n$, i.e.

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

In our ALE method, we will use the following discretization of (2.14), (2.5) (note that the role of ξ is played by indices i and $i+1/2$):

$$\begin{aligned}
&\frac{h_{i+1/2}^{n+1}\bar{u}_{i+1/2}^{n+1} - h_{i+1/2}^n \bar{u}_{i+1/2}^n}{\Delta t^n} - \frac{1}{2}\left(\left(u_{i+1}^n\right)^2 - \left(u_i^n\right)^2\right) = \varepsilon\left(\left[\frac{\delta \bar{u}^n}{\delta x}\right]_{i+1} - \left[\frac{\delta \bar{u}^n}{\delta x}\right]_i\right), \\
&x_i^{n+1} = x_i^n + \Delta t^n\, u_i^n\,.
\end{aligned} \qquad (3.1)$$

where $\Delta t^n = t^{n+1} - t^n$ denotes the time step, u_i^n denotes nodal velocity obtained by linear interpolation of cell-centered values,

$$u_i^n = \frac{h_{i-1/2}^n\, u_{i+1/2}^n + h_{i+1/2}^n\, u_{i-1/2}^n}{h_{i-1/2}^n + h_{i+1/2}^n}\,, \qquad (3.2)$$

and

$$\left[\frac{\delta \bar{u}^n}{\delta x}\right]_i = \frac{\bar{u}_{i+1/2}^n - \bar{u}_{i-1/2}^n}{0.5\,(h_{i+1/2}^n + h_{i-1/2}^n)} \approx \frac{\partial u}{\partial x}(x_i^n, t^n)\,.$$

The sequence of computations in the Lagrangian stage consists of three steps. First, we compute nodal velocities according to (3.2), then we compute new coordinates of mesh nodes using the second formula in (3.1), and finally we update cell-centered values of u:

$$\begin{aligned}
\bar{u}_{i+1/2}^{n+1} = \frac{h_{i+1/2}^n}{h_{i+1/2}^{n+1}}\,\bar{u}_{i+1/2}^n \quad &+ \quad \frac{\Delta t^n}{h_{i+1/2}^{n+1}}\;\frac{(u_{i+1}^n)^2 - (u_i^n)^2}{2} \\
&+ \quad \frac{\varepsilon \Delta t^n}{h_{i+1/2}^{n+1}}\left(\left[\frac{\delta \bar{u}^n}{\delta x}\right]_{i+1} - \left[\frac{\delta \bar{u}^n}{\delta x}\right]_i\right).
\end{aligned}$$

Time step Δt^n is chosen according to the following practical stability condition

$$\Delta t^n \le \min_i \left(\frac{|\bar{u}_{i+1/2}^n|}{h_{i+1/2}^n} + \frac{2\varepsilon}{(h_{i+1/2}^n)^2}\right)^{-1} \qquad (3.3)$$

which can be found, for example, in [65], [72].

4 Rezone Strategies

In this section, we describe rezone strategies which will be used in ALE simulations of Burgers' equation.

We assume that the initial grid x_i^0, at time moment $t = 0$, is chosen in such a way that all features of function $U(x)$, which specifies 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 [8] (for more details see Remark 4.1).

Now, we need to choose a grid to start a new Lagrangian step. It is obvious that at the time level t^n, we only know the mean values $\bar{u}^n_{i+1/2}$ on a grid $\{x_i^n\}$ (which is either the result of the previous Lagrangian step, or the initial mesh, if $n = 0$). We can start the new Lagrangian step using this data (that is, not doing any rezone) or use another rezoned mesh $\{\tilde{x}_i^n\}$ and the corresponding remapped (conservatively interpolated) mean values $\overline{\tilde{u}}^n_{i+1/2}$. In simulations presented in this paper, we use conservative linearity-and-bound preserving interpolation from [49]. The conservative property of the interpolation imply conservation of the overall ALE method.

Now, the time advance will look as follows

$$
\begin{aligned}
x_i^{n+1} &= \tilde{x}_i^n + \Delta t^n \, \tilde{u}_i^n \,, \qquad h^{n+1}_{i+1/2} = x^{n+1}_{i+1} - x^{n+1}_i \,, \\
\bar{u}^{n+1}_{i+1/2} &= \mathcal{L}^n_{i+1/2}\left(\{\overline{\tilde{u}}^n_{i+1/2}\}\right) \\
&= \frac{\tilde{h}^n_{i+1/2}}{h^{n+1}_{i+1/2}}\, \overline{\tilde{u}}^n_{i+1/2} + \frac{\Delta t^n}{h^{n+1}_{i+1/2}} \, \frac{(\tilde{u}^n_{i+1})^2 - (\tilde{u}^n_i)^2}{2} + \frac{\varepsilon \Delta t^n}{h^{n+1}_{i+1/2}} \left(\left[\frac{\delta \overline{\tilde{u}}^n}{\delta x}\right]_{i+1} - \left[\frac{\delta \overline{\tilde{u}}^n}{\delta x}\right]_i \right) ,
\end{aligned}
\tag{4.1}
$$

where Δt^n is defined by stability condition (3.3) for grid $\{\tilde{x}_i^n\}$.

Suggested logic assumes that the procedure for finding grid $\{\tilde{x}_i^n\}$ *does not* depend on Δt^n. The only assumption we have made is that Δt^n satisfies the stability condition. We shall address this question later in this paper.

4.1 Reference Jacobian Rezone strategy

In this section we briefly describe the Reference Jacobian rezone strategy [47] for 1D case. The goal of algorithm presented in [47] is to maintain geometrical quality of the mesh while keeping it as close as possible to the Lagrangian mesh.

The essential idea used in [47] 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 grid has a much shorter wavelength, and so 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 grid which is close to the Lagrangian grid but is "smoother" (i.e., it has better geometrical quality).

The most fundamental object describing a map is its Jacobian matrix (Jacobian, $\partial x/\partial \xi$, in 1D). Therefore, one can expect that if two maps have similar Jacobians, then the maps themselves must be similar and must produce grids close to each other. This leads to the following strategy. We first construct Jacobians —termed as reference Jacobians (RJs) — that are based on the geometry of the nearest neighbors of a Lagrangian cell and effectively smooth the short-wavelength grid deformation. Then, we construct a global functional that measures the difference between the RJ (which depends on the Lagrangian grid and the smoothing process) and a rezoned Jacobian (which is not known before the minimization process) over all mesh cells. By minimizing this functional over the entire grid (as a functional of the vertex coordinates), we get the rezoned grid. Note that in 1D the Jacobian is $h^n_{i+1/2}$, $h^n_{i+1/2} = x^n_{i+1} - x^n_i$.

Following the algorithm from [47], 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 size of virtual cells $i + 1/2$ and $i - 1/2$ the same. The reference position for node i is shown in Fig. 4.1 by a square shaded with diagonal lines.

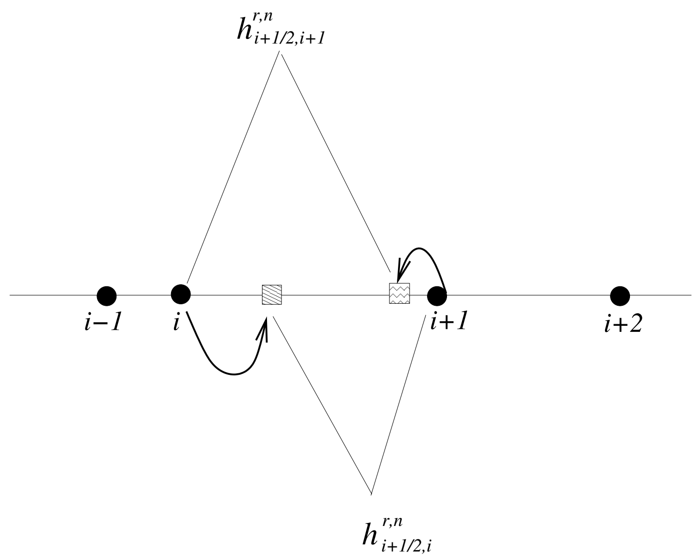


Figure 1: Construction of the Reference Jacobian. Solid circles are positions of nodes after the Lagrangian step. Diagonally shaded square is the reference position of node i, and square shaded by "waves" is the reference position of node $i + 1$.

It gives one reference Jacobian, $h^{r,n}_{i+1/2,i}$ for cell $i + 1/2$,

$$h^{r,n}_{i+1/2,i} = (h^n_{i+1/2} + h^n_{i-1/2})/2 \,. \tag{4.2}$$

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

$$h^{r,n}_{i+1/2,i+1} = (h^n_{i+1/2} + h^n_{i+3/2})/2 \,. \tag{4.3}$$

Finally, the global functional which has to be minimized is

$$F(\{\tilde{x}^n_i\}) = \sum_{cells} \left\{ \frac{\left[(\tilde{x}^n_{i+1} - \tilde{x}^n_i) - h^{r,n}_{i+1/2,i}\right]^2}{(\tilde{x}^n_{i+1} - \tilde{x}^n_i)\, h^{r,n}_{i+1/2,i}} + \frac{\left[(\tilde{x}^n_{i+1} - \tilde{x}^n_i) - h^{r,n}_{i+1/2,i+1}\right]^2}{(\tilde{x}^n_{i+1} - \tilde{x}^n_i)\, h^{r,n}_{i+1/2,i+1}} \right\} . \tag{4.4}$$

The denominators in (4.4) make each term in the functional dimensionless and introduce a barrier preventing mesh from folding. The details of optimization procedures minimizing functional (4.4) can be found in [47].

4.2 Error-Minimization-Based Rezone strategy

In the error-minimization-based (EMB) rezone strategy, the choice of grid $\{\tilde{x}_i^n\}$ is based on minimization of L_2 norm of error on new time level t^{n+1},

$$\min_{\{\tilde{x}_i^n\}} \sum_{i=0}^{M} \int\limits_{x_i^{n+1}}^{x_{i+1}^{n+1}} (u(x,\, t^{n+1}) - \bar{u}_{i+1/2}^{n+1})^2 \, \mathrm{d}x = \min_{\{\tilde{x}_i^n\}} \sum_{i=0}^{M} \int\limits_{x_i^{n+1}}^{x_{i+1}^{n+1}} (u(x,\, t^{n+1}) - \mathcal{L}_{i+1/2}^n(\{\overline{\tilde{u}}_{i+1/2}^n\}))^2 \, \mathrm{d}x \,, \tag{4.5}$$

where $x_i^{n+1} = \tilde{x}_i^n + \Delta t^n \, \tilde{u}_i^n$.

The L_2 norm of error on new time level t^{n+1} is a superposition of the interpolation error in computing mean values $\overline{\tilde{u}}_{i+1/2}^n$, an error due to the time advancing method (4.1), and the space discretization error. It is clear that if $\{\tilde{x}_i^n\} = \{x_i^n\}$, then the interpolation error is zero; however, the grid $\{x_i^{n+1}\} = \{\tilde{x}_i^n + \Delta t^n \, \tilde{u}_i^n\}$ may not be the best grid to represent features of the solution at time t^{n+1}. The goal of minimization problem (4.5) is to achieve balance between the errors (if possible) which will minimize the overall error at time t^{n+1}.

