Towards Simulation-Driven Optimization of High-Order Meshes by the Target-Matrix Optimization Paradigm

Veselin Dobrev, Patrick Knupp, Tzanio Kolev, Vladimir Tomov

Abstract We present a method for simulation-driven optimization of high-order curved meshes. This work builds on the results of [9], where we described a framework for controlling and improving the quality of high-order finite element meshes based on extensions of the Target-Matrix Optimization Paradigm (TMOP) of [19]. In contrast to [9], where all targets were based strictly on geometric information, in this work we blend physical information into the high-order mesh optimization process. The construction of target-matrices is enhanced by using discrete fields of interest, e.g., proximity to a particular region. As these discrete fields are defined only with respect to the initial mesh, their values on the intermediate meshes (produced during the optimization process) must be computed. We present two approaches for obtaining values on the intermediate meshes, namely, interpolation in physical space, and advection remap on the intermediate meshes. Our algorithm allows high-order applications to have precise control over local mesh quality, while still improving the mesh globally. The benefits of the new high-order TMOP meth-

Veselin Dobrev, Tzanio Kolev, Vladimir Tomov
Lawrence Livermore National Laboratory, 7000 East Avenue, Livermore, CA 94550
e-mail: dobrev1@llnl.gov,kolev1@llnl.gov,tomov2@llnl.gov

Patrick Knupp
Dihedral LLC, Bozeman, Montana, e-mail: zg37rd@gmail.com

Performed under the auspices of the U.S. Department of Energy under Contract DE-AC52-07NA27344 (LLNL-CONF-752038)

ods are illustrated on examples from a high-order arbitrary Lagrangian-Eulerian application [5].

1 Introduction

Discretizations that leverage high-order curved meshes have become increasingly popular in recent years, due to several mathematical and computational advantages. For example, high-order meshes are essential for achieving optimal convergence rates on domains with curved boundaries/interfaces and for symmetry preservation in radial flow [10, 8]. They develop naturally in high-order simulations with moving meshes, including Lagrangian formulations [10, 8, 17]. They play a pivotal role in high-order arbitrary Lagrangian-Eulerian (ALE) methods [7, 3], where maintaining high-order convergence (for smooth flows) and 3D symmetry is generally impossible with linear meshes. High-order meshes can be relatively coarse and still capture curved geometries adequately, leading to equivalent simulation quality for a smaller number of elements. Furthermore, such meshes have increased flexibility and great potential in the context of adaptivity to simulation features.

Our general framework for controlling and improving the quality of *high-order* finite element meshes was presented in [9]. The proposed approach is based on TMOP [19], which is distinguished from similar methods by its emphasis on target-matrix construction methods that permit a greater degree of control over the optimized mesh. Mesh positions are optimized via node movement. Pointwise mesh quality metrics are defined by utilizing sub-zonal information. These metrics are capable of measuring shape, size or alignment of the region around the point of interest. TMOP requires predefined target-matrices as a way for the users to incorporate application-specific physical information into the metric that is being optimized. The combination of targets and quality metrics is used to optimize the node positions, so that they are as close as possible to the shape/size/alignment of their targets. The resulting mathematical problem is posed as global or local optimization of the chosen metric over the mesh.

All optimization methods in [9] are driven explicitly by geometric information. Many high-order applications require an additional capability, namely, adaptivity to certain physical features. The emphasis in this paper is the study of simulation-driven optimization within the context of finite element r-adaptivity on high-order meshes. Copious literature exists for r-adaptivity on first-order finite element meshes (e.g., [18]), but very little for high-order meshes. We extend the methods from [9] by including information about discrete fields in the optimization process. An example of such discrete field is some material's location, which is represented as a finite element function on the starting mesh. Recent advances on this subject were presented in [22, 16]. In contrast, this study addresses the case of general high-order curved triangular/quadrilateral and tetrahedral/hexahedral grids. Furthermore, the concept of adaptivity can be applied towards a general target, e.g., we can adapt not only the size, but also the local shape of the elements.

We enable the use of simulation-specific information by developing target-matrix construction algorithms that include information about discrete solution fields and data of interest. The use of discrete field data is a major challenge, because it is defined only with respect to the initial mesh. The values of the discrete data on the intermediate meshes (produced during the optimization) are obtained either by interpolation in physical space, or by advection / reconstruction on the intermediate meshes [2, 16, 22]. After the discrete values on the current mesh are obtained and the final target-matrices are calculated, we proceed by optimizing a global nonlinear functional via moving the mesh node positions. This nonlinear functional is defined in terms of local quality metrics, which can measure shape / size or alignment in the neighborhood of a given sample point.

Section 2 reviews the representation of the high-order mesh, basic TMOP components, and the objective function. Section 3 describes the construction of adaptivity-based target-matrices. Section 4 considers ways to transition the discrete values of interest from the initial to the current mesh. In Section 5 we discuss approaches to control the frequency of mesh optimization steps in time-dependent simulations through leveraging local quality metrics. Section 6 presents numerical results which demonstrate the ability to adapt high-order meshes using basic TMOP target-matrices and quality metrics. Conclusions are presented in Section 7.

2 Preliminaries

The goal of this section is to introduce the discrete mesh representation that is used throughout the paper, and briefly recall the purely geometric-based approach for optimization of high-order meshes through TMOP. The complete details can be found in [9]. The changes in the TMOP components related to adaptivity to simulation features are discussed in the following sections.

