A Mesh Optimization Algorithm to Decrease the Maximum Error in Finite Element Computations

U. Hetmaniuk[1] and P. Knupp[2]

[1] Sandia National Laboratories*, P.O. Box 5800, MS 1320, Albuquerque, NM 87185-1320
ulhetma@sandia.gov

[2] Sandia National Laboratories, P.O. Box 5800, MS 1318, Albuquerque, NM 87185-1318
pknupp@sandia.gov

Abstract. We present a mesh optimization algorithm for adaptively improving the finite element interpolation of a function of interest. The algorithm minimizes an objective function by swapping edges and moving nodes. Numerical experiments are performed on model problems. The results illustrate that the mesh optimization algorithm can reduce the $W^{1,\infty}$ semi-norm of the interpolation error. For these examples, the L^2, L^∞, and H^1 norms decreased also.

1 Introduction

Engineering applications often involve solutions with different scales of variations along different directions. For these problems, recent works have illustrated the importance of anisotropic mesh adaptation, see [18, 21, 25] and the references therein. Courty et al. [12] show that meshes aligned with the solution characteristics allow to capture the solution details with fewer nodes than isotropic meshes. When using anisotropic meshes, they observe second-order convergence with respect to the number of degrees of freedom for representing a step function with continuous linear finite elements.

Several techniques exist for anisotropic mesh adaptation. Remeshing approaches define an anisotropic metric map that governs the generation of a new anisotropic mesh [1, 12, 18, 21]. Mesh modification techniques (refinement, coarsening, edge swapping, and node movement) operate locally to adapt the mesh [14, 25]. The approach followed in this paper is mesh optimization via edge swapping and node smoothing guided by the minimization of an appropriate objective function.

We can distinguish the objective functions available in the literature into two major categories. In the first category, the objective functions rely on a variational formulation or a governing partial differential equation. Examples of

* Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

such objective functions include the energy functional [15, 22, 28] and the least-squares norm of residuals [27, 29, 31]. For these examples, the exact solution minimizes the objective function. The functional provides a natural criterion for the design of computational grids adapted to this solution via node movement and edge swapping. In the second category, the objective functions rely on recovery of derivatives of the function of interest u. This second category is more function-centric and does not depend on the problem origin (variational formulation or partial differential equation). Typically, the objective function exploits an approximation to the Hessian matrix of u. Examples of such objective functions include geometric quality functionals in a transformed space [8, 14, 25] (where the metric map is built on the Hessian matrix of u) and functionals based on local interpolation errors [3, 6, 11, 23].

We choose here to focus on an Hessian-based objective function. Defining an optimization problem independent from a particular engineering application enables the development of a black box mesh optimization algorithm that can impact many different simulation groups and packages. Such a strategy has been implemented successfully in the Mesquite mesh optimization library [24]. Currently, though, Mesquite does not include an algorithm to adapt a mesh to the physical solution in an engineering simulation. The present work is a stepping-stone toward filling this gap.

Several mesh smoothing schemes have been designed to reduce the interpolation error. These works differ on how to measure the interpolation error. When measuring the interpolation error in the L^2 norm, the optimization [3, 8, 10, 11, 14, 25] aims to equidistribute the edge length under some metric map related to the Hessian matrix of the approximated function. Additional geometric criteria must be used in conjunction with the edge length criterion, otherwise inverted elements or sliver tetrahedra may appear. The approach is efficient at reducing the L^2 norm of the error. For many engineering applications, a good representation of the solution gradient is as important as a good representation of the solution itself. Unfortunately, a mesh optimized for reducing the L^2 norm of the interpolation error may be inappropriate for representing the gradient of the solution. Long thin elements are good for linear approximation if we measure the error in the L^2 norm [11, 26]. But such elements can be undesirable when we look for a good representation of the solution gradient (see [4]). Bank and Smith [6] (node movement) and Lagüe [23] (edge swapping and node movement) introduced two mesh optimization algorithms to minimize the H^1 norm of the interpolation error. Their objective functions incorporate naturally a "barrier" term, which helps prevent elements from becoming degenerate or tangled. To the best of our knowledge, these two works are the only ones optimizing for the gradient of the interpolation error.

