J. Math. Study
doi: 10.4208/jms.v48n2.15.03

Vol. **48**, No. 2, pp. 125-167
June 2015

R-Adaptive Reconnection-based Arbitrary Lagrangian Eulerian Method-R-ReALE

Wurigen Bo[1], Mikhail Shashkov[2,*]

[1] *Computational and Computer Science Division, CCS-2, Los Alamos National Laboratory, Los Alamos, NM 87545, USA*
[2] *X-Computational Physics, XCP-4, Los Alamos National Laboratory, Los Alamos, NM 87545, USA*

Received 9 January, 2015; Accepted (in revised version) 22 May, 2015

Abstract. We present a new R-adaptive Arbitrary Lagrangian Eulerian (ALE) method, based on the reconnection-based ALE - ReALE methodology [5, 41, 42]. The main elements in a standard ReALE method are: an explicit Lagrangian phase on an arbitrary polygonal (in 2D) mesh, followed by a rezoning phase in which a new grid is defined, and a remapping phase in which the Lagrangian solution is transferred onto the new grid. The rezoned mesh is smoothed by using one or several steps toward centroidal Voronoi tessellation, but it is not adapted to the solution in any way. We present a new R-adaptive ReALE method (R-ReALE, where R stands for Relocation). The new method is based on the following design principles. First, a monitor function (or error indicator) based on Hessian of some flow parameter(s), is utilized. Second, the new algorithm uses the equidistribution principle with respect to the monitor function as criterion for defining an adaptive mesh. Third, centroidal Voronoi tessellation is used for the construction of the adaptive mesh. Fourth, we modify the raw monitor function (scale it to avoid extremely small and large cells and smooth it to create a smooth mesh), in order to utilize theoretical results related to centroidal Voronoi tessellation. In the R-ReALE method, the number of mesh cells is chosen at the beginning of the calculation and does not change with time, but the mesh is adapted according to the modified monitor function during the rezone stage at each time step. We present all details required for implementation of the new adaptive R-ReALE method and demonstrate its performance relative to standard ReALE method on a series of numerical examples.

AMS subject classifications: 65M08, 65M55,76N99
Key words: Gas Dynamics, R-adaptation, Reconnection, ALE.

1 Background and rationale

As in most standard Arbitrary-Lagrangian-Eulerian (ALE) methods [28], the main elements in a ReALE [42] simulation are an explicit Lagrangian phase in which the solution

*Corresponding author. *Email addresses:* bowurigen@gmail.com (W. Bo), shashkov@lanl.gov (M. Shashkov)

and grid are updated (without changing its connectivity), a rezoning phase in which a new grid is defined, and a remapping phase in which the Lagrangian solution is transferred onto the new rezoned grid. The ReALE method described in [42] differs from standard ALE method in one element only - it allows connectivity changes during the rezone stage. The rezone phase of ReALE includes both mesh movement and the reconnection procedure, which is done using the machinery of Voronoi diagrams, [1]. The ReALE rezone strategy consists of a special movement of generators. It is similar to Lagrangian motion in some sense, but also include a smoothing procedure based on the notion of centroidal Voronoi diagrams [16]. By construction, a Voronoi mesh is a valid mesh and therefore each Lagrangian step starts with a valid mesh. The main objective of [42] was to develop a robust reconnection-based ALE method in which averaged cell movement is close to Lagrangian. The ReALE method allows running complex simulations to completion without user intervention, while maintaining reasonable accuracy. In [42] we also presented a comparison of Lagrangian methods with ReALE on problems for which pure Lagrangian methods can run without mesh tangling. Examples of ReALE simulations can be found in [5, 27, 41, 42].

The ReALE methods have significant potential with respect to adaptivity. First of all, repositioning (relocation or R-adaptivity) of the generators during the rezone stage can be related to some error indicator. Secondly, the number of generators (which defines the number of cells) can change with time to refine the mesh where it is needed (h-adaptivity). However, in [42] there was no attempt to explore adaptivity in framework of ReALE methods. In this paper we explore R-adaptivity in the framework of ReALE.

The need for adaptive methods is well recognized and there are numerous papers related to adaptation, see for example Chapter 14 in [40] and [30] and corresponding references herein, or the website `http://lsec.cc.ac.cn/~ttang/MMref`.

According to [40], any adaptive method is composed of three main ingredients: an error estimator or error indicator, an optimal-mesh criterion, and an algorithm of the strategy for mesh improvement. These ingredients answer the following questions: Where are mesh changes required? How should the optimal mesh be defined? How should the improved mesh be constructed?

Our adaptive ReALE methods are based on following well known basic design principles.[†]

The first design principle is to use a monitor (error indicator) function based on the Hessian of some flow parameter(s), which is a measure of interpolation error, [30, 40]. In general, a monitor function $\phi(\mathbf{x},t) > 0$ is some measure or indicator of the error. In an ideal case its construction is based on error estimates. However, in reality, especially for non-linear hyperbolic problems, practitioners use much simpler and readily computable indicators of errors. In our case we choose to use a monitor function based on estimates

[†]Let us note that to describe the main ideas we intentionally use a loose style of presentation to avoid lengthy definitions and explanations. In the main text of the article we give strict definition of all notions and notations that we use.

of spectral norm of a Hessian matrix, which are related to quadratic terms in the Taylor expansion of some flow parameter. The function $\phi(\mathbf{x},t)>0$ is known for any $\mathbf{x}$ only at initial time moment $t=0$. In general, a monitor function is given by its values, ϕ_i^n, on some mesh at a particular time moment t^n, where the index i is related to some mesh entity and the index n identifies the time step. To be used in an adaptation process the monitor function has to satisfy some requirements. We will formulate these requirements after we describe how a monitor function is used to construct an adaptive mesh.

The second design principle is to use the equidistribution principle for the monitor function as a criterion for the definition of an adaptive mesh, [30, 40]. If we are given the polygonal computational domain Ω and a monitor function ϕ_i^n then the mesh consisting of cells Ω_i, such that $\cup_i\Omega_i=\Omega$, satisfies the equidistribution principle if

$$\phi_i^n|\Omega_i|=E=const. \tag{1.1}$$

We will call E the equidistribution level. If the number of cells, N, is given (which is the case for R-adaptive methods) then, the equidistribution level E, is given by

$$E=\left(\sum_{i=1}^{N}\phi_i^n|\Omega_i|\right)\Big/N\approx\left(\int_\Omega\phi dV\right)\Big/N. \tag{1.2}$$

The third design principle is to use centroidal Voronoi tessellation as a tool for creating an adaptive mesh that satisfies the equidistribution principle, [16]. To explain this procedure, we need to recall some definitions [1, 16]. Given a set of N generators $\mathbf{G}=\{\mathbf{G}_i\}_{i=1}^N$, Voronoi tessellation is defined as the space decomposition into cells Ω_i

$$\Omega_i=\{\mathbf{x}\in\mathbb{R}^2:\|\mathbf{x}-\mathbf{G}_i\|\le\|\mathbf{x}-\mathbf{G}_j\|,\text{ for } j=1,...,N,j\neq i\},$$

where $\|\cdot\|$ is the Euclidean distance. The center of mass of cell Ω_i with respect to a weight function $\varphi(\mathbf{x})$ is

$$c_i^\varphi=\frac{\int_{\Omega_i}\mathbf{x}\varphi(\mathbf{x})d\mathbf{x}}{\int_{\Omega_i}\varphi(\mathbf{x})d\mathbf{x}}.$$

If $G_i=c_i^\varphi$, then the Voronoi tessellation is called a centroidal Voronoi tessellation (CVT) with respect to weight φ. That is, the positions of the generators coincide with the centers of mass of the corresponding Voronoi cells. The CVTs have several useful properties, in particular, in the asymptotic case of an infinite number of generators with a smooth weight function, the cells of the CVT are perfect hexagons in 2D and

$$\sqrt{\varphi(\mathbf{G}_i)}|\Omega_i|\approx\sqrt{\varphi(\mathbf{G}_j)}|\Omega_j|.$$

Therefore, if we set $\varphi=\phi^2$, we obtain a mesh which asymptotically satisfies the equidistribution principle

$$\phi(\mathbf{G}_i)|\Omega_i|\approx\phi(\mathbf{G}_j)|\Omega_j|,\quad\text{or}\quad\phi_i|\Omega_i|\approx\phi_j|\Omega_j|.$$

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

Most theoretical results about CVT have been proved with an assumption that the weight function is smooth and in the limit of an infinite number of generators. In practice, to be useful for adaptive mesh construction, we need to guarantee some level of smoothness of the weight function and use a reasonably large number of generators.

The fourth design principle is modification of the monitor function. This includes several steps. First, as it is seen from equation (1.1), if the monitor is too large then the corresponding cell will be too small - this can lead to a small time step, because we use explicit time integration and time step has to satisfy the CFL stability condition. If the monitor is too small then the corresponding cell can be too and this can lead to a loss of accuracy. Therefore, we would like to scale the monitor function to avoid extremely small and large cells. The details of scaling will be presented in Section 3. As we have already mentioned, the monitor function has to be sufficiently smooth to use CVT for construction of mesh which satisfies equidistribution principle. To construct a smooth monitor we use ideas presented in the seminal paper [14] - see Section 3.3 for details.

Now we are ready to give a brief characterization of our R-adaptive ReALE method (R-ReALE where R stands for Relocation). In this method the number of mesh cells is chosen at the beginning of the calculation and does not change with time, but the mesh is adapted to the modified monitor function during the rezone stage at each time step, such that the Lagrangian step always begins with a mesh which satisfies the equidistribution principle with respect to $\tilde{\phi}_i^n$.

Let us now make some general comments on adaptive methods. The standard ALE methods or moving mesh methods in general, [30, 52], use fixed mesh topology, and nodes are moved to refine the mesh in some areas of the computational domain at the expense of coarsening the mesh in other parts of the problem. Generally, the increase of mesh resolution is limited, and, most importantly it can degrade the mesh quality leading to robustness issues. One of the most cited papers in R-adaptive mesh adaptation is [4], where the authors use a variational approach to combine requirements related to maintaining the geometric quality of the mesh and mesh adaptation based on equidistribution of some error indicator. The functional which is responsible for mesh smoothness essentially can be considered as a variational form of the Winslow approach, [56], for which the corresponding optimization problem is well behaved. In contrast, the functional responsible for error equidistribution, if used by itself, has multiple local minima and its minimization can lead to a tangled mesh. A difficulty arises when one tries to combine these — how should one weight the relative importance of these separate goals and still obtain a well behaved optimization process? In particular, the two global functionals have distinct (physical) dimensions, and so, can only be combined with some dimensional constant. At present, there is no theoretical basis for choosing this constant, thus delegating the decision to the user. In practice, a bad choice of this parameter can lead to loss of accuracy (if the mesh is over-smoothed) or robustness problems (if the mesh becomes tangled). The main advantage of ReALE adaptive methods in comparison with standard adaptive ALE methods with fixed connectivity is that adaptive ReALE methods always produce smooth, valid and adaptive meshes.

Most papers related to adaptive methods start with statements like this one: "Solutions of PDEs arising in science and engineering frequently have large variations occurring over small portions of the physical domain, and major challenge when solving such problems is to appropriately resolve solution behavior there. ... fine mesh is required in those particular parts of the physical domain." - [30] or "Another characteristic of fluid flows is the very wide range of spatial scales often encountered: shocks in compressible flows, interfaces between immiscible liquids, turbulence intermittency, boundary layers and vorticity generation near solid boundaries are just a few examples. Consequently, in recent years a number of researchers have investigated the use of adaptive mesh refinement, where the spatial discretization is adjusted to follow the scale and temporal evolution of flow structures .... " - [48] This sounds like good motivation to develop new adaptive methods.

Authors then typically describe their adaptive method and present examples where mesh is adapted in some way in places where distinct features of the flow are present. In some papers results from other method(s) are shown and errors are compared. Usually there is no evidence given as to why the new adaptive method is more efficient than the non-adaptive method or some other adaptive method. Our opinion is that there are several reasons for this.

Clearly, each adaptive method comes with some overhead related to computing an error indicator, solving some type of optimization problem (R-adaptation), or dealing with more complicated data structures (connectivity change as in ReALE family of methods or h-adaptation). The percentage of required resources (CPU time, memory, complexity of coding, incorporating adaptation in existing solvers, etc.) related to overhead with respect to overall resources defines the efficiency of the adaptive method.

For some classes of adaptive methods the situation with regards to efficiency is relatively clear. For example, for classical Adaptive Mesh Refinement (AMR) type of methods, [3,10,29,47] (see also [22,45]), efficiency is usually measured with respect to *equivalent* uniform high-resolution static grid results, [35]. This metric is well defined, because in standard AMR methods there are only two regimes in which these methods can run - uniform mesh or adaptive mesh [‡]. However, even for these type of methods, the results of efficiency analysis will depend on the complexity of the flow at a particular time moment and may differ significantly for different error definitions.

In general, it is very hard to define "efficiency". We have had discussions with several experts in adaptive mesh refinement and the only aspect on which all experts agree is that adaptive mesh refinement is needed in regions where particular spatial resolution is required to resolve some important physics. In this case if you do not have a method that can achieve the required spatial resolution then you cannot obtain physically meaningful results, and therefore, there is no question about efficiency.

At first glance, the definition of efficiency should be obvious. Let us assume that we have some notion of accuracy[§], then the most efficient method is the one which allows

[‡]The assumption is that refinement/derefinement criterion is fixed and zero-level of refinement is chosen.
[§]In many cases, especially for solving complex multi-physics problems definition of accuracy is a problem

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

us to obtain the given level of accuracy with the least amount of resources. However, the notion of resources may include many different things: CPU time, computer memory, electricity used by computer, man-hours of user(s) needed to complete the calculation. Most metrics are hardware dependent and also depend on how efficiently the algorithm is coded with respect to the architecture of a particular computer. Man-hours needed to complete specific calculations strongly depends on user experience with that particular algorithm and that particular problem. Moreover, in many cases users are only interested in knowing what maximum accuracy can be achieved using their limited resources. All these considerations make direct and fair comparisons of computational algorithms a difficult problem.

The situation is more complicated when comparing standard non-adaptive pure Lagrangian methods with ALE methods with fixed mesh connectivity. In Lagrangian and ALE methods with fixed connectivity, accuracy and robustness strongly depends on the choice of an initial mesh. Practitioners usually run the same problem many times starting with different meshes to find one which allows the calculation to run to completion. In many cases, pure Lagrangian calculation cannot be completed with any choice of initial mesh and therefore it is not possible to compare relative efficiency of a pure Lagrangian and standard adaptive ALE method. In general, the initial mesh for an ALE calculation is already adapted to the initial data, but this initial mesh is not necessarily a good choice for non-adaptive Lagrangian methods. Therefore, the only feature which adaptive and non-adaptive method may have in common is the number of cells and the connectivity of the mesh. If the number of cells and the connectivity is fixed then one can find the best mesh for a Lagrangian calculation and the best mesh for adaptive ALE calculations and compare the results.

We have presented all of these arguments just to demonstrate that in general it may not be very useful to talk about the relative efficiency of new methods.

Our opinion is that what is useful is to consider a new method as a new tool and present some performance characteristics of the new method in different situations. This will allow a potential user to make their own conclusion about how useful this new tool could be to their application.

We would like to mention several recently published papers, which are in a similar spirit to our paper [2, 11, 20, 51].

The remainder of the paper is organized as follows. In Section 2 we present definitions related to Voronoi and centroidal Voronoi meshes, consider their properties and describe the related optimization problem which is the basis for construction of Voronoi meshes. In the same section we give some details related to construction of centroidal Voronoi meshes when a weight function is given on some background mesh. In Section 3 we define the monitor function and describe scaling and smoothing procedures for the monitor. Section 4 describes a family of ReALE methods, including standard non-adaptive ReALE methods as well as our new R-adaptive R-ReALE method. In Section 5

by itself.

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

we describe possible ways to compare the performance of ReALE and R-ReALE methods. In Section 6 we present numerical examples which demonstrate different aspects of the new R-ReALE method and allows comparison with standard non-adaptive ReALE methods. Conclusions and future work are presented in Section 7.

2 Constrained centroidal Voronoi tessellation

In this Section we describe the main tool with which we construct meshes during the rezone stage of the R-adaptive ReALE method.