2.1 Discrete Representation of the High-Order Mesh

Let $d \in \{1,2,3\}$ be the space dimension and consider a set of scalar basis functions $\{\bar{w}_i\}_{i=1}^{N}$ on a reference element $\bar{E}$ of dimension d. Typically, for simplicial elements (triangles, tets), the basis $\{\bar{w}_i\}$ spans P_k, the space of all polynomials of total degree at most k; for tensor product elements (quads, hexes) $\{\bar{w}_i\}$ is chosen to be a basis for Q_k, the space of all polynomials of degree at most k in each variable. The degree k is the mesh order, e.g. $k = 1$ corresponds to linear meshes, $k = 2$ to quadratic meshes, etc. The shape of any element E in the mesh is then fully described by a matrix $\mathbf{x}_E$ of size $d \times N$ whose columns represent the coordinates of the element control points (aka element degrees of freedom). Given $\mathbf{x}_E$, we introduce the map $\boldsymbol{\Phi}_E : \bar{E} \to \mathbb{R}^d$ whose image is the high-order element E:

$$x(\bar{x}) = \Phi_E(\bar{x}) \equiv \sum_{i=1}^{N} \mathbf{x}_{E,i} \bar{w}_i(\bar{x}), \qquad \bar{x} \in \bar{E}, \tag{1}$$

where we used $\mathbf{x}_{E,i}$ to denote the i-th column of $\mathbf{x}_E$. When two elements E and E' share a common mesh entity (vertex, edge, or face) then their control-point matrices $\mathbf{x}_E$ and $\mathbf{x}_{E'}$ are not independent because, to ensure mesh continuity, the descriptions of the common entity (through Φ_E on one hand and $\Phi_{E'}$ on the other) must coincide. This type of interdependence among mesh elements is typically expressed by defining a *global* vector $\mathbf{x}$ of control points (degrees of freedom) and a set of linear operators $\mathbf{x} \to \mathbf{x}_E$, one per mesh element, that define/extract the local coordinates $\mathbf{x}_E$ from the global vector $\mathbf{x}$. Thus, the global continuity of the mesh is ensured for any value of $\mathbf{x}$.

For any element E in the mesh, we can compute the Jacobian of the mapping Φ_E at any reference point $\bar{x} \in \bar{E}$ as

$$A_E(\bar{x}) = \frac{\partial \Phi_E}{\partial \bar{x}} = \sum_{i=1}^{N} \mathbf{x}_{E,i} [\nabla \bar{w}_i(\bar{x})]^T. \tag{2}$$

2.2 TMOP and Optimization of High-Order Meshes

At each quadrature point (inside each mesh element), TMOP uses three Jacobian matrices:

- The Jacobian matrix $A_{d\times d}$ of the transformation from reference to physical coordinates, given by (2).
- The *target-matrix*, $W_{d\times d}$, which is the Jacobian of the transformation from the reference to the *target* coordinates. The target-matrices are defined according to a user-specified method prior to the optimization; they define the desired properties in the optimal mesh.
- The *weighted Jacobian* matrix, $T_{d\times d}$, defined by $T = AW^{-1}$, represents the Jacobian of the transformation from the target to the physical coordinates.

The T matrix is used to define the *local quality measure*, $\mu(T)$. The quality measure can evaluate shape, size, or alignment of the quadrature point's neighborhood.

The goal of TMOP is to minimize a global *objective function* that depends on the local quality measure throughout the mesh. In this paper we focus on the following objective function:

$$F(x) := \sum_{E \in \mathscr{M}} \int_{E_t} \mu(T(x_t)) dx_t = \sum_{E \in \mathscr{M}} \sum_{x_q \in E_t} w_q \det(W(\bar{x}_q)) \mu(T(x_q)), \tag{3}$$

where $\mathscr{M}$ is the current mesh, E_t is the target element corresponding to the physical element E, w_q are the quadrature weights, and the point x_q is the image of the reference quadrature point $\bar{x}_q$ in the target element E_t. Note that the right-hand side in (3)

depends on the mesh positions x through the Jacobian matrices A used in the definition of T; in addition, we will also allow the target matrices $W(\bar{x}_q)$ to depend on the mesh positions x, see Section 3. The integration in (3) is performed over the target elements, enforcing that the integral contribution from a given element E, relative to the contributions from other elements, is only based on the difference with its target E_t (which is measured by $\mu(T)$), and not on its relative size compared to the other elements. In particular, very small elements will not be neglected by the optimization process due to their size. The existence of a minimum for the variational optimization problem (3) has been explored theoretically in [13, 14]. The objective function can be extended by using combinations of quality metrics, space-dependent weights for each metric, and limiting the amount of allowed mesh displacements, see details in [9].

As $F(x)$ is nonlinear, in this paper we minimize it by utilizing Newton's method to solve the critical point equations $\partial F(x)/\partial \mathbf{x} = 0$. This approach involves the computation of the first and second derivatives of $\mu(T)$ with respect to T. Furthermore, boundary nodes are enforced to stay fixed or move only in the boundary's tangential direction. Line search procedure is utilized to guarantee that the Newton updates do not lead to inverted meshes, see details in [9]. Alternative approach for minimizing $F(x)$ is presented in [21], where local optimization problems are solved for each node of the mesh.

In the context of ALE, we will also refer to the initial mesh, which is to be optimized, as the "Lagrangian" mesh. We denote the Lagrangian mesh by $\mathscr{M}_0$, and $\mathscr{M}_n$ is the mesh obtained after n Newton iterations.

2.3 Adaptivity Function

To represent the desired mesh features and to control the adaptivity process, we utilize an adaptivity field $\eta(x)$ (the terms monitor function and metric tensor are also used [18, 6]). In general, the definition of this field depends on some simulation-specific goal of the adaptivity process. Adaptivity fields are constructed by user-defined procedures that combine the simulation's available input. For example, $\eta(x)$ can be a tensor representing multiple (up to 4 in 2D and up to 9 in 3D) point-wise characteristics of the desired mesh element located at the point x.