Other choices of norms are possible. For example, one might consider constructing an objective function based on the L^∞-norm in order to reduce pointwise the interpolation error of some physical quantity of interest. Reducing the L^∞-norm of the error does not always guarantee a decrease in the error as measured by the L^p-norms (p finite), but it provides an upper bound for these error

norms when the function of interest is smooth. A similar remark holds for the $W^{1,\infty}$ and $W^{1,p}$ norms. By reducing the $W^{1,\infty}$-norm, we control point values of the error and its gradient. This control can provide an upper bound for the $W^{1,p}$-norms, like the H^1-norm ($H^1 = W^{1,2}$). Since engineering problems are often interested in a good representation of the solution gradient at a point, we here construct an objective function that can lead to a reduction of the $W^{1,\infty}$ semi-norm of the interpolation error.

To the best of our knowledge, no work has studied the max-norm for the gradient of the interpolation error. The goal of this paper is to fill this gap and to assess numerically whether local mesh modification techniques can reduce the $W^{1,\infty}$ semi-norm of the linear interpolation error. In Section 2, we motivate and derive the adaptation algorithm. Starting from the objective function of Bank and Smith [6], we will present a new algorithm targeting the $W^{1,\infty}$ semi-norm of the interpolation error. In Section 3, we study on model problems the capability of this algorithm to reduce the interpolation error measured in the L^2, L^∞, H^1, and $W^{1,\infty}$ norms. Throughout the paper, by mesh optimization, we mean a combination of edge swapping and node movement.

2 Optimization-Based Adaptation Algorithm

In this section, we describe our approach. Section 2.1 reviews the algorithm of Bank and Smith [6]. Next, Section 2.2 motivates and derives the new objective function. Section 2.3 describes the edge swapping and node smoothing algorithms.

The focus of this paper is on two-dimensional triangular meshes because they are an important stepping stone toward a practical algorithm. For other meshes (quadrilateral, tetrahedral, ...), the method may apply but important details would need to be worked out, like swapping for tetrahedral elements. In our study, we do not incorporate h-adaptivity via refining or coarsening elements. Several works have illustrated the importance of combining h-adaptivity with mesh smoothing to generate adaptively anisotropic meshes (see [6, 12, 25]). In general, h-adaptivity and mesh smoothing are applied sequentially or in a black-box mode. We think that our algorithm could be plugged into such an approach. Most likely, some work would be required in order to get these approaches to play nicely together.

2.1 Review of the Bank and Smith Algorithm [6]

Given a domain Ω in $\mathbb{R}^2$ and a function $u \in H^1(\Omega)$, Bank and Smith [6] introduced an optimization of node locations to compute approximately

$$\min_{\mathcal{T} \in \mathcal{F}} \int_\Omega |\nabla u - \nabla u_L|^2 \, d\Omega, \tag{1}$$

where $\mathcal{F}$ is the family of conforming triangulations with fixed number of vertices. Members of $\mathcal{F}$ differ by the positions of the vertices in the mesh and, possibly,

a finite number of edge flips. u_L denotes the continuous piecewise linear nodal interpolant of u defined on the triangulation $\mathcal{T}$.

Since problem (1) is expensive to solve, Bank and Smith [6] considered

$$\min_{\mathcal{T}\in\mathcal{F}} \int_\Omega |\nabla u_Q - \nabla u_L|^2 \, d\Omega, \tag{2}$$

where u_Q denotes the quadratic interpolant of u on K. They showed that

$$\int_K |\nabla u_Q - \nabla u_L|^2 \, dK \approx \frac{l_1^2 + l_2^2 + l_3^2}{192\,|K|} s(\mathbf{H}_K, K), \tag{3}$$

where the function $s(\mathbf{H}_K, K)$ is defined by

$$s(\mathbf{H}_K, K) = (l_1^2 \mathbf{t}_1^T \mathbf{H}_K \mathbf{t}_1)^2 + (l_2^2 \mathbf{t}_2^T \mathbf{H}_K \mathbf{t}_2)^2 + (l_3^2 \mathbf{t}_3^T \mathbf{H}_K \mathbf{t}_3)^2. \tag{4}$$