In Section 2.1 we define Voronoi tessellation for non-convex computational domains. The reader who is interested in more information about Voronoi tessellation can refer to [1, 16–18, 32, 46, 54]. In Section 2.2 we define centroidal Voronoi tessellation (CVT) and explain how the properties of such tessellation can be used to construct meshes which satisfy an equidistribution principle. The computation of CVT, based on a variational formulation, is described in Section 2.4.

2.1 Voronoi tessellation in general domain

Given a set of N generators $\mathbf{G}=\{\mathbf{G}_i\}_{i=1}^N$, and polygonal computational domain Ω, a Voronoi tessellation (VT) is defined as the decomposition of space into cells according to the nearest generator. Namely,

$$\Omega_i=\{\mathbf{x}\in\mathbb{R}^2|\ \|\mathbf{x}-\mathbf{G}_i\|\leq\|\mathbf{x}-\mathbf{G}_j\|,\text{ for } j=1,...,N,j\neq i\},\tag{2.1}$$

where $\|\cdot\|$ is the Euclidean distance. The region Ω_i is referred to as a Voronoi cell and the set $\{\Omega_i\}_{i=1}^N$ is referred to as a VT [9].

If a VT is used in a discretization method for a partial differential equation (PDE) on a bounded domain Ω, the boundary of the VT must conform with the boundary of Ω so that the boundary condition of the PDE can be enforced. To represent the domain boundary, we use the bounded VT [54]: assume that Ω is a bounded domain and its boundary $\partial\Omega$ is comprised of a set of line segments that satisfies the following properties.

- The interior of each line segment in $\partial\Omega$ intersects no other line segments in $\partial\Omega$.
- The endpoints of the line segments are the generators in $\mathbf{G}$.

Assume the set of generators is the union of two sets $\mathbf{G}=\mathbf{G}_I\cup\mathbf{G}_B$, $\mathbf{G}_I=\{\mathbf{G}_i\}_{i=1}^M$, $\mathbf{G}_B=\{\mathbf{G}_i\}_{i=M+1}^N$, where the generators in $\mathbf{G}_B$ lie on $\partial\Omega$ and those in $\mathbf{G}_I$ lie on the interior of Ω. The VT on Ω is defined as

$$\Omega_i=\{\mathbf{x}\in\Omega|\ d(\mathbf{x},\mathbf{G}_i)\leq d(\mathbf{x},\mathbf{G}_j),\text{ for } j=1,...,N,j\neq i\}.\tag{2.2}$$

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

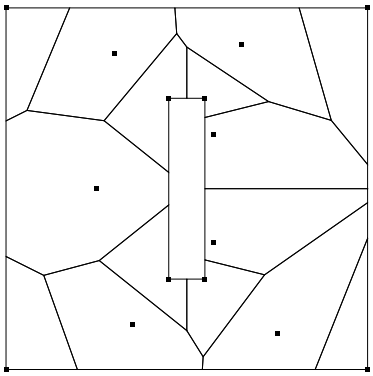


Figure 1: the VT with 15 generators. The 8 boundary generators form the inner and outer boundary of the domain.

where $d(\mathbf{x},\mathbf{y})$ is the distance which incorporates visibility constraints

$$d(\mathbf{x},\mathbf{y})=\begin{cases} \|\mathbf{x}-\mathbf{y}\| & \text{if } \mathbf{x} \text{ "sees" } \mathbf{y}, \\ +\infty & \text{otherwise.} \end{cases} \tag{2.3}$$

In this definition, $\mathbf{x}$ "sees" $\mathbf{y}$ when no line segment on $\partial\Omega$ intersects the line segment connecting $\mathbf{x}$ and $\mathbf{y}$. In Fig. 1 we present VT for square domain with rectangular hole in the middle - it has 8 boundary generators and 7 interior generators.

Let us note that Voronoi cells which correspond to internal generators are always convex and Voronoi cells which correspond to boundary generators can be non-convex, but each cell always consist of only one piece.

2.2 Constrained centroidal Voronoi tessellation

Given a weight function $\varphi(\mathbf{x})>0$ defined on Ω, for each Voronoi cell Ω_i, we can define its mass centroid $\mathbf{G}_i^*$ by

$$\mathbf{G}_i^*=\frac{\int_{\Omega_i}\mathbf{x}\varphi(\mathbf{x})d\mathbf{x}}{\int_{\Omega_i}\varphi(\mathbf{x})d\mathbf{x}},\ i=1,...,N. \tag{2.4}$$

We refer to a VT as a constrained centroidal Voronoi tessellation (CCVT) if and only if

$$\mathbf{G}_i=\mathbf{G}_i^*,\ i=1,...,M. \tag{2.5}$$

In this definition, the generators on the boundary are not subject to (2.5). It is worth noting that there are different definitions for CCVTs [46] depending on how to place the boundary generators. One example is the conforming CVT [32] in which the boundary generators are automatically placed on the boundary. In our case, the boundary generators are fixed, which is the simplest case of CCVTs in [46]. In Fig. 2, we give a VT and a

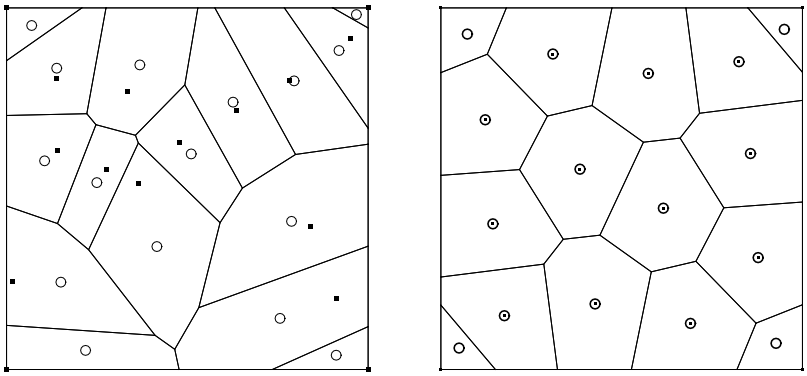


Figure 2: Four generators lie in the corners of the square. On the left, a VT corresponding to 12 randomly selected interior generators. On the right, a CCVT corresponding to 12 interior generators. In both figures, the weight function is a constant. The dots are the generators and the circles are the centroids of the corresponding Voronoi cells.

CCVT corresponding to 12 interior generators in a square. Note for the CCVT, (2.5) is not imposed on the four boundary generators.

Given any set of generators $\{\mathbf{x}_i\}_{i=1}^N$ on $\bar{\Omega}$ and any tessellation $\{V_i\}_{i=1}^N$, we define the corresponding energy functional by

$$\mathcal{F}(\{V_i\}_{i=1}^N,\{\mathbf{x}_i\}_{i=1}^N)=\sum_{i=1}^N\int_{V_i}\varphi(\mathbf{x})\|\mathbf{x}-\mathbf{x}_i\|^2d\mathbf{x}, \tag{2.6}$$

In [46], it is shown that CCVTs are solutions of the problem

$$\min_{\{V_i\}_{i=1}^N,\{\mathbf{x}_i\}_{i=1}^N}\mathcal{F}(\{V_i\}_{i=1}^N,\{\mathbf{x}_i\}_{i=1}^N),\quad \text{subject to } \mathbf{x}_j=\mathbf{G}_j,\ j=M+1,...,N. \tag{2.7}$$

If a CCVT with its generators is a global minimum of $\mathcal{F}$, it is called an optimal CCVT.

For an optimal CCVT with a smooth weight function, the internal Voronoi cells have many useful geometric properties as the number of generators becomes sufficiently large [21]:

- *Gersho's conjecture:* In the limit of an infinite number of generators, the internal Voronoi cells are congruent to a basic cell which depends on the dimension. The basic cell in 2D is the regular hexagon. Gersho's conjecture has been proved in 2D, [24], but it remains open for three and higher dimensions.

- Based on Gersho's conjecture, the following relation holds for the internal Voronoi cells:

$$\varphi(\mathbf{G}_i^*)h_i^{d+2}\approx\varphi(\mathbf{G}_j^*)h_j^{d+2}, \tag{2.8}$$

  where h_i is the diameter of the Voronoi cell i

$$h_i=2\max_{\mathbf{y}\in\Omega_i}\|\mathbf{y}-\mathbf{G}_i\| \tag{2.9}$$

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

and d is the spatial dimension. Since all of the internal Voronoi cells are congruent to a basic cell asymptotically, (2.8) can be rewritten in terms of the cell areas

$$\varphi(\mathbf{G}_i^*)|\Omega_i|^{1+2/d} \approx \varphi(\mathbf{G}_j^*)|\Omega_j|^{1+2/d}, \tag{2.10}$$

where $|\Omega_i|$ is the length/area/volume in 1D/2D/3D of the Voronoi cell i.

2.3 CCVT and construction of mesh satisfying equidistribution principle

The properties of CCVT summarized at the end of the previous Section suggest that an optimal CCVT can be used to construct polygonal meshes which satisfy the equidistribution condition. Given some monitor function ϕ on a 2D domain, we define a weight function $\varphi=\phi^2$. Then using (2.10), on the optimal CCVT with respect to φ, we have

$$\phi(\mathbf{G}_i^*)|\Omega_i| \approx \phi(\mathbf{G}_j^*)|\Omega_j|. \tag{2.11}$$

Therefore, the equidistribution condition with respect to ϕ is satisfied asymptotically.

If the number of generators, N, is given then according to (1.2) we will be able to achieve the following equidistribution level on CCVT mesh with weight function $\varphi=\phi^2$

$$E=\left(\int_\Omega \phi dV\right)\Big/ N.$$

2.4 Computation of CCVTs

2.4.1 Variational formulation

CCVTs as defined in (2.5) can be computed in the same way as CVTs. The computation of CVTs from a given set of generators has been well studied [16, 17]. Due to its simplicity and robustness, the most popular algorithm for the computation of CVTs is Lloyd's algorithm [39]. However, Lloyd's method may be too slow for practical applications. Recently, more efficient methods for CVT computation have been proposed, including the Lloyd-Newton method [15], a Newton-based multigrid algorithm [13] and a quasi-Newton method [38]. In this paper, we use a quasi-Newton method, the limited memory Broyden-Fletcher-Goldfarb-Shanno algorithm (L-BFGS).

To use L-BFGS for CCVT computation, we need to define the objective function to be minimized. From the previous section, any minimizer of the energy functional (2.6) forms a CCVT. However, it is hard to apply the L-BFGS algorithm for finding the minimizer of (2.6) because the $\mathbf{x}_i$ and V_i terms are independent in (2.6). For simplicity, we use the following energy functional in L-BFGS which has the same minimizer as (2.7)

$$\begin{aligned}&\mathcal{K}(\{\mathbf{x}_i\}_{i=1}^N)=\sum_{i=1}^N\int_{\Omega_i}\varphi(\mathbf{x})\|\mathbf{x}-\mathbf{x}_i\|^2 d\mathbf{x},\\ &\min_{\{\mathbf{x}_i\}_{i=1}^N}\mathcal{K}(\{\mathbf{x}_i\}_{i=1}^N), \text{ subject to } \mathbf{x}_j=\mathbf{G}_j,\ j=M+1,...,N.\end{aligned} \tag{2.12}$$

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

where Ω_i's are the Voronoi cells corresponding to the $\mathbf{x}_i$'s. Since the independent variables in (2.12) are the coordinates of the internal generators, we rewrite (2.6) as a new energy functional

$$\mathcal{E}(\{\mathbf{x}_i\}_{i=1}^M) = \mathcal{K}(\{\mathbf{x}_i\}_{i=1}^N). \tag{2.13}$$

The L-BFGS algorithm also requires the gradient of $\mathcal{E}$, which is given in [16]:

$$\frac{\partial \mathcal{E}}{\partial \mathbf{x}_i} = 2(\mathbf{x}_i - \mathbf{G}_i^*) \int_{\Omega_i} \varphi(\mathbf{x}) d\mathbf{x} \tag{2.14}$$

where $\mathbf{G}_i^*$ is the mass centroid of Ω_i. In order to describe the stopping criterion of the L-BFGS algorithm, we define an averaged deviation

$$D_{ave} = \frac{1}{M} \sum_{i=1}^{M} D_i, \tag{2.15}$$

where $D_i = \|\mathbf{x}_i - \mathbf{G}_i^*\| / h_i$ and h_i is defined in (2.9). It can be seen from (2.14) that $D_{ave} = 0$ if and only if $\partial \mathcal{E} / \partial \mathbf{x}_i = 0$, $i = 1...M$. Thus D_{ave} can be used to check the convergence of a CCVT computation. For the results in this paper, we use a stopping criterion $D_{ave} < 5 \cdot 10^{-4}$ with a maximum of 40 iterations.

The L-BFGS method requires an initial guess. The simplest choice is to use the centroids of the cells of the Lagrangian mesh. However, our tests show that there is a more efficient strategy based on insertion/deletion of generators using quad-tree and a local equidistribution principle to guide the insertion and deletion of generators. This algorithm will be described in a separate paper.

2.4.2 Numerical integration on the background mesh

The above CCVT computation relies on the evaluation of the integrals in (2.4) and (2.14) on Voronoi cells. We are constructing CCVT after a Lagrangian step and values of the monitor function can be associated with centroids of the cells of this Lagrangian mesh. The values of the monitor will be associated with a triangular *background* mesh $\mathcal{T}$, which is constructed from the Lagrangian mesh $\mathcal{M}^{\mathcal{L}}$.

Given the Lagrangian mesh $\mathcal{M}^{\mathcal{L}}$, we first construct the dual mesh connecting centroids of the neighboring cells of $\mathcal{M}^{\mathcal{L}}$. We then re-triangulate the boundary triangles of the dual mesh by inserting the boundary vertices of $\mathcal{M}^{\mathcal{L}}$. Fig. 3 shows an example for the Lagrangian mesh and the background mesh. To construct the monitor function at an interior vertex of $\mathcal{T}$, we use the Lagrangian state at that point. On a boundary vertex of $\mathcal{T}$, we use the physical state as defined by wall reflection.

To evaluate the integrals in (2.4) and (2.14), we subdivide each cell into triangles and use a composite one-point centroid rule to compute the integrals on each triangle [25] - see Fig. 4 for details. ¶

¶A triangle is subdivided into 16 equal-sized smaller triangles. In each smaller triangle, its centroid and area are used as the integration point and weight for the quadrature rule. See Fig. 4 for details. Our experiments show that our discretization of the integrals provides sufficient accuracy without unnecessarily increasing computation time.

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

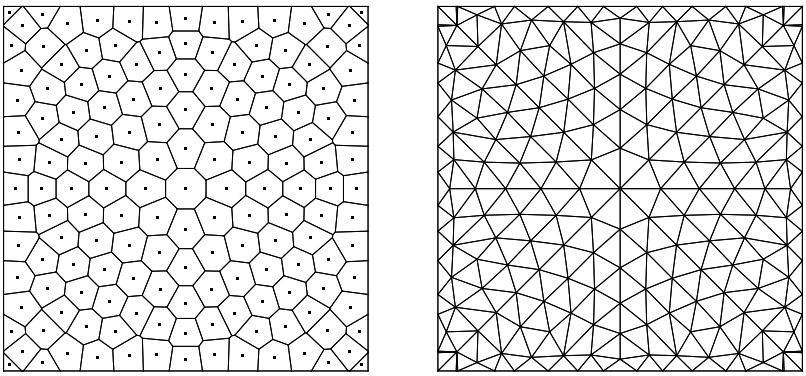


Figure 3: An illustration for the Lagrangian mesh and the background mesh. Left: the Lagrangian mesh $\mathcal{M}_{\mathcal{L}}$. The dots are the centroids of the cells. Right: the background mesh $\mathcal{T}$.

This method requires the evaluation of the monitor function at any integration point. For the purposes of this section we can assume that the monitor function ϕ is given by its values at the vertices of the background mesh. The detailed description of the computation of these values will be given in Section 3. Since the monitor function is defined on a triangular mesh, we need to find which triangle (we call it the containing triangle of the point), in the background mesh contains the integration point $\mathbf{x}$, then $\phi(\mathbf{x})$ is obtained by using a linear interpolation from the nodal values of ϕ on $\mathcal{T}$.

