Comparison of Algorithms for High-Order, Metric-Based Mesh Optimization

Devina P. Sanjaya*
Department of Mechanical, Biomedical, and Aerospace Engineering, University of Tennessee, Knoxville, TN 37996, USA

Krzysztof J. Fidkowski†
Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI 48109, USA

Scott M. Murman‡
NASA Ames Research Center, Moffett Field, CA, 94035, USA

We assess the performance of two iterative mesh optimization algorithms for generating high-order, metric-conforming meshes appropriate for use with high-order, finite-element methods. The three key steps in these metric-based approaches are: 1) obtain the desired high-order, Riemannian metric field, 2) generate a linear, metric-conforming mesh, and 3) place the high-order geometry nodes such that the mesh-implied metric conforms to the desired metric field. The procedure to obtain the desired high-order metric field is similar to the Mesh Optimization via Error Sampling and Synthesis (MOESS) developed by Yano[1] for linear meshes. In this work, we consider two approaches to obtain the desired, *higher-order* metric field: 1) performing MOESS on a refined mesh, and 2) extending native MOESS to high-order MOESS (HOMES). The mesh regeneration procedures used to obtain the metric-conforming property are also described in detail. The accuracy benefits offered by these two proposed algorithms are presented for a series of one-dimensional problems. A two-dimensional proof of concept is also provided to establish the value of extending the proposed metric-based optimization algorithm to higher dimensions and to demonstrate the feasibility of the proposed mesh regeneration procedure in higher dimensions.

I. Introduction

The rapid advance and the growing accessibility of affordable high-performance computing systems to the computational fluid dynamics (CFD) community during the last several decades have motivated several research projects that are aimed to improve the accuracy of CFD simulations. Indeed, the additional accuracy is becoming more important for increasingly many engineering applications. While improvement in numerical schemes and physical models yield in higher accuracy, improvement in these aspects of CFD alone is not suffice. Higher-quality meshes are required to get the best performance of these numerical schemes and physical models.[2] The work presented here focuses on generating optimal high-order meshes suitable for use with high-order, finite-element methods. The optimal placement of mesh resolution (i.e., the shape and size of a mesh element) is guided by a desired, higher-order metric field and *a posteriori* error estimates.

Encoding mesh resolution information into a metric field has been done in several previous works.[3,4] In the context of mesh adaptation and generation, the metric information can be derived through heuristic,[5–7] semi-heuristic,[8–11] and more recently, rigorous[1,12] ways. These methods have been successfully applied to generate meshes that minimize error or computational cost, and are known as metric-based mesh adaptation

*Assistant Professor, AIAA Member
†Associate Professor, AIAA Senior Member
‡Aerospace Engineer, AIAA Member

approaches. Another approach to improve the approximation power of a given mesh is to increase the approximation power of mesh elements. This is usually done by refining the mesh (h adaptation), increasing the polynomial order for solution approximation or output prediction (p adaptation), or combining the two (hp adaptation).[13–15] More accurate solution approximation or output prediction can also be obtained by optimizing the test space.[16–19] Yet, another option is to tailor finite-element basis functions to the problem of interest. Several methods, such as the partition of unity method,[20] the extended finite-element method,[21] isogeometric analysis,[22] and the discontinuous enrichment method,[23] tailors the basis functions *a priori*. Though it is possible for some problems, it is generally hard, especially for complex flows in which the locations of features such as shocks and shear layers are not known ahead of time. Tailoring basis functions *a posteriori* is a more robust alternative. Previous work by Sanjaya and Fidkowski[24] shows that tailoring basis functions *a posteriori* can be done by finding the optimal locations of the high-order geometry nodes within a mesh element.

The high-order, metric-based optimization algorithms presented in this paper combine the key ideas from previous works by Yano[1] and Sanjaya and Fidkowski.[24,25] Specifically, the procedures to obtain the desired, *higher-order* metric field in the proposed optimization algorithms follow similar steps as the one developed by Yano for linear mesh elements (i.e., MOESS) with some changes to incorporate the high-order geometry information. The fundamental concepts pertaining to extending key ideas of MOESS for high-order mesh elements will be discussed in details. Once we obtain the desired, high-order metric field, we solve a metric-based optimization problem to optimally place the vertices and high-order geometry nodes. This step essentially tailors the basis functions *a posteriori* on each element and generates warped mesh elements, which is the key idea of the warped-element refinement method developed by Sanjaya and Fidkwoski.[24,25] We refer to the resulting mesh as a high-order, metric-conforming mesh. Our eventual goal is to implement this algorithm as part of a mesh generation procedure to be used with a very high-order finite-element method (e.g., up to 16^{th} order).

The remainder of this paper is outlined as follows. In Section II, we give a brief review on our chosen discretization, a high-order, discontinuous-Galerkin method. In Section III, we present the mechanisms and benefits of warped-element refinement method. In Section IV, we describe the iterative mesh optimization algorithms used to determine the desired, high-order metric field. In Section V, we detail our approach for placing the geometry nodes such that the metric-conforming property is obtained. In Section VI, we showcase the accuracy benefits of high-order, metric-conforming meshes in one and two dimensions. Finally, we conclude with a summary and our plans for future work in Section VII.

II. Discretization

Our focus for the development of the high-order, metric-based mesh optimization algorithms is on the discontinuous-Galerkin (DG) method, mainly because we have experience with it, and because it is a relatively mature high-order method suitable for convection-dominated flows that are prevalent in aerospace engineering. DG,[26–28] as a finite-element method, approximates the state $\mathbf{u}(\vec{x}) \in \mathbb{R}^s$, where s is the state rank, in functional form using linear combinations of basis functions on each element. No continuity constraints are imposed between adjacent elements. Denoting by T_h the set of N_e elements in a non-overlapping tessellation of the domain Ω, the state on element e, Ω_e, is approximated as

$$\mathbf{u}_h(\vec{x}(\vec{\xi}))\Big|_{\Omega_e} = \sum_{n=1}^{N_p} \mathbf{U}_{en}\phi_{en}^{\text{glob}}(\vec{x}(\vec{\xi})). \tag{1}$$

In this equation, N_p is the number of basis functions per element, $\mathbf{U}_{en}$ is the vector of s coefficients for the n^{th} basis function on element e: $\phi_{en}^{\text{glob}}(\vec{x}(\vec{\xi}))$. $\vec{x}$ denotes the global coordinates, and $\vec{\xi}$ denotes the reference-space coordinates in a master element. Formally, we write $\mathbf{u}_h \in \boldsymbol{\mathcal{V}}_h = [\mathcal{V}_h]^s$, where, if the elements are not curved,

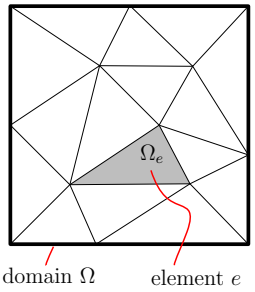


Figure 1. A partition of a square domain into triangular elements.

$$\mathcal{V}_h = \{u \in L_2(\Omega) : u|_{\Omega_e} \in \mathcal{P}^p \ \forall \Omega_e \in T_h\},$$

and $\mathcal{P}^p$ denotes polynomials of order p on the element. The reference-to-global mapping, $\vec{x}(\vec{\xi})$, is polynomials of order q ($\mathcal{P}^q$), where q is the geometry order of the element. A caveat here is that for warped (curved)

elements, the polynomial approximation is usually performed on a master reference element, so that following the reference-to-global mapping, the state approximation on curved elements is not strictly of order p in global space. In this work, we take advantage of this observation by placing the high-order geometry nodes at their optimal locations based on the desired metric field. Of course, this advantage can only be obtained if we properly encode the high-order geometry information into the desired metric field.

III. Warped-Element Refinement Method

Today, curving of mesh elements is mainly performed out of necessity in obtaining accurate boundary representation and/or in maintaining the validity of the mesh. That is, we usually only pay attention to the precise locations of the high-order geometry nodes that lie on the boundary. Heuristics, such as avoiding clustering of the nodes to maximize the validity of the elements, are also often used to determine the location of these high-order geometry nodes. Recently, Sanjaya and Fidkowski[24,25] investigated the importance of the precise location these high-order nodes for better solution approximation and output prediction. To obtain such benefits, some clustering of the high-order nodes within an element is allowed as long as the validity of the element is maintained. The resulting mesh will contain warped (curved) mesh elements in the regions where complex geometries and/or important flow features are present. This process is known as warped-element refinement (q adaptation) since we are only changing the inner shape of the mesh elements. Unlike the previous work,[24] where the warping of the mesh elements is performed by directly minimizing the solution or output error, we now use the desired, high-order metric field to guide the element warping process. Later, we will see that warping of mesh elements is indeed vital to obtaining the desired metric-conforming property. The duality between metric-based and solution-based optimization was previously observed in the work done by Sanjaya et al.[29]

Similar to solution approximation, a different set of basis functions is used to approximate the geometry. In this work, we choose to use a polynomial mapping from the reference element to the global element, as this allows us to take advantage of the existing infrastructure that is already available in most high-order codes. The formula for the mapping function is given in Figure 2, where q is the order of this polynomial, $N_q = (q+1)(q+2)/2$ is the total number of degrees of freedom in the mapping, $\vec{\xi} = [\xi, \eta]^{\mathrm{T}}$ is the coordinate in reference space, and $\vec{x} = [x, y]^{\mathrm{T}}$ is the coordinate in global space. Some results presented in this paper

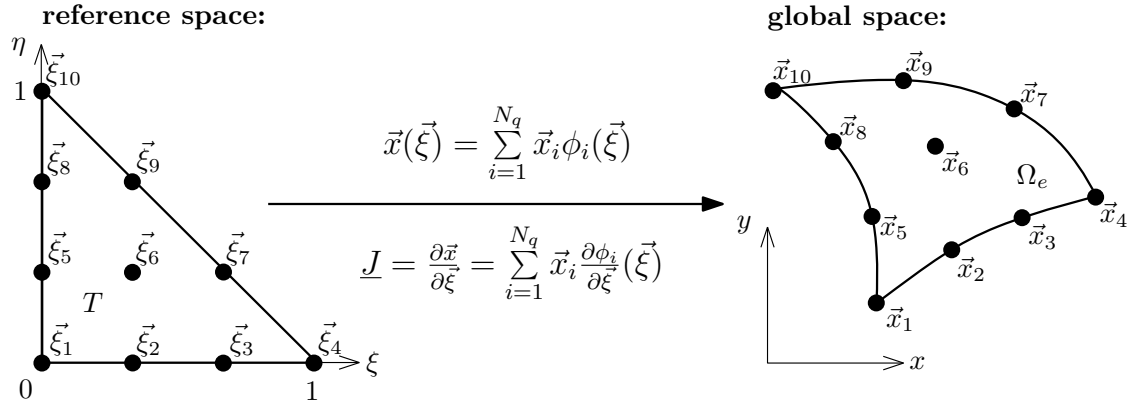


Figure 2. Example of a mapping from a unit reference triangle to a curved-element in global space. The mapping functions for the global coordinates x, y (rolled into $\vec{x}$) are polynomials of order $q = 3$ in this case, for a total of $N_q = 10$ geometry nodes.

are generated using Lagrange basis functions with uniform node distribution in reference space since they allow for an intuitive specification of the high-order element. We also explore the benefits of using Bernstein basis functions, which are commonly used for computer-aided design (CAD), with Gauss-Legendre-Lobato (GLL) node distribution in reference space. Other basis functions could also be used to define the element geometry, as long as the resulting mapped elements constitute a complete, non-overlapping tessellation of the domain.

The coordinates $\vec{x}_i$ should be chosen consistently with the corresponding reference-space nodes, $\vec{\xi}_i$, which are equally spaced on the reference element. For example, in Figure 2, $\vec{\xi}_6$ is the centroid of the reference triangle, so $\vec{x}_6$ should be located somewhere in the middle of the curved element. On edges/faces that are

on domain boundaries, these nodes are typically on the geometry. However, these requirements do not pin down their locations, and heuristics or quality metrics such as maximizing the Jacobian determinant are often used in high-order node placement.

In general, for arbitrary curved elements, a function that is an order p polynomial in reference space does not remain an order p polynomial, or even a polynomial at all, in the global space coordinates. Specifically, a polynomial basis function in reference space, $\phi^{\text{ref}}(\vec{\xi})$, maps to a global basis function according to

$$\phi^{\text{glob}}(\vec{x}(\vec{\xi})) = \phi^{\text{ref}}(\vec{\xi}), \tag{2}$$

where $\vec{x}(\vec{\xi})$ is the geometry mapping given in Figure 2. Thus, by moving an element's high-order geometry nodes, we distort this mapping and gain control over the appearance of the high-order basis functions. The proposed mesh optimization algorithms take advantage of this observation by incorporating the high-order geometry information into the desired metric field and placing all geometry nodes (i.e., vertices and high-order geometry nodes) according to this desired, high-order metric field.

IV. Iterative Metric-Based Mesh Optimization Algorithm

In this section, we provide a brief review of MOESS introduced by Yano[1] for linear meshes, and detail the required modifications to this approach in order to obtain the desired metric field suitable for high-order meshes. Two approaches for obtaining the desired, high-order metric field are considered: 1) performing MOESS on a refined mesh, and 2) natively extending MOESS to higher order (HOMES). For our first proposed approach, no modification is made to the MOESS algorithm, and the high-order geometry information is obtained by simply performing the original MOESS on a refined mesh. In one dimension, for example, the refined mesh will have q times the number of mesh elements of the original mesh of interest. For our second proposed approach, the high-order geometry information is directly incorporated into the desired metric field through a procedure that follows similar steps as in MOESS. The main modifications occur in the error model and the use a set of basis functions to construct a continuous approximation of the metric field. These basis functions are fitted through the metric at the vertices and the high-order geometry nodes. With these modifications, HOMES generates a more accurate high-order metric field than MOESS on a refined mesh, though the accuracy comes with a higher computational cost. Regardless of their differences, both optimization algorithms considered here are designed to iteratively determine the optimal change in the metric field given prescribed cost and error models. Additional details can also be found in our previous works.[12,25]

IV.A. Metric-Based Meshing

A Riemannian metric field, $\mathcal{M}(\vec{x})$, is a field of symmetric positive definite (SPD) tensors that functions as a directional "yardstick" for measuring distances. In multiple dimensions, the metric distance of points separated by $\delta\vec{x}$ is measured as

$$\delta\ell = \sqrt{\delta\vec{x}^T \mathcal{M} \delta\vec{x}}. \tag{3}$$

By definition, each edge of the elements in a metric-conforming mesh has approximately the same metric length, i.e., Equation 3 integrated along the edge.

The success of metric-based meshing algorithm relies on its ability to extract the mesh-implied metric information from the reference-to-global mapping Jacobian on each element, and to accurately determine the optimal changes to this mesh-implied metric, $\mathcal{M}_0(\vec{x})$, for the problem at hand. The metric field is then modified by applying affine-invariant changes[4] via a symmetric *step matrix*, $\mathcal{S} \in \mathbb{R}^{d\times d}$, according to:

$$\mathcal{M} = \mathcal{M}_0^{\frac{1}{2}} \exp(\mathcal{S}) \mathcal{M}_0^{\frac{1}{2}}. \tag{4}$$

This relation tells us that $\mathcal{S} = 0$ leaves the metric unchanged, while diagonal values in $\mathcal{S}$ of $\pm 2\log 2$ halve/double the metric stretching sizes. In MOESS and MOESS on a refined mesh, transfers of metrics between elements and nodes involve an averaging process. In HOMES, we have a continuous approximation of the metric field, and the sensitivities are computed with respect to the coefficients of the basis functions used to approximate the step matrix field. Although having a continuous representation of the metric field eliminates the need for averaging of the metric, we will see later that averaging of the mesh-implied metric

at the shared nodes can eliminate the undesirable oscillations present in the continuous representation of the desired metric field.