In this paper, we assume that $\eta(x)$ is derived from the simulation's discrete field data (thus, η is also a discrete field). To illustrate the basic properties of the method in a simplified framework, we define $\eta(x) \in \mathscr{V}$ as a scalar finite element function with values in $[0,1]$. We choose the discrete space $\mathscr{V} \subset H^1(\mathscr{M})$ to be the space of continuous finite element functions of degree k defined on the domain mesh $\mathscr{M}$:

$$\mathscr{V} = \begin{cases} \{v \in C^0(\mathscr{M}) \mid v_{|E} \in Q_k(E)\} & \text{for quadrilateral/hexahedral meshes}, \\ \{v \in C^0(\mathscr{M}) \mid v_{|E} \in P_k(E)\} & \text{for triangular/tetrahedral meshes}, \end{cases}$$

where E is an arbitrary mesh element, $Q_k(E)$ is the mapped space of polynomials of degree at most k in each variable, and $P_k(E)$ is the mapped space of polynomials of total degree at most k, i.e.

$$Q_k(E) = \{p \circ \Phi_E^{-1} \mid p \in Q_k\} \qquad P_k(E) = \{p \circ \Phi_E^{-1} \mid p \in P_k\}.$$

Unit values of η represent regions of most interest, e.g., smallest local mesh size in the case of size adaptivity, and zero values represent regions of least interest. The choice of the continuous finite element space $\mathscr{V}$ is made to avoid sharp transitions. Furthermore, $\eta \in \mathscr{V}$ uses the same degree k as the mesh, in order to represent adequately the sub-element resolution. One can also choose to discretize η with lower degree polynomials, depending on the level of detail that is expected on sub-element level. Specific formulations and usages of $\eta(x)$ in the context of shape and size adaptivity are presented in Section 6.

3 Adaptive Target-Matrices

Once the adaptivity function $\eta(x)$ is known on a given mesh, its information needs to be converted into a set of spatially-varying target-matrices $\{W(q_\ell)\}$ through a Target-Matrix construction algorithm. The set $\{q_\ell\}$ consists of all quadrature points (which depend on the mesh-node locations) in physical space. The specific construction algorithm depends on the type of data contained in the adaptivity field, the corresponding features desired in the adapted mesh, and on properties of the quality metric. In this work we illustrate this idea by utilizing targets of the form

$$W = \begin{pmatrix} W_{11} & 0 \\ 0 & W_{22} \end{pmatrix}, \text{where } W_{11} > 0 \text{ and } W_{22} > 0 \text{ both depend on } \eta(x).$$

Each of these target elements is orthogonal, has aspect ratio W_{22}/W_{11}, and area $W_{11}W_{22}$. This setup controls shape, if a Shape metric is used, and both shape and size, if a Shape+Size metric is used. Specific values of W_{11} and W_{22} for the above adaptation cases are presented in Section 6. Choosing W to be diagonal is not overly restrictive because this form of the target-matrix will be used in conjunction with rotation-invariant shape or shape+size quality metrics.

As W depends on the discrete function $\eta(x)$, a major difference in the resulting optimization problem (3), compared to our previous work on purely geometric mesh quality optimization, is that the gradient of the mesh quality functional, $\partial\mu(T)/\partial\mathbf{x}$, requires the computation of the derivatives of $\{W(q_\ell)\}$ with respect to the mesh-node locations. Namely, its first derivative, at a fixed quadrature point $\bar{x}_q \in \bar{E}$ and a fixed mesh element E such that $q_\ell = \Phi_E(\bar{x}_q)$, has the form (denoting $[\mathbf{x}_E]_{ji} = x_{ij}$, $i = 1,\dots,N$, $j = 1,\dots,d$)

$$\begin{aligned}\frac{\partial \mu(T)}{\partial x_{ij}} &= \sum_{k,l=1}^{d} \frac{\partial \mu(T)}{\partial T_{kl}} \frac{\partial T_{kl}}{\partial x_{ij}} = \frac{\partial T}{\partial x_{ij}} : \frac{\partial \mu}{\partial T} = \frac{\partial \left(AW^{-1}\right)}{\partial x_{ij}} : \frac{\partial \mu}{\partial T} \\ &= \left(\frac{\partial A(\mathbf{x})}{\partial x_{ij}} W^{-1}(\eta(q_\ell)) + A(\mathbf{x}) \frac{\partial W^{-1}(\eta(q_\ell))}{\partial x_{ij}} \right) : \frac{\partial \mu}{\partial T}\,,\end{aligned} \tag{4}$$

where $\frac{\partial \mu}{\partial T}$ is a matrix with components $\left[\frac{\partial \mu}{\partial T}\right]_{kl} = \frac{\partial \mu}{\partial T_{kl}}$ and we used the notation $X : Y$ to denote the inner product of two matrices X and Y of the same size, $m \times n$, defined as $X : Y = \sum_{s=1}^{m} \sum_{t=1}^{n} X_{st} Y_{st}$. In the above formula, we can further express

$$\frac{\partial W^{-1}(\eta(q_\ell))}{\partial x_{ij}} = \frac{\partial W^{-1}}{\partial \eta} \frac{\partial \eta}{\partial x_{ij}} = -W^{-1} \frac{\partial W}{\partial \eta} W^{-1} \frac{\partial \eta}{\partial x_{ij}}\,.$$

The Jacobian matrix $A(\mathbf{x})$ and its derivative with respect to the variable x_{ij} can be expressed as

$$[A(\mathbf{x})]_{kl} = \sum_{s=1}^{N} x_{sk} [\nabla \bar{w}_s(\bar{x}_q)]_l\,, \quad \frac{\partial A_{kl}}{\partial x_{ij}} = \delta_{kj} [\nabla \bar{w}_i(\bar{x}_q)]_l\,.$$

The minimization functional $F(x)$ also depends on $\det(W)$, so we need its derivative as well:

$$\frac{\partial [\det(W)]}{\partial x_{ij}} = \frac{\partial [\det(W)]}{\partial \eta} \frac{\partial \eta}{\partial x_{ij}} = \left(\mathrm{adj}(W)^T : \frac{\partial W}{\partial \eta} \right) \frac{\partial \eta}{\partial x_{ij}}\,.$$