It is obvious that the time advancing method (4.1), and therefore, L_2 error at t^{n+1} depends on a choice of Δt^n. At first glance, it looks like our assumption that the procedure for finding grid $\{\tilde{x}_i^n\}$ *does not* depend on Δt^n is violated. However, we shall prove in the next sections, that under additional assumptions about the order of interpolation and grid smoothness, the leading term in (4.5) *does not* depend on Δt^n if the latter satisfies the stability condition.

4.2.1 Error analysis

In order to simplify analysis of Burgers' equation, we assume that the data associated with grid $\{x_i^n\}$ are exact. In reality, the mean values $\bar{u}_{i+1/2}^n$ are approximate values and accumulate errors during time advance from the initial time moment $t = 0$ to the current time moment t^n. However, we shall ignore this fact in future analysis and assume that $\bar{u}_{i+1/2}^n$ are exact mean values on grid $\{x_i^n\}$. Our numerical experiments confirm that this is a valid assumption for examples considered in this paper. A rigorous theoretical justification of this assumption will be the topic of future research.

Let us introduce a small parameter $h = 1/(M+1)$. All accuracy estimates will be formulated with respect to this small parameter.

We assume that there exist constants c_h, C_h and $C_{\Delta h}$, independent of h, such that

$$c_h \, h \le \tilde{h}_{i+1/2}^n \le C_h \, h \qquad \text{and} \qquad |\tilde{h}_{i+1/2}^n - \tilde{h}_{i-1/2}^n| \le C_{\Delta h} \, h^2 \,. \tag{4.6}$$

That is, we assume that mesh steps are of order h and the change in mesh steps is of order h^2. The problem of building grids satisfying (4.6) will be addressed in Section 4.2.3.

Our next assumption is that the interpolation procedure from grid $\{x_i^n\}$ to grid $\{\tilde{x}_i^n\}$ is second-order accurate. In other words, the interpolated mean values are second-order approximations of the exact ones,

$$\overline{\tilde{u}}_{i+1/2}^n = \overline{u}([\tilde{x}_i^n, \tilde{x}_{i+1}^n], t^n) + O(h^2) \,. \tag{4.7}$$

In this paper, we use the second-order accurate conservative interpolation (remapping) described in [62], [49].

Let us now analyze the function

$$e_{i+1/2}(x,t^{n+1}) = u(x,t^{n+1}) - \overline{u}^{n+1}_{i+1/2}\,, \qquad x \in [x_i^{n+1}, x_{i+1}^{n+1}]\,, \tag{4.8}$$

representing the error between the solution at time t^{n+1} and its piecewise constant approximation. Making use of Lagrangian coordinate ξ, function (4.8) can be rewritten in equivalent form

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

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 (ξ, t^n):

$$u(x(\xi,t^{n+1}),t^{n+1}) = u(x(\xi,t^n),t^n) + \Delta t^n \,\frac{du}{dt}\,(x(\xi,t^n),t^n) + O\left((\Delta t^n)^2\right). \tag{4.9}$$

Because all continuous functions are computed at the same point (ξ, t^n), these arguments will be omitted in the sequel. Using (2.7), equation (4.9) can be rewritten as follows:

$$u(x(\xi,t^{n+1}),t^{n+1}) = u + \varepsilon\,\Delta t^n\,\frac{\partial^2 u}{\partial x^2} + O\left((\Delta t^n)^2\right)\,. \tag{4.10}$$

The stability conditions (3.3) implies that $\varepsilon\,\Delta t^n < (\tilde{h}^n_{i+1/2})^2/2$. Thus, the ε-term in (4.10) is of order h^2 and therefore

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

Now, let us consider the expression for $\overline{u}^{n+1}_{i+1/2}$,

$$\begin{aligned}\bar{u}^{n+1}_{i+1/2} &= \mathcal{L}^n_{i+1/2}\left(\{\overline{\tilde{u}}^n_{i+1/2}\}\right)\\ &= \frac{\tilde{h}^n_{i+1/2}}{h^{n+1}_{i+1/2}}\,\overline{\tilde{u}}^n_{i+1/2} + \frac{\Delta t^n}{h^{n+1}_{i+1/2}}\,\frac{(\tilde{u}^n_{i+1})^2-(\tilde{u}^n_i)^2}{2} + \frac{\varepsilon\Delta t^n}{h^{n+1}_{i+1/2}}\left(\left[\frac{\delta\overline{\tilde{u}}^n}{\delta x}\right]_{i+1} - \left[\frac{\delta\overline{\tilde{u}}^n}{\delta x}\right]_i\right).\end{aligned} \tag{4.12}$$

First, we note that the equation of grid motion (3.1) 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,

$$\frac{\tilde{h}^n_{i+1/2}}{h^{n+1}_{i+1/2}} = 1 - \Delta t^n\,\frac{\partial u}{\partial x}(\tilde{x}^n_{i+1/2},t^n) + O(h\,\Delta t^n) = 1 + O(\Delta t^n)\,. \tag{4.13}$$

Let us show that the last term in the right hand side of (4.12) is $O(h^2)$. Using equation (4.7) and the fact that exact mean value $\overline{u}([\tilde{x}^n_i,\tilde{x}^n_{i+1}],t^n)$ differs from the function value at the middle point $\tilde{x}^n_{i+1/2}$ by $O(h^2)$, we can prove that

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

This equation, the previously used inequality, $\varepsilon \Delta t^n < (\tilde{h}^n_{i+1/2})^2/2$, and estimate (4.13) imply that the ε-term in (4.12) is $O(h^2)$, i.e.

$$\frac{\varepsilon \Delta t^n}{h^{n+1}_{i+1/2}} \left(\left[\frac{\delta \overline{\tilde{u}}^n}{\delta x} \right]_{i+1} - \left[\frac{\delta \overline{\tilde{u}}^n}{\delta x} \right]_{i} \right) = O(h^2). \tag{4.14}$$

Using estimate (4.13) again, the first term in the right hand side of (4.12) can be estimated as follows

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

Let us analyze now the second term in the right hand side of (4.12). Using our assumptions about grid smoothness and accuracy of remapping, one can prove that

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

From (4.14), (4.15), and (4.16) we can conclude that

$$\bar{u}^{n+1}_{i+1/2} = \overline{\tilde{u}}^n_{i+1/2} + O(h^2 + h \, \Delta t^n) \, . \tag{4.17}$$

Finally, definition (4.8) and equations (4.11), (4.17) imply that the local error equals:

$$\begin{aligned} e_{i+1/2}(x(\xi, t^{n+1}), t^{n+1}) &= u(x(\xi, t^{n+1}), t^{n+1}) - \overline{u}^{n+1}_{i+1/2} \\ &= u(x(\xi, t^n), t^n) - \bar{u}[\tilde{x}^n_i, \, \tilde{x}^n_{i+1}] + O(h^2 + (\Delta t^n)^2 + \Delta t^n \, h). \end{aligned} \tag{4.18}$$

The formula (4.18) shows that the leading term in the error *does not* depend on Δt^n; that is, the *assumption* we have made in the beginning of Section 4 is valid. Summing up the local errors and using (4.13), we get

$$\begin{aligned} \sum_{i=0}^{M} \int_{x_i^{n+1}}^{x_{i+1}^{n+1}} \left| e_{i+1/2}(x, t^{n+1}) \right|^2 \, \mathrm{d}x &= \sum_{i=0}^{M} \int_{x_i^{n+1}}^{x_{i+1}^{n+1}} (u(x, \, t^{n+1}) - \bar{u}^{n+1}_{i+1/2})^2 \, \mathrm{d}x \\ &\approx \sum_{i=0}^{M} \int_{\tilde{x}_i^{n}}^{\tilde{x}_{i+1}^{n}} \left(u(x, t^n) - \bar{u}([\tilde{x}^n_i, \, \tilde{x}^n_{i+1}], t^n) \right)^2 \, \mathrm{d}x. \end{aligned} \tag{4.19}$$

To summarize, we have shown that under reasonable assumptions about grid smoothness, accuracy of interpolation, and time step Δt^n, the leading term in the error at time level t^{n+1} depends on accuracy of representation of function $u(x, t)$ at time level t^n by its exact mean values. This is nothing else but the standard interpolation problem of the best piecewise constant fit with adjustable nodes. In the next Section we shall consider this interpolation problem in more details and develop different techniques for its solution, which will be used later in implementation of our rezone strategy.

4.2.2 The best piecewise constant fit with adjustable nodes

Problem formulation

Let $v(x)$ be a given sufficiently smooth function over the unit interval [0, 1] and $\{\bar{v}_{i+1/2}\}$ be a discrete piecewise constant function on grid $\{x_i\}$,

$$0 = x_0 < x_1 < \ldots < x_{M+1} = 1.$$

We shall refer to the grid $\{x_i\}$ as the initial grid. We assume that $\bar{v}_{i+1/2}$ is the exact mean value of function $v(x)$ on interval $[x_i, x_{i+1}]$,

$$\bar{v}_{i+1/2} = \bar{v}[x_i,\, x_{i+1}]\,,$$

and that these mean values are the only information we know about the continuous function $v(x)$. This corresponds to assumptions made in Section 4.2.1 about the discrete function $\{\bar{u}^n_{i+1/2}\}$. The error functional defined by (4.19) results in problem of the best piecewise constant fit with adjustable nodes: Find positions of mesh nodes that minimize the L_2 norm of the error between function $v(x)$ and its piecewise constant representation $\{\bar{v}[\tilde{x}_i,\, \tilde{x}_{i+1}]\}$, i.e.

$$\min_{\tilde{x}_1,\ldots,\tilde{x}_M} \Phi_{ex}(\{\tilde{x}_i\}), \qquad \Phi_{ex}(\{\tilde{x}_i\}) = \sum_{i=0}^{M} \int\limits_{\tilde{x}_i}^{\tilde{x}_{i+1}} \left(v(x) - \bar{v}[\tilde{x}_i,\, \tilde{x}_{i+1}]\right)^2 \, \mathrm{d}x\,. \tag{4.20}$$

Remark 4.1 *In the context of this paper, the functional Φ_{ex} may be useful when the initial condition (2.2) for Burgers' equation is given analytically. The initial grid may be built by minimizing functional Φ_{ex} with $v(x) = U(x)$. The minimization problem (4.20) can be efficiently solved by methods from [67].*

We refer reader to Chapter 9 in book [8], and paper [7] for a more detailed exposition and references related to this problem.

Since, both the continuous function $v(x)$ and the exact mean values $\bar{v}[\tilde{x}_i,\, \tilde{x}_{i+1}]$ are not available for $t^n > t^0$, the functional $\Phi_{ex}(\{\tilde{x}_i\})$ has to be estimated. This will require data interpolation from grid $\{x_i\}$ to grid $\{\tilde{x}_i\}$. The assumptions made in the process of deriving the error functional (4.19) require that this interpolation should be exact for linear functions and conservative.