IV.B. Error Convergence Model

A metric-based mesh optimization algorithm requires an error model that accurately describes the error changes as the metric changes. In a practical setting, an error model involves refinement and sampling procedures. However, for our test cases, we choose to simply use the least-squares error between the approximated and exact solutions, or in the case of output predictions, we use the approximated and exact outputs. Since no sampling is performed, the convergence rate, $\mathcal{R}_e$, is determined by the asymptotic convergence rate based on the solution approximation order, p.

IV.B.1. MOESS on a Refined Mesh

For an element, Ω_e, with a current error $\mathcal{E}_{e0}$ and a proposed metric step matrix of $\mathcal{S}_e$, the general elemental error model can be written as:

$$\mathcal{E}_e = \mathcal{E}_{e0} \exp\left[\text{tr}(\mathcal{R}_e \mathcal{S}_e)\right] \quad \Rightarrow \quad \frac{\partial \mathcal{E}_e}{\partial \mathcal{S}_e} = \mathcal{E}_e \mathcal{R}_e. \tag{5}$$

The total error over the mesh is the sum of the elemental errors, $\mathcal{E} = \sum_{e=1}^{N_e} \mathcal{E}_e$. During optimization, we will be able to change the step matrices at the mesh vertices, $\mathcal{S}_v$, which will map to the step matrices at the elements, $\mathcal{S}_e$.

IV.B.2. HOMES

In order to directly encode the high-order geometry information into the metric field, the error model for HOMES must be able to accurately model the error changes within an element. We do so by tracking the error kernel $E^S_{\mathcal{M}_0}$. Thus, the current error on element e ($\mathcal{E}_{e0}$) and the general elemental error model ($\mathcal{E}_e$) can be written as:

$$\mathcal{E}_{e0} = \int E^S_{\mathcal{M}_0} d\vec{x}, \tag{6}$$

$$\mathcal{E}_e = \int E^S_{\mathcal{M}_0} \exp\left[\text{tr}(R_e S_e)\right] d\vec{x}. \tag{7}$$

This error model can be derived from Theorem D.3 in Yano's thesis.[1] In one dimension, the error kernel simplifies to

$$E^S_{\mathcal{M}_0} = \underbrace{\left[\frac{\partial^{p+1} u}{\partial x^{p+1}}\right]^2}_{f(x)} \mathcal{M}_0^{-(p+1)}. \tag{8}$$

In practice, the error kernel is computed through a sampling procedure. The goal of this sampling procedure is to solve for $f(x)$. The sampling procedure consists of two steps. First, we split each element into sub-elements, and for each sub-element i, we have

$$\mathcal{E}_{e_{0i}} = \int_{e_{0i}} f(x) \mathcal{M}_0^{-(p+1)} dx \approx f(x_i) \int_{e_{0i}} \mathcal{M}_0^{-(p+1)} dx \tag{9}$$

$$\Rightarrow f(x_i) = \frac{\mathcal{E}_{e_{0i}}}{\int_{e_{0i}} \mathcal{M}_0^{-(p+1)} dx}.$$

Second, we interpolate linearly between the data points obtained in the first step.

IV.C. Cost Model

We base our cost model on degrees of freedom, `dof`, which on each element simply depends on the approximation order p, assumed constant over the elements. In one dimension, this cost is $(p+1)$ `dof` per element. By Equation 4 and properties of the metric tensor, when the step matrix S_e is applied to the metric of element e, the area of the element decreases by $\exp\left[\frac{1}{2}\mathrm{tr}(S_e)\right]$. Equivalently, the number of new elements, and hence degrees of freedom, occupying the original area Ω_e increases by this factor. So, the elemental cost model is

$$C_e = C_{e0}\exp\left[\frac{1}{2}\mathrm{tr}(\mathcal{S}_e)\right] \quad \Rightarrow \quad \frac{\partial \mathcal{C}_e}{\partial \mathcal{S}_e} = \mathcal{C}_e\frac{1}{2}\mathcal{I}, \tag{10}$$

where $C_{e0} = \texttt{dof}_{e0}$ is the current number of degrees of freedom on element e, and $\mathcal{I}$ is the identity tensor. The total cost over the mesh is the sum of the elemental costs, $\mathcal{C} = \sum_{e=1}^{N_e} C_e$. Note that we use the same cost model for MOESS and HOMES.

IV.D. Metric Optimization Algorithm

As previously mentioned, both MOESS and HOMES follow a similar procedure. Thus, to avoid repetition, we start with a review of the MOESS algorithm and simply point out the modifications that are required for HOMES.

IV.D.1. MOESS on a Refined Mesh

Given a refined mesh with its mesh-implied metric $\mathcal{M}_0(\vec{x})$, elemental error indicators $\mathcal{E}_{e0}$, and elemental rate tensor estimates, $\mathcal{R}_e$, the goal of the metric optimization algorithm is to determine the step matrix field, $\mathcal{S}(\vec{x})$, that minimizes the error at a fixed cost. The step matrix field is approximated by values at the mesh vertices, $\mathcal{S}_v$, which are arithmetically-averaged to adjacent elements,

$$\mathcal{S}_e = \frac{1}{|V_e|}\sum_{v\in V_e}\mathcal{S}_v, \tag{11}$$

where V_e is the set of vertices ($|V_e|$ is the number of them) adjacent to element e. At the end of the optimization, we will obtain the required $\mathcal{S}_v$ that gives us the minimum total error $\mathcal{E}$ at a targeted total cost $\mathcal{C}$. For MOESS is a gradient-based optimization algorithm, it requires derivatives of the error and cost with respect to $\mathcal{S}_v$, which are computed using the chain rule and Equation 11. And since the cost only depends on the *trace* of the step matrix, we can separate the vertex step matrices into trace ($s_v\mathcal{I}$) and trace-free ($\widetilde{\mathcal{S}}_v$) parts,

$$\mathcal{S}_v = s_v\mathcal{I} + \widetilde{\mathcal{S}}_v. \tag{12}$$

The derivatives of the error with respect to s_v and $\widetilde{\mathcal{S}}_v$ are

$$\frac{\partial \mathcal{E}}{\partial s_v} = \mathrm{tr}\left(\frac{\partial \mathcal{E}}{\partial \mathcal{S}_v}\right), \qquad \frac{\partial \mathcal{E}}{\partial \widetilde{\mathcal{S}}_v} = \frac{\partial \mathcal{E}}{\partial \mathcal{S}_v} - \frac{\partial \mathcal{E}}{\partial s_v}\frac{\mathcal{I}}{d}. \tag{13}$$

The MOESS algorithm[1] is as follows:

1. Given a mesh, solution, and adjoint, calculate $\mathcal{E}_e, \mathcal{C}_e, \mathcal{R}_e$ for each element e.
2. Set $\delta s = \delta s_{\max}/n_{\text{step}}$, $\mathcal{S}_v = 0$.
3. Begin iteration: $i = 1 \ldots n_{\text{step}}$
   (a) Calculate $\mathcal{S}_e$ from Equation 11, $\frac{\partial \mathcal{E}_e}{\partial \mathcal{S}_e}$ from Equation 5, and $\frac{\partial \mathcal{C}_e}{\partial \mathcal{S}_e}$ from Equation 10.
   (b) Calculate derivatives of $\mathcal{E}$ and $\mathcal{C}$ with respect to s_v and $\widetilde{\mathcal{S}}_v$ using Equation 13.
   (c) At each vertex form the ratio $\lambda_v = \frac{\partial \mathcal{E}/\partial s_v}{\partial \mathcal{C}/\partial s_v}$ and
      - Refine the metric for 30% of the vertices with the largest $|\lambda_v|$: $\mathcal{S}_v = \mathcal{S}_v + \delta s\mathcal{I}$
      - Coarsen the metric for 30% of the vertices with the smallest $|\lambda_v|$: $\mathcal{S}_v = \mathcal{S}_v - \delta s\mathcal{I}$
   (d) Update the trace-free part of S_v to enforce stationarity with respect to shape changes at fixed area: $S_v = S_v + \delta s(\partial\mathcal{E}/\partial\widetilde{\mathcal{S}}_v)/(\partial\mathcal{E}/\partial s_v)$. Note that we skip this step in one dimension.

(e) Rescale $S_v \rightarrow S_v + \beta\mathcal{I}$, where β is a global constant calculated from Equation 10 to constrain the total cost to the desired `dof` value: $\beta = \frac{2}{d}\log\frac{\mathcal{C}_{\text{target}}}{\mathcal{C}}$, where $\mathcal{C}_{\text{target}}$ is the target cost.

Note that λ_v is a Lagrange multiplier in the optimization, which is the ratio of the marginal error to marginal cost of a step matrix trace increase. The above algorithm iteratively equidistributes λ_v globally so that, at optimum, all elements have the same marginal error to cost ratio.

IV.D.2. HOMES

The HOMES algorithm follows the similar steps as the MOESS algorithm with some fundamental modifications. The first modification is to create continuous representations of the metric and step matrix fields using two different sets of basis functions, one set for each field. With these continuous representations, the metric and step matrix can now be computed at any points in the computational domain. Furthermore, all sensitivities in the HOMES algorithm are now computed with respect to the coefficients of the basis functions used to approximate the step matrix field. The second modification is to formulate the error model such that it also models the sub-element error behavior. We do so by tracking the error kernel, as previously described in Section IV.B.2.

V. Mesh Regeneration Procedure

So far, we have introduced the idea of element warping to improve the approximation power of mesh elements and the ease of using metric fields to store the mesh resolution information. In this section, we describe our mesh regeneration approach that combines these two concepts. We are now exploring two mesh regeneration procedures to place all geometry nodes at their optimal locations without solving any global optimization problems and compromising greatly on the accuracy benefits.

V.A. MOESS on a Refined Mesh

The mesh regeneration procedure that is coupled with MOESS on a refined mesh does not require us to solve any post-processs optimization problems. Since the desired, high-order metric field obtained from MOESS on a refined mesh is merely a crude approximation, we believe that a simple, less precise mesh generation algorithm is sufficient to obtain the benefits that MOESS on a refined mesh can offer.

The refined mesh on which we perform MOESS is generated by splitting each edge of the original mesh elements q times. This means, in one dimension, the refined mesh is a linear mesh that has q times the number of mesh elements of our original mesh. Referring back to the basic idea of the metric-based meshing, we place all geometry nodes on the refined mesh such that $\sqrt{\Delta\vec{x}^{\mathrm{T}}\mathcal{M}\Delta\vec{x}} = 1$. In one dimension, this simplifies to $\int\sqrt{\mathcal{M}}dx = 1$. The last step in the mesh regeneration process is then to take some of the geometry nodes in the refined mesh as high-order geometry nodes in the original mesh of the interest. For example, for a $q = 2$ mesh, every other nodes in the refined mesh are $q = 2$ nodes in the mesh of interest. Here, we want to note that this mesh regeneration process indeed does not guarantee to produce a valid mesh, and as we will see in Section VI, that this approach is not as robust as our second approach, especially for $q > 2$. Furthermore, we want to note that this method cannot be directly extended to higher dimensions.

V.B. HOMES

Our main purpose of extending native MOESS to higher order is to obtain a more accurate high-order metric field. To achieve this goal, we use a more sophisticated mesh regeneration procedure for HOMES. Here, the mesh regeneration procedure is done in two steps. At the first step, the vertices ($q = 1$) nodes are placed such that each edge length of mesh elements has approximately the same metric length. This results in a linear, metric-conforming mesh. At the second step, the high-order geometry nodes are placed such that the mesh-implied metric conforms to the desired metric field. This step requires us to solve an optimization problem locally on each mesh element. The optimization problem formulation is detailed as follows.

V.B.1. Design Variables

Our design variables are the locations of the high-order geometry nodes ($\boldsymbol{\delta}$) within an element. For our one-dimensional tests, we have relatively small to moderate number of design variables. So far, we have not

observed any difficulty in finding the optimum solution. However, in two dimensions, finding the optimum solution for large number of design variables might be a challenge, though, we expect to observe some benefits.

V.B.2. Objective Function

Our goal is to place the high-order geometry nodes at their optimal location such that the high-order mesh is metric conforming. To do so, the objective function is computed based on the least-squares error in the metric approximation on each element ($\varepsilon_{\mathcal{M},e}$):

$$(\varepsilon_{\mathcal{M},e})^2 = \int_{\Omega_e} \left| \underline{J}(\boldsymbol{\delta})^{-T} \underline{J}(\boldsymbol{\delta})^{-1} - \mathcal{M}_{\text{desired}}(\boldsymbol{\delta}) \right| dA_e. \tag{14}$$

The desired metric is obtained through HOMES, while the approximated metric is calculated based on the reference-to-global mapping Jacobian matrix (defined in Figure 2).

V.B.3. Constraints

The proposed optimization problem involves a volume constraint:

$$c_e \equiv \frac{\min(|\underline{J}(\vec{\xi}_{\text{int}})|)}{V_0} - \eta_V > 0 \tag{15}$$

The volume constraint is set by enforcing a limit in how small the Jacobian determinant can be as a fraction (η_V) of initial element volume (V_0). The Jacobian determinant is calculated at interrogation points in the reference space ($\vec{\xi}_{\text{int}}$), which consist of a large number of equally-spaced points within the element. Note that a non-negative Jacobian determinant is needed to ensure that all mesh elements are valid so that we obtain physical solution. The more interrogation points that are used, the better the chances are that the element is valid everywhere, although this is not guaranteed for for a finite number of points.

V.C. Optimization Problem Formulation

We formulate our constrained optimization problem as follows:

$$\begin{aligned} &\text{minimize} \int_{\Omega_e} \left| \underline{J}(\boldsymbol{\delta})^{-T} \underline{J}(\boldsymbol{\delta})^{-1} - \mathcal{M}_{v,\text{desired}}(\boldsymbol{\delta}) \right| dA_e, \quad \text{w.r.t. } \boldsymbol{\delta}, \\ &\text{subject to } c_e \equiv \frac{\min(|\underline{J}(\vec{\xi}_{\text{int}})|)}{V_0} - \eta_V > 0 \end{aligned} \tag{16}$$

For our one-dimensional test cases, we solve the constrained optimization problem using Matlab's `fmincon` function with the sequential quadratic programming (SQP) algorithm. An automatic tuning procedure is implemented to determine η_V for each element. Currently, gradient calculations are performed using finite differences.

VI. Results

The test cases presented here are formulated to demonstrate the benefits of high-order mesh elements and to give us a deeper understanding of important concepts pertaining to high-order, metric-conforming mesh optimization. We will start with a series of one-dimensional test cases to thoroughly assess the performance of MOESS on a refined mesh and high-order MOESS (HOMES) in terms of numerical benefits, robustness, convergence, and automation. Due to the simplicity of one-dimensional problems, we are able to observe and analyze some behaviors that otherwise would be hard to see in higher dimensions. We then present a two-dimensional proof of concept that further emphasizes the benefits high-order, metric-conforming meshes and demonstrates the feasibility and benefits of extending one of the proposed metric-based approaches to higher dimensions.