In the above formulas, the term $\frac{\partial \eta}{\partial x_{ij}}$ can be computed as

$$\frac{\partial \eta}{\partial x_{ij}} \approx \nabla \eta(q_\ell) \cdot \frac{\partial q_\ell}{\partial x_{ij}}\,, \qquad \text{where} \quad [q_\ell]_k = \sum_{s=1}^{N} x_{sk} \bar{w}_s(\bar{x}_q)\,.$$

Note that the latter formula is an approximation of the derivative $\frac{\partial \eta}{\partial x_{ij}}$ because it does not take into account the fact that when the mesh nodes move, the discrete values of η on the new mesh will change, as discussed in Section 4. Computing the exact numerical derivatives is non-trivial, as the calculation must take into account the transition procedure of the discrete η values. We are planning to address the computations of exact derivatives in our future work. More complicated expressions for η, e.g., tensors, would lead to additional challenges in the computation of the above derivatives. Furthermore, when the Newton's method is used, one also needs to compute second derivatives of the quantities discussed above.

4 Obtaining Discrete Field Values

Adaptive target-matrices must be evaluated on the current mesh configuration $\mathscr{M}_n$. Computing these matrices is not trivial, because the discrete finite element func-

tion $\eta(x)$ is typically given on the initial Lagrangian mesh $\mathscr{M}_0$. In this section we propose two alternative methods for evaluating $\eta(x)$ on the mesh $\mathscr{M}_n$.

4.1 Interpolation

One way to evaluate $\eta(x)$ is to find the logical image of x on the Lagrangian mesh $\mathscr{M}_0$. More specifically, for a given physical point x, we must find the corresponding element $E_0 \in \mathscr{M}_0$ and reference point $\bar{x}_0 \in \bar{E}_0$, so that

$$\Phi_{E_0}(\bar{x}_0) = x\,.$$

Once the image $\bar{x}_0$ of x is found in some element $E_0 \in \mathscr{M}_0$, we simply perform a standard finite element evaluation of η on the Lagrangian mesh $\mathscr{M}_0$.

The first step in this procedure is to determine a set $\mathscr{S}$ of candidate elements in $\mathscr{M}_0$. Each of these elements would be tested for containing the physical point x. We form this set by including: (i) the element E_0^* whose center point is closest to x, and (ii) all vertex-neighbor elements of E_0^*. The above set can also be formed by more advanced algorithms, e.g., Section 7.1.5 in [12].

Testing whether a given element $E_0 \in \mathscr{S}$ contains x, consists of inverting the nonlinear transformation Φ, given by (1). Once solved, if $\bar{x}_0 \in \bar{E}_0$, then the point is found in E_0. Otherwise, the point is outside of E_0 and we proceed with the remaining elements in the candidate set. The inversion problem is solved by a Newton method:

$$\bar{x}_0^{n+1} = \bar{x}_0^n + A^{-1}(\bar{x}_0^n)\left[x - \Phi_{E_0}(\bar{x}_0^n)\right]\,.$$

In general there is no guarantee that the above nonlinear iteration will find an element that contains x. Specific implementations of the above interpolation procedure can trade performance for accuracy and robustness, e.g., adjusting convergence tolerances, choosing initial guess within the reference cell, correction strategy when the Newton update overshoots the element boundaries. The specifics of our implementation of the above inversion method can be found in the MFEM finite element library [20], in particular the function *FindPoints* and the class *InverseElementTransformation*.

This interpolation procedure is used to obtain the value of η at a single point. To reconstruct the whole discrete finite element function $\eta(x)$ on $\mathscr{M}_n$, we interpolate the position (with respect to $\mathscr{M}_n$) of every node of the space $\mathscr{V}$.

4.2 Advection Remap

In this section we utilize ideas from [2] to transfer the discrete adaptivity function $\eta_0(x_0)$, defined on the Lagrangian mesh $\mathscr{M}_0$, to the function $\eta(x)$, defined on the current mesh $\mathscr{M}$. The problem is formulated in terms of transporting (or advecting)

the adaptivity function between the two meshes over a fictitious (or pseudo) time interval determined by the motion of the mesh between the two configurations.

Assuming that every point $x_0 \in \mathscr{M}_0$ is associated with a unique point $x \in \mathscr{M}$, we can define a continuous transition function $P(x_0,\tau) : \mathscr{M}_0 \times [0,1] \to \mathbb{R}^d$, such that

$$P(x_0,0) = x_0 \quad \text{and} \quad P(x_0,1) = x\,, \quad \forall x_0 \in \mathscr{M}_0\,.$$

We refer to the parameter $\tau \in [0,1]$ as "pseudo-time". The intermediate meshes, $\mathscr{M}_\tau = \{P(x_0,\tau) : x_0 \in \mathscr{M}_0\}$, are obtained by linear interpolation:

$$P(x_0,\tau) = x_0 + \tau(x - x_0)\,.$$

Next we define the "pseudo-time velocity", u, as

$$u(x_0,\tau) := \frac{\partial P}{\partial \tau} = x - x_0\,.$$

Note that $u(x_0)$ is constant in pseudo-time along each transition trajectory $x_0 \to x$. Having defined such velocity, we can track the values of η in pseudo-time along the trajectories $x_\tau = P(x_0,\tau)$, by utilizing the concept of the advective derivatives (also known as "material" or "Lagrangian" derivatives):

$$\frac{d\eta(x_\tau,\tau)}{d\tau} = \frac{\partial \eta}{\partial \tau} + u \cdot \nabla \eta\,. \tag{5}$$

As the goal of this transfer procedure is to extend η_0 to η without changing its values with respect to pseudo-time, i.e., $\partial\eta/\partial\tau = 0$, we enforce this requirement in (5). That is, we transfer η by solving in pseudo-time the following equation:

$$\frac{d\eta}{d\tau} = u \cdot \nabla \eta\,, \quad \eta(x_0,0) = \eta_0(x_0)\,. \tag{6}$$