The conservative second-order accurate interpolation is based on a piecewise linear reconstruction on the grid $\{x_i\}$ followed by exact integration of the reconstructed function over cells of grid $\{\tilde{x}_i\}$ (see, e.g., [49]). A piecewise linear reconstruction $\tilde{v}(x)$, which we use in this paper, has the following form:

$$\tilde{v}(x) = \bar{v}_{i+1/2} + s_{i+1/2}\,(x - x_{i+1/2}), \qquad x \in [x_i,\, x_{i+1}],$$

where $\bar{v}_{i+1/2}$ are the mean values given on the initial grid, $x_{i+1/2} = (x_i + x_{i+1})/2$, and slopes $s_{i+1/2}$ are defined using the minmod limiter (see, e.g., [50]).

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

$$\int_0^1 \tilde{v}(x)\, dx = \sum_{i=0}^{M} \bar{v}_{i+1/2}\, h_{i+1/2}\,.$$

Using this reconstruction, we define the approximation, $\bar{\bar{v}}_{i+1/2}$, for $\bar{v}[\tilde{x}_i,\, \tilde{x}_{i+1}]$ by computing the mean value of the reconstructed function $\tilde{v}(x)$ over interval $[\tilde{x}_i,\, \tilde{x}_{i+1}]$:

$$\bar{\bar{v}}_{i+1/2} = \frac{1}{\tilde{h}_{i+1/2}} \int\limits_{\tilde{x}_i}^{\tilde{x}_{i+1}} \tilde{v}(x)\ \mathrm{d}x\,. \tag{4.21}$$

The conservative property of the reconstruction gives

$$\sum_{i=0}^{M} \bar{\bar{v}}_{i+1/2}\, \tilde{h}_{i+1/2} \ = \sum_{i=0}^{M} \bar{v}_{i+1/2}\, h_{i+1/2}\,.$$

Since we know how to interpolate data, let us take a more closer look at the functional Φ_{ex}. Using the Taylor expansion in (4.20) with the Lagrange remainder, it is easy to show that the minimization problem is equivalent to the following one:

$$\min_{\tilde{x}_1,\dots,\tilde{x}_M} \Phi_{ex}(\{\tilde{x}_i\}), \qquad \Phi_{ex}(\{\tilde{x}_i\}) = \frac{1}{12}\sum_{i=0}^{M}\left(\left.\frac{\partial v}{\partial x}\right|_{\tilde{x}^*_{i+1/2}}\right)^2 \tilde{h}^3_{i+1/2} \tag{4.22}$$

where $\tilde{x}^*_{i+1/2}$ is a point from interval the $(\tilde{x}_i,\, \tilde{x}_{i+1})$.

Since the point $\tilde{x}^*_{i+1/2}$ is unknown, we have to evaluate the solution gradient using, for example, methods of *a posteriori* error estimates [1, 81]. In 1D, these estimates are quite simple. First, we approximate $\partial v/\partial x$ on the initial grid by a piecewise constant function $\{[\delta v/\delta x]_{i+1/2}\}$. Second, we interpolate this discrete function onto the current grid $\{\tilde{x}_i\}$ using the interpolation operator defined by (4.21). Formula (4.22) implies that in our estimates, we can use any first-order approximation of $\partial v/\partial x$ because it will produce high-order terms of the same order as the terms neglected in derivation of (4.19). However, in numerical experiments, we use the second order formula for the gradient of function v:

$$\left[\frac{\delta v}{\delta x}\right]_{i+1/2} = \frac{1}{\bar{h}_{i+1/2}}\left(\left[\frac{\delta v}{\delta x}\right]_{i+1}\left(\frac{h_{i+1/2}}{6}+\frac{h_{i-1/2}}{3}\right)+\left[\frac{\delta v}{\delta x}\right]_{i}\left(\frac{h_{i+1/2}}{6}+\frac{h_{i+3/2}}{3}\right)\right) \tag{4.23}$$

where

$$\bar{h}_{i+1/2} = \frac{1}{3}(h_{i-1/2}+h_{i+1/2}+h_{i+3/2})$$

and

$$\left[\frac{\delta v}{\delta x}\right]_{i} = \frac{\bar{v}_{i+1/2}-\bar{v}_{i-1/2}}{(h_{i+1/2}+h_{i-1/2})/2}.$$

The following algorithm describes a numerical method for solving the optimization problem (4.22).

Algorithm 1 *[Approximate optimization]*

1. *For the initial grid $\{x_i\}$, compute values $[\delta v/\delta x]_{i+1/2}$ and set $\{x_i^{(0)}\} = \{x_i\}$.*
2. *Choose a positive number K_{max}. Then, for $k = 1,\dots,K_{max}$ do*

   2a. Perform one Gauss-Seidel sweep

$$\min_{x_i^{(k+1)}}\left\{\left[\widehat{\frac{\delta v}{\delta x}}\right]^2_{i+1/2}(x^{(k)}_{i+1}-x^{(k+1)}_i)^3+\left[\widehat{\frac{\delta v}{\delta x}}\right]^2_{i-1/2}(x^{(k+1)}_{i}-x^{(k+1)}_{i-1})^3\right\},$$

where $i = 1, \ldots, M$ and $\{\widehat{[\delta v/\delta x]}_{i+1/2}\}$ is the conservative interpolant of $\{[\delta v/\delta x]_{i+1/2}\}$ from the initial grid to the intermediate grid

$$0 = x_0^{(k+1)} < \ldots < x_i^{(k+1)} < x_{i+1}^{(k)} < \ldots < x_{M+1}^{(k)} = 1.$$

2b. Stop iterations if

$$\max_i \frac{|x_i^{(k)} - x_i^{(k+1)}|}{x_{i+1}^{(k)} - x_{i-1}^{(k)}} \leq TOL$$

where TOL is the user given tolerance. Set $\{\tilde{x}_i\} = \{x_i^{(k+1)}\}$.

Numerical experiments

Here, we compare meshes and discrete functions produced by minimization of the exact functional (4.20) with ones obtain by using an approximate optimization (Algorithm 1).

Problem (4.20) can be solved using some modifications of nonlinear optimization methods from [67]. Modifications are required, because (4.20) is the constrained nonlinear optimization problem where constraints, $\tilde{x}_{i+1} > \tilde{x}_i$, express validity of the mesh. One can also use methods from [8]. In our numerical experiments, we use a high-order quadrature to compute integrals in (4.20) and the nonlinear Gauss-Seidel (coordinate descent) method to solve the problem.

Let us consider a test function $v(x)$ from [66], given by

$$v(x) = 1 - \frac{9\,r_1(x) + 5\,r_1(x)^5}{10(r_1(x) + r_1(x)^5 + r_2(x))}, \quad r_1(x) = \exp\frac{1/2 - x}{20\varepsilon}, \quad r_2(x) = \exp\frac{3/8 - x}{2\varepsilon}, \tag{4.24}$$

with $\varepsilon = 0.005$ (see Fig. 2).

The results of numerical experiments are shown in Table 1 and in Fig. 2, 3. In these experiments the initial mesh for approximate optimization algorithm is uniform. We use the L_2-norm of error

$$E(\{\tilde{x}_i\}) = \sqrt{\Phi_{ex}(\{\tilde{x}_i\})}$$

as the measure of grid quality. Let $\{\tilde{x}_i^{ex}\}$ be the grid that minimizes functional Φ_{ex}, $\{\tilde{x}_i^{Alg1}\}$ be the grid produced by Algorithm 1, and $\{x_i^{un}\}$ be the uniform grid. We shall use similar notations for the corresponding piecewise constant functions, for instance, $v^{ex}(x)$ denotes the best piecewise constant approximation of $v(x)$ on grid $\{\tilde{x}_i^{ex}\}$.

All three columns in Table 1 demonstrate the first order convergence rate for L_2 norm of the error. However, optimal distribution of mesh points gives us approximately 3 times smaller errors. We observe that meshes, and the corresponding discrete functions, produced by Algorithm 1 are as good as that of the exact minimization problem. For instance, the errors on the coarse grid with only 32 mesh intervals differ by 3.8%. The asymptotic convergence of errors to each other confirms efficiency of formula (4.23) in recovering the gradient of function $v(x)$.

The results of approximate and exact optimizations are further compared in Fig. 2. On the top, we plot function $v(x)$ and its piecewise constant approximations. On the bottom, we plot the mesh steps as function of x.

In order to emphasize the importance of adaptation, we compare solutions on adaptive grids given by Algorithm 1 with piecewise constant representation on a uniform grid (see Fig. 3). It is clear that the adaptive grid provides much better resolution of sharp features of the function.

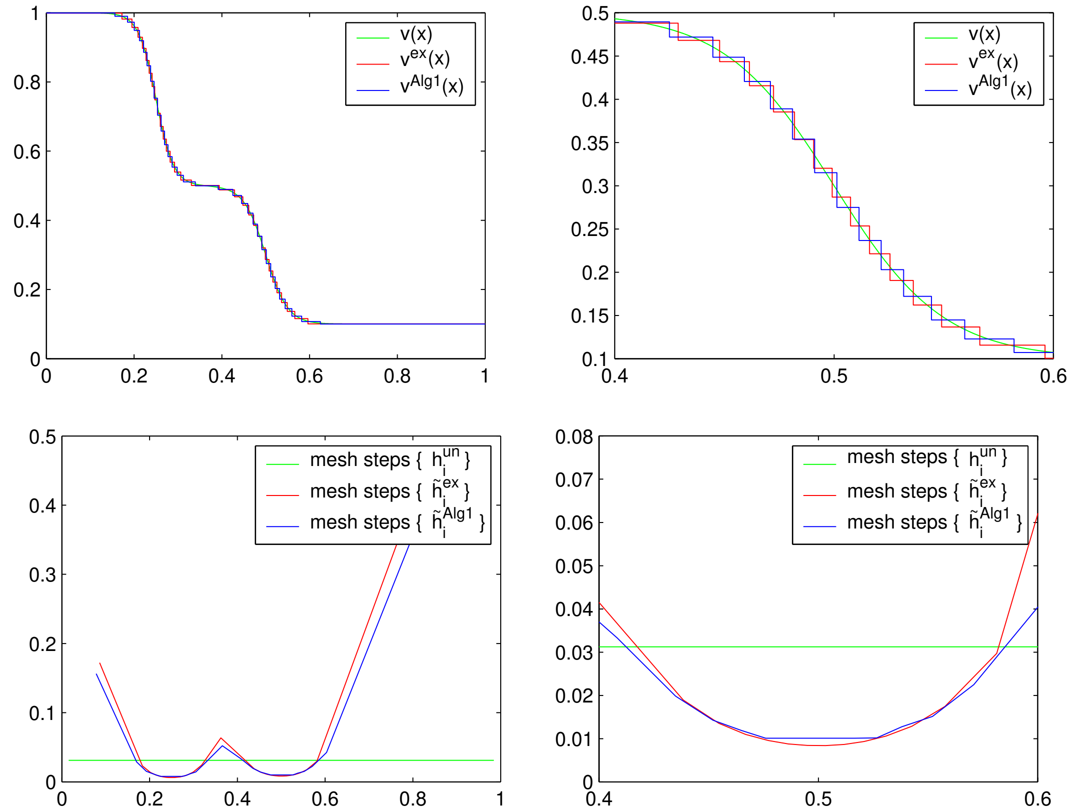


Figure 2: Results of approximate and exact optimizations for $M = 32$: functions (on the top) and mesh steps as functions of x (on the bottom).