At quad-tree initialization there are two types of generators. For the first, from the set $\{\mathbf{G}_i^L\}$, we know that they are in the vertex set of the specific triangles of the background mesh. The second type of generators are generators which are added in a specific cell of mesh $\mathcal{M}^{\mathcal{L}}$ and a simple local search allow determination of the containing triangle. In the process of L-BFGS iterations, if the position of a generator $\mathbf{G}_j$ is changed, we find the containing triangle of $\mathbf{G}_j$ by searching the nearby triangles of its previous containing triangle. Finally, finding the containing triangle for for an integration point on a cell j is done by searching the nearby triangles of the containing triangle of $\mathbf{G}_j$.

Let us also mention that for convex domains one can use *walking* algorithms for the point location problem, [12]. However, simple *walking* algorithms do not work for non-convex domains.

3 Monitor function

3.1 Raw monitor

We base construction of the monitor function on estimate of the interpolation error. Although it may not be directly related to the solution error, interpolation error provides

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

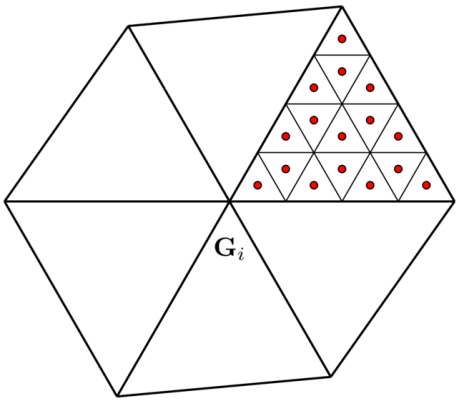


Figure 4: The discretization used to compute the integral of a function on a Voronoi cell. The Voronoi cell is divided into triangles. In each triangle, the integral is computed with a composite one-point centroid rule. The red dots are the integration points on a triangle.

important information on whether a mesh is suitable to approximate the solution of the PDE [6].

One of the option is to use the monitor function based on the L1 norm error of a linear interpolation [30]. This monitor is related to Hessian matrix. Let H_u be the Hessian matrix of a scalar physical quantity u. H_u is decomposed as

$$H_u = \begin{pmatrix} \frac{\partial^2 u}{\partial x^2} & \frac{\partial^2 u}{\partial x \partial y} \\ \frac{\partial^2 u}{\partial x \partial y} & \frac{\partial^2 u}{\partial y^2} \end{pmatrix}. \tag{3.1}$$

The monitor function based on the L1 norm error of a linear interpolation is given in [30] and chosen to be the square root of the spectral norm of Hessian matrix

$$\phi = \sqrt{||H_u||_2}. \tag{3.2}$$

Knowing the nodal values of the physical state u, many methods to compute the Hessian matrix components (second derivatives of u) at vertices can be used. Here, we use a finite volume-like approach [8]. The advantage of this approach is that it computes the Hessian matrix on a vertex by only using the nodal values of its adjacent vertices. Thus it is much faster than the classic use of least square approximation.

Let us present how raw monitor based of Hessian looks like in 1D for exact solution of well-known Sod problem [50] for $t = 0.2$ - Fig. 5 (left panel).

The raw monitor in this case is $\sqrt{|\partial^2 u/\partial x^2|}$. For demonstration purposes we use uniform background mesh and approximate second derivative using simple finite difference formula. The raw monitor corresponding to Hessian is presented in the right panel in Fig. 5.

It is important to mention the following facts about the behavior of the monitor function for the solutions which have discontinuity. For the smooth regions, like constant

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

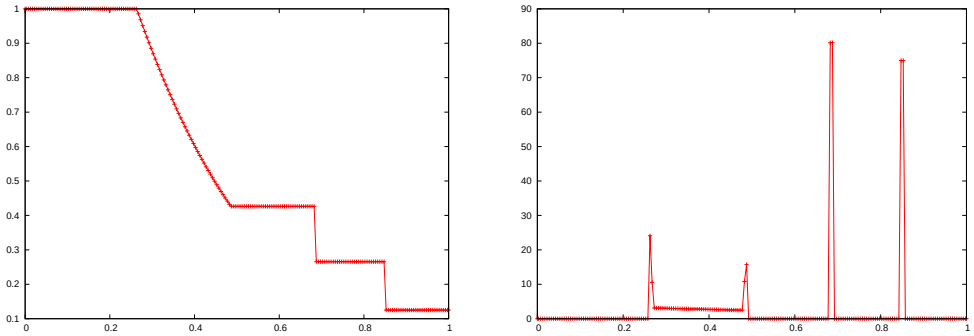


Figure 5: Sod problem: left panel - exact solution at $t=0.2$ on the uniform mesh of 200 cells; right - raw monitor using Hessian.

states regions or rarefaction region in Sod problem, the value of the monitor function only slightly changes with the mesh resolution - this is because numerical derivatives used to compute monitor are well behaved and converge to analytical derivatives with mesh refinement. Because the monitor reaches its minimum at smooth regions we can assume that

$$\phi^h_{min} \sim \phi^{h/k}_{min}, \tag{3.3}$$

where h is characteristic cell size and $k>1$ defines mesh refinement.

At the discontinuity analytical derivatives do not exist and numerical derivatives increase with increasing mesh resolution. Let us consider the simplest situation in 1D when we have to states u_l and u_r at the left and on the right side of the discontinuity and denote jump by $\Delta u = |u_r - u_l|$. Then numerical second derivative on mesh with cell size resolution h is proportional to

$$\left(\frac{\delta^2 u}{\delta x^2}\right)_h \sim \frac{\Delta u}{h^2}. \tag{3.4}$$

Therefore, maximum value of the monitor on mesh with characteristic size h is

$$\phi^h_{max} \sim \sqrt{\left(\frac{\delta^2 u}{\delta x^2}\right)_h} \sim \frac{\sqrt{\Delta u}}{h}. \tag{3.5}$$

For refined mesh with characteristic cell size h/k then

$$\phi^{h/k}_{max} \sim \sqrt{\left(\frac{\delta^2 u}{\delta x^2}\right)_{h/k}} \sim \sqrt{\frac{\Delta u}{(h/k)^2}} = k\frac{\sqrt{\Delta u}}{h} \sim k\phi^h_{max}. \tag{3.6}$$

Therefore, for solutions with discontinuities we have

$$\frac{\phi^{h/k}_{min}}{\phi^{h/k}_{max}} \sim \frac{1}{k}\frac{\phi^h_{min}}{\phi^h_{max}}, \tag{3.7}$$

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

That is ratio of ϕ_{min}/ϕ_{max} for solutions with discontinuities reduces with the same rate as characteristic mesh size.

3.2 Monitor scaling

For our example raw monitor is zero in regions where function is constant. In general, as it is seen from equation (1.1), if the monitor is too large then the corresponding cell will be too small - this can lead to a small time step, because we use explicit time integration and the time step has to satisfy a CFL stability condition. If the monitor is too small then the corresponding cell can be too large and this can lead to loss of accuracy. Therefore, we would like to scale the monitor function to avoid extremely large and small cells. However, we would like to scale it in such a way that the scaled monitor function preserves the "shape" of the original monitor - we will call this process "shape preserving scaling" or just "scaling". We will denote the raw monitor by ϕ and scaled monitor by $\hat{\phi}$.

To construct $\hat{\phi}$ we first compute average value of the raw monitor as follows

$$\bar{\phi}=\sum_i \phi_i|\Omega_i| \Big/ \sum_i |\Omega_i| \approx \int_\Omega \phi dV \Big/ |\Omega|. \tag{3.8}$$

If we use $\bar{\phi}$ as monitor function we will obtain uniform mesh consisting of hexagons in 2D (uniform mesh in 1D). Because

$$\bar{\Phi}=\int_\Omega \bar{\phi} dV=\Phi=\int_\Omega \phi dV, \tag{3.9}$$

then to achieve the same equidistribution level $E=\Phi/N=\bar{\Phi}/N$ using $\bar{\phi}$ or ϕ we will need the same number of generators N. For sake of brevity for the rest of the paper we will denote areas of the cells by A, for example, $A_i=|\Omega_i|$. For uniform mesh we have

$$A_i=\bar{A}=|\Omega|/N, \tag{3.10}$$

and

$$\bar{A}\bar{\phi}=E. \tag{3.11}$$

For mesh obtained from raw monitor we have

$$A_i\phi_i=E=A_{min}\phi_{max}=A_{max}\phi_{min}. \tag{3.12}$$

Therefore variation of area size for mesh corresponding to raw monitor is

$$A_{min}/A_{max}=\phi_{min}/\phi_{max}. \tag{3.13}$$

We construct scaled monitor function as follows

$$\hat{\phi}_i=\bar{\phi}+\beta(\phi_i-\bar{\phi}), \quad 0\le\beta\le 1, \tag{3.14}$$

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

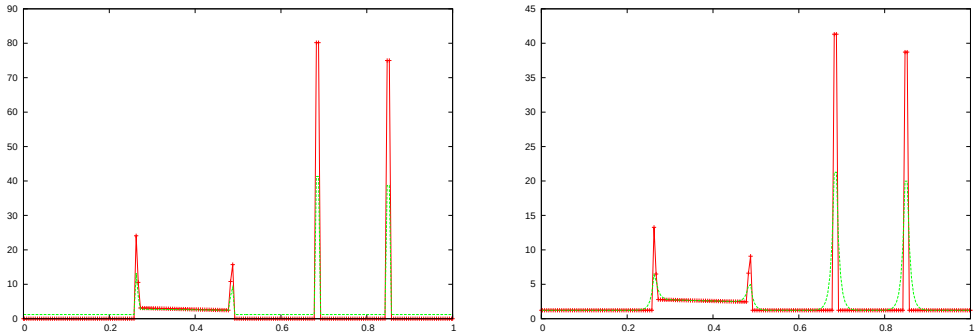


Figure 6: Sod problem: left panel - raw (red) and scaled (green) monitors; right panel - scaled (green) and smoothed (black) monitors

such that β=0 will correspond to uniform mesh and β=1 corresponds to most "adaptive" mesh. The continuous analog of this definition is

$$\hat{\phi}(\mathbf{x}) = \bar{\phi} + \beta(\phi(\mathbf{x}) - \bar{\phi}). \tag{3.15}$$

From equation (3.15) we can conclude that

$$\nabla_{\mathbf{L}}\hat{\phi} = \beta\nabla_{\mathbf{L}}\phi, \quad \forall \mathbf{L} \tag{3.16}$$

where $\nabla_{\mathbf{L}}$ is directional derivative in $\mathbf{L}$ direction. The equation (3.16) is definition of *shape preservation*. It is important to note that

$$\hat{\Phi} = \int_{\Omega} \hat{\phi} dV = \Phi = \int_{\Omega} \phi dV = \bar{\Phi} = \int_{\Omega} \bar{\phi} dV. \tag{3.17}$$

In left panel in Fig. 6 we superimpose raw and scaled monitors for Sod solution for β=0.5. One can see that scaled monitor is "shape" preserving and has a larger minimum and a smaller maximum than the raw monitor, and that in particular, its minimum is not zero.

The scaling also can be used to enforce the prescribed ratio of $(A_{min}/A_{max})^{pres}$. In fact, for $0 \le \beta \le 1$ we have

$$\frac{\hat{A}_{min}}{\hat{A}_{max}}(\beta) = \frac{\hat{\phi}_{min}}{\hat{\phi}_{max}}(\beta) = \frac{\bar{\phi} + \beta(\phi_{min} - \bar{\phi})}{\bar{\phi} + \beta(\phi_{max} - \bar{\phi})}. \tag{3.18}$$

It is easy to show that

$$\frac{\hat{A}_{min}}{\hat{A}_{max}}(0) = 1, \quad \frac{\hat{A}_{min}}{\hat{A}_{max}}(1) = \frac{\phi_{min}}{\phi_{max}}, \tag{3.19}$$

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

and function $\frac{\hat{A}_{min}}{\hat{A}_{max}}(\beta)$ is monotone decreasing with increasing β. In fact,

$$\frac{d}{d\beta}\left(\frac{\hat{A}_{min}}{\hat{A}_{max}}(\beta)\right)=\frac{\bar{\phi}(\phi_{min}-\phi_{max})}{(\bar{\phi}+\beta(\phi_{max}-\bar{\phi}))^2}\leq 0. \tag{3.20}$$

Therefore, to achieve prescribed ratio $(A_{min}/A_{max})^{pres}$ one need to solve linear equation

$$\left(\frac{\hat{A}_{min}}{\hat{A}_{max}}\right)^{pres}=\frac{\bar{\phi}+\beta^{pres}(\phi_{min}-\bar{\phi})}{\bar{\phi}+\beta^{pres}(\phi_{max}-\bar{\phi})}, \tag{3.21}$$

which gives

$$\beta^{pres}=\frac{\bar{\phi}\left(1-\left(\frac{\hat{A}_{min}}{\hat{A}_{max}}\right)^{pres}\right)}{\left(\frac{\hat{A}_{min}}{\hat{A}_{max}}\right)^{pres}(\phi_{max}-\bar{\phi})-(\phi_{min}-\bar{\phi})}. \tag{3.22}$$

3.3 Monitor smoothing

As we have already mentioned monitor function has to be sufficiently smooth to use CCVT for construction of mesh which satisfies equidistribution principle. To construct smooth monitor we use ideas presented in seminal paper [14]. Authors of [14] suggest to solve

$$(I-\alpha(1+\alpha)\delta^2)\tilde{\phi}=\hat{\phi}$$

equation for spatial smoothing, where I is identity operator and δ^2 is Laplacian in logical space, to obtain smooth monitor function $\tilde{\phi}$ from scaled monitor function $\hat{\phi}$. According to [14] the smoothed monitor will approximately satisfy the following conditions

$$\alpha/(1+\alpha)\leq\tilde{\phi}_i/\tilde{\phi}_j\leq(1+\alpha)/\alpha, \tag{3.23}$$

where j is index of neighbor cell of the cell i. This process is also "shape" preserving. In right panel in Fig. 6 we superimpose scaled and smoothed monitors for Sod solution for $\alpha=1$.

In Fig. 7 we plot ratio of values of the smoothed monitor in two neighboring cells as function of the coordinates of the vertex which shared by these cells. According to the inequality (3.23) ratio of values of smoothed monitor in neighboring cells is between 2 and $1/2$ for $\alpha=1$.

4 ReALE family of methods

In this section we will describe reconnection-based ALE (ReALE) family of methods. We will start with description of the standard ReALE method introduced in [42]. Then we will describe our new adaptive ReALE method: R-ReALE, in which number of cells does

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

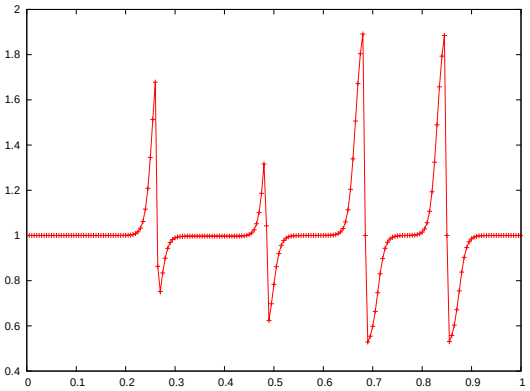


Figure 7: Sod problem: ratio of values of the smoothed monitor in two neighboring cells as function of the coordinates of the vertex which shared by these cells.

not change but mesh is adapting (by relocation of the generator's positions) to the monitor function according to equidistribution principle.

Let us mention that the main motivation for developing the original ReALE method was to create a robust method, that would allow the computation of complex flows with large shear deformation without user intervention and while maintaining "Lagrangian" characteristics. The key to achieving these goals was to allow connectivity changes during the rezone stage of the method.

Both methods use the same Lagrangian phase - it can be any method which can deal with arbitrary polygonal meshes. In [42] we present results for ReALE using both staggered and cell-centered discretizations - one can find relevant references to Lagrangian discretizations in that paper; we will not repeat it here. In this paper our Lagrangian phase is based on cell-centered discretization introduced in [43, 44] and also briefly described in [42].