On semi-discrete level, we discretize equation (6) by a standard continuous Galerkin formulation (recall that η is considered a continuous finite element function). Our discretization utilizes basis functions that follow the mesh motion, i.e., $dw/dt = 0$. The semi-discrete form of (6) is

$$M\frac{d\eta}{d\tau} = K\eta\,, \quad \text{where} \quad M_{ij} = \int_{\mathscr{M}_\tau} w_i w_j\,, \quad K_{ij} = \int_{\mathscr{M}_\tau} w_i (u \cdot \nabla w_j)\,. \tag{7}$$

This formulation does not need to include any boundary integrals, because every Newton step of our mesh optimization preserves the boundaries of the Lagrangian mesh or only moves nodes in the tangential direction of the boundary, i.e.,

$$(x - x_0) \cdot n = u \cdot n = 0\,.$$

Note that both the mass matrix M and the advection matrix K depend on τ, as the Jacobians of the reference $\rightarrow$ physical transformations depend on τ. Thus, these matrices must be reassembled at every pseudo-time step.

The discretization in time of (7) is performed by a standard explicit Runge-Kutta method. Our default choice for all numerical tests in this paper is RK4. The advection time step $\Delta\tau$ is computed by considering the mesh size of the original elements and the magnitude of the mesh advection velocity u as

$$\Delta\tau = 0.5 \min_{\mathscr{M}_0} \frac{\Delta x}{|u|}.$$

The above ratio is taken at various quadrature points within each element, and Δx is computed as the determinant (to power $1/d$) of the reference $\rightarrow$ physical Jacobian at a given quadrature point.

An important choice that must be considered in this advection approach is the starting point of the advection. When the Newton iteration computes a new mesh $\mathscr{M}_{n+1}$, there are two major options for the calculation of η_{n+1}:

1. Advect from the Lagrangian mesh, i.e., η_0 to η_{n+1}, with $u = x_{n+1} - x_0$.
2. Advect from the latest iteration, i.e., η_n to η_{n+1}, with $u = x_{n+1} - x_n$.

The main advantage of the first approach is that it does not accumulate numerical "noise". Without any nonlinear enhancements, linear transport methods usually produce oscillations [15], especially in the case of high-order finite elements [1]. Our goal in this paper, however, is to derive an advection method that is simple and gives a reasonable representation of η on the new mesh. The first approach will always advect η using the shortest path between x_0 and x_{n+1}, thus avoiding any intermediate stages. Such stages appear during the second approach, when numerous Newton iterations are taken. Each of these intermediate stages would introduce and accumulate oscillations in η, when the second approach is used. For thorough discussion and methods for treating oscillations resulting during transport of high-order finite element fields, see [1]. The advantage of the second approach is that every advection would usually perform less time steps, as x_{n+1} is expected to be closer to x_n than to x_0. In our numerical tests we always utilize the first approach.

After the time evolution of (6) is completed and $\eta(x)$ is computed on $\mathscr{M}_n$, we post-process the solution by adjusting $\eta(x) = \max(1, \eta(x))$ and $\eta(x) = \min(0, \eta(x))$. This is required, because our methods assume the range of η is always $[0,1]$, and the remap procedure might violate this range, as explained in the previous paragraph.

5 Mesh Optimization Triggers

Moving mesh simulations, such as ALE, typically need to periodically perform mesh optimization steps (referred to as remeshing in ALE), because deterioration of the mesh quality can cause simulation failure. Performing such steps too often, however, usually results in (i) slower computations, and (ii) additional numerical

errors, because ALE steps are purely numerical and not based on the physics of the problem. In order to trigger remesh steps, applications need procedures to track accurately the mesh quality during the moving mesh phase, in order to produce robust, efficient, and accurate calculations.

The TMOP concepts can be utilized to define such mesh optimization triggers, both in the purely geometric and the simulation-driven case. Similar to specifying target-matrices, the user can specify *admissible* Jacobian matrix, S, which defines the transformation from the reference element to the worst element that can be used during the moving mesh phase. The Jacobian of the transformation between target and admissible coordinates becomes $U = SW^{-1}$. This matrix is then used to calculate $\mu(U)$, the highest admissible mesh quality metric value for the metric of interest μ. Then remesh is triggered whenever $\mu(T) > \mu(U)$ at any mesh quadrature point.

The specific formulation of S is, of course, problem dependent, e.g., it can be used to trigger remesh steps whenever particular mesh configurations occur. Just as the target-matrices W, the admissible matrices can contain information about the local shape, size or alignment, at any quadrature point. They can vary in space/time by depending on a known analytic function or the simulation's discrete fields. As an example, we can define S as

$$S = \begin{pmatrix} 1 & 0 \\ 0 & S_{22} \end{pmatrix}, \text{where } S_{22} > 0 \text{ is a user-specified parameter}. \tag{8}$$

When μ is a Shape metric, remesh is triggered based on local aspect ratio. When μ is a Shape+Size metric, remesh is triggered based on local size and aspect ratio. Furthermore, the mesh optimization trigger can be made adaptive by making S_{22} space/time dependent.

6 Numerical Examples

In this section we report numerical results from the algorithms in the previous sections as implemented in the MFEM finite element library [20]. This implementation is freely available at http://mfem.org.

All results below are calculated by utilizing the objective function (3) and Gauss-Legendre quadrature for the resulting integrals. Newton's method is used to solve the nonlinear optimization problems with the modification that the derivatives of the target matrices W are assumed to be zero. The Newton relative tolerance is set to 10^{-12}, and all of the presented results are fully converged to this tolerance. The linear solve inside each Newton step is performed by the standard minimum residual (MINRES) algorithm. Boundary nodes are allowed to move as long as the motion does not perturb the initial domain.

For all tests, values of η on intermediate meshes are obtained by the advection remap method from Section 4.2. The interpolation method from Section 4.1 was also tested, and it produced similar results. The advection remap method is currently

preferred because in our implementation it has better computational performance (it admits easier parallelization). Improvements of our interpolation routines and detailed comparisons between the two methods will be addressed in the future.