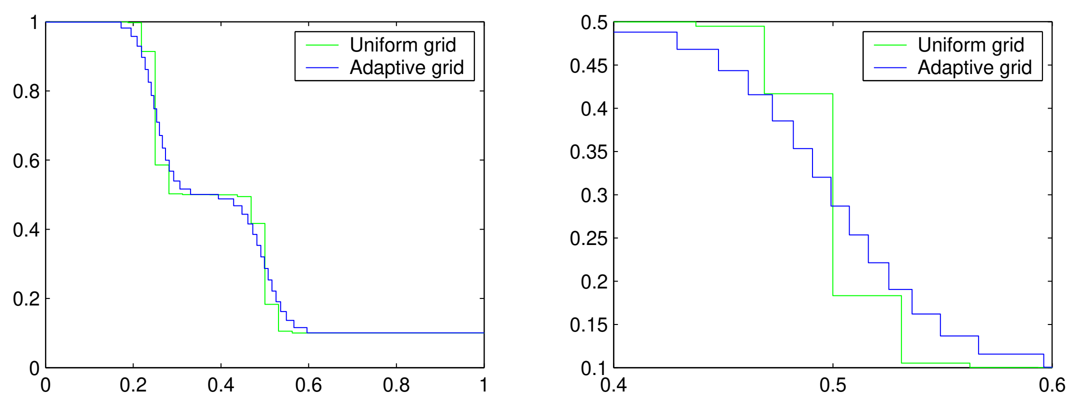


Figure 3: Piecewise constant approximations on uniform and adaptive grids for $M = 32$.

Table 1: Comparison of errors: $\{\tilde{x}_i^{ex}\}$ is the grid minimizing functional Φ_{ex}, $\{\tilde{x}_i^{Alg1}\}$ is the grid produced by Algorithm 1, and $\{x_i^{un}\}$ is the uniform grid.

M	$E(\{x_i^{un}\})$	$E(\{\tilde{x}_i^{ex}\})$	$E(\{\tilde{x}_i^{Alg1}\})$
16	2.99e-2	1.01e-2	1.19e-2
32	1.59e-2	4.99e-3	5.18e-3
64	7.99e-3	2.48e-3	2.50e-3
128	4.00e-3	1.24e-3	1.24e-3

4.2.3 Achieving mesh regularity

The algorithms described above produce grids that are the optimal from viewpoint of error minimization. However, these grids may not be appropriate for solving time dependent problems due to strong disproportionality of neighboring mesh cells, which may affect stability of the overall ALE method. Moreover, error analysis of Burgers' equation performed in Section 4.2.1 assumes that mesh is smooth. Therefore, we need a algorithm which allows us to produce meshes which are simultaneously smooth and adapted to a solution. Especially it is important to find a way to enforce mesh smoothness in the regions where the solution is constant, which implies that the error is zero for any mesh in these regions.

In this paper, we use the idea of guaranteed smoothing proposed in [32]. Let α be a positive number. We define a smooth grid as the grid whose mesh steps satisfy the following condition:

$$\frac{\alpha}{\alpha+1} \leq \frac{h_{i-1/2}}{h_{i+1/2}} \leq \frac{\alpha+1}{\alpha}, \qquad i = 1, \ldots, M. \tag{4.25}$$

One way to build a smooth adapted mesh is to apply algorithms from [32] to the grid $\{x_i^{Alg1}\}$. Another approach is to modify problem (4.22) such that its solution will be a smooth grid. 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 error functional and the faster convergence of optimization algorithms resulting thereby.

We begin with deriving the necessary conditions for extrema of functional (4.20). The straightforward differentiation of Φ_{ex} with respect to $\tilde{x}_i$ gives

$$\frac{\partial \Phi_{ex}}{\partial \tilde{x}_i} = 2\, v(\tilde{x}_i) - \bar{v}[\tilde{x}_{i-1},\, \tilde{x}_i] - \bar{v}[\tilde{x}_i,\, \tilde{x}_{i+1}] = 0. \tag{4.26}$$

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

$$\left.\frac{\partial v}{\partial x}\right|_{\tilde{x}_i} \left(\tilde{h}_{i+1/2} - \tilde{h}_{i-1/2}\right) + \frac{1}{3} \left.\frac{\partial^2 v}{\partial x^2}\right|_{\tilde{x}_i} \left(\tilde{h}^2_{i+1/2} + \tilde{h}^2_{i-1/2}\right) = 0. \tag{4.27}$$

Now we assume that the mesh satisfying (4.27) is described by a smooth function $\tilde{h}(x)$, i.e. $\tilde{h}_{i+1/2} = \tilde{h}(x_{i+1/2})$. Then

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

Substituting these estimates into (4.27), we get the following equation at point $\tilde{x}_i$:

$$\frac{2}{3}\frac{\partial}{\partial x}\left(\ln\frac{\partial v}{\partial x}\right) = -\frac{\partial}{\partial x}\left(\ln\tilde{h}(x)\right). \tag{4.28}$$

Equation (4.28) means that the mesh $\{\tilde{x}_i\}$, minimizing functional Φ_{ex} satisfies the following condition approximately:

$$\left(\left.\frac{\partial v}{\partial x}\right|_{\tilde{x}_{i+1/2}}\right)^{2/3}\tilde{h}_{i+1/2} = const.$$

In other words, in order to build a grid satisfying (4.25), we may to apply algorithms of guaranteed smoothing to a piecewise constant function $\{[\delta v/\delta x]^{2/3}_{i+1/2}\}$. Let $\{\widetilde{[\delta v/\delta x]}_{i+1/2}\}$ be the smoothed discrete function (see Algorithm 2 for details). Then the modified minimization problem reads:

$$\min_{\tilde{x}_1,\ldots,\tilde{x}_M}\Phi_{sm}(\{\tilde{x}_i\}), \qquad \tilde{\Phi}_{sm}(\{\tilde{x}_i\}) = \sum_{i=0}^{M}\left[\widetilde{\frac{\delta v}{\delta x}}\right]^2_{i+1/2}\tilde{h}^3_{i+1/2}. \tag{4.29}$$

Guaranteed smoothing procedure follows original ideas of Dorfi and Drury [32] and is based on the following proposition.

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

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

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

$$\tilde{\omega}_{i+1/2} - \alpha(\alpha+1)(\tilde{\omega}_{i+3/2} - 2\tilde{\omega}_{i+1/2} + \tilde{\omega}_{i-1/2}) = \omega_{i+1/2}, \tag{4.30}$$

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

It is clear that the left hand side of equation (4.30) can be written as:

$$\left(I - (\Delta\xi)^2\,\alpha\,(\alpha+1)\,\frac{\Delta^2}{\Delta\xi^2}\right)\tilde{\omega}\,, \quad \left(\frac{\Delta^2\tilde{\omega}}{\Delta\xi^2}\right)_{i+1/2} = \frac{\tilde{\omega}_{i+3/2} - 2\tilde{\omega}_{i+1/2} + \tilde{\omega}_{i-1/2}}{(\Delta\xi)^2}\,.$$

Since $\tilde{\omega}$ is smoother than ω, $\left(\frac{\Delta^2\tilde{\omega}}{\Delta\xi^2}\right)$ is a bounded function and the second term in the left expression is $O\left((\Delta\xi)^2\right)$. For the same reason, the whole operator acting on $\tilde{\omega}$ is close to the identity operator. This consideration gives a simple explanation of why is equation (4.30) indeed produces $\tilde{\omega}$ which is smoother than original ω but still preserves features of ω.

The proof of Proposition 4.1 is presented in [58] and is similar to one presented in [87]. The proposition provides a simple way for merging the functional minimization with the grid smoothing.

Algorithm 2 *[Smooth Optimization]*

1. Set $\{x_i^{(0)}\} = \{x_i\}$ *and compute numbers* $\omega_{i+1/2}^{(0)} = [\delta v/\delta x]_{i+1/2}^{2/3}$ *using formula (4.23).*

2. Choose a positive K_{max}*. Then, for* $k = 1, \ldots, K_{max}$ *do*

2a. For the current grid $\{x_i^{(k)}\}$*, interpolate piecewise constant function* $\{\omega_{i+1/2}^{(0)}\}$ *from the initial grid. Let* $\{\omega_{i+1/2}\}$ *be the interpolated discrete function.*

2b. Compute the smoothed function $\{\tilde{\omega}_{i+1/2}\}$ *by solving system (4.30) and set*

$$\left[\widetilde{\frac{\delta v}{\delta x}}\right]_{i+1/2} = \tilde{\omega}_{i+1/2}^{3/2}, \qquad i = 0, \ldots, M.$$

2c. Perform one Gauss-Seidel sweep

$$\min_{x_i^{k+1}} \left\{ \widehat{\omega}_{i+1/2}^3 \, (x_{i+1}^{(k)} - x_i^{(k+1)})^3 \, + \, \widehat{\omega}_{i-1/2}^3 \, (x_i^{(k+1)} - x_{i-1}^{(k+1)})^3 \right\},$$

where $i = 1, \ldots, M$ *and* $\{\widehat{\omega}_{i+1/2}\}$ *is the conservative interpolant of* $\{\tilde{\omega}_{i+1/2}\}$ *from grid* $\{x_i^{(k)}\}$ *to the intermediate grid*

$$0 = x_0^{(k+1)} < \ldots < x_i^{(k+1)} < x_{i+1}^{(k)} < \ldots < x_{M+1}^{(k)} = 1.$$

2d. Stop iterations if

$$\max_i \frac{|x_i^{(k)} - x_i^{(k+1)}|}{x_{i+1}^{(k)} - x_{i-1}^{(k)}} \leq TOL$$

where TOL *is the user given tolerance. Set* $\{\tilde{x}_i\} = \{x_i^{(k+1)}\}$.

The principle difference between Algorithms 1 and 2 is that the smoothing has to be performed on each loop of k-cycle. If we smooth the discrete function $\{\omega_{i+1/2}^{(0)}\}$ and use the result inside the Gauss-Seidel sweep instead of $\{\tilde{\omega}_{i+1/2}\}$, the final grid will not satisfy condition (4.25).

In order to study the effect of smoothing, we consider the test function (4.24) again. The results of numerical experiments are shown in Table 2 and in Fig. 4. We use superscript $Alg2$ for grids generated by Algorithm 2.

Note that the loss in the accuracy of the solution is about 3% on the finest grid. In Fig. 4, we compare results of the exact minimization problem (4.20) with similar results of the approximate problem (4.29). The neighboring mesh steps satisfy inequality (4.25), with $\alpha = 1$. Observe that coefficients of the smoothed functional preserve the main features of the original functional.