VI.A. Reproducing MOESS with $q = 1$ HOMES

The first test case is a benchmark test case to ensure that HOMES with $q = 1$ can reproduce MOESS. Two exact solutions are considered. First, we consider a smoothly varying exponential function,

$$u^{\text{exact}}_{\text{smooth}} = \exp(-5x).$$

This function is chosen to simulate a "boundary layer"-type feature, which must be well approximated in order to obtain an accurate output prediction in flows governed by the compressible Navier-Stokes equations. Furthermore, the smoothness and the monotonic behavior of the function allow us to have an educated guess on what the optimal node distribution would be. In this case, we expect the geometry nodes to be moved toward the left domain with monotonically increasing spacings between the nodes as we move towards the right domain. Second, we consider a smooth function with rapid changes (shown in Figure 3),

$$u^{\text{exact}}_{\text{rapid changes}} = \frac{\cosh(\pi x)}{(1 + 0.75\cos(2\pi x))}.$$

This function is chosen to test the ability of our proposed optimization algorithms to capture sharp solution gradients and to recover the symmetry of the problem.

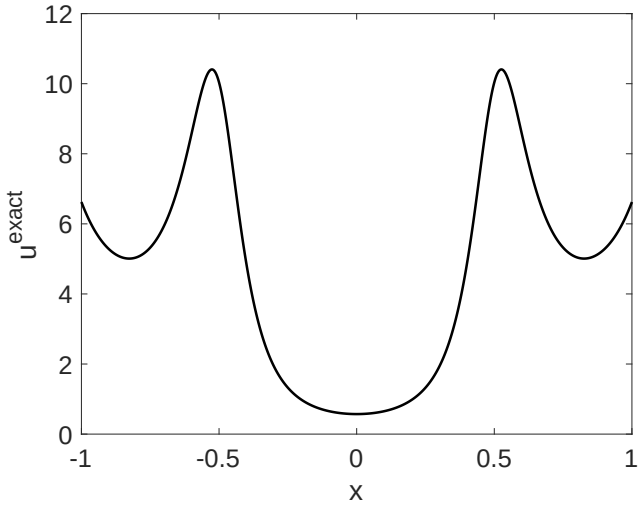
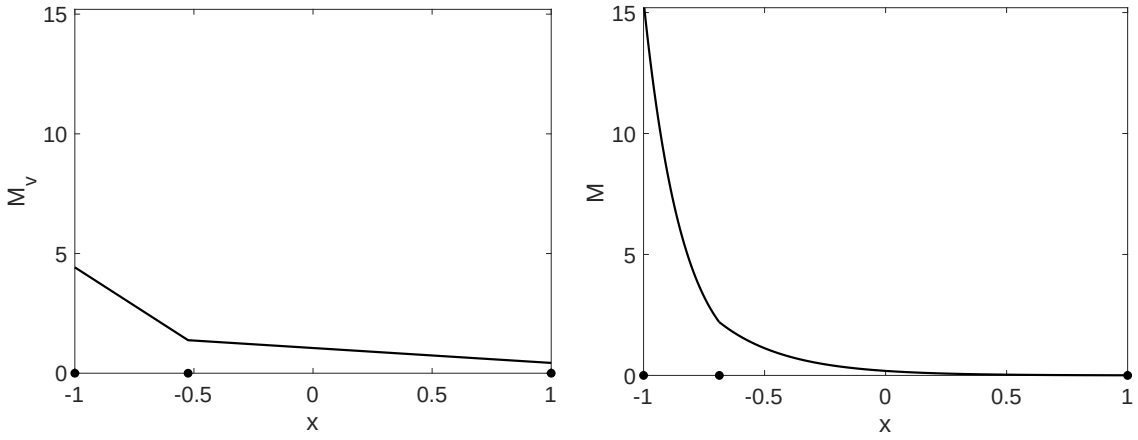


Figure 3. Exact solution with rapid changes $\left(u^{\text{exact}}_{\text{rapid changes}}\right)$

The metric fields obtained from MOESS and HOMES with $q = 1$ for these exact solutions are shown in Figures 4 and 5, respectively. Here, we can see that both methods produce similar metric fields, and hence, converge to similar meshes, as expected.

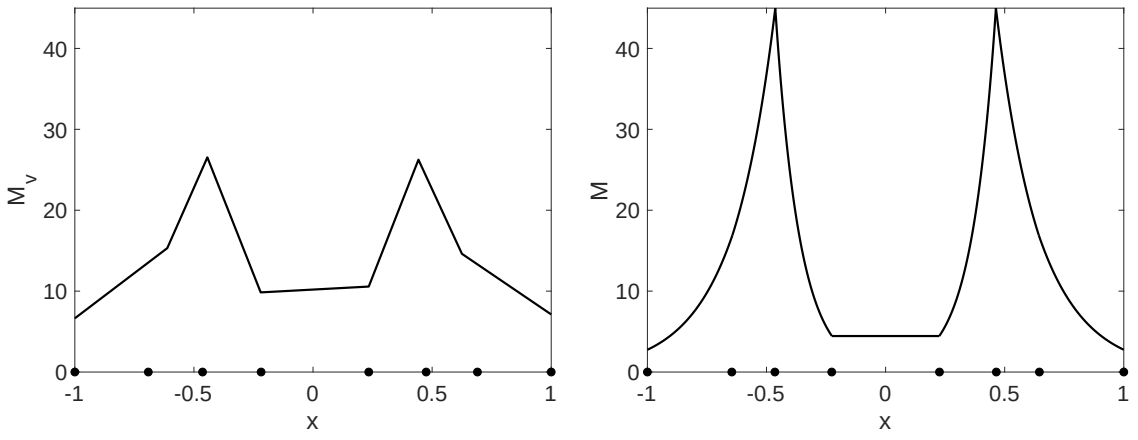
VI.B. Higher-Order Metric Representation with HOMES

Now that we have seen the performance of $q = 1$ HOMES, we are in a position to assess the performance of HOMES in obtaining the desired, higher-order metric field. Figure 6 shows the $p = 4$ metric for $u^{\text{exact}}_{\text{rapid changes}}$ with $q = [1, 2, 3]$. These metrics are generated based on geometry and metric approximations using Lagrange basis functions with uniform node distributions in the reference space. First, notice that all metrics shown in Figure 6 have the same baseline shape and preserve the symmetry of the solution despite the minor differences. Second, increasing q gives us better resolution around the two peaks, and this results in lower solution error (ε) for higher q.



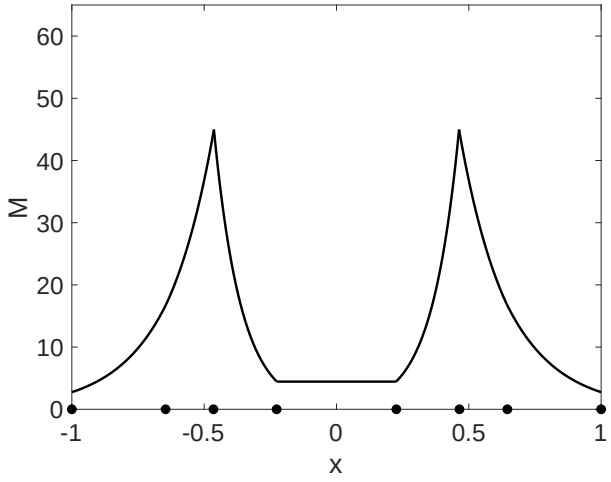
(a) MOESS (b) HOMES with $q = 1$

Figure 4. $p = 1$ **metric comparison for** $u^{\text{exact}}_{\text{smooth}}$ **on two linear mesh elements. The metric generated using MOESS is plotted based on the metric at vertices. For ease of reading, the geometry nodes are shown as black dots on the** x**-axis.**

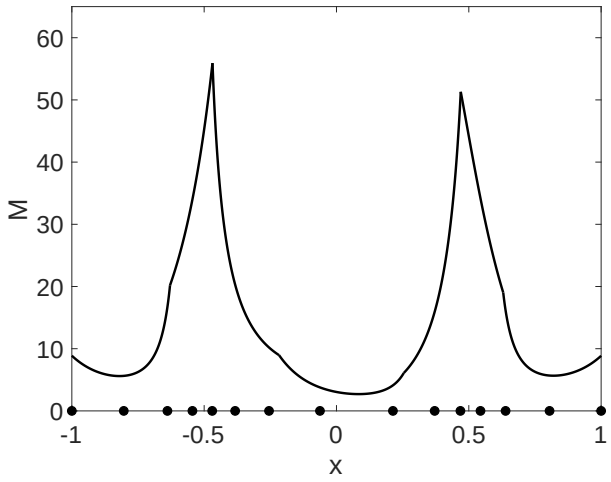


(a) MOESS (b) HOMES with $q = 1$

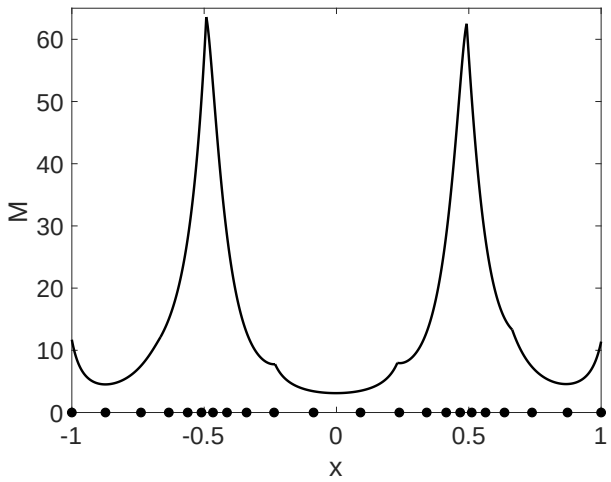
Figure 5. $p = 4$ **metric comparison for** $u^{\text{exact}}_{\text{rapid changes}}$ **on seven linear mesh elements. The metric generated using MOESS is plotted based on the metric at vertices. For ease of reading, the geometry nodes are shown as black dots on the** x**-axis.**



(a) $q = 1$ ($\varepsilon = 1.934 \times 10^{-2}$)



(b) $q = 2$ ($\varepsilon = 9.695 \times 10^{-3}$)



(c) $q = 3$ ($\varepsilon = 6.428 \times 10^{-3}$)

Figure 6. $p = 4$ **metric for** $u^{\text{exact}}_{\text{rapid changes}}$ **on seven mesh elements with** $q = [1, 2, 3]$**. For ease of reading, the geometry nodes are shown as black dots on the** x**-axis.**

VI.C. MOESS on a Refined Mesh vs. HOMES

The goal of this test case is to compare the accuracy benefits obtained from MOESS on a refined mesh and HOMES for increasing p and/or q. This study is conducted using Lagrange basis functions with uniform node distribution in the reference space for geometry and metric approximations. To provide a baseline comparison for our proposed approaches, we provide comparison with results obtained on uniform meshes and on meshes that are optimized by directly minimizing the least-squares error between the approximated and exact solutions. This solution-based optimization is performed globally (i.e., all geometry nodes are movable, except the boundary nodes).

Figure 7 shows the least-squares error in approximating $u^{\text{exact}}_{\text{smooth}}$ with two linear mesh elements. As expected, we observe lower error as p increases. Furthermore, notice that HOMES gives us slightly lower error than MOESS. Finally, the global solution-based optimization gives us the lowest error. This is not surprising since our metric-based optimization is performed locally on each element. Although global optimization will give us lower error, the cost of solving the optimization problem globally might be too large for practical engineering purposes.

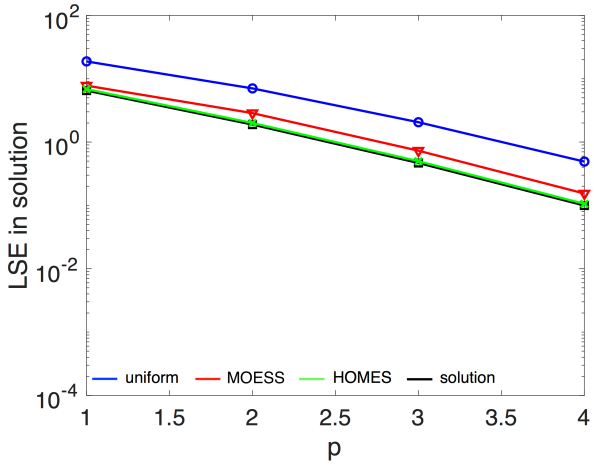
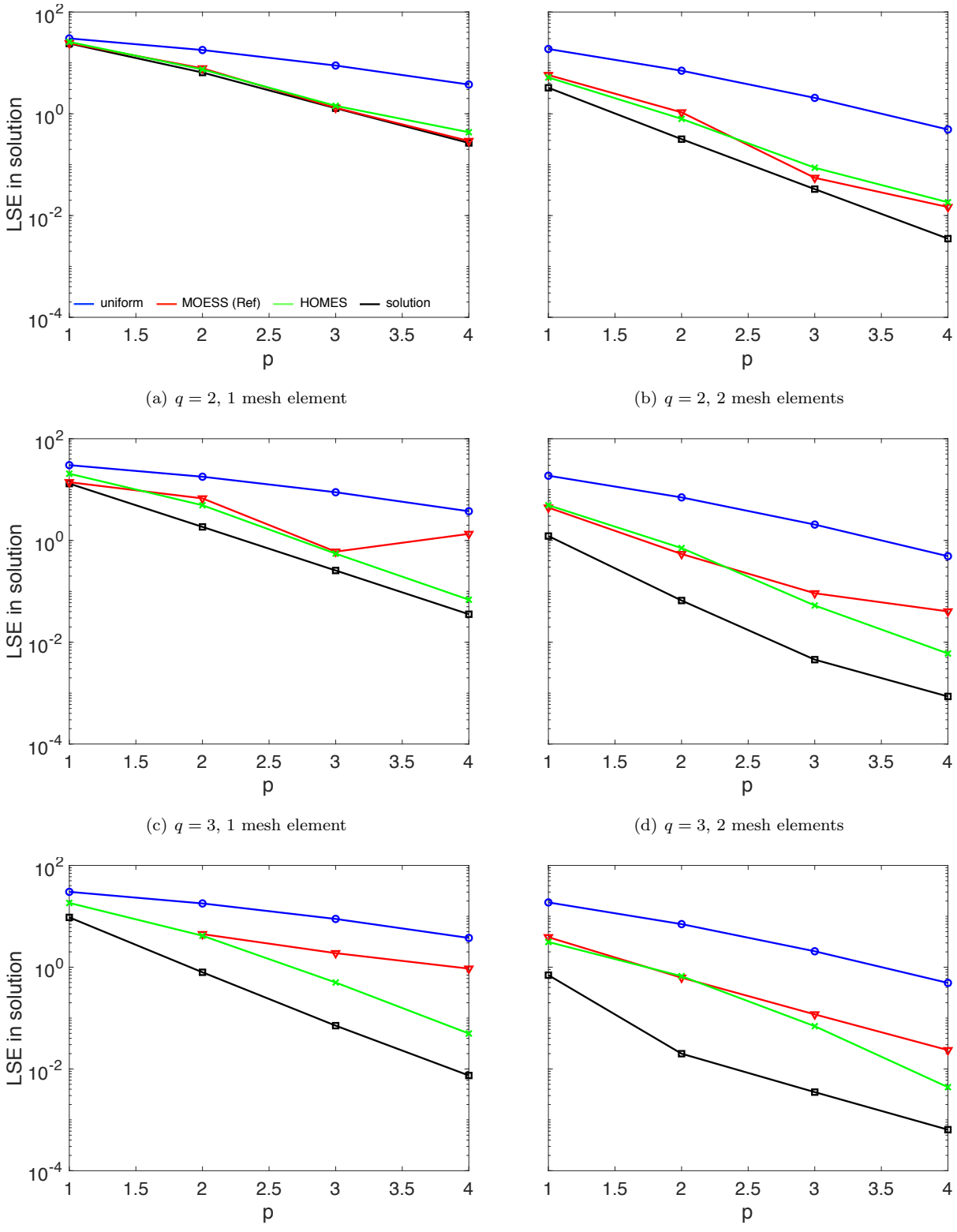


Figure 7. Least-squares error between the approximated and exact solutions ($u^{\text{exact}}_{\text{smooth}}$) with $q = 1$ and 2 mesh elements. The blue line is obtained using a uniform mesh and the black line is obtained by performing global solution-based optimization.