6.1 Adaptation to a Static Sinusoidal Region

This example is used to illustrate the utilization of a discrete solution field in TMOP and compare different adaptivity-based target construction methods. We consider three different target constructions. They are oriented towards adaptivity with respect to Size, Shape and coupled Size+Shape. We always start with a Cartesian 16×16 third order mesh. For all of the following tests, our goal is to achieve three times smaller area in regions where $\eta \approx 1$, compared to regions where $\eta \approx 0$.

In all cases, the discrete adaptivity function η is initialized on the initial mesh $\mathscr{M}_0$ by interpolating the following smooth analytic function:

$$\tanh(10(y_0 - 0.5) + \sin(4\pi x_0) + 1) - \tanh(10(y_0 - 0.5) + \sin(4\pi x_0) - 1)\,. \quad (9)$$

Once the above function is interpolated on the initial mesh, the algorithm uses only its finite element version, which is shown in Figure 1.

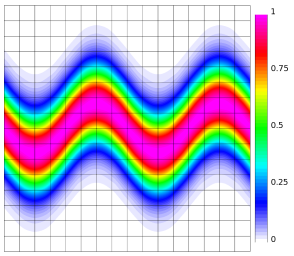

Fig. 1 Third order finite element adaptivity function η on the initial mesh.

Size Adaptation As a first attempt to adapt the local mesh size, we use the following targets and quality metric:

$$W_1(x) = [\eta(x)s + (1 - \eta(x))\alpha s]^{1/2} I\,, \quad \mu_7(T) = |T - T^{-t}|^2\,,$$

where $|T|^2 = \mathrm{tr}(T^t T)$, $\alpha = 7$ is a user-specified size ratio between big and small local size, and s is a local mesh size. Note that μ_7 is a Shape+Size metric. Thus, higher η values should result in smaller local mesh size. The size s is approximated by taking into account the total volume V of the domain, the volume V_η occupied by η, the number N of available elements, and the specified α:

$$\frac{V_\eta}{s} + \frac{V - V_\eta}{\alpha s} = N\,, \quad \text{where} \quad V_\eta = \int_{\mathscr{M}_0} \eta_0(x_0)\,,\ V = \int_{\mathscr{M}_0} 1\,. \quad (10)$$

Note that the calculation of s is done once using the initial mesh. As usual, the integrals in (10) are approximated by a quadrature rule of order comparable to the order of the mesh and η.

The results of this test are shown in Figure 2. The objective function (3) is reduced by approximately 47%, from $F(x_0) = 2.165$ to $F(x) = 1.136$. We observe that the mesh size is adapted towards the bigger η values, but to achieve the desired 3-times reduction in size, we need to use a higher ratio in the target construction ($\alpha = 7$). This is caused by the fact that the combination (μ_7, W_1) optimizes not only the local size, but also tries to achieve equal aspect ratio. Nevertheless, combining shape and size optimization is recommended, as pure size optimization usually results in highly distorted meshes, which causes divergence of the Newton iterations.

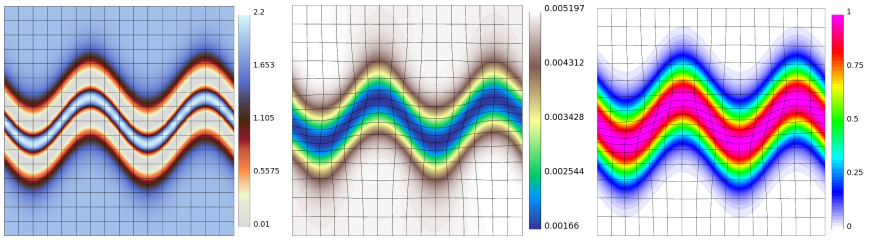

Fig. 2 Size adaptation: initial (μ_9, W_1) metric values (left), $\det(A)$ values on the optimized mesh (middle), and remapped η values on the optimized mesh (right).

Shape Adaptation As the above case suggested, to adapt successfully to η, our method must be able to adapt the local aspect ratio. This can be achieved by the following targets and quality metric:

$$W_2(x) = \begin{pmatrix} \eta(x)\beta + (1-\eta(x)) & 0 \\ 0 & 1 \end{pmatrix}, \quad \mu_{58}(T) = \frac{|T^t T|^2}{\det(T)^2} - 2\frac{|T|^2}{\det(T)} + 2,$$

where $\beta = 3$ is a user-specified desired aspect ratio. Note that μ_{58} is a pure Shape metric. Thus, higher η values should result in higher resolution in the y direction, while smaller η values should lead to a 1:1 aspect ratio.

The results of this test are shown in Figure 3. The objective function (3) is reduced by approximately 22%, from $F(x_0) = 1545.92$ to $F(x) = 1207.33$. We observe that the 3-times reduction in area is achieved in the region specified by η, through the aspect ratio adaptivity. However, optimizing μ_{58} produces bigger local sizes around the top and bottom boundaries, as μ_{58} is not influenced by local size.

Shape+Size Adaptation Our best result for the above problem is achieved by adapting shape and size simultaneously, through the following targets and metric:

$$W_3(x) = \begin{pmatrix} \sqrt{\gamma s} & 0 \\ 0 & \eta(x)\frac{\sqrt{\gamma s}}{\gamma} + (1-\eta(x))\sqrt{\gamma s} \end{pmatrix}, \quad \mu_9(T) = \det(T)|T - T^{-t}|^2,$$

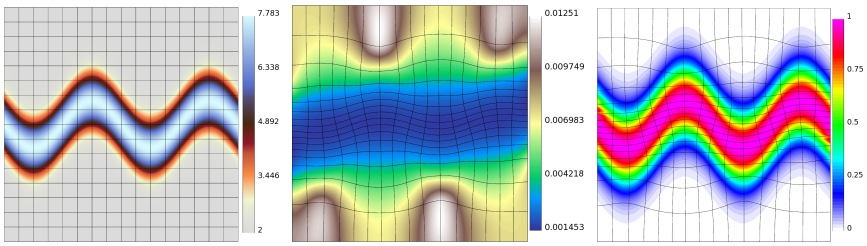

Fig. 3 Shape adaptation: initial (μ_{58}, W_2) metric values (left), $\det(A)$ values on the optimized mesh (middle), and remapped η values on the optimized mesh (right).

where $\gamma = 3$ is a user-specified desired aspect ratio, and s is computed by (10). Note that μ_9 is a Shape+Size metric. Thus, $\eta = 1$ should result in 3:1 resolution in the y direction, and 3 times smaller local size than $\eta = 0$ values.