Table 2: Accuracy analysis of Algorithm 2.

M	$E(\{\tilde{x}_i^{ex}\})$	$E(\{\tilde{x}_i^{Alg1}\})$	$E(\{\tilde{x}_i^{Alg2}\})$
16	1.01e-2	1.19e-2	1.75e-2
32	4.99e-3	5.18e-3	6.28e-3
64	2.48e-3	2.50e-3	2.70e-3
128	1.24e-3	1.24e-3	1.28e-3

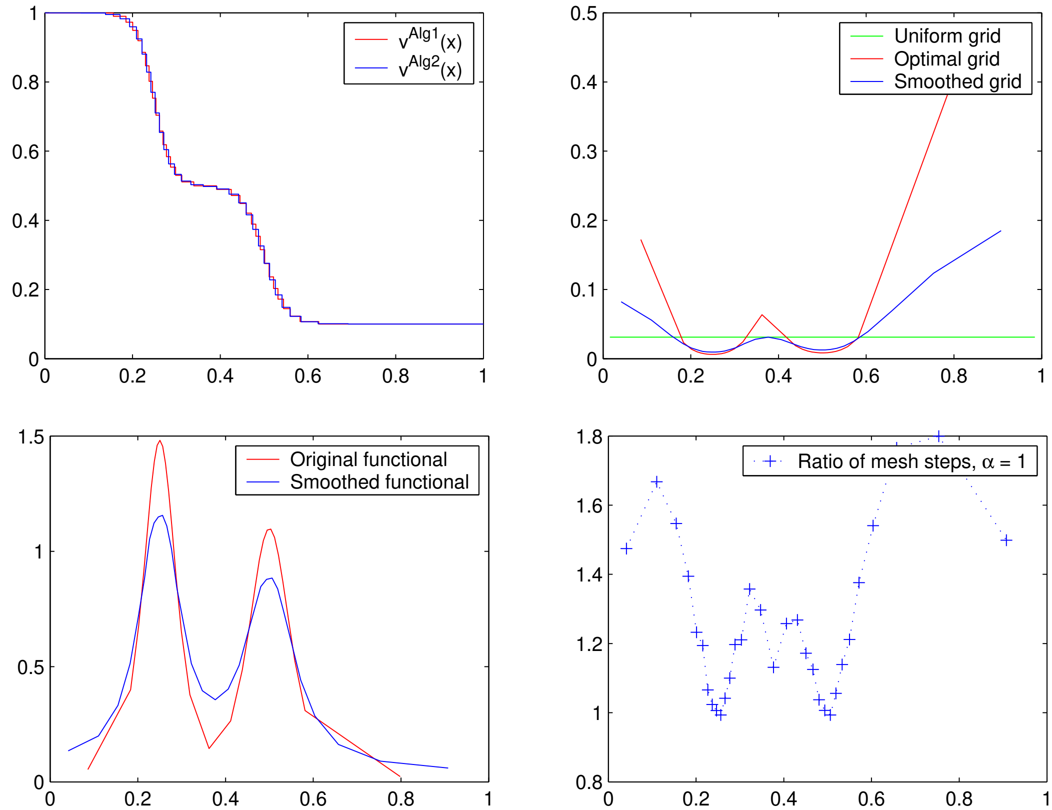


Figure 4: Comparison of optimization algorithms with and without smoothing for $M = 32$ and $\alpha = 1$: solutions (top-left), mesh steps as functions of x (top-right), coefficients $[\delta v/\delta x]^{2/3}_{i+1/2}$ and $[\widetilde{\delta v/\delta x}]^{2/3}_{i+1/2}$ of the functionals Φ_{ex} and Φ_{sm}, respectively (bottom-left), and ratio of neighboring mesh steps in the smooth grid (bottom-right).

Let us show that Smooth Optimization algorithm does not produce grids with very small space intervals. Later, this will guarantee that the time step in the numerical solution of Burgers' equation is bounded from below. Condition (4.25) allows us to derive an estimate for the minimal mesh step $\tilde{h}_{i+1/2}$. Let us consider the extreme case when the grid is geometrically refined to $x = 1$ in such a way that

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

This immediately implies that the minimal mesh step is given by

$$\tilde{h}_{M+1/2} = \frac{\alpha^M}{(1+\alpha)^{M+1} - \alpha^{M+1}}\,.$$

4.3 Summary of EMB Rezone strategy

At initial time moment, mesh $\{x_i^0\}$ is chosen to minimize functional Φ_{ex} with $v(x) = U(x)$ where $U(x) = u(x, 0)$ defines the initial condition and is usually given analytically. Mesh $\{\tilde{x}_i^0\}$ is obtained by using Smooth Optimization Algorithm 2 described in the previous section.

For $n \geq 1$, mesh $\{x_i^n\}$ is the result of Lagrangian step (4.1) while mesh $\{\tilde{x}_i^n\}$ is again obtained with the Smooth Optimization algorithm.

5 Numerical experiments for Burgers' equation

In this section we describe numerical results for viscous Burgers' equation obtained by the pure Lagrangian method, an ALE method with the RJ rezone strategy, and an ALE method with the EMB rezone strategy. We set $K_{max} = 500\,M$, $TOL = 10^{-3}$, and $\alpha = 1$ for the EMB ALE method. In all calculations, we use a time step which is half the size recommended by stability condition (3.3).

The viscous Burgers' equation has an exact solution when the initial condition $U(x)$ is given by function (4.24). The exact solution (see, e.g., [66]) is

$$\begin{aligned}
u(x,t) &= 1 - \frac{0.9\,r_1(x,t) + 0.5\,r_2(x,t)}{r_1(x,t) + r_2(x,t) + r_3(x,t)}\,, \\
r_1(x,t) &= \exp\left(-\frac{x-0.5}{20\,\varepsilon} - \frac{99\,t}{400\,\varepsilon}\right), \\
r_2(x,t) &= \exp\left(-\frac{x-0.5}{4\,\varepsilon} - \frac{3\,t}{16\,\varepsilon}\right), \quad r_3(x,t) = \exp\left(-\frac{x-3/8}{2\,\varepsilon}\right).
\end{aligned} \tag{5.31}$$

The exact solution for different time moments is presented in Fig. 5. Initially it consist

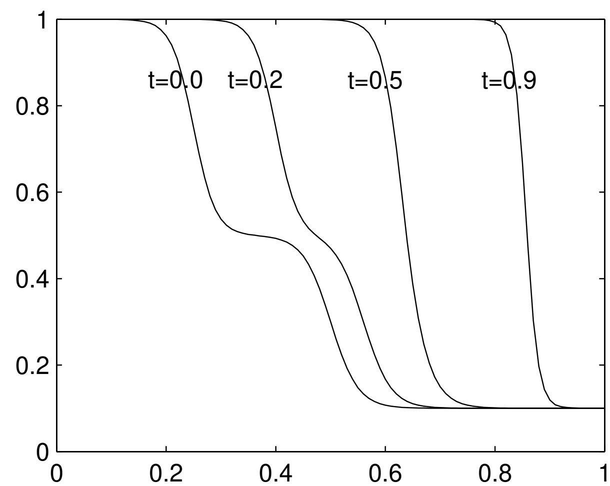


Figure 5: Viscous Burgers' equation with $\varepsilon = 0.005$. Exact solution for different t.

of two shock-like structures, which then move with different speeds and finally merge at $t \approx 0.5$.

The trajectories of mesh nodes for the pure Lagrangian method are presented in Fig. 6a). Horizontal slices represent the mesh at the corresponding time moments. As one can see from this picture, the node get progressively closer to each other, that is, the size of each cell diminishes. In regions where solution has big negative gradient, the size of the mesh cells diminishes faster. 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 confirms statement from [79] (Section 37.2.2) that Method of

Characteristics (MoC) is not well-suited for general hyperbolic partial differential equations (PDE's) and in particular for Burgers' equation. From a practical point of view, constantly diminishing size of the mesh cells means that time step is goes to zero according to the stability condition and becomes so small that the calculation practically stalls.

In Fig. 6b) we present trajectories and numerical solution corresponding to ALE method with the RJ rezone strategy. It is clear that trajectories roughly follow the Lagrangian trajectories. The mesh is smooth and does not have problems of pure Lagrangian mesh; however, main features of the solution are under-resolved.

In Fig. 6c) we present trajectories and numerical solution corresponding to ALE method with the EMB rezone strategy. At each time moment, the mesh is smooth and resolves features of the solution.

In Fig. 7 we plot the error

$$E(\{x_i^N\}) = \left[\sum_{i=0}^{M} \int_{x_i^N}^{x_{i+1}^N} (u(x,T) - \bar{u}_{i+1/2}^N)^2 \, \mathrm{d}x \right]^{1/2} .$$

Fig. 7a) demonstrates error behavior for the pure Lagrangian, RJ ALE, and EMB ALE simulations as function of time for the case $M = 32$. Pure Lagrangian method is accurate up to about $t = 0.1$. After this moment, dynamics of the cells do not correspond to dynamics of features and accuracy starts to degrade. For RJ ALE method, the error first starts to grow because the RJ method makes the mesh smoother compared to the original mesh and by doing this, it reduces resolution of features. Then, the error stays approximately constant. After $t = 0.3$ the accuracy of RJ ALE is significantly better then accuracy of pure Lagrangian method. For EMB ALE method, the error is about the same as for pure Lagrangian method for $t \leq 0.15$, but after this moment, it is significantly less as the error for both pure Lagrangian and RJ ALE methods.

The Fig 7b) demonstrates the linear convergence of L_2-norm of the error with respect to the number, M, of mesh steps both for RJ ALE and EMB ALE methods. However, for the given accuracy, e.g. $2 \cdot 10^{-3}$, we need about 2 times less mesh points for EMB ALE method then for RJ ALE method. This difference grows when ε decreases and the solution profile becomes more sharp. For $\varepsilon = 0.002$ (see Fig. 8) and the given accuracy, e.g. $2 \cdot 10^{-3}$, we need about 3 times less mesh points for EMB ALE method then for RJ ALE method.