Figure 8 shows the least-squares solution error in approximating $u^{\text{exact}}_{\text{smooth}}$ on a single element and two mesh elements with $q = [2, 3, 4]$. The single-element cases are used to assess the robustness of HOMES and MOESS on a refined mesh. For $q = 2$, we see that MOESS on a refined mesh and HOMES have comparable performance, but as q increases, the performance of MOESS on a refined mesh deteriorates. In fact, for $p = 1$ and $q = 4$ on a single mesh element, MOESS on a refined mesh failed to converge to a valid mesh. This is not surprising since MOESS on a refined mesh produces a crude approximation of the metric field, which might not be sufficiently accurate for higher q. Figure 9 shows sample $p = [2, 3]$ approximations of $u^{\text{exact}}_{\text{smooth}}$ obtained on a single element mesh with $q = [2, 3]$. For $p = 2$, HOMES performs better than MOESS on a refined mesh. However, for $p = 3$, both methods converge to similar meshes.

Figure 10 compares the least-squares error in approximating $u^{\text{exact}}_{\text{rapid changes}}$ on uniform meshes and on meshes generated using MOESS on a refined mesh, HOMES, and global solution-based optimization. For this, we consider $q = [1, 2, 3]$ and seven mesh elements. Although we observe similar error trend as our test cases with $u^{\text{exact}}_{\text{smooth}}$, here the metric-based optimization gives us noticeably less accuracy benefits compared to the solution-based optimization. Furthermore, notice that MOESS on a refined mesh does not guarantee lower error for increasing q. Finally, we find that increasing q does not change the convergence rate, but it does give us more accurate solution approximation.

In summary, while MOESS on a refined mesh is significantly less expensive than HOMES and requires minimal development time, the current formulation of MOESS on a refined mesh does not extend well beyond $q = 2$. For $q = 2$, using MOESS on a refined mesh can be beneficial, especially considering the cost of HOMES. Although more expensive, HOMES gives us the extra robustness and accuracy that MOESS on a refined mesh lacks of. In particular, HOMES continues to offer some accuracy benefits for higher q.



(a) $q = 2$, 1 mesh element

(b) $q = 2$, 2 mesh elements

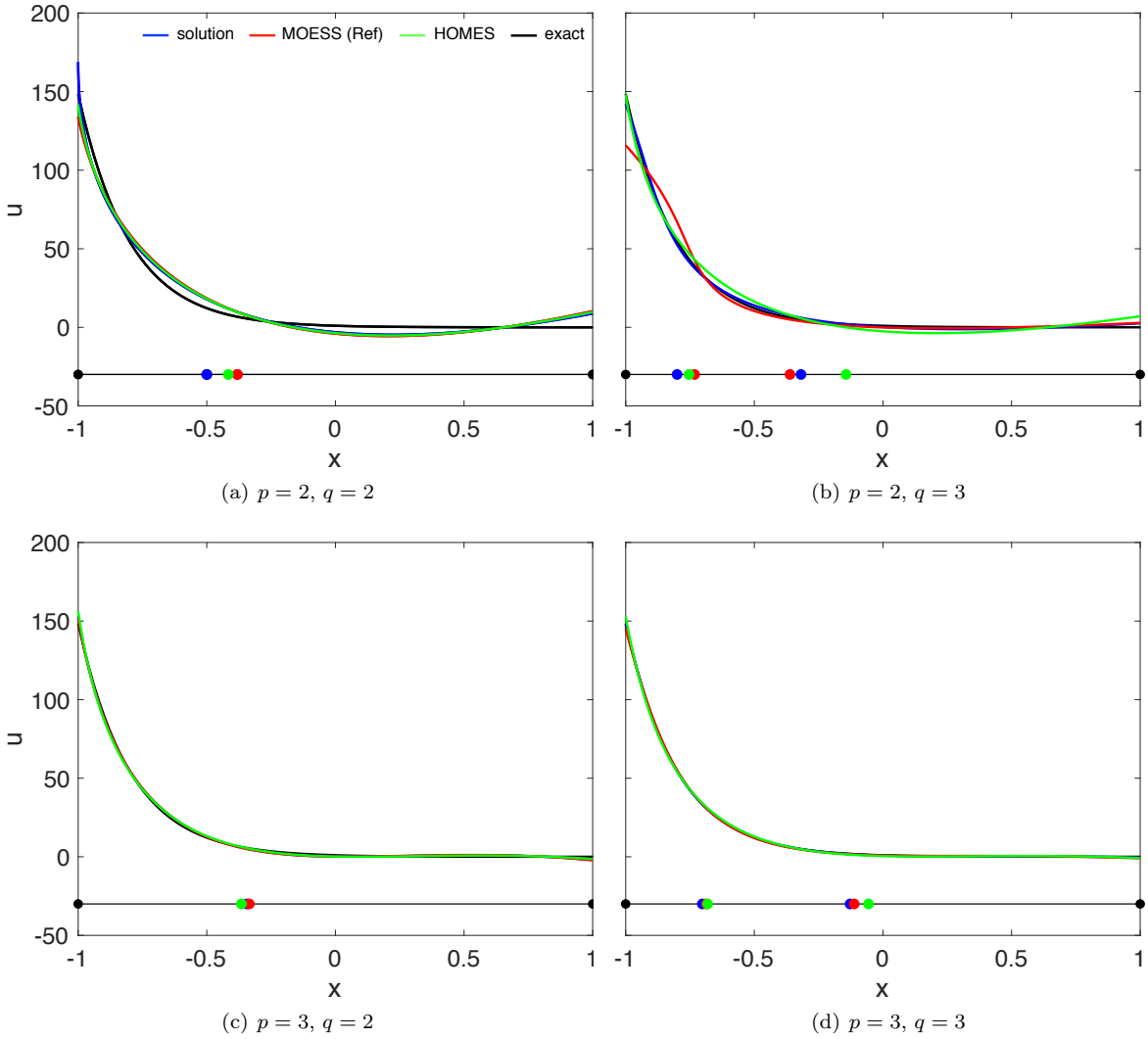
(c) $q = 3$, 1 mesh element

(d) $q = 3$, 2 mesh elements

(e) $q = 4$, 1 mesh element

(f) $q = 4$, 2 mesh elements

Figure 8. Least-squares error between the approximated and exact solutions for $(u^{\text{exact}}_{\text{smooth}})$ with $q = [2, 3, 4]$. The blue line is obtained using a uniform mesh and the black line is obtained by performing global solution-based optimization.



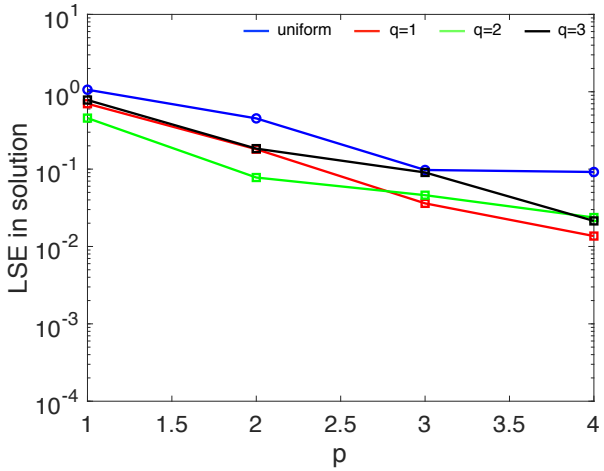
(a) $p = 2$, $q = 2$

(b) $p = 2$, $q = 3$

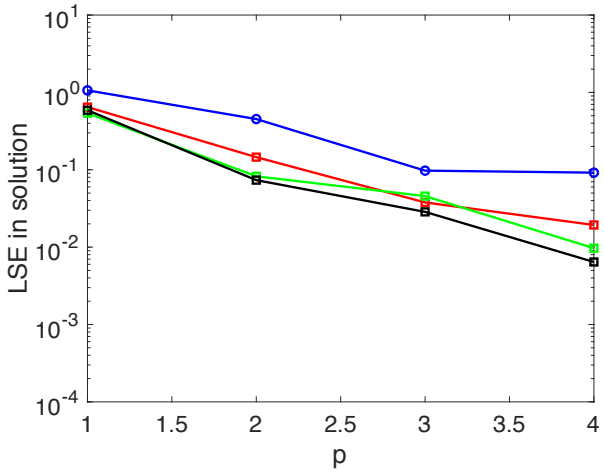
(c) $p = 3$, $q = 2$

(d) $p = 3$, $q = 3$

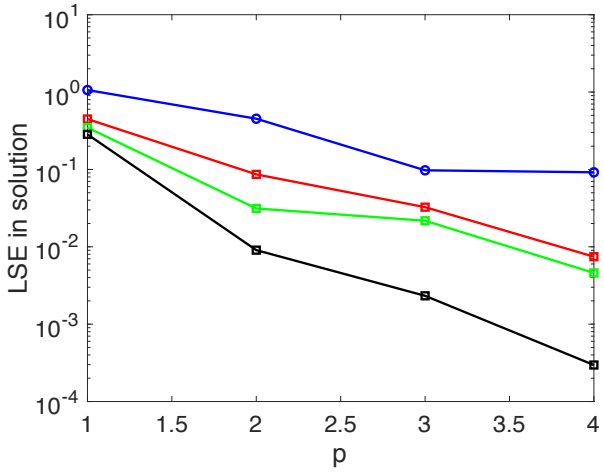
Figure 9. Sample approximated solutions for $(u^{\text{exact}}_{\text{smooth}})$ on a single mesh element with $q = [2, 3]$. The blue line is obtained by performing global solution-based optimization. For ease of reading, the geometry nodes are shown as dots on the horizontal black line near the x-axis. The solution curves for $q = 3$ overlay on top of each other.



(a) MOESS on a refined mesh



(b) HOMES



(c) Global solution-based optimization

Figure 10. Least-squares error between the approximated and exact solutions for $u^{\text{exact}}_{\text{rapid changes}}$ and $q = [1, 2, 3]$ on seven mesh elements.

Nevertheless, the current formulation of HOMES can benefit from more research.

VI.D. The Power of a Single, High-Order Mesh Element

Next, we want to test HOMES performance for higher p and/or q orders. For this, we introduce a "shock"-like feature into the exact solution. Furthermore, in order to demonstrate the approximation power of a high-order mesh element when its high-order geometry nodes are placed optimally, we approximate this function on a single mesh element. The exact solution considered here is

$$u_{\text{shock}}^{\text{exact}} = (1 + \sin(0.8\pi x)) \tanh(15 - 30\pi(x + 0.24)).$$

Figure 11 shows that the approximated solution becomes a better representation of the exact solution when the high-order geometry nodes are placed at their optimal locations. Moreover, we see that the accuracy of the approximated solution improves as we increase p and/or q, as expected. Finally, we want to point out that this observation is analogous to the ability of the warped-element refinement method to better approximate the shock location due to the slight change in the angle of attack.

VI.E. Improving Robustness, Convergence, and Automation of HOMES

So far, we have established the need for a desired, higher-order metric representation and observed the superior performance of HOMES in terms of accuracy benefits. With this, we will now shift our focus solely on HOMES. In particular, we aim to improve its robustness, convergence, and automation.

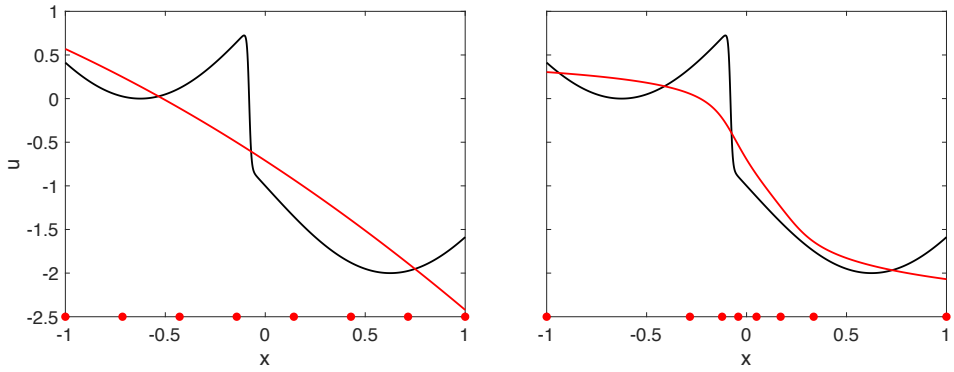
VI.E.1. Metric at Shared Geometry Nodes

Extracting mesh-implied metric information and constructing a high-quality, desired metric field are key steps in the metric-based mesh optimization algorithm. Thus, it is important for HOMES to have the ability to build a continuous metric approximation that is free from noise and undesirable oscillations. To do so, we implement a simple average of the metric at shared nodes. Although having a continuous representation of the metric field eliminates the need for averaging (i.e., such as one that is implemented in MOESS), moving the high-order geometry nodes within an element can cause discontinuities in the reference-to-global mapping Jacobian at the shared nodes, and these discontinuities can greatly affect the quality of the continuous metric representation. Indeed, we find that a least-squares fit (with logarithm barrier) through these discontinuous Jacobian does not work well since it allows for undesirable oscillations in the continuous metric approximation. By averaging the metric at these shared nodes, HOMES can produce higher-quality metric field.

VI.E.2. Choice of Basis Functions and Node Distribution for Geometry and Metric Approximations

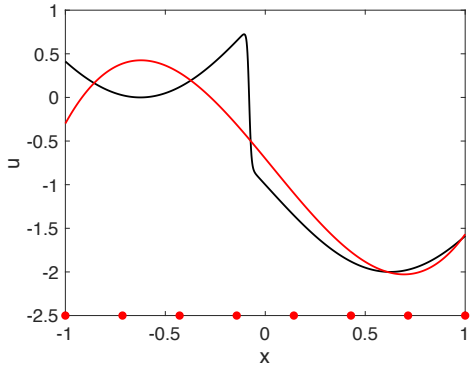
The geometry and metric approximations in the previous sections are computed using Lagrange basis functions with uniform node distribution in reference space. While this combination of basis functions and node distribution shows promise in terms of accuracy, it does not give us the necessary robustness, convergence, and automation for applying HOMES to one-dimensional output predictions on high-order meshes. This finding motivates the study on how the combination of basis functions and node distribution for geometry and metric approximations affects HOMES performance in terms of accuracy benefits, quality of desired metric field, and sensitivity to initial mesh and optimization parameters. For this purpose, we test four possible basis functions and node distribution pairs, involving Lagrange basis functions, Bernstein basis functions, uniform node distribution, and GLL node distribution. This study shows that the use of uniform node distribution causes the lack of robustness and convergence, and that Bernstein basis functions possess desirable properties that are needed for generating higher-quality, desired metric field.