The results of this test are shown in Figure 4. The objective function (3) is reduced by approximately 65%, from $F(x_0) = 0.756$ to $F(x) = 0.258$. We observe that the 3-times reduction in local mesh size and 3:1 aspect ratio is achieved in the region specified by the highest η values, while the 1:1 aspect ratio is preserved in the rest of the domain.

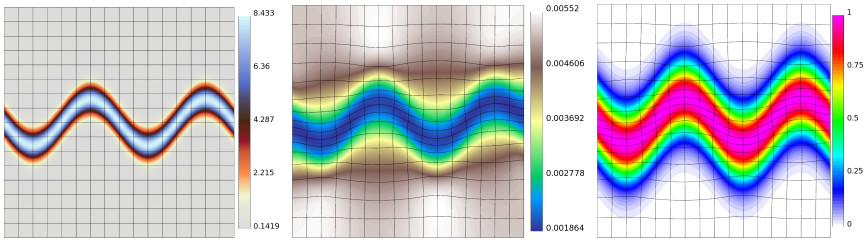

Fig. 4 Shape+Size adaptation: initial (μ_9, W_3) metric values (left), $\det(A)$ values on the optimized mesh (middle), and remapped η values on the optimized mesh (right).

6.2 2D Full ALE Simulation of Gas Impact

This problem represents a complete multi-material ALE simulation of the compressible Euler equations. The test is a simplified high velocity impact introduced in [4]. There are three materials that represent an impactor, a wall, and the background. This problem tests the mesh optimization method's ability to couple to full time dependent ALE simulations that use high-order finite elements. The presented simulation is performed by the BLAST code [5], which utilizes our implementation in the MFEM finite element library.

The domain is $[0,2] \times [0,2]$ with $v \cdot n = 0$ boundary conditions. The material regions are $0.15 \leq x \leq 0.65$ and $0.9 \leq y \leq 1.1$ for the Impactor, $0.1 \leq x \leq 1.0$ for the Wall, and the rest is Background. The problem is run to a final time of $t = 10$. We use a 60×60 second order mesh. The complete thermodynamic setup of this problem and additional details about our multi-material finite element discretization and overall ALE method can be found in [11, 3].

Remesh steps are performed at different times during the simulation, according to the algorithm presented in Section 5. The goal of the mesh optimization is to adapt the size of the mesh towards the locations of the Impactor and the Wall at any given remesh step. To achieve this, we use the following targets and quality metric:

$$W(x) = [\eta(x)s + (1-\eta(x))\alpha s]^{1/2} I, \quad \mu_9(T) = \det(T)|T - T^{-t}|^2,$$

where $\alpha = 10$, and the local size s is computed by (10). The adaptivity field η is constructed from the simulation's discrete volume fractions η_{wall} and η_{imp}. These fields have values in $[0,1]$, and they represent the positions of the Wall and Impactor at any given spatial point. The adaptivity field η is computed as

$$\eta(x) = \max(\eta_{wall}(x), \eta_{imp}(x)).$$

The time evolution of the material positions and the corresponding mesh can be seen in Figure 5. We observe that the algorithm adapts well to the moving materials, while preserving good overall shape throughout the domain.

7 Conclusions

In this paper we outline a framework for optimization of high-order curved meshes that is driven by the state of the overall simulation in which the mesh is being used. This framework is build on the Target-Matrix Optimization Paradigm [19] and [9], and allows high-order applications to have precise control over local mesh quality, while still improving the mesh globally.

While the current paper is intended as an initial extension of the pure geometry-based methods of [9] to the simulation-driven case, we do make the important choice to require only discrete description of the simulation feature to which to adapt to (e.g. provided as a finite element function on the mesh). This is a critical step for the practical applicability of the algorithms we propose and distinguishes us from approaches that require analytical information.

As discrete fields are defined only with respect to the initial mesh, one of the challenges with our approach is that we need to obtain their values on the intermediate meshes produced during the optimization process. We propose two methods to address this challenge. The numerical experiments reported here illustrate results which can be obtained from the second (advection remap) approach. We plan to explore and compare these results to alternative or improved algorithms and per-