6 Related Methods, Conclusion and Future Plans

Note that there are many rezone strategies whose primary goal is to maintain geometrical quality of the mesh rather than a direct error minimization. One of the typical examples can be found in [60, Ch. 12, Sec. 12.3] (where author talks about interaction of flow with structures) in which, smoothing of the nodal velocity field is achieved by solving a Laplace equation with diffusion coefficient depending on distance from moving bodies. Another example can be found in recent paper [42] where the mesh moves with velocity equal $\beta \vec{U}_L$ where β is a parameter and $\vec{U}_L$ is fluid velocity (Lagrangian velocity). Review of similar rezone strategies for ALE methods is given in [47], and we will not repeat it here. We

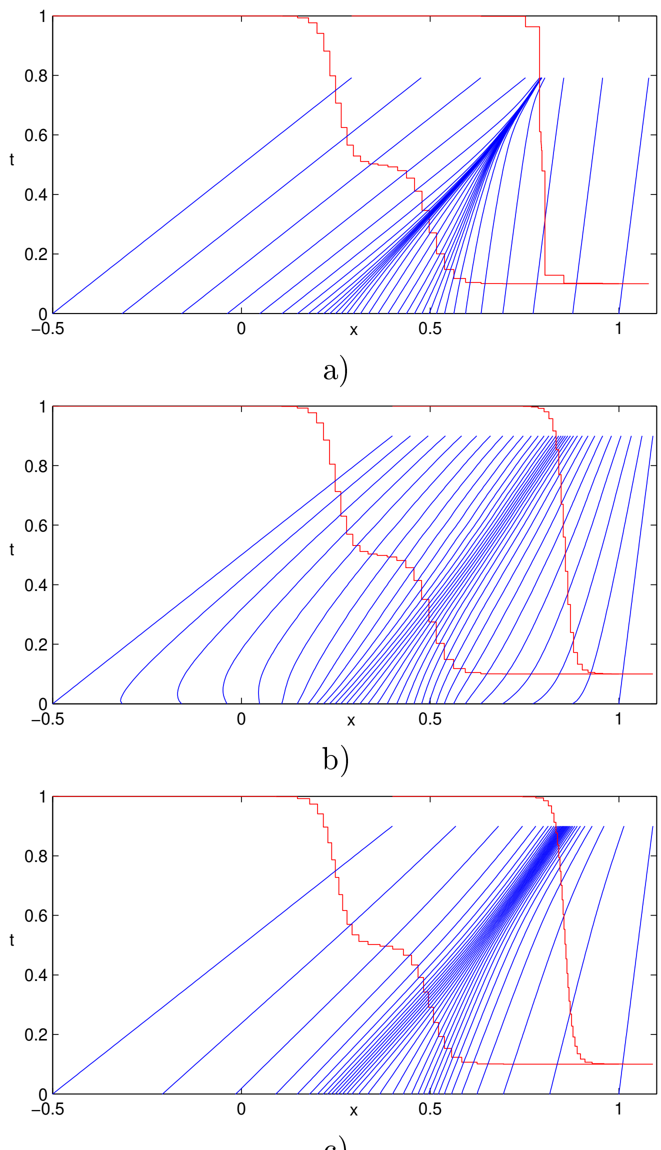
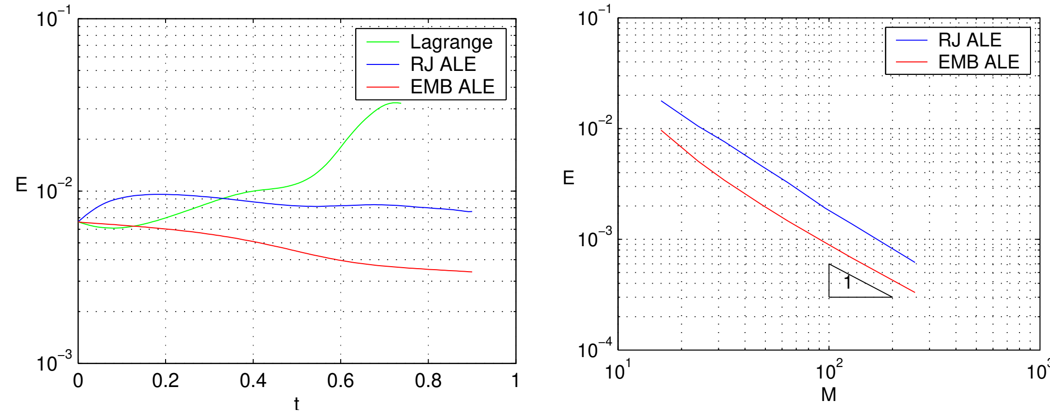


Figure 6: Viscous Burgers' equation with $\varepsilon = 0.005$. Trajectories of mesh points and discrete solutions at initial and final time moments: a) pure Lagrangian method; b) ALE method with RJ rezone strategy; c) ALE method with EMB rezone strategy.



a) b)

Figure 7: Viscous Burgers' equation with $\varepsilon = 0.005$. The picture on the left shows accuracy of the pure Lagrangian, RJ ALE, and EMB ALE simulations versus the time for the case $M = 32$. The picture on the right shows convergence of the RJ ALE and EMB ALE simulations for final time moment $t^N = T$ versus the number of mesh points, M.

only want to add one more historically important reference, [86], in which one of the first moving grid methods is described. The authors' objective is similar to one described in Section 4.1 for the RJ ALE method: movement with flow preventing grid distortion.

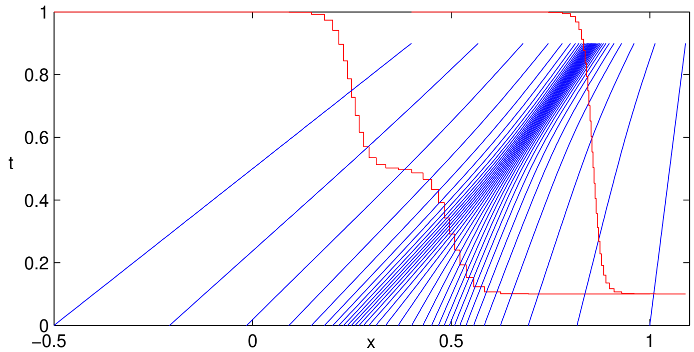
6.1 Related Methods

According to [60, Ch. 14, p. 261], any adaptive scheme is composed of three main *ingredients*: an optimal-mesh criterion, an error indicator, and an algorithm or strategy for mesh improvement. These ingredients answer to the following *questions*: How should the optimal mesh be defined, where is mesh change required, and how should the improved mesh be constructed?

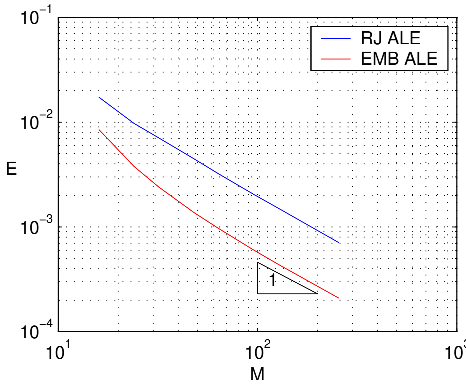
Arbitrary Lagrangian-Eulerian methods described in this paper can be considered as members of much wider class of so-called *moving mesh* methods, or *r-refinement (r-adaptation)* methods (where "r" stands for relocation, redistribution, or repositioning). For this class of methods, a strategy for mesh improvement is based on moving the spatial grid continuously and automatically in space-time domain. Moving-grid techniques use the fixed number of grid points, fixed connectivity and let grid points dynamically move with underlying features of PDEs. Extensive review of this class of methods can be found in [38, 10, 9, 76, 63, 21], [79, Ch. 33-37 in Part IV], [25, Ch. 7,8], [59, Ch. 7,8,10], [36, 2], and [13, Ch. 7.10].

Optimal-mesh criteria. It is clear that the optimal mesh can be defined as a 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 to mesh adaptation and movement in 1D is given by the equidistribution principle (see, e.g., [29, 71]). The goal is to find a grid



a)



b)

Figure 8: EMB ALE simulations of viscous Burgers' equation with $\varepsilon = 0.002$: trajectories of mesh points and discrete solutions at times $t = 0$ and $t = T$ (left), convergence rate (right).

on which error is uniformly distributed in space. There are a lot of papers which use some form of this concept. The list includes moving mesh partial differential equations (MMPDE) [40], method based on geometric conservation law (GCL) [20], deformation method [74, 15, 56, 57] and [59, Sec. 7.3.4 and 7.5.3], methods based on harmonic maps [52, 53, 78, 5, 4, 6], etc.

Another important concept in r-adaptivity is the moving finite element methods (MFE), [64, 26, 27, 8], where movement of nodal points is driven by the residual of the finite element approximation. A simplified adaptive mesh technique based on MFE is presented in [33].

An interesting idea is introduced in [7, 8, 9] and is termed as Moving Best Fit (MBF) in [40]. The goal of MBF approach is to find a mesh which minimizes L_2 norm of error of representation of a continuous function by either a piecewise constant or piecewise linear function. In our paper we have shown that in the case of Burgers' equation, the leading term in the error does, in fact, depend on the accuracy of a piecewise constant representation.

A very different approach is introduced in [43, 70], where one of the goals is to find trajectories for mesh nodes along which the time rate of the solution change is minimized. In [44] mesh velocities are derived so that solution at the mesh points (in reference frame moving with mesh) varies slowly. It is done by combining two strategies: one reduces a time variation and the other reduces a spatial variation. The weights are chosen to optimize the total truncation error. An approach with a similar goal — to find a mesh on which the nearly singular physical solution is better behaved – is presented in [28].

There are many other specific approaches for hyperbolic conservation laws. For instance, in [37] authors suggest to determine grid motion by averaging local characteristic velocities of the equations with respect to amplitude of signals. This approach requires solution of local Riemann problems and, in particular, if the solution is a single shock or a contact discontinuity, the mesh algorithm automatically places one mesh point on the path of this discontinuity enabling its perfect resolution.

Error indicators and error estimators. Error indicators and error estimators are the basis by which one can decide where mesh change is required. Because the exact solution of PDEs and therefore the corresponding error are not known, one needs to use computable error estimators or error indicators. These error indicators are used by mesh improvement strategies. According to [60], the most popular error indicators presently used in production codes may be grouped into the following categories: jump in the monitored variable (such as density or entropy); interpolation theory, which allows estimations of error by comparing approximations of different order; comparison of significant derivatives using schemes of different order; residual of PDEs over adjacent cells; more physics oriented indicators such as energy norms; etc. A well-developed theory of *a posteriori* error estimates does exist for elliptic equations (see, e.g., [1, 81]); however that is not the case for hyperbolic equations. In most moving mesh algorithms, 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 $u(x,t)$, $M(x,t) = \sqrt{1 + \alpha\,|\mathrm{grad}\, u(x,t)|^p}$. Here $\mathrm{grad}\, u$ plays the role of error indicator, α controls the adaptivity, p is responsible for some smoothing and 1 is added to prevent possible degenerate behavior in regions where function is constant and gradient is zero. Any reasonable error indicator can be put in place of the gradient.

In most methods based on some form of equidistribution, the equidistribution principle is applied to a 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 [18, 19, 41]. Some discussion of how to choose monitor function for hyperbolic conservation laws is presented in [77]. In particular, the authors address the question: How to create monitor function for systems of PDEs which will be sensitive to both shocks and contact discontinuities.

An interesting approach is suggested in [35] where the notion of *adaptive metric* of space, which in turn defines notion of distance, is introduced. The metric is constructed using some error indicator. Then, this metric is used to create a *unit mesh* where all edges have unit lengths with respect to the introduced metric. This approach is usually used for generating adaptive triangular and tetrahedral meshes and the adaptation strategy combines elements of h-adaptation with some elements of r-adaptation.