Figure 12 shows that the combination of Bernstein basis functions and GLL node distribution further improve upon the previously observed accuracy benefits. These additional accuracy benefits can also be obtained at lower computational cost. We suspect that the higher-quality, desired metric field contributes greatly in reducing the time spent in the mesh regeneration process and in allowing for the use of larger δs and smaller n_{step}, resulting in lower overall runtime. The change in the choice of basis functions and node distribution also reduces HOMES sensitivity to initial mesh and optimization parameters. With this, we

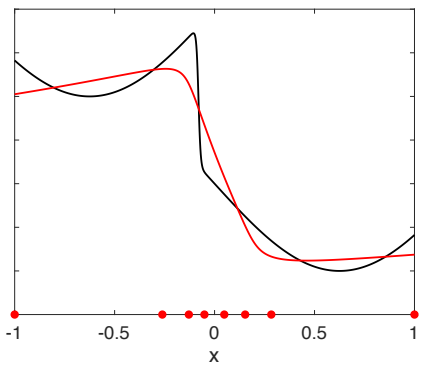


(a) $p = 2$, $q = 7$, initial mesh

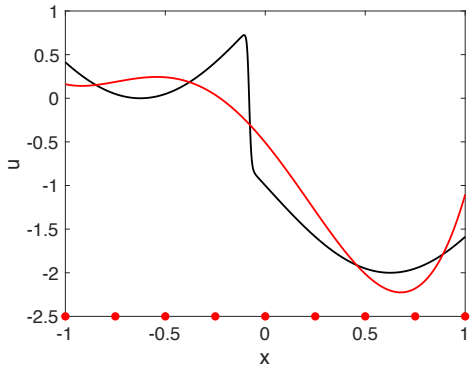
(b) $p = 2$, $q = 7$, optimized mesh



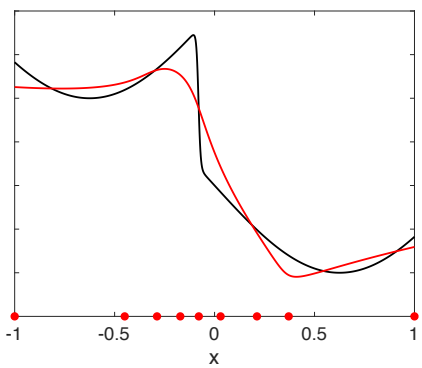
(c) $p = 3$, $q = 7$, initial mesh



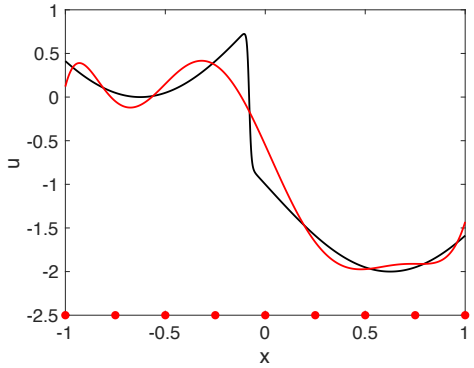
(d) $p = 3$, $q = 7$, optimized mesh



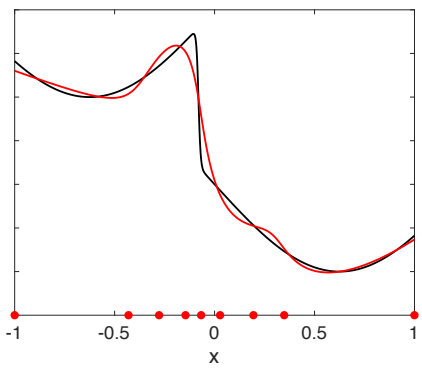
(e) $p = 4$, $q = 8$, initial mesh



(f) $p = 4$, $q = 8$, optimized mesh

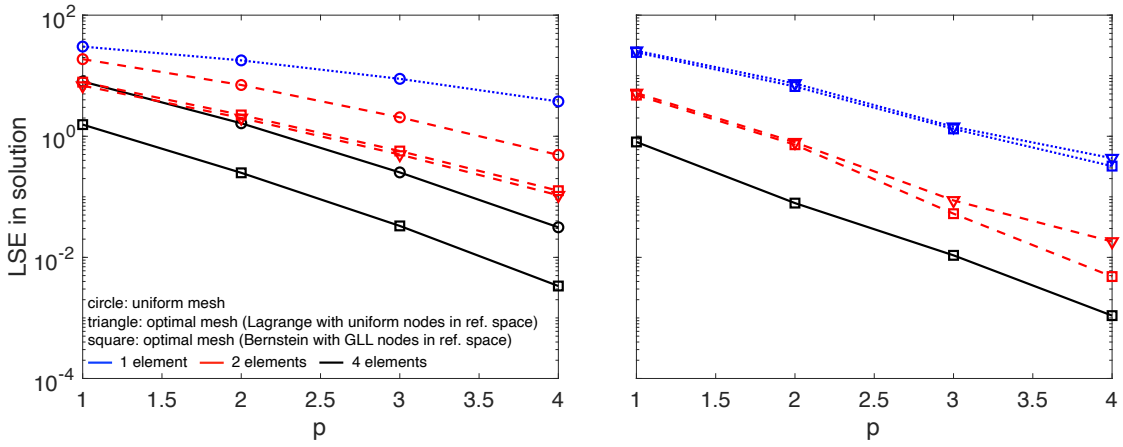


(g) $p = 8$, $q = 8$, initial mesh

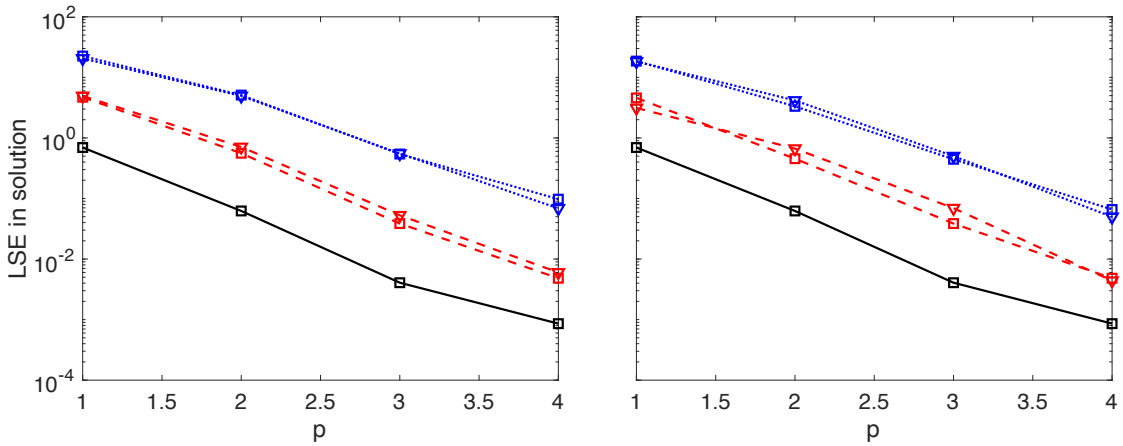


(h) $p = 8$, $q = 8$, optimized mesh

Figure 11. Approximated solution (shown in red) to the "shock"-like function (shown in black) for different p and q combinations. The desired metric field is obtained using HOMES. The geometry nodes are shown as red dots on the x-axis.



(a) $q = 1$ (b) $q = 2$



(c) $q = 3$ (d) $q = 4$

Figure 12. Comparison of least-squares error between the approximated and exact solutions for $(u^{\text{exact}}_{\text{smooth}})$ with $p = [1, 2, 3, 4]$ and $q = [1, 2, 3, 4]$ on different meshes and two different pairs of basis functions and node distribution. The test cases on four mesh elements are strictly performed with the combination of Bernstein basis functions and GLL node distribution in order to further test its benefits.

decided to run the solution approximation test case on a finer mesh (i.e., 4 mesh elements), and we continued to observe this improvement.

Figures 13 to 15 further highlight the benefits of the combination of Bernstein basis functions and GLL node distribution. In particular, these figures visually show higher-quality, desired metric field compared to the ones shown in Figure 6. Notice how the change in the choice of basis functions and node distribution reduces the undesirable oscillations and better preserves the symmetry of the problem. As expected, the improvement in the quality of the desired metric field leads to better accuracy in solution approximation. Lastly, these figures confirm the possibility of using larger δs, smaller n_{step}, and a generalized set of optimization parameters, showing promise for cost reduction and automation. Note that the results in previous sections are obtained with small δs and large n_{step}.

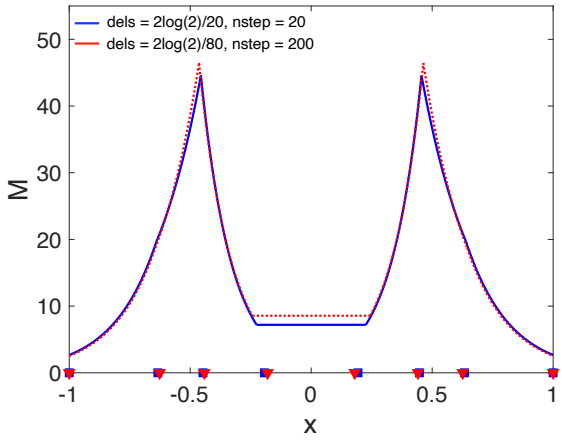


Figure 13. Updated $p = 4$ metric for $u^{\text{exact}}_{\text{rapid changes}}$ on seven, $q = 1$ mesh elements due to the change in the choice of basis functions and node distribution for geometry and metric approximations. Blue curve: $\varepsilon = 1.966 \times 10^{-2}$ and red curve: $\varepsilon = 1.718 \times 10^{-2}$. The red curve is generated using the same settings as in Figure 6(a).

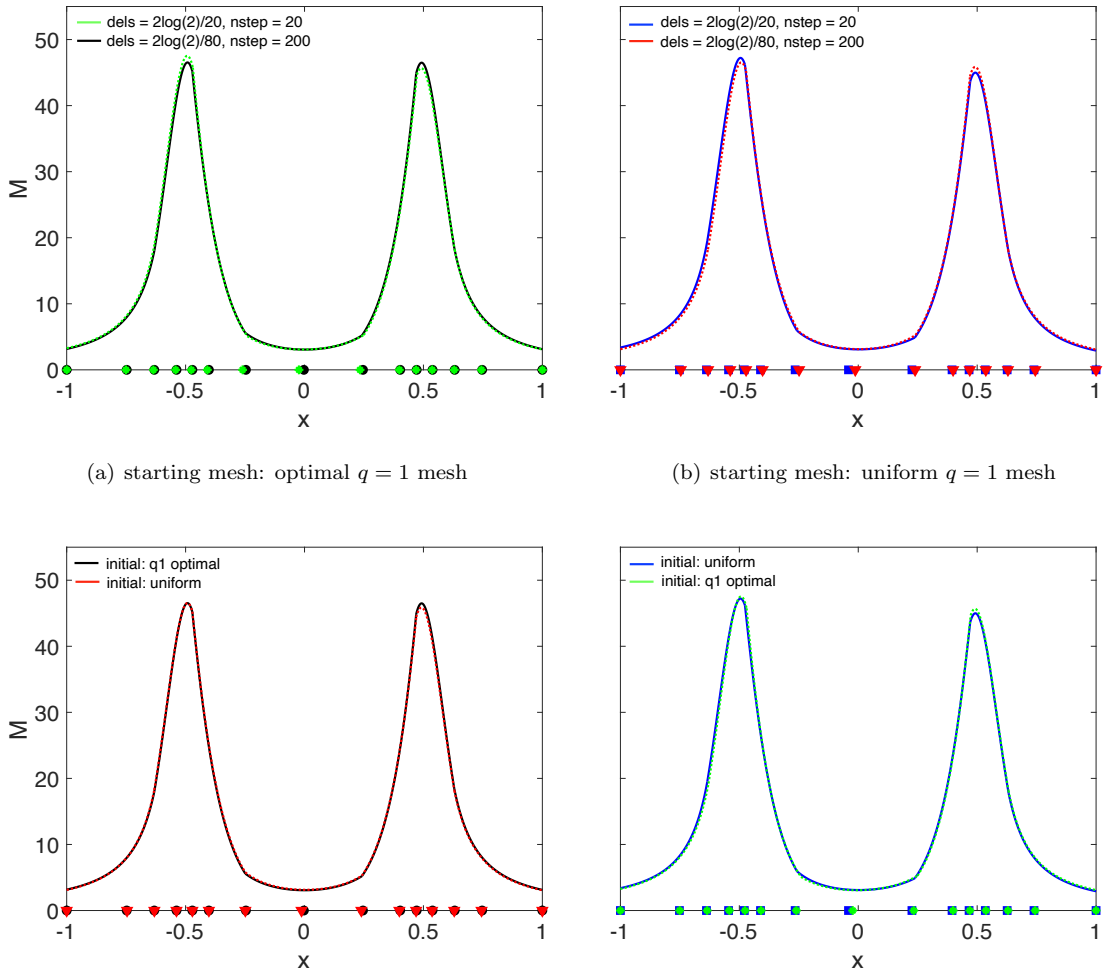
VI.F. Predicting an Output of Interest in One Dimension

For our last one-dimensional test case, we consider a one-dimensional, advection-diffusion equation,

$$a\frac{\partial u}{\partial x} - \nu\frac{\partial^2 u}{\partial x^2} = 0, \quad x \in [0, L], \quad u(0) = 0, u(L) = 1, \tag{17}$$

where $a = 1$, $L = 1$, and the Peclet number is $Pe \equiv aL/\nu = 1$. Our output of interest is the viscous flux on the right boundary, $\mathcal{J} = \nu\frac{\partial u}{\partial x}\Big|_{x=L}$. Three meshes with 8, 16, and 32 elements are optimized using HOMES with Bernstein basis functions and GLL node distribution for geometry and metric approximations. The step metric approximation order is set to one.

Figure 16 shows significant improvement in the accuracy of the output prediction (at least a factor of two) compared to the uniform mesh. In general, we also see the expected error trend in that finer mesh, higher p, and/or higher q result in more accurate output prediction, though this is not always guaranteed. The underlying problem lies in the failure of HOMES to produce a high-quality, desired metric field due to its sensitivity to the noise in the error kernel sampling procedure. Figure 17 shows a sample desired metric field and the mesh-implied metric of the final mesh along with the optimal geometry node distributions. Here, we can see the degradation of the quality of the desired metric field as the mesh gets finer, which greatly affects the accuracy of the output prediction. Nevertheless, our mesh regeneration procedure performs well in creating a mesh that conforms to a desired metric field. Finally, we note that the geometry nodes are shifted towards the right boundary, as expected.