Both methods use intersection-based remap [19, 23, 36]. The remapping phase is conceptually fairly simple. The quantities on the old Lagrangian mesh are cell-centered density, velocity and total energy that must be transferred on the rezoned mesh. First piecewise linear representations of cell-centered variables are constructed on the Lagrangian mesh. Then a slope limiting process [53] is performed to enforce physically justified bounds. Conservative quantities, namely mass, momentum and total energy, are obtained by integration of these representations. New conservative quantities are calculated by integration over polygons of intersection of new (rezoned) and old (Lagrangian) meshes. Finally, primary variables are simply recovered by division by new volume (for density) or new mass (for momentum and energy). Clearly, because in general we have to remap between two meshes with different connectivity, we need to know with which cells of old mesh cell of new mesh has to be intersected. In our case it does not require global searches because we trace how generators which define new mesh are moving

throughout the old mesh during adaptation process.

The real difference between ReALE and R-ReALE is in the rezone phases. However, connectivity changes in both methods are performed using the machinery of the Voronoi diagrams.

The comparison of the accuracy and performance of two methods will be demonstrated on numerical examples in the special section.

Now we describe the rezone stages for these two methods.

4.1 Rezone phase of standard ReALE method

The details of the rezone phase for standard ReALE method can be found in [42]. In this Section we only give some short description of this phase to emphasize difference between rezone stages of standard and adaptive ReALE. do not change in time. At time t^n mesh is constructed using position of generators $\mathbf{G}_i^n$, that is Lagrangian step starts with mesh which is Voronoi mesh, which correspond to $\mathbf{G}_i^n$, we denote this mesh L^n. During Lagrangian step generators play no role. The result of the Lagrangian phase is mesh L^{n+1} on which we have all flow data. Mesh L^{n+1} is not Voronoi mesh, but has the same connectivity as mesh L^n. The rezoned mesh R^{n+1} is obtained as follows. First, one defines "Lagrangian" position of generators $\mathbf{G}_i^{L,n+1}$ at t^{n+1} these positions are obtain by moving generators from their positions $\mathbf{G}^n$ with some "Lagrangian" velocity averaged from vertices of the cell i - $\mathbf{u}_i^L$

$$\mathbf{G}_i^{L,n+1} = \mathbf{G}_i^n + \mathbf{u}_i^L \Delta t.$$

Then one computes position of centroids $\mathbf{C}_i^{L,n+1}$ of the cell i of the mesh L^{n+1}. Finally, positions of generators $\mathbf{G}_i^{n+1}$ which will define rezoned mesh R^{n+1} are defined as follows

$$\mathbf{G}_i^{n+1} = \mathbf{G}_i^{L,n+1} + \omega_i(\mathbf{C}_i^{L,n+1} - \mathbf{G}_i^{L,n+1}), \quad 0 \le \omega_i \le 1, \tag{4.1}$$

where weight ω_i defined based on eigen values of deformation tensor for cell i and ω_i is zero under translation or solid rotation. There are several design principles for using (4.1) to define positions of the generators: first - it keeps mesh close to Lagrangian, second - it allows to maintain smooth mesh, because $\mathbf{C}_i^{L,n+1}$ correspond to one Lloyds iterations toward centroidal Voronoi mesh, which is smooth mesh, and finally choice of the parameter ω_i allows us not to perturb initial mesh until time when mesh start to deform at particular location.

In the context of adaptation we want to emphasize that the rezone strategy suggested in [42] keeps cells "Lagrangian" on average, that is shape and connectivity of the cells are changing but movement of the cell is "Lagrangian" on average. This is clearly demonstrated in [42] (Section 9). Therefore, standard ReALE has the adaptive properties of typical Lagrangian method. That is, it preserves initial contact discontinuity, and adapt to shock. However, they do not adapt to rarefaction and do not adapt to complicated

flow structures which may appear during calculations, like interaction of the shock with contact, see for example, [37].

4.2 Rezone phase of adaptive R-ReALE method

For R-ReALE the number of cell does not change in time. The monitor function at each time step is given by its value on mesh L^{n+1}. Initial mesh and mesh on each time on rezone phase is adapting to current monitor according to equidistribution principle. The construction of the mesh which satisfies equidistribution principle is performed as described in Section 2.4.

The input data is monitor function and number of generators, N. On each time step it is reaching different level of equidistribution

$$E(t) = \left(\int_{\Omega(t)} \phi(\mathbf{x},t)\, dV \right) \Big/ N,$$

because number of cells *does not* change, but global integral of monitor function *does* change because monitor and potentially computational domain may change in time.

5 How to compare ReALE and R-ReALE methods?

As we have mentioned in the Background and Rationale Section, the CPU time for different methods strongly depend on implementation and computer architecture - for this reason we do not think that presenting CPU time adds any value to this paper.

However, we still want to compare methods. We have chosen to compare different characteristics of the methods to obtain numerical solution for chosen final time T. Clearly one of most important characteristics is error at time T. However, one need to consider how many time steps one need to take to reach time T; how many cells were used in overall process (that is, sum of number of cells used on all time steps), spatial resolution achieved, and so on. The most universal measure of the method is convergence rate. However, the investigation of convergence rate assumes some error model. The usual approach for Eulerian methods is to choose some parameter h and use model of form $\varepsilon = C_h h^p + C_t^s \delta t$. Because for explicit methods $\delta t \sim h$ one can use the following model $\varepsilon = \tilde{C}_h h^q$. The choice of h for Eulerian methods on uniform mesh is obvious, however if one considers adaptive Eulerian methods which use adaptive mesh refinement the choice of h is not obvious. The situation is more complicated for pure Lagrangian methods because even number of cells in the initial mesh can be the same, the results strongly depend on choice of initial mesh. The situation with standard ALE methods even more complicated because results depend on rezone strategy, which may change depending on the resolution.

In R-ReALE method there are several parameters. If parameters β and α, which participate in raw monitor transformation are fixed, the there is only one parameter N -

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

which is number of cells, which is fixed in time. Therefore, it looks like that comparison of ReALE and R-ReALE methods suppose to be easy because both methods use the fixed number of generators. However, one still need to decide what is initial mesh. For R-ReALE initial mesh is adapted to initial data according to monitor and equidistribution principle. It is almost obvious that such mesh will not be optimal for standard ReALE method and uniform initial mesh can be better choice, because it will adapt to the solution according to averaged Lagrangian motion.

For both methods we can compute some estimate of the norm of the spatial error at the particular time moment using exact solution (if it is known) or reference solution obtained using high-resolution calculations. For example, we can compute L1 error for density as follows

$$e(t^n)=\int_\Omega |\rho(\mathbf{x},t^n)-\rho_{ext}(\mathbf{x},t^n)|d\mathbf{x}=\sum_i \int_{\Omega_i} |\rho(\mathbf{x},t^n)-\rho_{ext}(\mathbf{x},t^n)|d\mathbf{x}, \tag{5.1}$$

where $\rho_{ext}(x,t^n)$ is exact or reference solution and numerical solution $\rho(\mathbf{x},t^n)$ is obtained ρ_i^n by some piece-wise linear reconstruction. The integrals on the right hand side of (5.1) are computed using the numerical integration introduced in Section 2.4.2.

Any convergence analysis assume some error model. In our case, it is natural to use the following model

$$e(T)=C(T)\left[N^{-1/2}\right]^{q(T)}. \tag{5.2}$$

In this formula $N^{-1/2}$ plays role of h, $q(T)$ is order of convergence and $C(T)$ constant, which does not depend on mesh. Let us mention that, in general, order of convergence may depend on T.

Assuming that this error model is correct and knowing errors $e^1(T)$ and $e^2(T)$ and corresponding N^1 and N^2 for two different calculation using the same method, but different number of cells, we can estimate convergence rate as follows

$$q(T)=2\log\left(\frac{e^1(T)}{e^2(T)}\right)\Big/\log\left(\frac{N^1}{N^2}\right). \tag{5.3}$$

To prove that error model is correct one need to use at least three calculations with different number of generators $N^1<N^2<N^3$ and compute $q(t)$ using pairs $e^1(T)$ and $e^2(T)$ $e^2(T)$ and $e^3(T)$. If two computed in such way $q(T)$ are close to each other then it is indication that model can be used for estimating convergence rate.

After $q(t)$ is defined $C(T)$ is deduced as follows

$$C(T)=e(T)/\left[N^{-1/2}\right]^{q(T)}. \tag{5.4}$$

The methods can be compared first with respect to convergence rate $q(T)$. If convergence rate is the same then one can compare $C(T)$.

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

Here we need to make some comments about how to choose sequence of meshes to estimate convergence. For ReALE method if we choose initial mesh to be uniform then there is only one parameter N - the total number of cells in the mesh - therefore to perform convergence analysis for ReALE method we can choose meshes with N, $2N$ and $4N$ number of cells.

For R-ReALE method the situation is not that simple because it has also parameters α and β. Convergence analysis assumes that with mesh refinement meshes stay "similar" in the sense that refinement pattern are similar. There is no problem with "similarity" with respect to parameter α because it controls smoothness of the mesh and it make sense to keep it the same with mesh refinement.

There are two main reasons why we may want to choose $\beta < 1$. First reason is to avoid zero values of the monitor. For this purpose we can choose some β close to 1, let say $\beta = 0.9$. This is what we do in Section 6.1. The second reason is to avoid small A_{min} which leads to a small time step for problems with high sound speeds. This is the case for the Sedov problem considered in Section 6.2. In this problem at $t = 0$ there is a large sound speed in the small area where the mesh is refined. To mitigate the problem of small time step we can choose some β which increases ratio A_{min}/A_{max} as described in Section 3.2, which effectively increases A_{min} and decreases A_{max}. For each problem the corresponding parameter β can be chosen differently according to user specification and the desired level of adaptation; note that $\beta = 0$ corresponds to uniform mesh.

Now the question is do we need to change ratio A_{min}/A_{max} when we investigate convergence and refine a mesh to keep meshes of different resolution "similar". Let us assume that we have chosen β^N at coarsest level to enforce the chosen ratio A^N_{min}/A^N_{max}. Our design principle for definition of $A^{2N}_{min}/A^{2N}_{max}$ and $A^{4N}_{min}/A^{4N}_{max}$ is based on desire to keep mesh "similar". To do this we choose to change this ratio as it will change for raw monitor. According to (3.7) for raw monitor we have

$$\frac{\phi^{h/k}_{min}}{\phi^{h/k}_{max}} \sim \frac{1}{k}\frac{\phi^{h}_{min}}{\phi^{h}_{max}}, \tag{5.5}$$

and therefore

$$\frac{A^{h/k}_{min}}{A^{h/k}_{max}} \sim \frac{1}{k}\frac{A^{h}_{min}}{A^{h}_{max}}. \tag{5.6}$$

Now we need to decide what is h and what is k in our case. We have already decided that role of h played by $N^{-1/2}$. Therefore, h/k will be $N^{-1/2}/\sqrt{k}$. It means that $h/2$ corresponds to $2N$ and therefore we want to reduce ratio of A_{min}/A_{max} by $\sqrt{2}$ when we go from mesh with N cells to mesh with $2N$ cells:

$$\frac{A^{2N}_{min}}{A^{2N}_{max}} \sim \frac{1}{\sqrt{2}}\frac{A^{N}_{min}}{A^{N}_{max}}. \tag{5.7}$$

In general, when investigating convergence of R-ReALE method we will choose β to

enforce that

$$\frac{A^{M*N}_{min}}{A^{M*N}_{max}} \sim \frac{1}{\sqrt{M}} \frac{A^{N}_{min}}{A^{N}_{max}}. \tag{5.8}$$

The useful information for comparison of the different methods also can be obtained if one choose number of cells in different methods such that error at final time is the same. Then one can compare number of needed generators in different methods as well as number of time steps needed to reach final time. This information can be useful for analysis of relative efficiency of different methods.

In next Sections we will compare ReALE and R-ReALE methods.

6 Numerical examples

In all R-ReALE calculations pressure is used for computing monitor. The parameter α used in smoothing is set to 1. The scaling parameter β is different for different problems and specified in the appropriate Section.

6.1 "Smooth-to-Shock" problem

We consider a 1D flow of ideal gas with $\gamma=3$ on a domain $[-9,9]\times[-0.9,0.9]$. The initial states are

$$\rho^0(x,y)=\frac{1}{\sqrt{\gamma}}\left(2+0.5e^{-(x/0.4)^2}\right), \quad p^0=(\rho^0)^\gamma, \quad \mathbf{u}^0=\mathbf{0}.$$

The domain boundary is treated as reflective walls. According to (A.8) of Appendix A, smooth solution for the test problem exists until $t_{shock}\approx 0.932663$. After $t=t_{shock}$, the solution develops shock waves. We perform ReALE and R-ReALE simulations to $t=2$. Since the analytical solution can only be derived for $t<t_{shock}$ (see Appendix A for details), we use a 1D staggered Lagrangian code based on compatible discretization methodology,

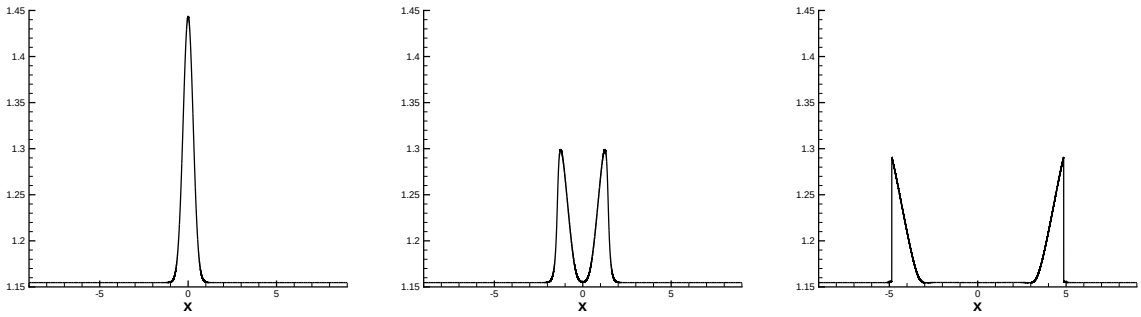


Figure 8: The density profile of the reference solution for the "smooth-to-shock" solution test at $t=0,0.5,2$ (from the left to the right).

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

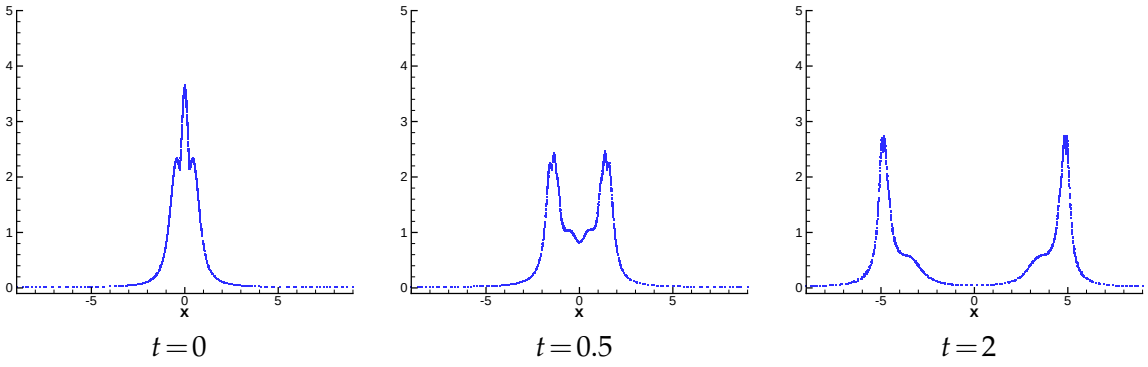


Figure 9: Smooth-to-shock problem. The scatter plot of the monitor function for R-ReALE with 1000 generators.