Algorithms to improve the mesh. As we have mentioned earlier in this paper, we are interested in methods where mesh improvement is achieved by mesh movement without changing number of points and connectivity (r-adaptation). Moving mesh methods can be divided in two main groups with respect to how mesh movement is incorporated into the overall solution algorithm. In the first group of methods (e.g., MFE, MMPDEs), a mesh movement equation is coupled with governing physical PDEs. This coupling leads to large system of nonlinear equations even if the original PDEs were linear. Moreover, such coupling introduces new terms in governing PDEs which require development of new discretizations and new analysis (e.g., stability) of overall nonlinear system. On the other hand, this class of method does not require any explicit interpolation of data between grids. In the second group of methods, the mesh movement is decoupled from governing PDEs. It means that spatial and temporal discretization of governing physical PDEs is done using standard methods and no new developments are required. When a new time step is completed, the obtained numerical solution is used in the mesh movement algorithm to produce a grid which is adapted to the solution. After that the solution is interpolated to the adapted grid and time integration is repeated. In ALE framework, the first group of methods is related to methods where equations are written in a moving coordinate system which introduces additional terms in governing equations [31, 45], while the second group of methods corresponds to methods where Lagrangian, rezone and remap phases are separated [39, 61].

This classification of moving mesh methods roughly follows the one presented at T. Tang's web-page (lsec.cc.ac.cn/ ~ttang/MMref) where he propose to distinguish interpolation-free and interpolation-based methods. Examples of interpolation-free methods are the classical 1D method by Dorfi and Drury [32], method from [3], previously mentioned MMPDE family of methods which includes also methods described in [12, 11, 77], deformation method, MFE methods, methods described in [70, 54, 55], method from [69], methods from [43, 44, 37] and many other methods. For all these methods, the governing PDEs are written in a moving coordinate frame that introduces additional terms in the original equations depending on mesh velocity. The variety of methods comes from a choice of a mesh movement equation (short review of mesh movement equations will be presented later in this section) and a solution algorithm. A short review of algorithms where method of lines (MOL) is used to solve the coupled system can be found in [79, Ch. 37] and [84, Ch. 4,5 and 8]. In MOL, the PDEs are discretized in space yielding a large (stiff) system of ordinary differential equations (ODEs). Time-integration of these ODEs is performed by using some ODE integrators. In particular, in [79, Ch. 37], the author describes results obtained with ODE integrators from [70]. Other authors (see, e.g., [77]) believe that there is no reason to solve the physical PDE and mesh equations with the same spatial discretization or to the same level of accuracy, because they are equations of different type (e.g., hyperbolic and parabolic equations in the MMPDE method for hyperbolic conservation laws) and have fundamentally different interpretations (one is physical while the other is artificial). For this reason, they suggest a semi-implicit scheme which behaves as an explicit predictor-corrector method for the coupled system. Similarly, the authors of [12] suggest to integrate forward in time in an iterative manner solving alternately for the grid and physical solution.

Examples of interpolation-based methods (PDEs and mesh movement equations are decoupled) can be found in [83, 17, 48, 52, 53, 78, 5, 4, 6, 73]. Because the space-time discretization of governing physical PDEs is standard, the methods differ by a choice of a mesh movement equation and an interpolation algorithm. It is clear that the interpolation procedure has to be conservative to maintain conservation property of overall method. This is the obvious requirement for methods solving hyperbolic conservation laws; however, in context of general moving mesh methods it is still considered as a "new" idea (see, e.g., introduction in [78]). The conservation property is also important for other types of governing equations, because interpolation is just a way to transfer data from one grid to another. In simple language, conservation means preservation of some norm of a discrete function.

Strategies for mesh movement. One can distinguish methods depending on how new positions of the mesh points are found. In the first group of methods, these positions are defined directly, while in the second group of methods nodal velocities are defined to update the positions. Such classification of rezone strategies for ALE methods is used in [47] and applied to moving mesh methods in [21]. Velocity-based approaches are usually used in methods where mesh movement is coupled with the governing PDEs and location-based approaches are used in methods where mesh movement is decoupled from the governing PDEs.

Typical velocity-based approaches are: the classical Lagrangian method (see, e.g., [75, Ch. 5], [22]), the moving finite-element method [64, 26, 27, 8], methods based on geometric conservation law (GCL) [20]; deformation methods [74, 15, 56, 57], [59, Sec. 7.3.4, 7.5.3]; the method presented in the seminal paper [32] and called the moving finite-difference (MFD) method in [79, Ch. 37.3.1]; MMPDE family of methods [40]; etc.

In classical Lagrangian methods, the velocities of mesh nodes are equal to fluid velocity. In the MFE method, equations for mesh movement are 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. It assures a direct volume control through the Jacobian determinant. The node velocity field is equal to $M\,\nabla w$ where M is a monitor function and w is calculated using the solution of the scalar Poisson equation with a right-hand side depending on the time derivative of M. The most attractive feature of this method is that it assures the existence of a valid grid, because it produces the grid with strictly positive ($M > 0$) determinant of transformation. The GCL is an extension of the deformation method where a more general equation is solved for w. In particular, it allows generation of an irrotational mesh velocity by a special choice of parameters. The MFD method and its extension MMPDE use a parabolic PDE (with coefficients depending on monitor function) for mesh movement. Its special feature is achievement of mesh smoothness in time and space. As it was shown in this paper, mesh smoothness is very important and it will be addressed again later in this section.

Typical location-based approaches are: equidistribution methods in 1D (see, e.g., [59, Sec. 7.2]); Winslow's variable diffusion functional [83, Sec. 7.10.6]; harmonic mapping [34, 4], [79, Ch. 8]; the functional of Brackbill and Saltzman [17]; directional control functional [16]; Jacobian-weighted elliptic grid generation [48]; Thompson's method [80]; variational approach suggested in [21]; etc. Among all the above, the methods based on a harmonic mapping are most promising. They always produce unfolded meshes, mesh adaptation is controlled only by a monitor function and there are no other parameters. 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 role of a *barrier*. The value of the functional approaches infinity when the map is degenerate which explains why the harmonic mapping produces unfolded grids. A detailed description of variational methods can be found in [59, Ch. 8].

Other examples of location-based methods are various methods using physical analogies with spring systems and modified elasticity equations (see, e.g., [10] and references therein, [36], [2], [30], [13]) where stiffness of springs and properties of artificial elastic media depend on a monitor function.

The location-based method close to the one presented in this paper is the moving best fit method [7, 9] where mesh is found by direct optimization of a functional representing the error of best piecewise constant L_2 fit with adjustable nodes.

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, almost all methods use the original idea of a guaranteed smoothing introduced in the seminal paper by Dorfi and Drury [32]. Detailed exposition of this idea is presented in dissertation [87]. 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 overall stability of moving mesh methods. The following papers directly address stability issues: [71, 55]. A guaranteed mesh smoothing in multidimensions is a very active research area.

It worth to note the interesting discussion on accuracy of adaptive grid methods for captured shocks, [85]. In particular, the authors question the validity of equidistribution principle for problems with strong discontinuities, analyze the influence of mesh non-uniformity on the overall accuracy, and show that for the problem they have analyzed, one may get no improvement by using many adaptive strategies.

Finally, we want to mention papers and books that deal with moving meshes for realistic problems of computational fluid dynamics and computational mechanics: [10, 60, 13, 76, 63, 36, 2, 6, 69].

6.2 Conclusion and Future Plans

We have developed an ALE method with an error-minimization-based rezone strategy for 1D viscous Burgers' equation. In the Lagrangian stage, we use the simplest first-order in time and space discretization of the conservative Lagrangian form of Burgers' equation. Primary unknowns in our discretization are mean values over mesh cells. Between time levels, the mesh nodes are moving using the simplest explicit time integration. The goal of the rezoning stage of the EMB ALE method is to reduce L_2 norm of error at new time level t^{n+1}. For remapping, we use the conservative linearity and local-bound-preserving method which is second-order accurate for smooth meshes and solutions.

The *optimal-mesh criterion* is minimization of L_2 norm of error of piecewise constant representation of exact solution at new time level t^{n+1} (Section 4.2).

The *error estimator* is the leading term in L_2 norm of error derived under assumptions of mesh smoothness, accuracy of remapping, and time step satisfying the stability condition (Section 4.2.1). Let us note that our estimator is optimal, because it converges to the real error when spatial and temporal meshes are refined.

The *algorithm for mesh improvement* is decoupled from the governing PDE (Section 4). The *strategy for mesh movement* is based on direct optimization of approximate error functional which is based on our error estimator. The guaranteed mesh smoothing is embedded in the minimization algorithm (Algorithm 2).

We have demonstrated superiority of our EMB ALE method in comparison with the pure Lagrangian and RJ ALE methods (Section 5). It is typical for 1D hyperbolic problems that the arithmetical complexity for a fixed accuracy is less for uniform grids 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 the convergence rates for uniform and adaptive grids will be different. In other words, the slopes of graphs in Fig. 6 and 8 will be different resulting in computational efficiency of adaptive grids.

In future we plan to extend our EMB ALE method to full system of gasdynamics equations in 2D.

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 acknowledge valuable comments made by B. Wendroff, L. Margolin, M. Hyman, S. Li and R. Garimella.

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. *International Journal for Numerical Methods in Fluids*, 39:657–673, 2002.

[3] D. C. Arney and J. E. Flaherty. A two dimensional mesh moving technique for time dependent partial differential equations. *J. Comput. Phys.*, 67:124–144, 1986.

[4] B. N. Azarenok. Variational barrier method of adaptive grid generation in hyperbolic problems of gas dynamics. *SIAM J. Numer. Anal.*, 40:651–682, 2002.

[5] B. N. Azarenok and S. A. Ivanenko. Application of adaptive grids in numerical analysis of time-dependent problems in gas dynamics. *Computational Mathematics and Mathematical Physics*, 40(9):1330–1349, 2000.

[6] B. N. Azarenok, S. A. Ivanenko, and T. Tang. Application of adaptive grids in numerical analysis of time-dependent problems in gas dynamics. *Communications in Mathematical Sciences*, 1(1):152–179, 2003.

[7] 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. of Comput.*, 62(206):645–669, 1994.

[8] M. J. Baines. *Moving Finite Elements.* Oxford University Press, New York, 1994.

[9] M. J. Baines. Grid adaptation via node movement. *Appl. Numer. Math.*, 26:77–96, 1998.

[10] T. J. Baker. Mesh adaptation strategies for problems in fluid dynamics. *Finite Elements in Analysus and Design*, 25:243–273, 1997.

[11] 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.

[12] 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.

[13] T. Belytschko, W. K. Liu, and B. Moran. *Nonlinear Finite Elements for Continua and Structures.* John Wiley & Sons, Ltd., Chichester, 2000.

[14] D. J. Benson. Computational methods in Lagrangian and Eulerian hydrocodes. *Computer Methods in Applied Mechanics and Engineering*, 99:235–394, 1992.