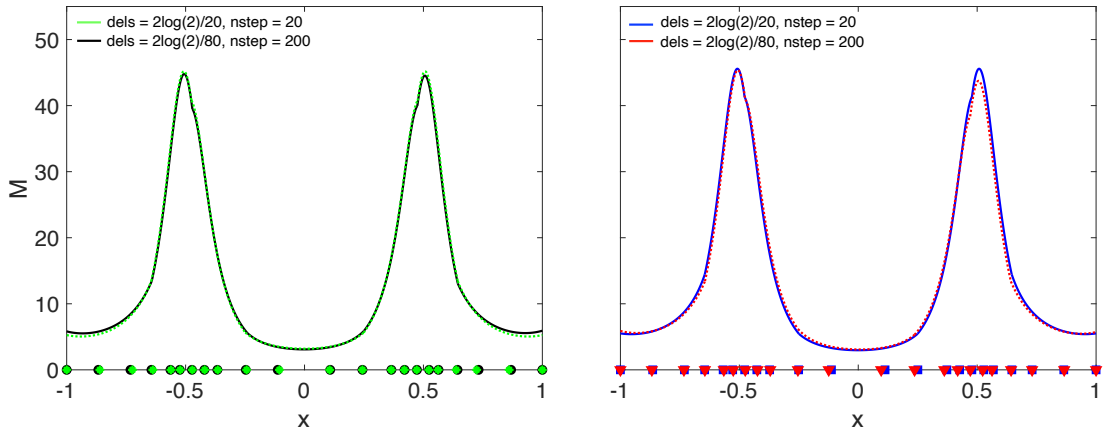
(a) starting mesh: optimal $q = 1$ mesh

(b) starting mesh: uniform $q = 1$ mesh

(c) $\delta s = 2\log(2)/80$, $n_{\text{step}} = 200$

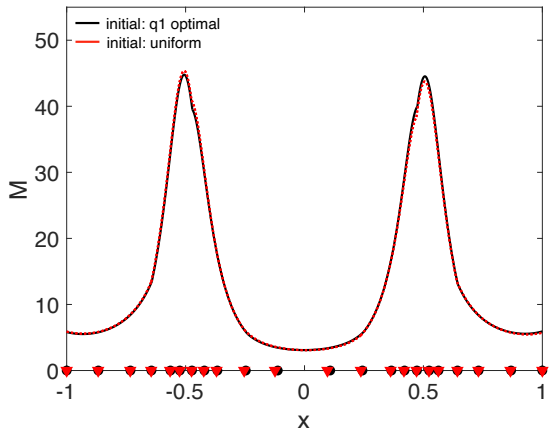
(d) $\delta s = 2\log(2)/20$, $n_{\text{step}} = 20$

Figure 14. Updated $p = 4$ metric for $u^{\text{exact}}_{\text{rapid changes}}$ on seven, $q = 2$ mesh elements due to the change in the choice of basis functions and node distribution for geometry and metric approximations. Green curve: $\varepsilon = 1.047 \times 10^{-2}$, black curve: $\varepsilon = 1.046 \times 10^{-2}$, blue curve: $\varepsilon = 1.067 \times 10^{-2}$, and red curve: $\varepsilon = 1.064 \times 10^{-2}$. The red curve is generated using the same settings as in Figure 6(b).

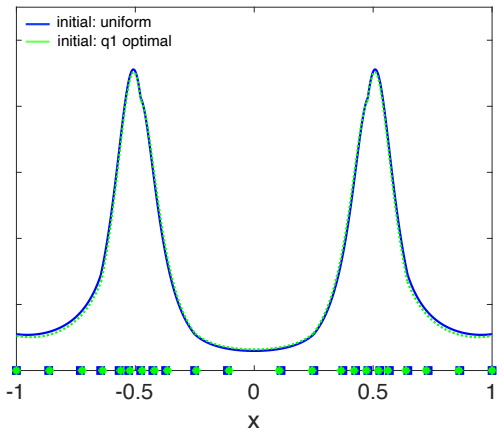


(a) starting mesh: optimal $q = 1$ mesh

(b) starting mesh: uniform $q = 1$ mesh

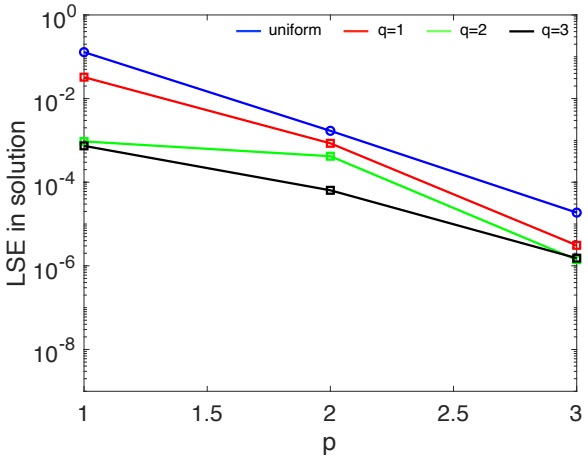


(c) $\delta s = 2\log(2)/80$, $n_{\text{step}} = 200$

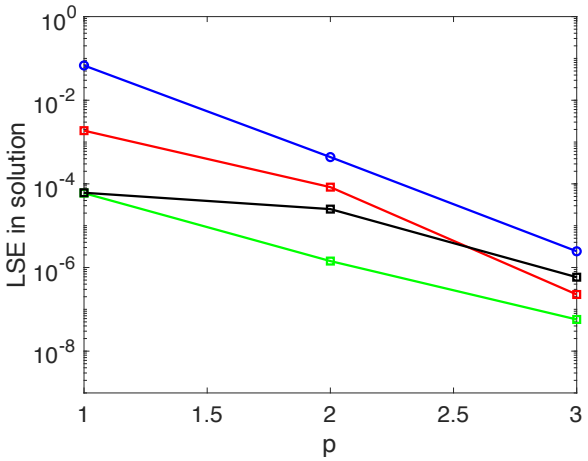


(d) $\delta s = 2\log(2)/20$, $n_{\text{step}} = 20$

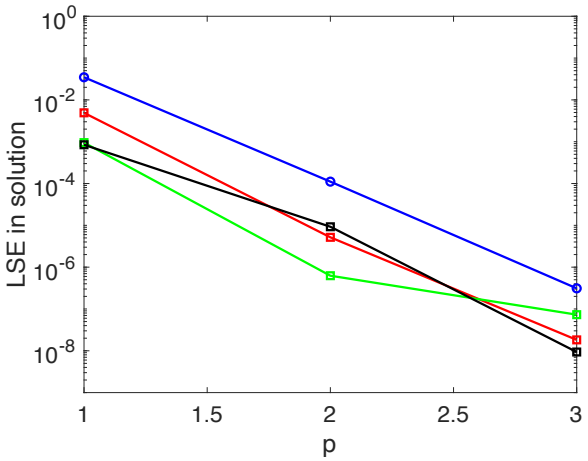
Figure 15. Updated $p = 4$ metric for $u^{\text{exact}}_{\text{rapid changes}}$ on seven, $q = 3$ mesh elements due to the change in the choice of basis functions and node distribution for geometry and metric approximations. Green curve: $\varepsilon = 5.344 \times 10^{-3}$, black curve: $\varepsilon = 4.464 \times 10^{-3}$, blue curve: $\varepsilon = 4.820 \times 10^{-3}$, and red curve: $\varepsilon = 4.536 \times 10^{-3}$. The red curve is generated using the same settings as in Figure 6(c).



(a) 8 mesh elements

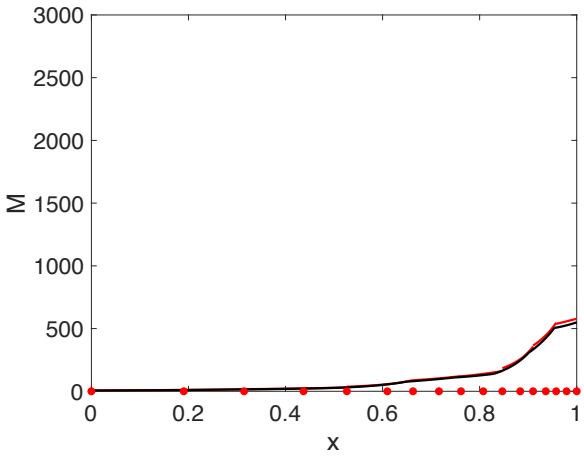


(b) 16 mesh elements

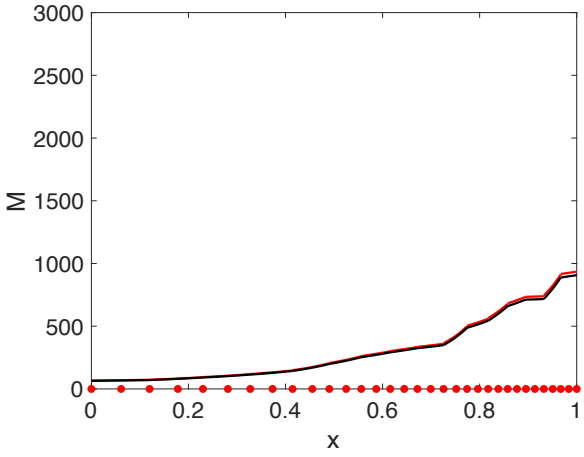


(c) 32 mesh elements

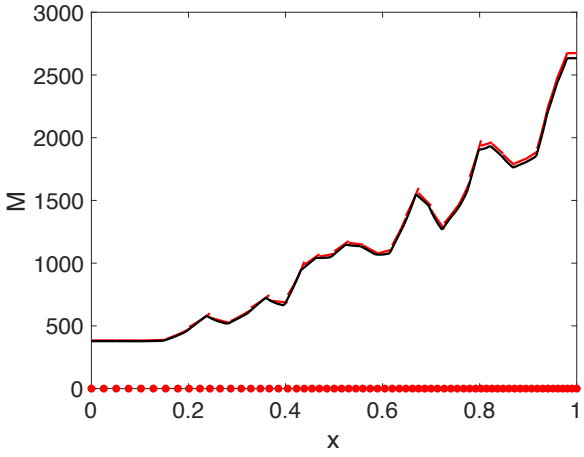
Figure 16. Output error in predicting the viscous flux on the right boundary on $q = [1, 2, 3]$ meshes. The current version of HOMES has potential in improving the accuracy of output prediction, but further development is necessary to continue observing the accuracy benefits for finer meshes, higher p, and/or higher q.



(a) 8 mesh elements



(b) 16 mesh elements



(c) 32 mesh elements

Figure 17. $p = 3$ **approximated metric (shown in red) and the corresponding desired metric (shown in black) on** $q = 2$ **meshes. The geometry nodes are shown as red dots on the** x**-axis. The noise in the desired metric is responsible for lower accuracy benefits observed on finer meshes, higher** p**, and/or higher** q**.**

VI.G. Two-Dimensional Proof of Concept

Our focus here is on quantifying the benefits of $q = 2$, metric conforming meshes for solution approximation in two dimensions, establishing the importance of extending one-dimensional HOMES to higher dimensions, and demonstrating the feasibility of implementing our mesh regeneration procedure in two dimensions. As in one dimension, the $q = 2$ geometry nodes will be placed such that the mesh-implied metric conforms to a desired Riemannian metric field, and the $q = 2$, metric-conforming mesh will be generated by staggering linear, metric-conforming, mesh generation (i.e., via Bi-dimensional Anisotropic Mesh Generator (BAMG)[3,30,31]) with a post-process, metric-based optimization for placing the $q = 2$ geometry nodes. However, unlike in one dimension, the desired metric field here will be specified analytically.

VI.G.1. Obtaining the Desired Metric Field

Extending the one-dimensional HOMES to two dimensions is not trivial. For this reason, we opt to use the analytical metric field derived by Loseille and Alauzet.[32] Of particular interest is the continuous metric field given by

$$\mathcal{M}_{1,\alpha} = F(x,y) \begin{bmatrix} \alpha^2 h_1^{-2}(x,y) & 0 \\ 0 & \alpha h_2^{-2}(x,y) \end{bmatrix} F(x,y)^{\mathrm{T}}, \tag{18}$$

where

$$F(x,y) = \frac{1}{\sqrt{x^2+y^2}} \begin{bmatrix} x & -y \\ y & x \end{bmatrix}, \quad h_1^{-2}(x,y) = 4(x^2+y^2), \quad h_2^{-2}(x,y) = \frac{1}{2\sqrt{x^2+y^2}}.$$

The parameter α in the equation above is user-specified and controls the coarseness and anisotropy of the mesh.

VI.G.2. Optimal Placement of the $q = 2$ Geometry Nodes

The placement of $q = 2$ nodes within an element is obtained by solving a metric-based optimization problem. Since a $q = 2$ element has exactly one $q = 2$ node on each edge, the metric-based optimization problem can be reduced to one dimension. The following unconstrained optimization problem is solved independently on each internal edge ($\partial\Omega_e \backslash \partial\Omega$):

$$\begin{aligned} &\text{minimize} \int_{\partial\Omega_e \backslash \partial\Omega} \sqrt{d\vec{x}_{\text{edge}}^{\mathrm{T}} \mathcal{M} d\vec{x}_{\text{edge}}}, \\ &\text{w.r.t.} \quad \boldsymbol{\delta}, \end{aligned} \tag{19}$$

where the locations of the geometry nodes on the edge can be approximated using one-dimensional basis functions:

$$\vec{x}_{\text{edge}}(\sigma) = \sum_{i=1}^{N_q} \vec{x}_i \phi_i^{\text{1D}}(\sigma),$$

$\sigma \in [0,1]$ is the reference node coordinates along the edge, $\mathcal{M}$ is the analytical metric field, and $\boldsymbol{\delta}$ is the location of the $q = 2$ nodes on that edge. This optimization problem is solved using Matlab's `fmincon` with the sequential quadratic programming (SQP) algorithm. Once all internal edges are optimized, we check for the validity of the mesh elements by ensuring that all Jacobian determinants at interrogation points are non negative. The interrogation points are defined in the reference space, which consist of a large number of equally-spaced points within the element. In the case where $q = 2$ node placement fails to maintain the validity of the element, the node displacements are adjusted by a certain factor, η. For our test cases, we find that η retains about 80-90% of the optimal displacements of the $q = 2$ nodes.

The initial input meshes to the metric-based optimization problem shown in Equation 19 are linear, metric-conforming meshes generated via BAMG. Here, we consider a square domain $\Omega = [-1,1]^2$ with $\alpha = [5, 10, 20]$. The first mesh (mesh A) does not split all internal edges with two boundary vertices in two, while the second (mesh B) does; this can be done using `-splitpbedge` option in BAMG.[30] We also want to note that the analytical metric defined in Equation 18 is undefined at $(x,y) = (0,0)$. Although

this might affect the node placement around the center of the square domain, a valid mesh can still be generated. Figures 18 and 19 respectively show meshes A and B. In comparison to the linear, metric-conforming meshes, we can visually see that the high-order, metric-conforming meshes have smoother circular patterns, as dictated by the analytical metric field. Next, we will see that better conformity to the metric field corresponds to more accurate solution approximation for functions with circular contour.

VI.G.3. Numerical Benefits

We quantify the approximation power of high-order, metric-conforming meshes by measuring the least-squares error in the solution approximation. The exact functions are shown in Figure 20. The parabolic and the parabolic hyperbolic tangent functions are chosen based on the fact that the analytical metric field places the mesh elements to form circular patterns.

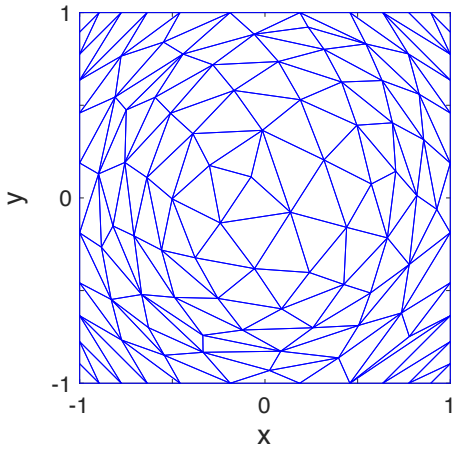
We consider the $p = 1$ solution approximation of the parabolic and parabolic hyperbolic tangent functions. Figure 21 shows the least-squares error in the solution approximation of these functions. Here, we can immediately see that the solution errors are consitently lower on the high-order ($q = 2$) meshes than on the linear ($q = 1$) meshes. In fact, the convergence rate is slightly higher on the $q = 2$ meshes. Also, notice that meshes A and B have similar solution approximation power for these two solutions. The solution error is reduced by a factor of 2 for $\alpha = 5$, by a factor of 3 for $\alpha = 10$, and by a factor of 4 for $\alpha = 20$. Furthermore, the accuracy in the solution approximation improves as the mesh gets finer (i.e., larger α), as expected. These observations are indeed consistent with our one-dimensional results. Sample approximated parabolic solutions on meshes A and B with $\alpha = 5$ and $\alpha = 20$ are shown in Figures 22 and 23, respectively. As one can expect, refining the mesh results in a higher quality of solution approximation. But more importantly, these figures show that approximating the solution on high-order, metric-conforming meshes significantly improves the quality of the approximated solution. That is, the discontinuity in the solution approximation is less pronounced on the higher-order meshes and/or on finer meshes.