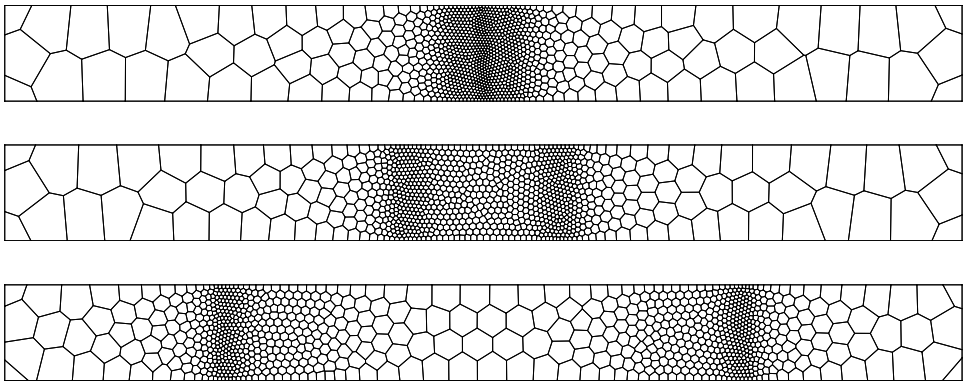


Figure 10: Smooth-to-shock problem. The meshes of R-ReALE with 1000 generators. $t=0,0.5,2$ from top to bottom.

[7], with 160000 cells to generate a reference solution. Fig. 8 shows the density profile of the reference solution at $t=0,0.5,2$.

To compare the accuracy of ReALE and R-ReALE for both smooth flow and shocks, we compute the L1 density error at $t=0.5$ and $t=2$. The simulations with 1000, 4000 and 16000 generators for both ReALE and R-ReALE are performed. For this test problem we use fixed $\beta=0.9$.

Fig. 9 shows the monitor function for case of 1000 generators at $t=0,0.5,2$. The meshes for the corresponding time are shown in Fig. 10. One can see that the monitor function controls the distribution of the cell area through the equidistribution principle. Since the integral of the monitor function on each cell is approximately a constant (Fig. 11), the area of the corresponding cell on a CCVT is approximately inverse proportional the monitor function as it can be seen in Fig. 12.

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

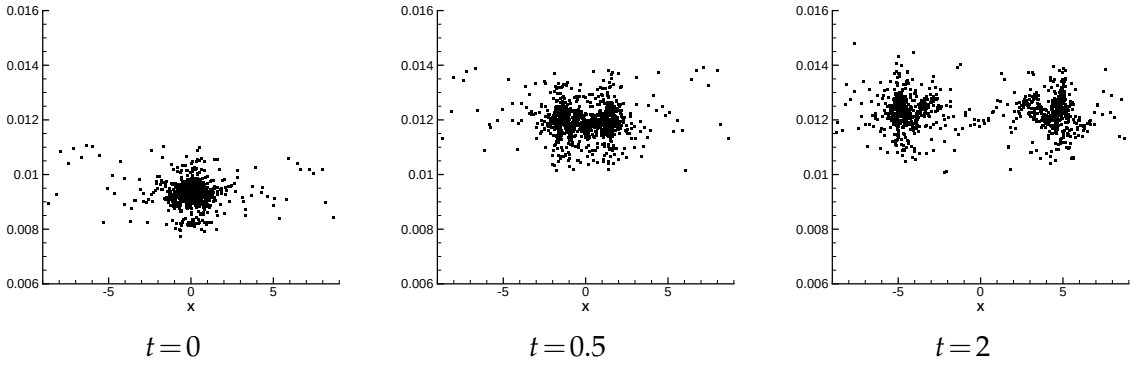


Figure 11: Smooth-to-shock problem. The scatter plot of the cell equidistribution level E_i.

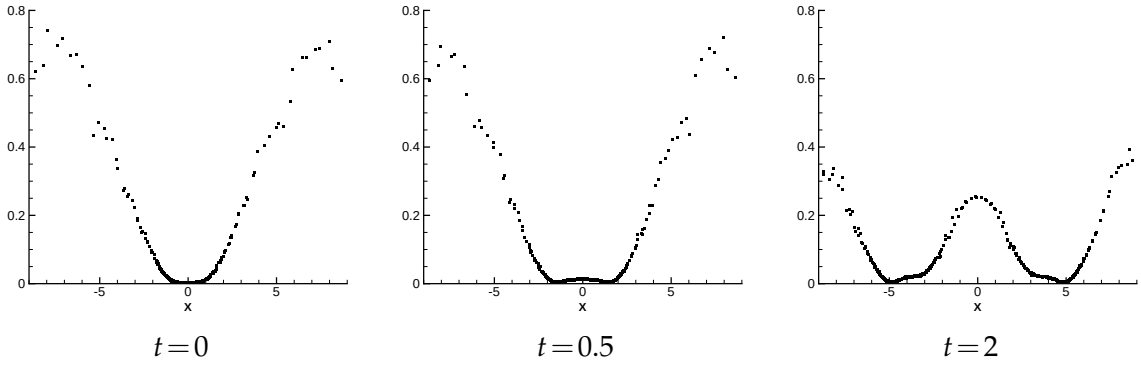


Figure 12: Smooth-to-shock problem. The scatter plot of the cell area.

In this test case, we solve a 1D problem in a 2D domain and CCVT algorithm producing 2D meshes when used in 2D domain. Therefore, is is important to check how well the numerical algorithm preserves the 1D symmetry of the solution. In Fig. 13 we plot the density isolines - it can be seen that the 1D symmetry of the solution is well preserved.

We compare the above results with the ReALE results using 1000 generators. One can see from Fig. 14 that the meshes show little distortion from the uniform one and the cell area on the region with non-constant density is much larger than that of R-ReALE. The Fig. 15 shows comparison of the density profiles for ReALE and R-ReALE on zoomed regions. It can be seen that R-ReALE has better accuracy due to smaller cells on the large curvature region.

To compare the rate of convergence of ReALE and R-ReALE, we perform simulations using 1000, 4000 and 16000 generators. Fig. 16 shows the history of the L1 density error. When $t < t_{shock}$, the solution is smooth and the error is increasing over time. After the shock forms, the error does not increase.

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

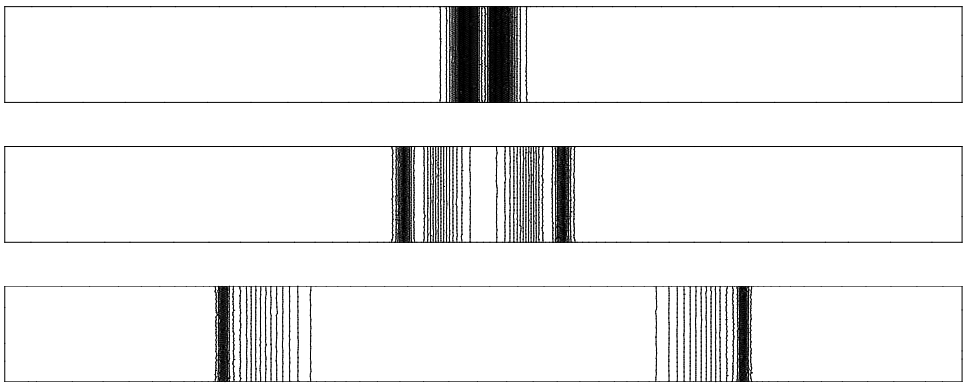


Figure 13: Smooth-to-shock problem. The density isolines from the R-ReALE with 1000 generators. $t=0,0.5,2$ from top to bottom.

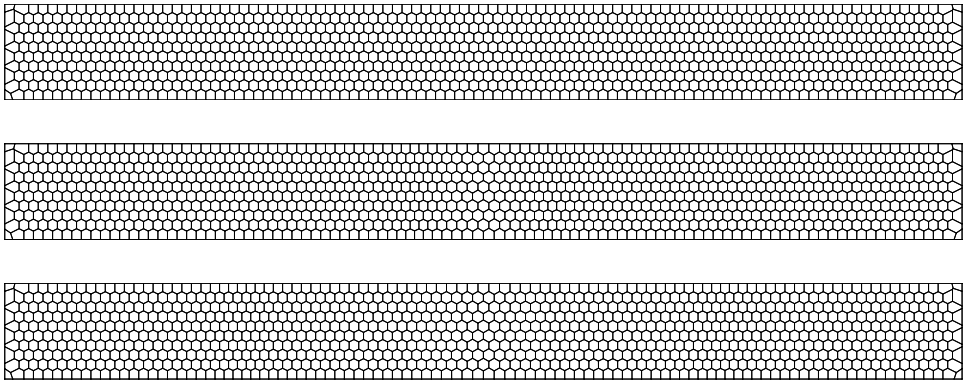


Figure 14: Smooth-to-shock problem. The mesh of ReALE with 1000 generators. $t=0,0.5,2$ from top to bottom.

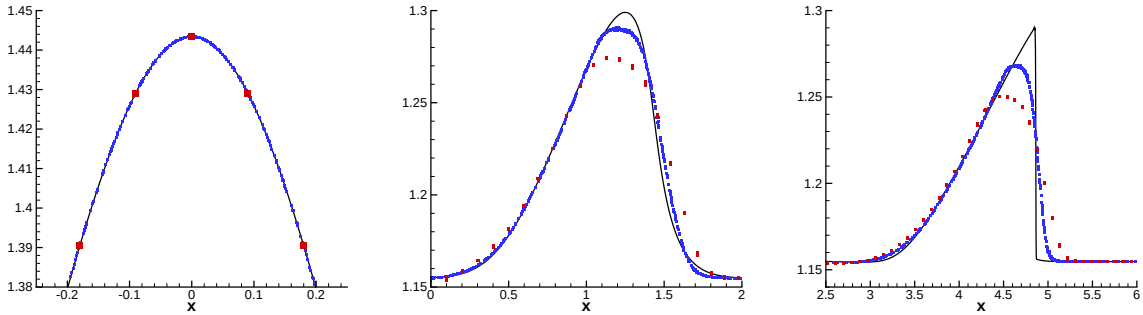
To further study the convergence property for both ReALE and R-ReALE, we fit $C(t)$ and $q(t)$ of the error model using the L1 density error from $N=4000$ and $N=16000$ calculations - Table 1. For smooth solution at $t=0.5$, both methods show 2-nd order convergence. However, the coefficient $C(t)$ for R-ReALE is 4 times smaller than for ReALE. We offer the following qualitative explanation of this fact as follows. We define [‖] a cell as a active cell if

$$\phi(\mathbf{x}_c,t)\geq 0.1\bar{\phi}(t)$$

where $\mathbf{x}_c$ is the centroid of the cell. At a fixed time t, the number of inactive cells can be

[‖] Let us mention that factor 0.1 in this definition has been chosen arbitrary.

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47



$t=0$ $t=0.5$ $t=2$

Figure 15: Smooth-to-shock problem. The scatter plot of density with ReALE and R-ReALE with 1000 generators. Black line: the reference solution. Red dots: ReALE, Blue dots: R-ReALE.

Table 1: Smooth-to-shock problem. The constants of the error model for ReALE and R-ReALE.

Time	method	$C(t)$	$q(t)$
0.5	ReALE	79.2	1.96
	R-ReALE	20.5	2.06
2	ReALE	4.7	1.13
	R-ReALE	56.3	2.05

estimated as

$$N_{inactive} \approx \frac{1}{E}\int_{\{\mathbf{x}|\phi(\mathbf{x},t)<0.1\bar{\phi}(t)\}} \phi(\mathbf{x},t)d\mathbf{x} \leq 0.1\frac{1}{E}\int_{\{\mathbf{x}|\phi(\mathbf{x},t)<0.1\bar{\phi}(t)\}} \bar{\phi}(t)d\mathbf{x}$$
$$\leq 0.1\frac{1}{E}\int \bar{\phi}(t)d\mathbf{x} = 0.1\frac{1}{E}\int \phi(\mathbf{x},t)d\mathbf{x} = 0.1N.$$

The total number of active cells at time t can be estimated as

$$N_{active} = N - N_{inactive} = 0.9N. \tag{6.1}$$

This means more than 90% cells are active for R-ReALE at any time step. For ReALE, because the meshes are close to a uniform one, the number of active cells depends on the solution. For this test problem, number of the active cells for ReALE at $t=0.5$ is less then 20%. Therefore, the ratio of active cells between R-ReALE and ReALE is $90\%/20\% = 4.5$, which is consistent with the ratio of $C(t)$ at $t=0.5$.

At $t=2$, shock forms and ReALE shows 1-st order convergence and R-ReALE shows 2-nd order convergence. We plan to give the rigorous explanation of this fact in the future paper.

To give an idea about relative efficiency of ReALE and R-ReALE, we calibrate the number of generators of ReALE so that ReALE gives similar L1 error as R-ReALE does

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

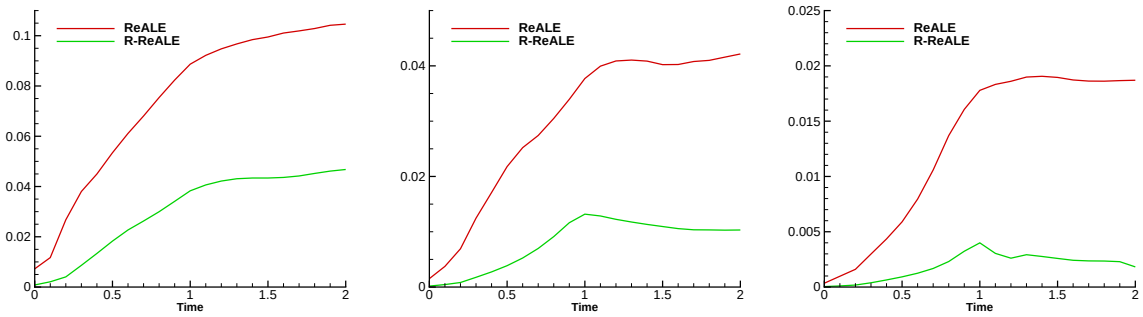


Figure 16: Smooth-to-shock problem. L1 density error vs. time for ReALE and R-ReALE

Table 2: Smooth-to-shock problem. Number of generators needed for ReALE to reach a similar L1 error as R-ReALE.

Time	method	N	$e(t)$	ratio of N	time steps
0	R-ReALE	4000	1.74e-04		0
	ReALE	31360	1.70e-04	7.84	0
0.5	R-ReALE	4000	3.86e-03		191
	ReALE	25000	3.81e-03	6.25	62
2	R-ReALE	4000	1.03e-02		957
	ReALE	49000	9.98e-03	12.25	243

at $t=0,0.5,2$. From Table 2, one can see that at $t=0.5$, the ratio of the generator number is 6.25 and for $t=2$ it is 12.25. That is R-ReALE more "efficient" with respect to ReALE for shocks then for smooth solutions.

Another metric related to efficiency is notion "equivalent" uniform mesh. This metric widely used in AMR community, [47]. Given a mesh from a R-ReALE simulation at a fixed time, we define the equivalent number of generators in a uniform mesh as

$$N_{eq}=A/A_{\min},$$

where A is the area of the domain and $A_{\min}$ is the minimal cell area of the R-ReALE mesh. The N_{eq} and ratio N_{eq}/N are presented in Table 3; For 1000 generators, the R-ReALE mesh is too coarse and the shock is smeared, which leads to a lower ratio at $t=2$ than at $t=0$. For smooth solution at $t=0$ and $t=0.5$, the ratio is almost constant for $N=4000$ and $N=16000$. For discontinuous solution at $t=2$, the ratio increases as we refine the mesh, which means R-ReALE becomes more "efficient" on the fine mesh. This is consistent with the data in the Table 1 - R-ReALE has higher convergence order than ReALE for shock waves.

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

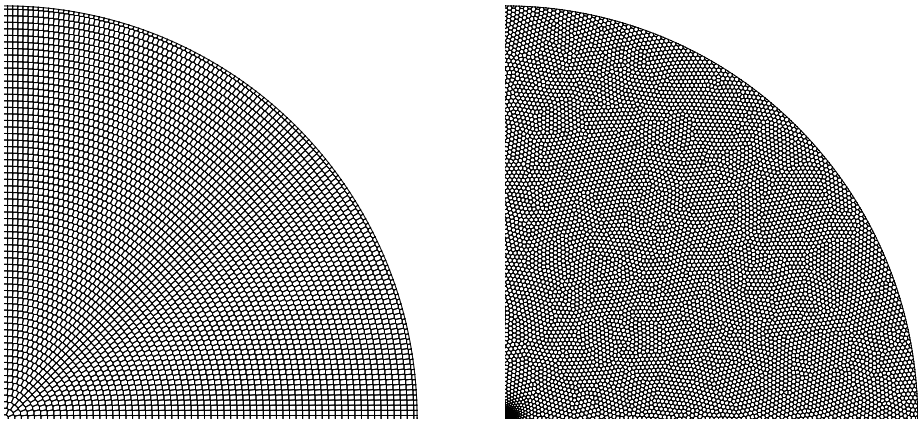


Figure 17: Sedov problem. The initial mesh with 16000 generators on a quarter of the domain: ReALE - left panel, R-ReALE - right panel

Table 3: Smooth-to-shock problem. N_{eq}: the equivalent number of generators in a uniform mesh and ratio N_{eq}/N - in parenthesis.