[15] P. Bochev, G. Liao, and G. Pena. Analysis and computation of adaptive moving grids by deformation. *Numer. Meth. PDEs*, 12:489–506, 1996.

[16] J. U. Brackbill. An adaptive grid with directional control. *J. Comput. Phys.*, 108:38–50, 1993.

[17] J. U. Brackbill and J. S. Saltzman. Adaptive zoning for singular problems in two dimensions. *J. Comput. Phys.*, 46:342–368, 1982.

[18] 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.

[19] 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.

[20] 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.

[21] W. Cao, W. Huang, and R. D. Russell. Approaches for generating moving adaptive meshes: Location versus velocity. *Appl. Numer. Math.*, 2003.

[22] 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.

[23] E. J. Caramana and M. J. Shashkov. Elimination of artificial grid distortion and hourglass-type motions by means of Lagarangian subzonal masses and pressures. *J. Comput. Phys.*, 142:521–561, 1998.

[24] E. J. Caramana, M. J. Shashkov, and P. P. Whalen. Formulations of artificial viscosity for multi-dimensional shock wave computations. *J. Comput. Phys.*, 144:70–97, 1998.

[25] G. F. Carey. *Computational Grids. Generation, Adaptation and Solution Strategy.* Taylor & Francis, Washington, 1997.

[26] 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.

[27] N. N. Carlson and K. Miller. Design and application of gradient-weighted moving finite element code i: In two dimension. *SIAM J. Sci. Comp.*, 19:766–798, 1998.

[28] H. D. Ceniceros and T. Y. Hou. An efficient dynamically adaptive mesh for potentially singular solutions. *J. Comput. Phys.*, 172:609–639, 2001.

[29] C. de Boor. *Good Approximation by Splines with Variable Knots, II.* Springer Lecture Series 363, Springer-Verlag, NY, 1973.

[30] C. Degand and C. Farhat. A three-dimensional torsional spring analogy method for unstructured dynamic meshes. *Computers and Structures*, 80:305–316, 2002.

[31] J. Donea, S. Guliani, and J. P. Halleux. An arbitrary Lagrangian-Eulerian finite-element method for transient dynamic fluid structure interaction. *Computer Methods in Applied Mechanics and Engineering*, 33:689–723, 1982.

[32] E. Dorfi and L. Drury. Simple adaptive grids for 1-D initial value problems. *J. Comput. Phys.*, 69:175–195, 1987.

[33] J. K. Dukowicz. A simplified adaptive mesh technique derived from the moving finite element method. *J. Comput. Phys.*, 56:324–342, 2002.

[34] A. S. Dvinsky. Adaptive grid generation from harmonic maps on riemannian manifolds. *J. Comput. Phys.*, 95:450–476, 1991.

[35] P.-L. George and H. Borouchaki. *Delaunay Triangulation and Meshing.* Hermes, Paris, 1998.

[36] W. G. Habashi, J. Dompierre, Y. Bourgault, D. Ait-Ali-Yahia, M. Fortin, and M.-G. Vallet. Anisotropic mesh adaptation: Toward user-independent, mesh-independent and solver-independent cfd. part i: General principles. *International Journal for Numerical Methods in Fluids*, 32:725–744, 2000.

[37] A. Harten and J. M. Hyman. Self-adjusting grid methods for one-dimensional hyperbolic conservation laws. *J. Comput. Phys.*, 50:235–269, 1983.

[38] D. F. Hawken, J. J. Gottleib, and J. S. Hansen. Review of some adaptive node-movement techniques in finite-element and finite-difference solution of partial differential equations. *J. Comput. Phys.*, 89:254–302, 1991.

[39] 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.

[40] W. Huang and R. D. Russell. Adaptive mesh movement – the MMPDE approach and its applications. *J. Comput. Appl. Math.*, 128:383–398, 2001.

[41] W. Huang and W. Sun. Variational mesh adaptation ii: Error estimates and monitor functions. *J. Comput. Phys.*, 184:619–648, 2003.

[42] W. H. Hui and S. Kudriakov. A unified coordinate system for solving the three-dimensional euler equations. *J. Comput. Phys.*, 172:235–260, 2001.

[43] J. M. Hyman. *Adaptive Moving Meshes for Partial Differential Equations.* Report LA-UR-82-3690, Los Alamos National Laboratory, 1982.

[44] J. M. Hyman and B. Larrouturou. Dynamic rezone methods for partial differential equations in one space dimension. *Appl. Numer. Math.*, 5:435–450, 1989.

[45] 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. *Computer Methods in Applied Mechanics and Engineering*, 158:81–116, 1998.

[46] P. Kjellgren and J.Hyvarien. An arbitrary Lagrangian-Eulerian finite element method. *Computational Mechanics*, 21:81–90, 1998.

[47] P. Knupp, L. G. Margolin, and M. Shashkov. Optimization-based rezone strategies for arbitrary Lagrangian Eulerian methods. *J. Comput. Phys.*, 176:93–128, 2002.

[48] P. M. Knupp. Jacobian-weighted elliptic grid generation. *SIAM J. Sci. Comput.*, 17:1475–1490, 1996.

[49] M. Kucharik, M. Shashkov, and B. Wendroff. An efficient linearity-and-bound-preserving remapping methods. *J. Comput. Phys.*, 188:462–471, 2003.

[50] C. B. Laney. *Computational Gasdynamics.* Cambridge University Press, 1998.

[51] R. J. LeVeque. *Numerical Methods for Conservation Laws.* Birkhauser Verlag, Basel, Boston, Berlin, 1992.

[52] R. Li, T. Tang, and P. Zhang. Moving mesh methods in multiple dimensions based on harmonic maps. *J. Comput. Phys.*, 170:562–588, 2001.

[53] R. Li, T. Tang, and P. Zhang. A moving mesh finite element algorithm for singular problems in two and three space dimensions. *J. Comput. Phys.*, 177:365–393, 2002.

[54] S. Li and L. Petzold. Moving mesh method with upwinding schemes for time-dependent pdes. *J. Comput. Phys.*, 131:368–377, 1997.

[55] S. Li, L. Petzold, and Y. Ren. Stability of moving mesh systems of partial differential equations. *SIAM J. Sci. Comput.*, 20:719–738, 1998.

[56] G. Liao, Z. Lei, and G. C. de la Pena. Adaptive grids for resolution enhancement. *Shock Waves*, 12:153–156, 2002.

[57] 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.

[58] K. Lipnikov and M. Shashkov. *Moving meshes for the Burgers equation.* Report LA-UR-03-7605, Los Alamos National Laboratory, 2003.

[59] V. Liseikin. *Grid Generation Methods.* Springer, Berlin, Heidelberg, New York, 1999.

[60] R. Löhner. *Applied CFD Techniques. An Introduction Based on Finite Element Methods.* John Wiley & Sons, Ltd., Chichester, 2001.

[61] L. G. Margolin. Introduction to "an arbitrary Lagrangian-Eulerian computing method for all flow speeds". *J. Comput. Phys.*, 135:202, 1997.

[62] L. G. Margolin and M. Shashkov. Second-order sign-preserving conservative interpolation (remapping) on general grids. *J. Comput. Phys.*, 184:226–298, 2003.

[63] D. S. McRae. r-refinement grid adaptation algorithms and issues. *Computer Methods in Applied Mechanics and Engineering*, 189:1161–1182, 2000.

[64] K. Miller and R. N. Miller. Moving finite elements i. *SIAM J. Numer. Anal.*, 18:1019–1032, 1981.

[65] K. Morton. *Numerical Solution of Convection-Diffusion Problems.* Chapmann & Hall, London, 1996.

[66] L. S. Mulholland, Y. Qui, and D. M. Sloan. Solution of evolutionary partial differential equations using adaptive finite differences with pdeudospectral post-processing. *J. Comput. Phys.*, 131:280–298, 1997.

[67] J. Nocedal and S. Wright. *Numerical Optimization.* Springer, Berlin, Heidelberg, New York, 1999.

[68] J. S. Peery and D. E. Carroll. Multi-material ALE methods in unstructured grids. *Computer Methods in Applied Mechanics and Engineering*, 187:591–619, 2000.

[69] B. Perot and R. Nallapati. A moving unstructured staggered mesh method for the simulation of incompressible free-surface flows. *J. Comput. Phys.*, 184:192–214, 2003.

[70] L. R. Petzold. *A Description of DASSL: A Differentila/Algebraic System Solver.* IMACS Trans. on Scientific Computations, ed. R. S. Stepleman, 1983.

[71] Y. Ren and R. D. Russell. Moving mesh techniques based upon equidistribution, and their stability. *SIAM J. Sci. Comput.*, 13:1265–1286, 1992.

[72] A. A. Samarskii. *The Theory of Finite Difference Schemes.* Marcel Dekker, New York, 2001.

[73] 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.

[74] B. Semper and G. Liao. A moving grid finite-element method using grid deformation. *Numer. Meth. PDEs*, 11:603–615, 1995.

[75] M. Shashkov. *Conservative Finite-Difference Methods on General Grids.* CRC Press, Boca Raton, 1996.

[76] B. K. Soni, R. Koomullil, D. S. Thompson, and H. Thornburg. 3D unstructured mesh ALE hydrodynamics with the upwind discontinuous finite element method. *Computer Methods in Applied Mechanics and Engineering*, 189:1183–1204, 2000.

[77] 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.

[78] H. Tang and T. Tang. Adaptive mesh methods for one- and two-dimensional hyperbolic conservation laws. *SIAM J. Numer. Anal.*, 41:487–515, 2003.

[79] J. F. Thompson, B. K. Soni, and N. P. Weatherill, editors. *Handbook of Grid Generation.* CRC Press, Boca Raton, 1999.

[80] J. F. Thompson, Z. A. Warsi, and C. W. Mastin, editors. *Numerical Grid Generation. Foundations and Applications.* North-Holland, New York, 1985.

[81] R. Verfürth. *A Review of a Posteriori Error Estimation and Adaptive Mesh Refinement Techniques.* Teubner Verlag and J. Wiley, Stuttgard, 1996.

[82] Z. U. A. Warsi. *Fluid Dynamics. Theoretical and Computational Approaches.* CRC Press, Boca Raton, 1993.

[83] A. Winslow. *Adaptive Mesh Zoning by Equipotential Method.* Report UCID-19062, Lawrence Livermore Laboratory, 1981.

[84] A. Wouwer, P. Sausez, and W. Schiesser, editors. *Adaptive Method of Lines.* Chapman & Hall, Boca Raton, 2001.

[85] N. K. Yamaleev and M. H. Carpenter. On accuracy of adaptive grid methods for captured shocks. *J. Comput. Phys.*, 181:280–316, 2002.

[86] 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, June 28 - July 2, 1976, Twente University, Enschede/The Netherlands, pp. 454–459.* Lecture Notes in Physics, Volume 59, 1976, eds. A. I. van de Vooren and P. J. Zandbergen, 1976.

[87] P. A. Zegeling. *Moving-Grid Methods for Time-Dependent Partial Differential Equations.* Centrum voor Wiskunde en Informatica, Amsterdam, 1992.