VII. Conclusions and Future Work

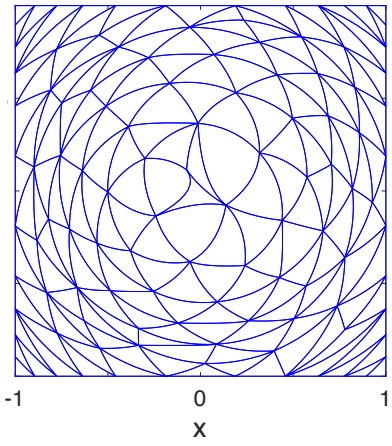
In this paper, we present two mesh optimization algorithms, called MOESS on a refined mesh and high-order MOESS (HOMES), for generating optimal high-order, metric-conforming meshes to better improve the accuracy of high-order, finite-element methods. The success of these metric-based optimization algorithms greatly relies on their ability to extract the mesh-implied metric information and construct a desired metric field. Our one-dimensional results highlight the superior potential of HOMES in this aspect, which is reflected in the larger error reduction in solution approximation and output prediction. Some suggestions to improve the overall robustness, convergence, and automation of HOMES are also provided in details. Furthermore, we demonstrate the benefits of extending the one-dimensional HOMES to two dimensions. Finally, an affordable metric-based mesh regeneration procedure to place the vertices and high-order geometry nodes according to a desired metric field is formulated. The feasibility of this mesh regeneration procedure is also demonstrated in two dimensions. Future work includes finalizing the HOMES algorithm, reducing the cost, and extending the one-dimensional HOMES to two dimensions. Our long-term goal is to include HOMES as part of high-order mesh generation process for use with very high-order, finite-element methods (e.g., up to 16^{th} order).

Acknowledgments

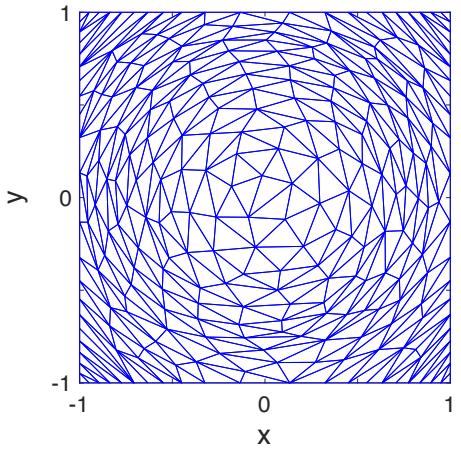
The authors acknowledge the financial support of the University of Tennessee, François–Xavier Bagnoud Fellowship, and the NASA Entry Systems Modeling project at the NASA Ames Research Center.



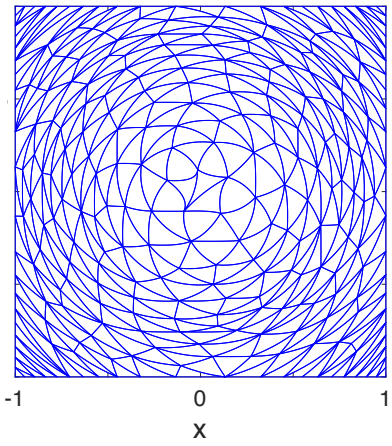
(a) $q = 1$, $\alpha = 5$, 176 mesh elements



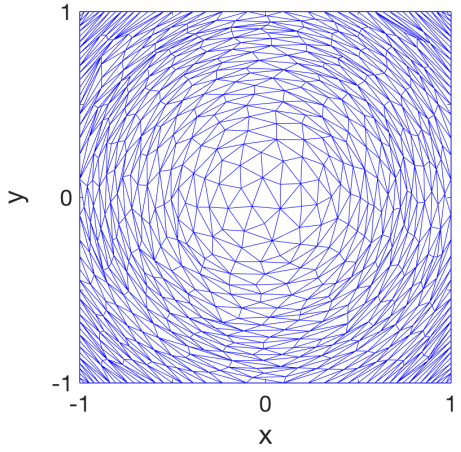
(b) $q = 2$, $\alpha = 5$, 176 mesh elements



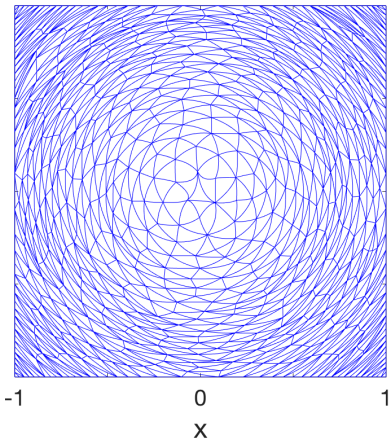
(c) $q = 1$, $\alpha = 10$, 465 mesh elements



(d) $q = 2$, $\alpha = 10$, 465 mesh elements

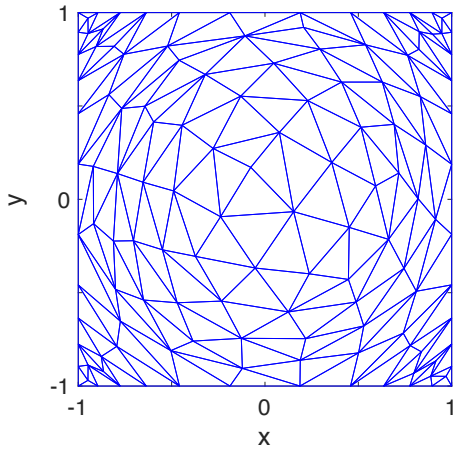


(e) $q = 1$, $\alpha = 20$, 1252 mesh elements

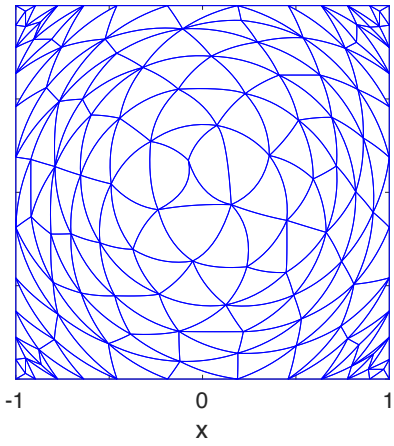


(f) $q = 2$, $\alpha = 20$, 1252 mesh elements

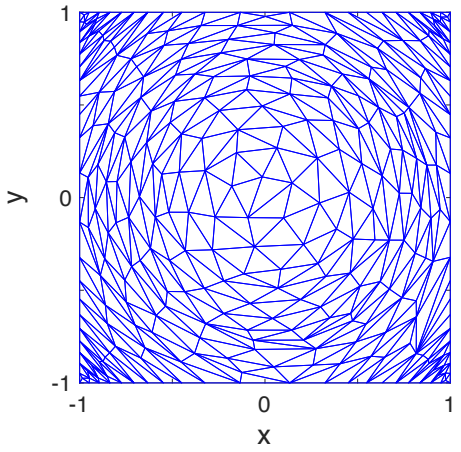
Figure 18. Meshes conforming to the metric field $\mathcal{M}_{1,\alpha}$ with `-splitpbedge` option turned off. We refer to these meshes as meshes of type A.



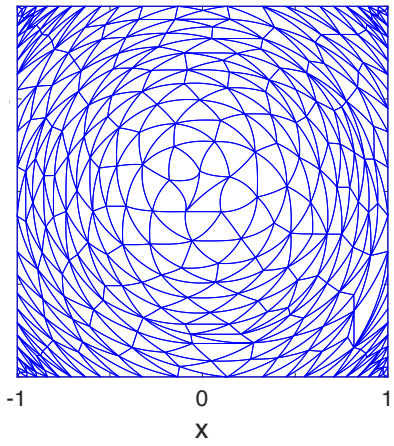
(a) $q = 1$, $\alpha = 5$, 214 mesh elements



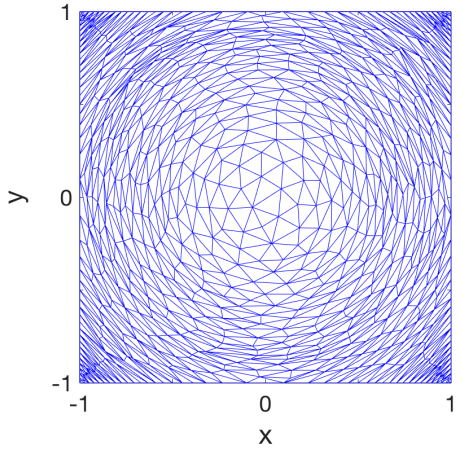
(b) $q = 2$, $\alpha = 5$, 214 mesh elements



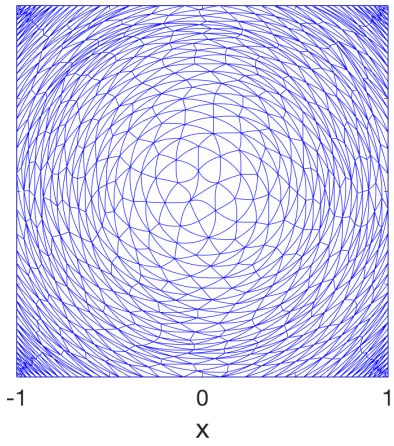
(c) $q = 1$, $\alpha = 10$, 536 mesh elements



(d) $q = 2$, $\alpha = 10$, 536 mesh elements

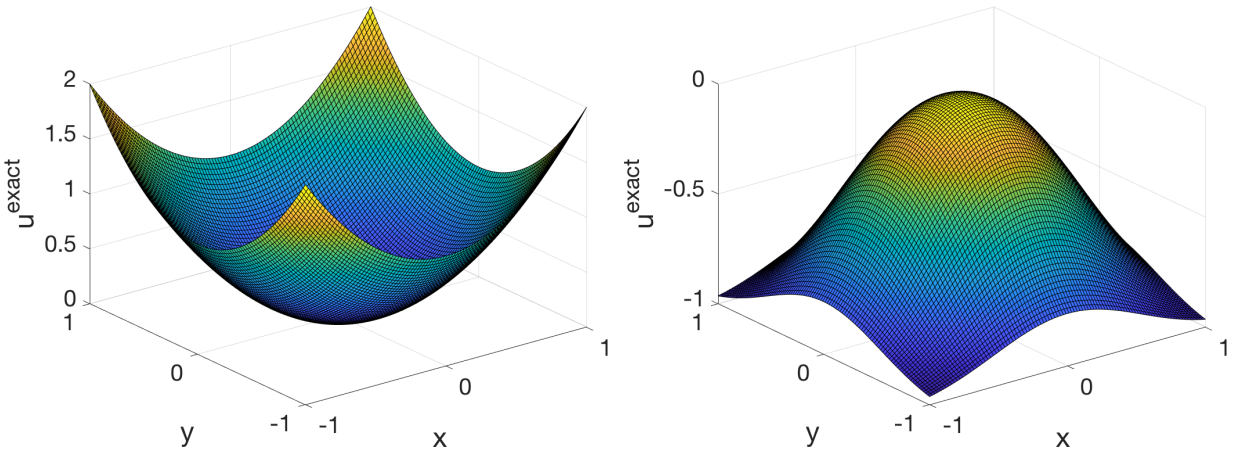


(e) $q = 1$, $\alpha = 20$, 1390 mesh elements



(f) $q = 2$, $\alpha = 20$, 1390 mesh elements

Figure 19. Meshes conforming to the metric field $\mathcal{M}_{1,\alpha}$ with `-splitpbedge` option turned on. We refer to these meshes as meshes of type B.



(a) parabolic: $u(x,y) = x^2 + y^2$ (b) parabolic tanh: $u(x,y) = \tanh(-x^2 - y^2)$

Figure 20. Exact solutions approximated on meshes conforming to the metric field $\mathcal{M}_{1,\alpha}$.

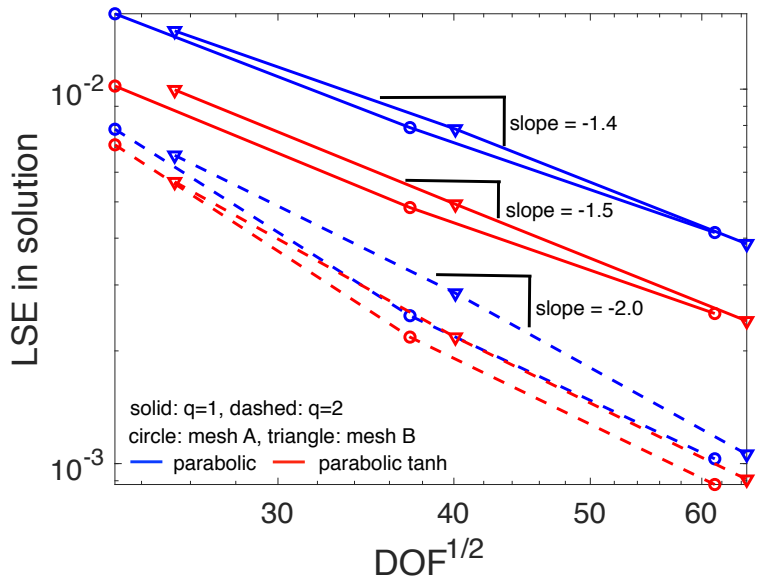
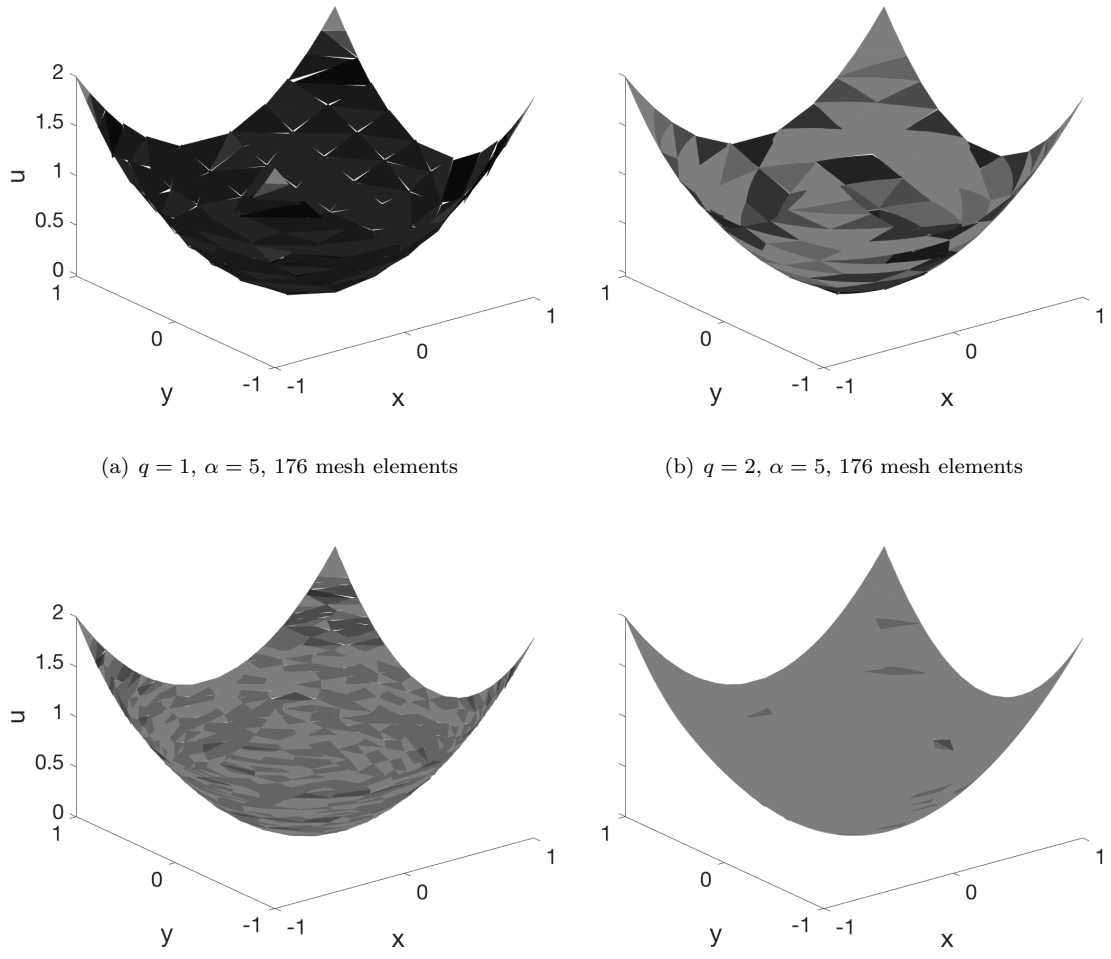


Figure 21. Least-squares error in the $p = 1$ solution approximation of the parabolic and parabolic hyperbolic tangent functions on meshes conforming to the metric field $\mathcal{M}_{1,\alpha}$, where $\alpha = [5, 10, 20]$. Lower solution errors are obtained on the $q = 2$, metric-conforming meshes.