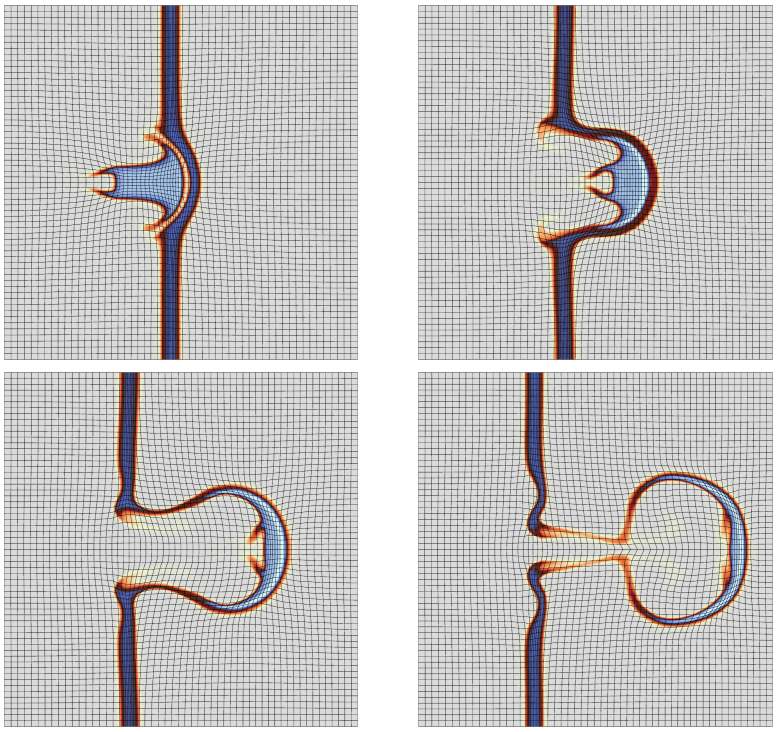

Fig. 5 Time evolution of the materials and mesh positions at times 2.5 (top left), 5 (top right), 7.5 (bottom left) and 10 (bottom right) in the 2D gas impact test case.

form detailed comparison of accuracy and performance in the future. In addition, we will explore a wider range of applications of our methods beyond ALE. These will require more sophisticated target construction methods.

References

1. R. W. Anderson, V. A. Dobrev, T. V. Kolev, D. Kuzmin, M. Q. de Luna, R. N. Rieben, and V. Z. Tomov. High-order local maximum principle preserving (MPP) discontinuous Galerkin finite element method for the transport equation. *J. Comput. Phys.*, 334:102–124, 2017. 10
2. R. W. Anderson, V. A. Dobrev, T. V. Kolev, and R. N. Rieben. Monotonicity in high-order curvilinear finite element arbitrary Lagrangian–Eulerian remap. *Int. J. Numer. Meth. Fluids*, 77(5):249–273, 2015. 3, 8
3. R. W. Anderson, V. A. Dobrev, T. V. Kolev, R. N. Rieben, and V. Z. Tomov. High-order multi-material ALE hydrodynamics. *SIAM Journal on Scientific Computing*, 40(1):B32–B58, 2018. 2, 15
4. A. Barlow, R. Hill, and M. J. Shashkov. Constrained optimization framework for interface-aware sub-scale dynamics closure model for multimaterial cells in Lagrangian and arbitrary Lagrangian-Eulerian hydrodynamics. *J. Comput. Phys.*, 276(0):92–135, 2014. 14

5. BLAST: High-order curvilinear finite elements for shock hydrodynamics. LLNL code, 2018. http://www.llnl.gov/CASC/blast. 2, 14
6. H. Borouchaki, P. L. George, F. Hecht, P. Laug, and E. Saltel. Delaunay mesh generation governed by metric specifications. Part I. Algorithms. *Finite Elements in Analysis and Design*, 25(12):61 – 83, 1997. Adaptive Meshing, Part 1. 5
7. W. Boscheri and M. Dumbser. High order accurate direct arbitrary-Lagrangian-Eulerian ADER-WENO finite volume schemes on moving curvilinear unstructured meshes. *Computers & Fluids*, 136:48 – 66, 2016. 2
8. V. Dobrev, T. Ellis, T. Kolev, and R. Rieben. High-order curvilinear finite elements for axisymmetric Lagrangian hydrodynamics. *Computers & Fluids*, pages 58–69, 2013. 2
9. V. Dobrev, P. Knupp, T. Kolev, K. Mittal, and V. Tomov. The target-matrix optimization paradigm for high-order meshes. *ArXiv e-prints*, 2018. https://arxiv.org/abs/1807.09807. 1, 2, 3, 5, 15
10. V. Dobrev, T. Kolev, and R. Rieben. High-order curvilinear finite element methods for Lagrangian hydrodynamics. *SIAM J. Sci. Comp.*, 34(5):606–641, 2012. 2
11. V. A. Dobrev, T. V. Kolev, R. N. Rieben, and V. Z. Tomov. Multi-material closure model for high-order finite element Lagrangian hydrodynamics. *Int. J. Numer. Meth. Fluids*, 82(10):689–706, 2016. 15
12. C. Ericson. *Real-Time Collision Detection*. CRC Press, Inc., Boca Raton, FL, USA, 2004. 8
13. V. Garanzha, L. Kudryavtseva, and S. Utyuzhnikov. Variational method for untangling and optimization of spatial meshes. *J. Comput. Appl. Math.*, 269:24–41, 2014. 5
14. V. A. Garanzha. Polyconvex potentials, invertible deformations, and thermodynamically consistent formulation of the nonlinear elasticity equations. *Comp. Math. Math. Phys.*, 50(9):1561–1587, 2010. 5
15. S. K. Godunov. A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics. *Matematicheskii Sbornik*, 89(3):271–306, 1959. 10
16. P. T. Greene, S. P. Schofield, and R. Nourgaliev. Dynamic mesh adaptation for front evolution using discontinuous Galerkin based weighted condition number relaxation. *Journal of Computational Physics*, 335:664–687, 2017. 2, 3
17. J.-L. Guermond, B. Popov, and V. Tomov. Entropy-viscosity method for the single material Euler equations in Lagrangian frame. *Computer Methods in Applied Mechanics and Engineering*, 300:402 – 426, 2016. 2
18. W. Huang and R. Russell. *Adaptive Moving Mesh Methods*. Springer, 2011. 2, 5
19. P. Knupp. Introducing the target-matrix paradigm for mesh optimization by node movement. *Engr. with Comptr.*, 28(4):419–429, 2012. 1, 2, 15
20. MFEM: Modular parallel finite element methods library, 2018. http://mfem.org. 8, 11
21. M. Turner, J. Peiró, and D. Moxey. Curvilinear mesh generation using a variational framework. *Computer-Aided Design*, 103:73–91, 2018. 5
22. P. Váchal and P.-H. Maire. Discretizations for weighted condition number smoothing on general unstructured meshes. *Computers & Fluids*, 46(1):479 – 485, 2011. 2, 3