N	$t=0$	$t=0.5$	$t=2$
1000	14087 (14.08)	6480 (6.48)	7363 (7.36)
4000	68790 (17.9)	40500 (10.12)	115710 (28.92)
16000	308570 (19.28)	185140 (11.57)	1604000 (100.2)

6.2 Sedov blastwave

We consider the Sedov blastwave in a uniform medium. In the initial time, the total amount of released energy equals to 1 and it is deposited in the center of the domain. The exact solution based on self-similarity arguments is available [34]. In numerical simulations, we deposit the released energy uniformly on a disk with radius $r_0=0.02$. The initial mesh for ReALE calculation is a "pseudo-polar" mesh which is Voronoi mesh for generators distributed along the circles, such that the distance between them is approximately equal along the circle and distance between circles is the same. In Fig. 17 (left panel) we show such mesh in one quarter of computational domain for 16000 generators. The same mesh is used as reference mesh to construct monitor at $t=0$ for R-ReALE.

The initial states are given by

$$(\rho,e,\mathbf{u})=\begin{cases}(1,1/(\pi r^2),\mathbf{0}) & r\le r_0\\ (1,10^{-6},\mathbf{0}) & r>r_0\end{cases},$$

where $r=(x^2+y^2)^{1/2}$. These states are projected to "pseudo-polar" mesh mesh. The final time of the simulations is $t=1$. The computational domain is a disk with radius 1.2.

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

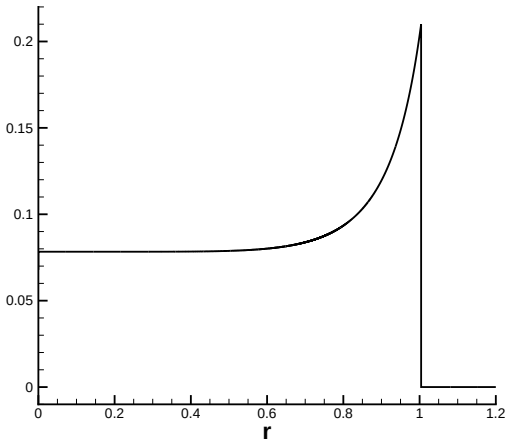


Figure 18: Sedov problem. The pressure of the reference solution along radial direction at $t=1$.

Table 4: Sedov problem. L1 norm pressure error for ReALE and R-ReALE: gens. is the number of generators. $e(1)$, $q(1)$ are the L1 pressure error and order at $t=1$.

method	gens.	$e(1)$	$q(1)$	time steps
ReALE	16000	1.349e-02		1236
	32000	8.554e-03	1.31	1492
	64000	5.851e-03	1.09	1716
R-ReALE	16000	1.006e-02		1894
	32000	5.235e-03	1.68	3273
	64000	2.917e-03	1.68	5740

The initial adapted mesh (in one quarter of the computational domain) for R-ReALE (β is chosen such that $A_{max}/A_{min}=115$) is shown in Fig. 17 (right panel).

The exact solution depends only on r, [49]. To check the accuracy of the numerical solution, we compute a reference solution with 20000 cells using the 1D code from [34] - see Fig. 18 for the pressure of the reference solution at $t=1$.

We first compare ReALE and R-ReALE based on the pressure from the simulations with 16000 generators. The top left panel in Fig. 19 shows the mesh from ReALE. Due to the shock compression, the cell area on the shock is smaller than the initial one and is increasing gradually towards the center of the domain. The bottom left panel in Fig. 19 shows the mesh from R-ReALE. All the small cells are on the shock. The mesh is adapted to the solution. From the right column in Fig. 19, one can see that both ReALE and R-ReALE preserve symmetry well.

To compare the performance of ReALE and R-ReALE, we present the simulation results with 16000, 32000 and 64000 generators. The β is chosen in such way that A_{max}/A_{min}

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

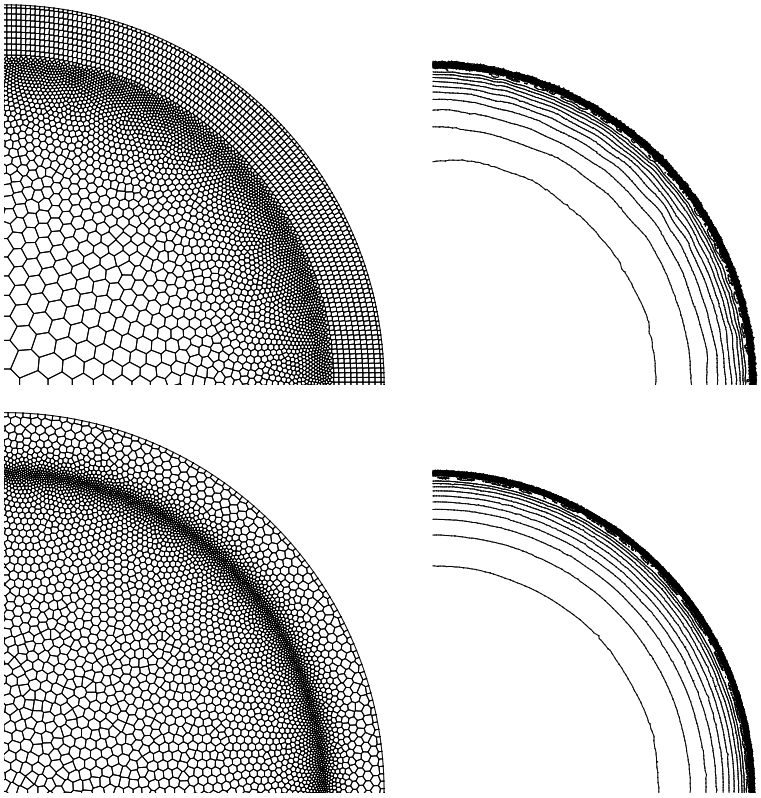


Figure 19: Sedov problem - $t=1$, 16000 generators. The mesh and pressure contour on a quarter of the domain. Top: ReALE, Bottom: R-ReALE.

for R-ReALE are 115, 160 and 230 correspondingly. To show how the mesh is adapted to the solution in both cases, we present the zoomed view of the mesh and the pressure color map for calculation with 64000 generators in Fig. 20. One can see that R-ReALE has smaller cells near the shock and cells are more smooth. The Fig. 21 shows the cell area plots for ReALE and R-ReALE calculations along the radial direction.

To check if an approximate CCVT is obtained at $t=1$, the scatter plots of the monitor function and its integrals over cell for all cells are given in Fig. 22. One can see that the deviation of the integrals from equidistribution is about 15%. This number can be reduced if one increases accuracy of CCVT iteration.

The scatter plot of the pressure along radial direction is given in Fig. 23. The ReALE shows overshoot near the shock and worse symmetry in comparison with R-ReALE, which may be due to the fact that mesh for ReALE is not smooth "enough". The Fig. 24 shows the history of the L1 error in pressure. The L1 pressure error of ReALE is about twice that of R-ReALE.

The errors at $t=1$ for different level of mesh refinement are given in Table 4. One

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

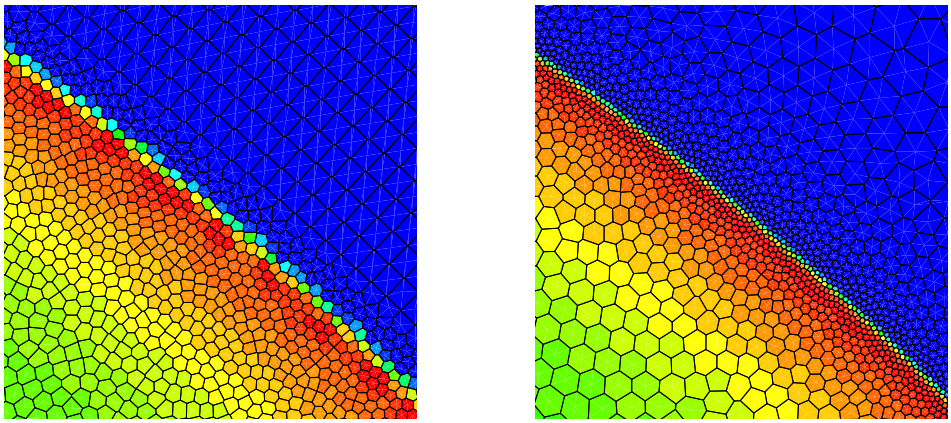


Figure 20: Sedov problem - $t=1$, 64000 generators. The zoomed view of mesh and pressure contour. The view window is $[0.56,0.7]\times[0.7,0.84]$.

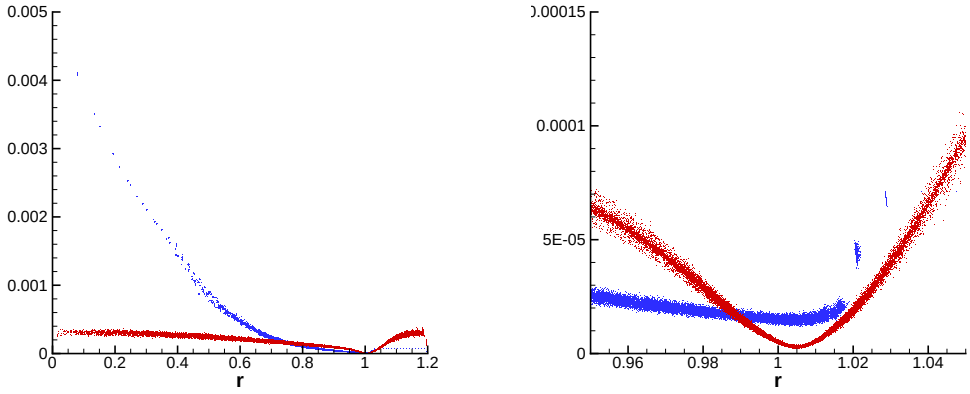


Figure 21: Sedov problem - $t=1$, 64000 generators. The scatter plot of the cell areas along the radial direction. Blue dots: ReALE, Red dots: R-ReALE. Left: on the entire domain. Right: the zoomed view on the shock.

can see that ReALE shows approximately 1-st order convergence and R-ReALE shows approximately 2-nd order convergence.

In the same Table we present number of time steps needed to reach final time. For Sedov blastwave, the specific internal energy reaches its maxima in the center. Thus the time step is determined by the cell diameter and the specific internal energy for the cell in the center due to CFL condition. Since R-ReALE has smaller cell on the center, it needs more time steps for each simulation.

The equivalent number of generators on a uniform mesh N_{eq} and the ratio N_{eq}/N are given in Table 5. One can see that the "efficiency" of R-ReALE is decreasing over time

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

Table 5: Sedov Problem. N_{eq}: the equivalent number of generators on a uniform mesh and ratio N_{eq}/N - in parenthesis.

N	$t=0$	$t=0.5$	$t=1$
16000	205630 (12.8)	141370 (8.83)	122270 (7.64)
32000	887040 (27.7)	514080 (16.0)	430850 (13.4)
64000	3231400 (50.4)	1966900 (30.7)	1809600 (28.7)

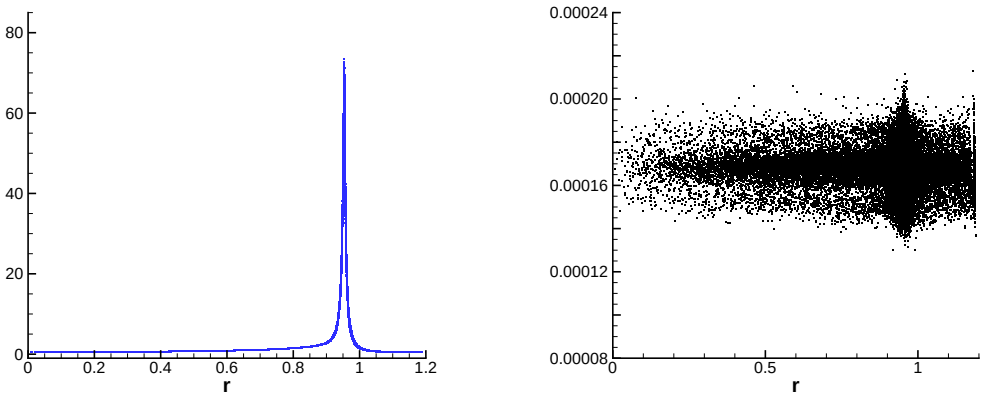


Figure 22: Sedov problem - $t=1$, 64000 generators. The scatter plot of the monitor function - left panel, and its integral over the cells along the radial direction - right panel.

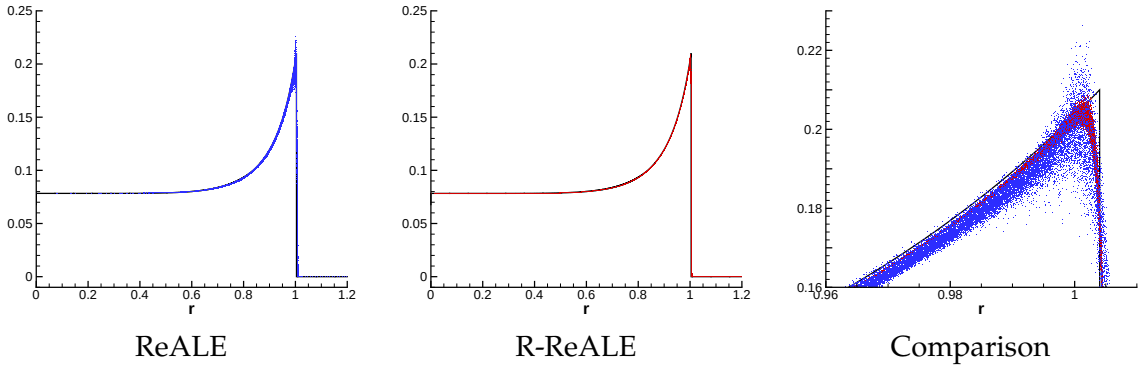


Figure 23: Sedov problem - $t=1$, 64000 generators. The scatter plot of pressure along the radial direction. Black line: reference solution. Blue dots: ReALE, Red dots: R-ReALE. The right figure compares ReALE and R-ReALE near a zoomed region on the shock.

for fixed N, however, the "efficiency" of R-ReALE is increasing by a factor of 2 when the number of generators increases by a factor of 2.

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

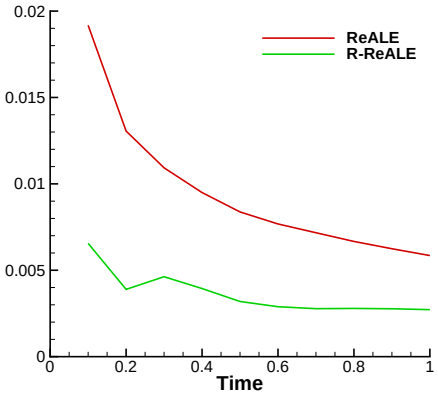


Figure 24: Sedov problem - 64000 generators: L1 pressure error vs. time.