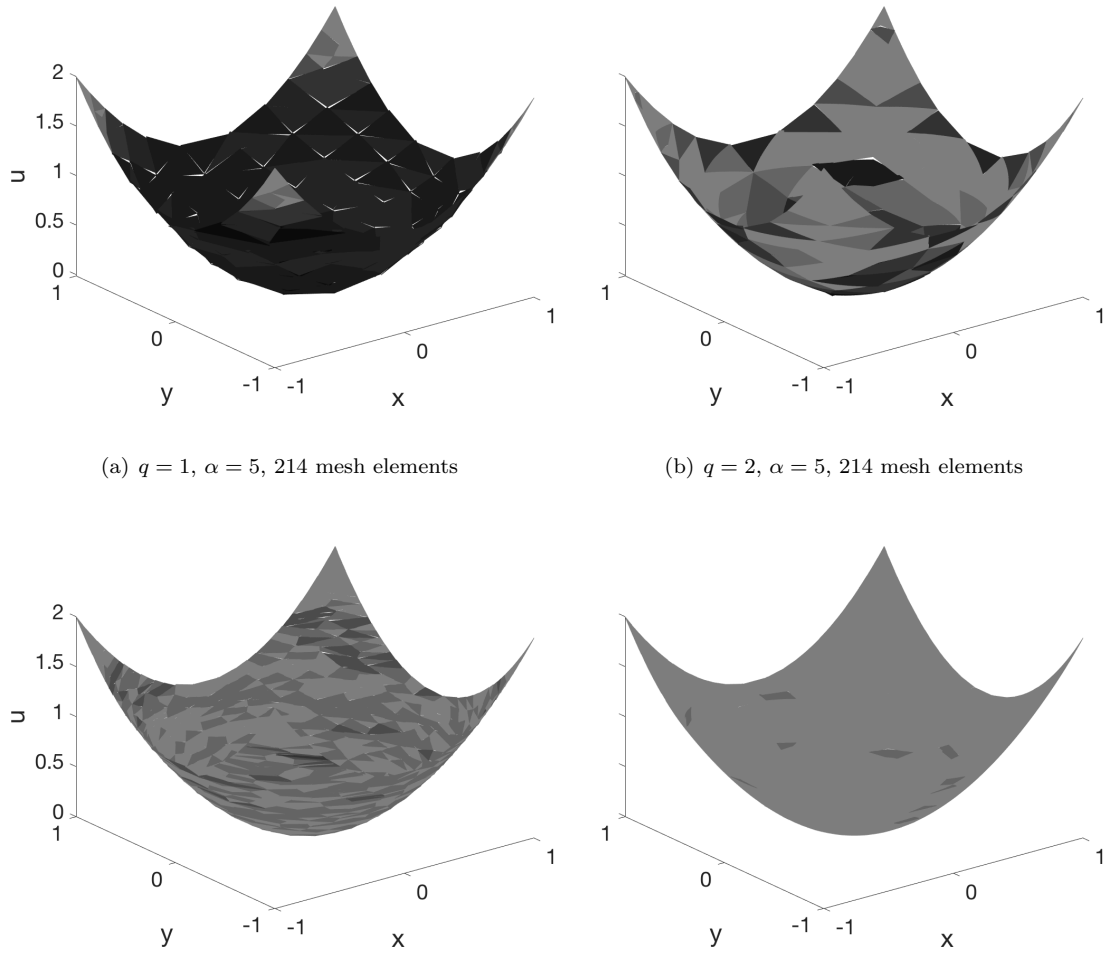
(a) $q = 1$, $\alpha = 5$, 176 mesh elements

(b) $q = 2$, $\alpha = 5$, 176 mesh elements

(c) $q = 1$, $\alpha = 20$, 1252 mesh elements

(d) $q = 2$, $\alpha = 20$, 1252 mesh elements

Figure 22. Sample $p = 1$ approximated solution of the parabolic function on meshes of type A conforming to the metric field $\mathcal{M}_{1,\alpha}$. The quality of solution approximation improves as we increase the q order and/or refine the mesh. Darker shades of gray correspond to higher solution error. The discontinuity in the solution approximation appears in white.



(a) $q = 1$, $\alpha = 5$, 214 mesh elements

(b) $q = 2$, $\alpha = 5$, 214 mesh elements

(c) $q = 1$, $\alpha = 20$, 1390 mesh elements

(d) $q = 2$, $\alpha = 20$, 1390 mesh elements

Figure 23. Sample $p = 1$ approximated solution of the parabolic function on meshes of type B conforming to the metric field $\mathcal{M}_{1,\alpha}$. The quality of solution approximation improves as we increase the q order and/or refine the mesh. Darker shades of gray correspond to higher solution error. The discontinuity in the solution approximation appears in white.

References

[1] Yano, M., *An Optimization Framework for Adaptive Higher-Order Discretizations of Partial Differential Equations on Anisotropic Simplex Meshes*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 2012.

[2] Slotnick, J., Khodadoust, A., Alonso, J., Darmofal, D., Gropp, W., Lurie, E., and Mavriplis, D., "CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences," NASA CR-2014–218178, 2014.

[3] Borouchaki, H., George, P. L., Hecht, F., Laug, P., and Saltel, E., "Delauny Mesh Generation Governed by Metric Specifications. Part I. Algorithms." *Finite Elements in Analysis and Design*, Vol. 25, No. 1–2, 1997, pp. 61–83, doi:10.1016/S0168-874X(96)00057-1.

[4] Pennec, X., Fillard, P., and Ayache, N., "A Riemannian Framework for Tensor Computing," *International Journal of Computer Vision*, Vol. 66, No. 1, 2006, pp. 41–66, doi:10.1007/s11263-005-3222-z.

[5] Castro-Díaz, M. J., Hecht, F., Mohammadi, B., and Pironneau, O., "Anisotropic Unstructured Mesh Adaptation for Flow Simulations," *International Journal for Numerical Methods in Fluids*, Vol. 25, No. 4, 1997, pp. 475–491, doi:10.1002/(SICI)1097-0363(19970830)25:4<475::AID-FLD575>3.0.CO;2-6.

[6] Baker, T. J., "Mesh Adaptation Strategies for Problems in Fluid Dynamics," *Finite Elements in Analysis and Design*, Vol. 25, No. 3–4, 1997, pp. 243–273, doi:10.1016/S0168-874X(96)00032-7.

[7] Buscaglia, G. C. and Dari, E. A., "Anisotropic Mesh Optimization and its Application in Adaptivity," *International Journal for Numerical Methods in Engineering*, Vol. 40, No. 22, 1997, pp. 4119–4136, doi:10.1002/(SICI)1097-0207(19971130)40:22<4119::AID-NME254>3.0.CO;2-R.

[8] Venditti, D. A. and Darmofal, D. L., "Anisotropic Grid Adaptation for Functional Outputs: Application to Two-Dimensional Viscous Flows," *Journal of Computational Physics*, Vol. 187, No. 1, 2003, pp. 22–46, doi:10.1016/S0021-9991(03)00074-3.

[9] Park, M. A., "Three-Dimensional Turbulent RANS Adjoint-Based Error Correction," AIAA Paper 2003–3849, 2003, doi:10.2514/6.2003-3849.

[10] Fidkowski, K. J. and Darmofal, D. L., "A Triangular Cut-Cell Adaptive Method for High-Order Discretizations of the Compressible Navier-Stokes Equations," *Journal for Computational Physics*, Vol. 225, No. 2, 2007, pp. 1653–1672, doi:10.1016/j.jcp.2007.02.007.

[11] Yano, M., Modisette, J. M., and Darmofal, D. L., "The Importance of Mesh Adaptation for Higher-Order Discretizations of Aerodynamics Flows," AIAA Paper 2011–3852, 2011, doi:10.2514/6.2011-3852.

[12] Fidkowski, K. J., "A Local Sampling Approach to Anisotropic Metric-Based Mesh Optimization," AIAA Paper 2016–0835, 2016, doi:10.2514/6.2016-0835.

[13] Houston, P., Senior, B., and Süli, E., "*hp*-Discontinuous Galerkin Finite Element Methods for Hyperbolic Problems: Error Analysis and Adaptivity," *International Journal for Numerical Methods in Fluids*, Vol. 40, No. 1–2, 2002, pp. 153–169, doi:10.1002/fld.271.

[14] Fidkowski, K. J. and Darmofal, D. L., "Review of Output-Based Error Estimation and Mesh Adaptation in Computational Fluid Dynamics," *AIAA Journal*, Vol. 49, No. 4, 2011, pp. 673–694, doi:10.2514/1.J050073.

[15] Wang, L. and Mavriplis, D. J., "Adjoint-Based *h–p* Adaptive Discontinuous Galerkin Methods for the 2D Compressible Euler Equations," *Journal of Computational Physics*, Vol. 228, No. 20, 2009, pp. 7643–7661, doi:10.1016/j.jcp.2009.07.012.

[16] Demkowicz, L. and Gopalakrishnan, J., "A Class of Discontinuous Petrov-Galerkin Methods. Part I: The Transport Equation," *Computer Methods in Applied Mechanics and Engineering*, Vol. 199, No. 23-24, 2010, pp. 1558–1572, doi:10.1016/j.cma.2010.01.003.

[17] Demkowicz, L. and Gopalakrishnan, J., "A Class of Discontinuous Petrov-Galerkin Methods. Part II: Optimal Test Functions," *Numerical Methods for Partial Differential Equations*, Vol. 27, No. 1, 2011, pp. 70–105, doi:10.1002/num.20640.

[18] Demkowicz, L., Gopalakrishnan, J., and Niemi, A., "A Class of Discontinuous Petrov-Galerkin Methods. Part III: Adaptivity," *Applied Numerical Mathematics*, Vol. 62, No. 4, 2012, pp. 396–427, doi:10.1016/j.apnum.2011.09.002.

[19] Kast, S. M., Dahm, J. P. S., and Fidkowski, K. J., "Optimal Test Functions for Boundary Accuracy in Discontinuous Finite Element Methods," *Journal of Computational Physics*, Vol. 298, No. 1, 2015, pp. 360–386, doi:10.1016/j.jcp.2015.05.048.

[20] Melenk, J. and Babuška, I., "The Partition of Unity Finite Element Method: Basic Theory and Applications," *Computer Methods in Applied Mechanics and Engineering*, Vol. 139, No. 1–4, 1996, pp. 289–314, doi:10.1016/S0045-7825(96)01087-0.

[21] Dolbow, J. and Belytschko, T., "A Finite Element Method for Crack Growth without Remeshing," *International Journal for Numerical Methods in Engineering*, Vol. 46, No. 1, 1999, pp. 131–150, doi:10.1002/(SICI)1097-0207(19990910)46:1<131::AID-NME726>3.0.CO;2-J.

[22] Benson, D. J., Bazilevs, Y., E. De Luycker, E., M.-C.Hsu, Scott, M., Hughes, T. J. R., and Belytschko, T., "A Generalized Finite Element Formulation for Arbitrary Basis Functions: From Isogeometric Analysis to XFEM," *International Journal for Numerical Methods in Engineering*, Vol. 83, No. 6, 2010, pp. 765–785, doi:10.1002/nme.2864.

[23] Farhat, C., Harari, I., and Franca, L. P., "The Discontinuous Enrichment Method," *Computer Methods in Applied Mechanics and Engineering*, Vol. 190, No. 48, 2001, pp. 6455–6479, doi:10.1016/S0045-7825(01)00232-8.

[24] Sanjaya, D. P. and Fidkowski, K. J., "Improving High-Order Finite Element Approximation Through Geometrical Warping," *AIAA Journal*, Vol. 54, No. 12, 2016, pp. 3994–4010, doi:10.2514/1.J055071.

[25] Sanjaya, D. P., *Towards Automated, Metric-Conforming, Mesh Optimization for High-Order, Finite-Element Methods*, Ph.D. thesis, University of Michigan: Ann Arbor, 2019.

[26] Reed, W. and Hill, T., "Triangular Mesh Methods for the Neutron Transport Equation," Los Alamos Scientific Laboratory Technical Report LA-UR-73-479, 1973.

[27] Cockburn, B., Karniadakis, G. E., and Shu, C.-W., "The Development of Discontinuous Galerkin Methods," *Discontinuous Galerkin Methods: Theory, Computation, and Applications*, edited by B. Cockburn, G. E. Karniadakis, and C.-W. Shu, Springer, Berlin, Heidelberg, 2000, pp. 3–50, doi:10.1007/978-3-642-59721-3_1.

[28] Fidkowski, K. J., Oliver, T. A., Lu, J., and Darmofal, D. L., "p-Multigrid Solution of High-Order Discontinuous Galerkin Discretizations of the Compressible Navier-Stokes Equations," *Journal of Computational Physics*, Vol. 207, No. 1, 2005, pp. 92–113, doi:10.1016/j.jcp.2005.01.005.

[29] Sanjaya, D. P., Fidkowski, K. J., Diosady, L. T., and Murman, S. M., "Error Minimization via Metric-Based Curved-Mesh Adaptation," AIAA Paper 2017–3099, 2017, doi:10.2514/6.2017-3099.

[30] Hecht, F., "BAMG: bidimensional Anisotropic Mesh Generator," User Guide, 2006, ftp://ftp.sleepgate.ru/pub/FreeBSD/ports/distfiles/bamg.pdf.

[31] Hecht, F., "New Development in Freefem++," *Journal of Numerical Mathematics*, Vol. 20, No. 3–4, 2012, pp. 251–266, doi:10.1515/jnum-2012-0013.

[32] Loseille, A. and Alauzet, F., "Continuous Mesh Framework Part II: Validations and Applications," *SIAM Journal on Numerical Analysis*, Vol. 49, No. 1, 2011, pp. 61–86, doi:10.1137/10078654X.