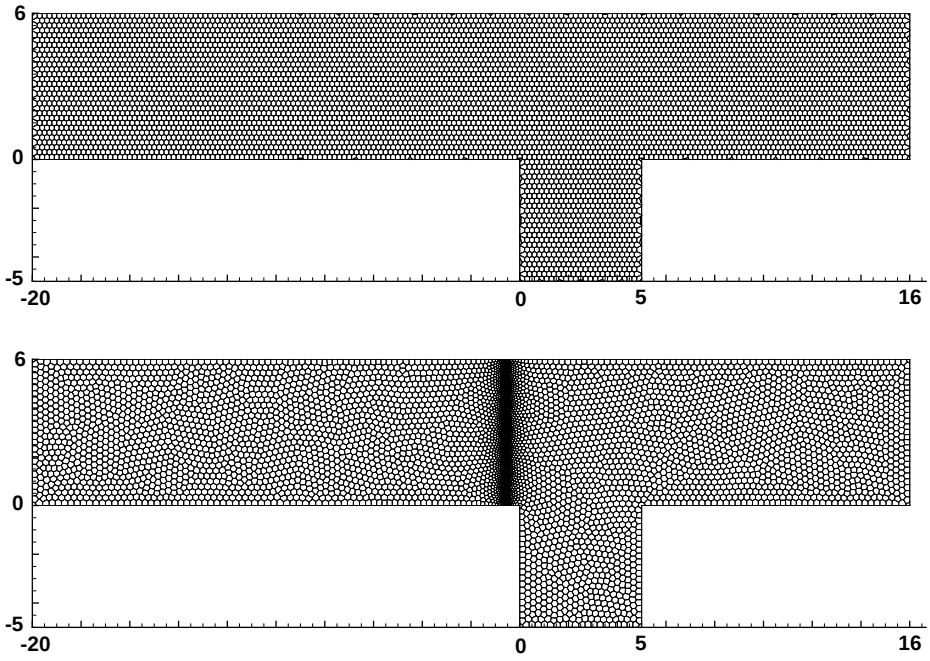


Figure 25: Shock cavity interaction. The initial mesh with 6000 generators. Top: ReALE, Bottom: R-ReALE.

6.3 Shock cavity interaction

In this section we present results for the problem of interaction between a planar shock wave and a square cavity, [31]. Numerical study of this problem using ReALE was performed in [26]. Here, we compare the simulation results from ReALE and R-ReALE with

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

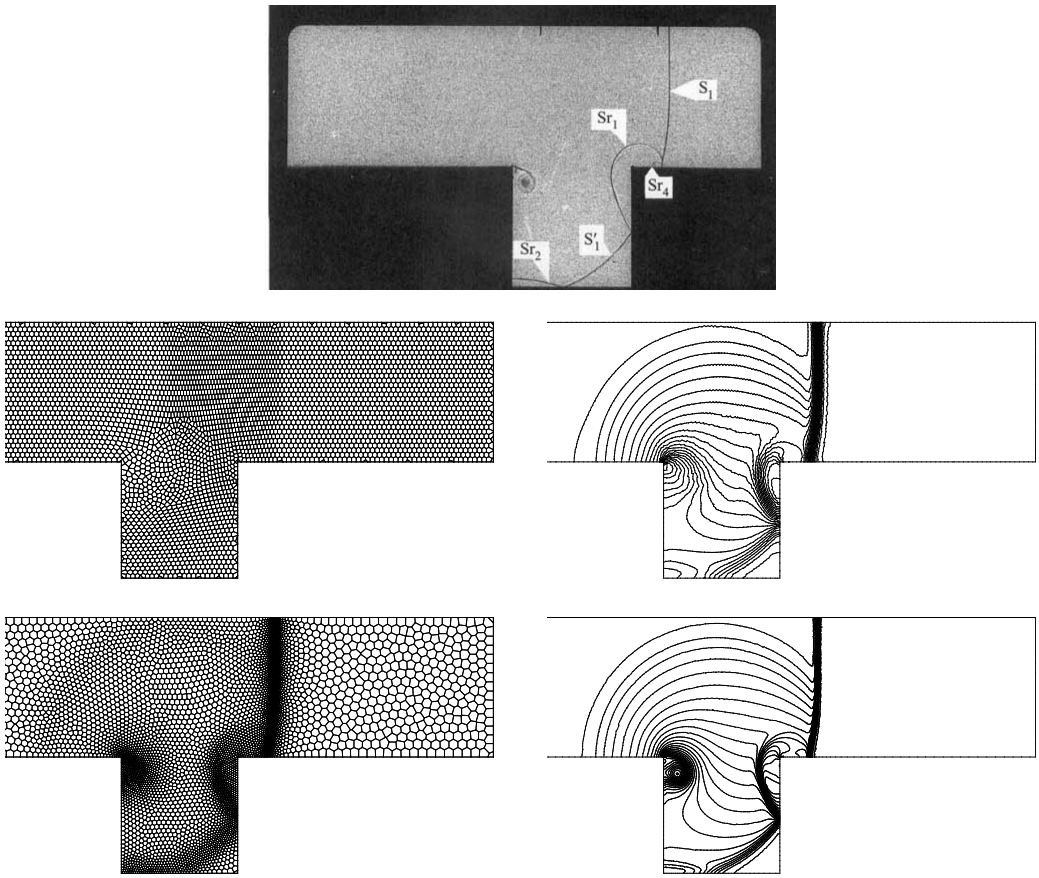


Figure 26: Shock cavity interaction. Top: shadowgraph from [31]. Mesh and pressure contour plots at $t=160$ with 6000 generators: middle - ReALE; bottom - R-ReALE.

the experimental shadowgraphs from [31]. The computational domain is shown in Fig. 25. The initial states are

$$(\rho,p,u,v)=\begin{cases} (1.73\times10^{-3},1.75,15.3,0) & x<-0.575 \\ (1.14\times10^{-3},0.97,0,0) & x\geq-0.575 \end{cases},$$

which correspond to an incident shock with Mach number $M_s=1.3$. The equation of state is ideal gas with γ = 1.4. On the left-most boundary, a piston condition is used. It moves with velocity $(15.3,0)$ to maintain shock. All other boundaries are reflective walls. Three simulations with 6000, 12000 and 24000 generators are performed with ReALE and R-ReALE. The β is chosen in such way that A_{max}/A_{min} for R-ReALE are 160, 226 and 320 correspondingly. The initial meshes for ReALE and R-ReALE with 6000 generators are given in Fig. 25.

The Fig. 26 shows the mesh and pressure contour plots at $t=160$ as well as the

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

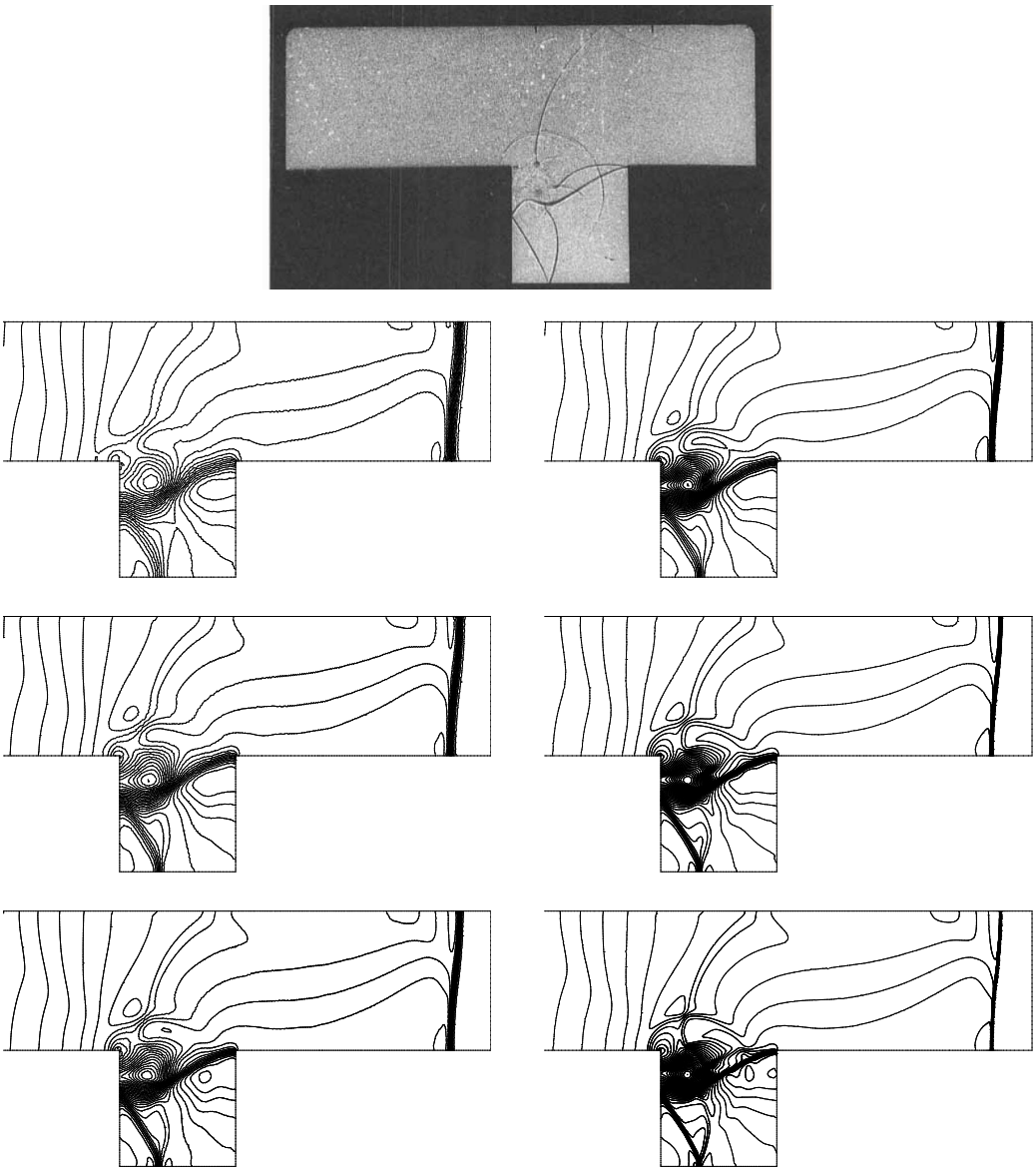


Figure 27: Shock cavity interaction. Very top: shadowgraph from [31]. Pressure contour plots at $t=340$. Left: ReALE. Right: R-ReALE. Top to bottom: $N=6000,12000,24000$.

experimental shadowgraph from [31]. The original shock is reflected by the wall of the cavity and forms three reflected shocks. The incident shock S_1 is strongest among all the shocks. A vortex forms near the upper-left corner of the cavity. From the left column of Fig. 26 one can see that on the shocks, the mesh from R-ReALE has smaller cells than ReALE. One major difference between the two simulations is that R-ReALE has much

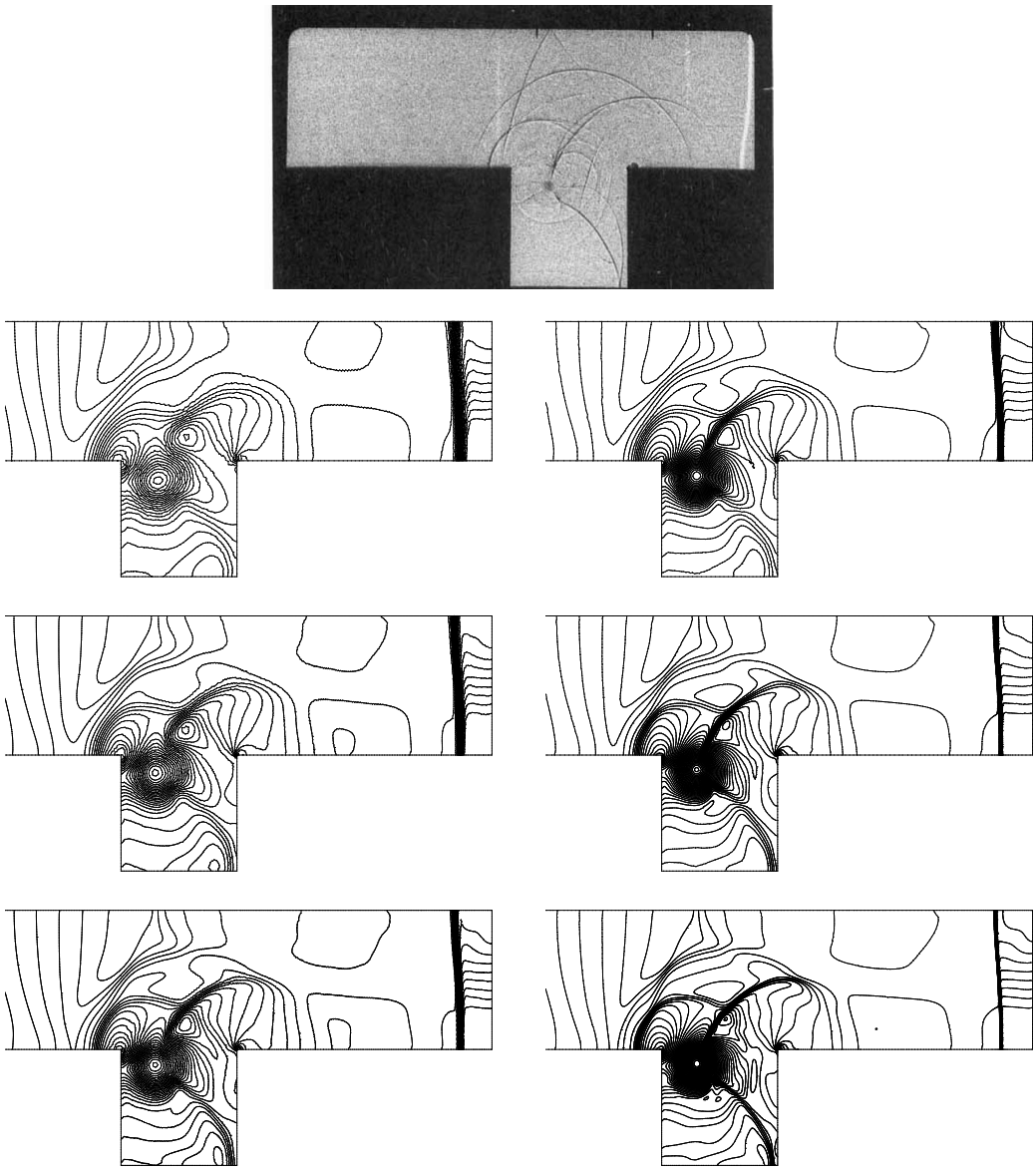


Figure 28: Shock cavity interaction. Very top: shadowgraph from [31]. Pressure contour plots at $t=420$. Left: ReALE. Right: R-ReALE. Top to bottom: $N=6000,12000,24000$.

better resolution for the vortex. This is because in the ReALE simulation, the diffracting shock moves generators into the shocked regions, which makes the mesh coarser on the vortex. For the R-ReALE simulation, the large density variation of vortex is detected by the monitor function and more generators are placed on the vortex. The shocks are also better resolved by R-ReALE then by ReALE.

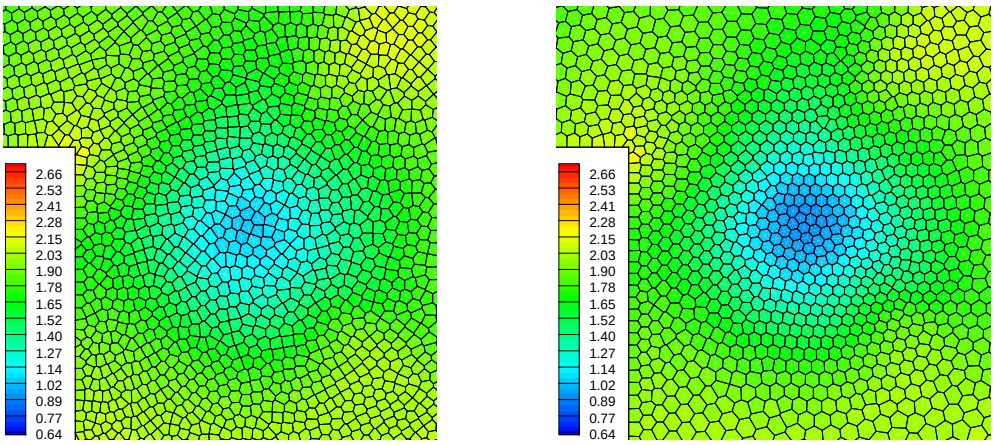


Figure 29: Shock cavity interaction. The mesh and pressure contour plots on a zoomed region on the vortex at $t=420$. Left: ReALE with 24000 generators. Right: R-ReALE with 6000 generators.

The Fig. 27 and Fig. 28 allow to compare the pressure contours at $t=340$ and 420 under mesh refinement. One can observe that the size of the vortex has increased while its center has slowly moved to the right of the cavity. With the same number of generators, R-ReALE gives better results. R-ReALE with 6000 generators resolves the vortex better than ReALE with 24000 generators. The Fig. 29 gives a zoomed view of the mesh on the vortex. One can see that although R-ReALE uses much less generators, it has almost the same cell area as ReALE has on the vortex.

Comparing the numerical results with the shadowgraphs from the experiment, one can see that both ReALE and R-ReALE results are in a good agreement with experimental data. This numerical experiment constitutes a validation of both methods by demonstrating their capabilities of producing physically accurate simulations of complex shock wave structures.

7 Conclusion and Future Work

We have described a new adaptive reconnection-based arbitrary Lagrangian-Eulerian method - R-ReALE. The R-ReALE method uses a monitor function as an indicator of the error. The raw monitor function is scaled to avoid extremely small and large cells and smoothed to guarantee gradual changes in cell size. We use the equidistribution principle for the monitor function and centroidal Voronoi tessellation machinery as a tool to create adaptive meshes.

In the R-ReALE method the number of mesh cells is chosen at the beginning of the calculation and does not change with time, but the mesh is adapted based on the modified monitor function during the rezone stage of each time step.

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

Using a series of test problems we have demonstrated the superiority of our new adaptive R-ReALE method in comparison with the standard non-adaptive ReALE method.

In our opinion, the main strength of the R-ReALE method is that it combines the robustness of the ReALE method (due to reconnection) with an adaptation capability.

The construction of the monitor function is problem and goal dependent and we do not present a new monitor in this paper. However, there are new developments even for Hessian-based monitors [33], and we are planning to explore other monitors in the future.

Work has already begun on creating an A-ReALE method in which we will be able to add and remove generators during the rezone stage; this method has obtained promising results and will be published in in a future paper.

8 Acknowledgments

This work was performed under the auspices of the National Nuclear Security Administration of the US Department of Energy at Los Alamos National Laboratory under Contract No. DE-AC52-06NA25396. The authors gratefully acknowledge the partial support of the US Department of Energy Office of Science Advanced Scientific Computing Research (ASCR) Program in Applied Mathematics Research and the partial support of the US Department of Energy National Nuclear Security Administration Advanced Simulation and Computing (ASC) Program. LA-UR-15-2094.

We would like to thank R. Loubere, P.-H. Maire, J. Breil, S. Galera, and S. Sambasivan who participated in developing the ReALE methodology as well as M. Owen, D. Miller, D. Strains for numerous discussions related to ReALE.

We also thankful to S. Diot and S. Mao, who participated in initial stages of the developing adaptive ReALE ideas.

We thank J. Quirk, S. Li, S. Popinet S. Del Pino and A. Barlow for sharing their thoughts on the efficiency of adaptive methods.

Finally, we would like to thank T. Masser for proofreading the draft version of this paper.

A Appendix: "Smooth-to-Shock" solution

Here, we follow the method in [55] to construct a smooth solution of the 1D gas dynamics equations. We will show that for smooth initial conditions, the gas dynamics equations have smooth solutions before a critical time t_{shock} and develop shock(s) after t_{shock}. We consider gamma-law equation of state and the isentropic fluid with $p=\rho^{\gamma}$. The two Riemann invariants of the gas dynamics equations are

$$\alpha_{\pm}=u\pm\frac{2}{\gamma-1}a, \tag{A.1}$$

where u is the velocity, a is the sound speed and γ is the adiabatic constant. Along each characteristic line, we have

$$\frac{d\alpha_{\pm}}{dt}=0 \quad \text{along} \quad \frac{dx}{dt}=u\pm a. \tag{A.2}$$

Take $\gamma=3$, (A.1) is rewritten as

$$\alpha_{\pm}=u\pm a. \tag{A.3}$$

Substitute the above equations into (A.2), we have

$$\frac{d\alpha_{\pm}}{dt}=0 \quad \text{along} \quad \frac{dx}{dt}=\alpha_{\pm}. \tag{A.4}$$

From (A.4), it can be seen that all characteristic lines are straight lines. Moreover, (A.4) are equivalent to two Burgers equations

$$\frac{\partial\alpha_{\pm}}{\partial t}+\alpha_{\pm}\frac{\partial\alpha_{\pm}}{\partial x}=0. \tag{A.5}$$

The characteristic lines of the two Burgers equations (A.5) are

$$x=\eta\pm\alpha_{\pm}(\eta,0)t, \tag{A.6}$$

where η is the starting point of the characteristic lines at time $t=0$. In order for (A.5) to have a smooth solution, the characteristic lines do not intersect with each other:

$$\frac{\partial x}{\partial\eta}=1\pm\frac{\partial\alpha_{\pm}(\eta,0)}{\partial\eta}t>0. \tag{A.7}$$

From (A.7), we compute the time of shock formation

$$t_{shock}=\min_{\eta}\left|\frac{\partial\alpha_{\pm}(\eta,0)}{\partial\eta}\right|^{-1}. \tag{A.8}$$

The above methods can be used to solve the gas dynamics equations when the initial states are smooth. Assume the initial states are given by $\rho(x,0)$, $u(x,0)$, $a(x,0)$, The initial conditions for (A.5) can be obtained from (A.3):

$$\alpha_{\pm}(x,0)=u(x,0)\pm a(x,0). \tag{A.9}$$

Given a 1D region and reflecting boundary conditions, the exact solution of the system (A.5), (A.9) can be solved using methods of characteristics. The solution of the gas dynamics equations (remember $\gamma=3$) can be obtained using (A.3) and isentropic conditions:

$$u(x,t)=\frac{\alpha_{+}(x,t)+\alpha_{-}(x,t)}{2}, \quad a(x,t)=\frac{\alpha_{+}(x,t)-\alpha_{-}(x,t)}{2}, \quad \rho(x,t)=\frac{1}{\sqrt{\gamma}}a(x,t). \tag{A.10}$$

Then pressure is obtained from isentropic conditions

$$p(x,t)=\rho(x,t)^3, \tag{A.11}$$

and internal energy is obtained from equation of state

$$e(x,t)=p(x,t)/(2\rho(x,t))=\rho(x,t)^2/2. \tag{A.12}$$

DOI https://doi.org/10.4208/jms.v48n2.15.03 | Generated on 2025-04-20 21:22:47

References

[1] F. Aurenhammer. Voronoi diagram-a survey of fundamental geometric data structures. ACM Computing Surveys, 23(3): 345–405, 1991.

[2] A. Belme, A. Dervieux and F. Alauzet. Time accurate anisotropic goal-oriented mesh adaptation for unsteady flows. Journal of Computational Physics, 231(19): 6323–6348, 2012.

[3] M. J. Berger and J. Oliger. Adaptive mesh refinement for shock hydrodynamics. J. Comput. Phys., 53: 484–512, 1984.

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

[5] J. Breil, T. Harribey, P.-H. Maire and M. Shashkov. A multi-material ReALE method with MOF interface reconstruction. Computers & Fluids, 83: 115–125, 2013.

[6] C. J. Budd, W. Huang and R. D. Russell. Adaptivity with moving grids. Acta Numerica, 18: 111–241, 2009.

[7] E. J. Caramana, D. E. Burton, M. J. Shashkov and P. P. Whalen. The construction of compatible hydrodynamics algorithms utilizing conservation of total energy. Journal of Computational Physics, 146(1): 227–262, 1998.

[8] G. Carré, S. Del Pino, B. Després and E. Labourasse. A cell-centered Lagrangian hydrodynamics scheme on general unstructured meshes in arbitrary dimension. Journal of Computational Physics, 228(14): 5160–5183, 2009.

[9] S-.W. Cheng, T. K. Dey and J. R. Shewchuk. Delaunay mesh generation. CRC Press, 2012.

[10] W. J. Coirier and K. G. Powell. Solution-adaptive Cartesian cell approach for viscous and inviscid flows. AIAA Journal, 24(5): 938–945, 1996.

[11] S. Del Pino. Metric-based mesh adaptation for 2D Lagrangian compressible flows. Journal of Computational Physics, 230(5): 1793–1821, 2011.

[12] O. Devillers, S. Pion, and M. Teillaud. Walking in a triangulation. International Journal of Foundations of Computer Science, 13(02): 181–199, 2002.

[13] Z. Di, M. Emelianenko and S. Nash. Truncated newton-based multigrid algorithm for centroidal Voronoi diagram calculation. Numerical Mathematics: Theory, Methods & Applications, 5(2), 2012.

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

[15] Q. Du and M. Emelianenko. Acceleration schemes for computing centroidal Voronoi tessellations. Numerical Linear Algebra with Applications, 13(2-3): 173–192, 2006.

[16] Q. Du, V. Faber and M. Gunzburger. Centroidal Voronoi tessellations: Applications and algorithms. SIAM Review, 41(4): 637–676, 1999.

[17] Q. Du, M. Gunzburger and L. Ju. Advances in studies and applications of centroidal Voronoi tessellations. Numerical Mathematics: Theory, Methods and Applications, 3(2): 119–142, 2010.

[18] Q. Du and D. Wang. Tetrahedral mesh generation and optimization based on centroidal Voronoi tessellations. International Journal for Numerical Methods in Engineering, 56(9): 1355–1373, 2003.

[19] J. K. Dukowicz and J. W. Kodis. Accurate conservative remapping (rezoning) for arbitrary Lagrangian-Eulerian computations. SIAM Journal on Scientific and Statistical Computing, 8(3): 305–321, 1987.

[20] P.-J. Frey and F. Alauzet. Anisotropic mesh adaptation for CFD computations. Computer Methods in Applied Mechanics and Engineering, 194(48): 5068–5082, 2005.

[21] A. Gersho. Asymptotically optimal block quantization. Information Theory, IEEE Transactions on, 25(4): 373–380, 1979.
[22] M. Gittings, R. Weaver, M. Clover, T. Betlach, N. Byrne, R. Coker, E. Dendy, R. Hueckstaedt, K. New, W. R. Oakes, D. Ranta and R. Stefan. The RAGE radiation-hydrodynamic code. Computational Science & Discovery, 1: 015005, 2008.
[23] J. Grandy. Conservative remapping and region overlays by intersecting arbitrary polyhedra. J. Comput. Phys., 148(2): 133–166, 1999.
[24] P. M. Gruber. Optimum quantization and its applications. Advances in Mathematics, 186(2): 456–497, 2004.
[25] M. D. Gunzburger. Finite Element Methods for Viscous Incompressible Flows: A Guide to Theory, Practice, and Algorithms. Elsevier, 2012.
[26] T. Harribey, J. Breil, P.-H. Maire and M. Shashkov. A swept-intersection-based remapping method in a reale framework. International Journal for Numerical Methods in Fluids, 72(6): 697–708, 2013.
[27] T. Harribey, J. Breil, P-H. Maire M. Shashkov. A swept intersection based remapping method in a ReALE framework. International Journal for Numerical Methods in Fluids, 72: 697–708, 2013.
[28] C. Hirt, A. Amsden and J. J. L. Cook. An arbitrary Lgrangian-Eulerian computing method for all flow speeds. Journal of Computational Physics, 14(3): 227–253, 1974.
[29] L. H. Howell and J. B. Bell. An adaptive mesh projection method for viscous incompressible flow. SIAM, J. Sci. Comput., 18(4): 996–1013, 1997.
[30] W. Huang and R. D. Russell. Adaptive moving mesh methods. Springer, 2010.
[31] O. Igra, J. Falcovitz, H. Reichenbach and W. Heilig. Experimental and numerical study of the interaction between a planar shock wave and a square cavity. Journal of Fluid Mechanics, 313(1): 105–130, 1996.
[32] L. Ju, M. Gunzburger and W. Zhao. Adaptive finite element methods for elliptic PDEs based on conforming centroidal Voronoi-Delaunay triangulations. SIAM Journal on Scientific Computing, 28(6): 2023–2053, 2006.
[33] L. Kamenski and W. Huang. How a nonconvergent recovered Hessian works in mesh adaptation. SIAM J. Numer. Anal., 52: 1692–1708, 2014.
[34] J. R. Kamm and F. X. Timmes. On efficient generation of numerically robust Sedov solutions. submitted to Astrophys. J. Suppl. Ser., in review, 2007.
[35] R. Keppens, M. Noll, G. Toth and J. P. Goedbloed. Adaptive mesh refinement for conservative systems: multi-dimensional efficiency evaluations. Computer Physics Communications, 153: 317–339, 2003.
[36] M. Kucharik, M. Shashkov and B. Wendroff. An efficient linearity-and-bound-preserving remapping methods. J. Comput. Phys., 188: 462–471, 2003.
[37] K. Lipnikov and M. Shashkov. The error-minimization-based strategy for moving mesh methods. Commun. Comput. Phys., 1(1): 53–80, 2006.
[38] Y. Liu, W. Wang, B. Lévy, F. Sun, D.-M. Yan, L. Lu and C. Yang. On centroidal Voronoi tessellation-energy smoothness and fast computation. ACM Transactions on Graphics (ToG), 28(4): 101, 2009.
[39] S. Lloyd. Least squares quantization in PCM. Information Theory, IEEE Transactions on, 28(2): 129–137, 1982.
[40] R. Löhner. Applied CFD Techniques. An Intoroduction based on Finite Element Methods. Wiley, 2001.
[41] R. Loubère, P.-H. Maire, and M. Shashkov. ReALE: A Reconnection Arbitrary-Lagrangian-

Eulerian method in cylindrical geometry. Computers & Fluids, 46(1): 59–69, 2011.
[42] R. Loubère, P.-H. Maire, M. Shashkov, J. Breil, and S. Galera. ReALE: A Reconnection-based Arbitrary-Lagrangian-Eulerian method. Journal of Computational Physics, 229(12): 4724–4761, 2010.
[43] P.-H. Maire. A high-order cell-centered Lagrangian scheme for two-dimensional compressible fluid flows on unstructured meshes. Journal of Computational Physics, 228(7): 2391–2425, 2009.
[44] P.-H. Maire, R. Abgrall, J. Breil, and J. Ovadia. A cell-centered Lagrangian scheme for two-dimensional compressible flow problems. SIAM Journal on Scientific Computing, 29(4): 1781–1824, 2007.
[45] J. M. Morrell, P. K. Sweby and A. Barlow. A cell by cell anisotropic adaptive mesh ALE method. Int. J. Numer. Meth. Fluids, 56: 1441–1447, 2008.
[46] H. Nguyen, J. Burkardt, M. Gunzburger, L. Ju and Y. Saka. Constrained CVT meshes and a comparison of triangular mesh generators. Computational Geometry, 42(1): 1–19, 2009.
[47] T. Plewa, T. Linde and V. G. Weirs, editors. Adaptive mesh refinement-Theory and applications. Springer, 2005.
[48] S. Popinet. Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries. J. Comput. Phys., 190:5 72–600, 2003.
[49] L. I. Sedov. Propagation of strong shock waves. Journal of Applied Mathematics and Mechanics, 10: 241–250, 1946.
[50] G. A. Sod. Numerical Methods in Fluid Dynamics. Initial and Initial Boundary-Value Problems. Cambridge University Press, 1985.
[51] V. Springel. E pur si muove: Galiliean-invariant cosmological hydrodynamical simulations on a moving mesh. Mon. Not. R. Astron. Soc., 401: 791–851, 2010.
[52] T. Tang. Moving mesh methods for computational fluid dynamics. Contemporary Mathematics, 383: 141–174, 2005.
[53] T. J. Barth. Numerical methods for gasdynamic systems on unstructured meshes. In D. Kroner, M. Ohlberger and Rohde C., editors, An introduction to Recent Developments in Theory and Numerics for Conservation Laws, Proceedings of the International School on Theory and Numerics for Conservation Laws, pages 195–284, Berlin, 1997. Lecture Notes in Computational Science and Engineering, Springer.
[54] J. Tournois, P. Alliez and O. Devillers. 2D centroidal Voronoi tessellations with constraints. Numerical Mathematics: Theory, Methods & Applications, 3(2), 2010.
[55] François Vilar. A high-order Discontinuous Galerkin discretization for solving two-dimensional Lagrangian hydrodynamics. Theses, Université Sciences et Technologies - Bordeaux I, November 2012. https://tel.archives-ouvertes.fr/tel-00765575.
[56] A. M. Winslow. Numerical solution of the quasilinear poisson equation in a nonuniform triangle mesh. J. Comput. Phys., 1(2): 149–172, 1966.