$\mathbf{t}_i$ is a unit tangent vector for edge i and l_i is the length of the same edge. $\mathbf{H}_K$ is the Hessian matrix for the quadratic interpolant u_Q. Their objective function approximates the squared H^1 semi-norm for the linear interpolation error in Ω,

$$\sum_{K\in\mathcal{T}} \frac{l_1^2 + l_2^2 + l_3^2}{192\,|K|} s(\mathbf{H}_K, K) \approx \int_\Omega |\nabla u_Q - \nabla u_L|^2 \, d\Omega \approx \int_\Omega |\nabla u - \nabla u_L|^2 \, d\Omega. \tag{5}$$

In objective function (5), the factor

$$\frac{l_1^2 + l_2^2 + l_3^2}{|K|}$$

is proportional to a well-known triangle quality metric [16]. It acts as a "barrier" function that guarantees untangled elements provided the initial mesh is untangled. Bank and Smith [6] combine a few sweeps of mesh smoothing with refinement and coarsening. They do not include edge swapping. Numerical experiments by Bank and Xu [7] illustrate reductions for the L^2 and H^1 interpolation error norms. However, they do not study the point-wise maximum norm. It is conceivable that the H^1-norm of the interpolation error decreases while the $W^{1,\infty}$-norm increases.

2.2 New Objective Function

To derive our objective function, we notice that

$$\max_{\mathbf{x}\in K} |\nabla u(\mathbf{x}) - \nabla u_L(\mathbf{x})|^2 \approx \frac{1}{|K|} \int_K |\nabla u - \nabla u_L|^2 \, d\Omega$$

when $u \in H^1(\Omega) \cap W^{1,\infty}(\Omega)$ and the mesh is fine enough. After exploiting the result of Bank and Smith (3), we define the local adaptive quality metric μ by

$$\mu(K) = \frac{l_1^2 + l_2^2 + l_3^2}{192\,|K|^2} s(\mathbf{H}_K, K) \approx \max_{\mathbf{x}\in K} |\nabla u_Q(\mathbf{x}) - \nabla u_L(\mathbf{x})|^2 . \tag{6}$$

$\mu(K)$ differs from (3) only in that the power of $|K|$ is two instead of one. The first term in $\mu(K)$ still acts as a "barrier" function, but it is no longer a triangle shape-metric since it is not scale-invariant. A different but related approach to derive objective functions is used in [20], where we construct objective functions from upper bounds for the interpolation error.

Bank and Smith did not prove that objective function (5) is convex or has a unique minimum. To study the convexity of $\mu(K)$, we plot level curves for μ. In Figure 1, the triangle K is composed of two fixed vertices at $(0,0)$ and at $(1,0)$ and a free vertex at (x,y). Two different choices of matrix $\mathbf{H}_K$ are presented. When the free vertex gets closer to the horizontal axis, $\mu(K)$ increases highlighting the effect of the barrier function. The choice of matrix $\mathbf{H}_K$ modifies the level curves of μ. When $\mathbf{H}_K$ has both positive and negative eigenvalues, the contour lines are not convex and, consequently, the function μ is not a convex function. On the other hand, when the matrix $\mathbf{H}_K$ is positive definite, the contour lines are convex.

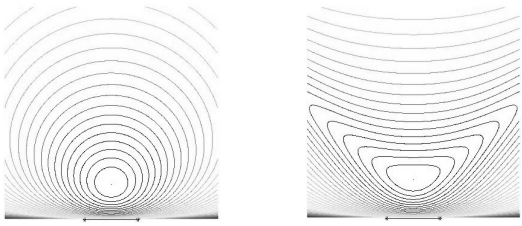

Fig. 1. Contour lines with $\mathbf{H}_K = diag(1,1)$ (left) and $\mathbf{H}_K = diag(1,-1)$ (right)

In Figure 2, showing contour lines for a local patch of elements, the interior node is free while the corner nodes are fixed. The plotted isolines are for

$$\max_{K_1,K_2,K_3,K_4} \mu(K), \tag{7}$$

when the interior node moves in the rectangle. The matrices vary per element such that

$$\mathbf{H}_{K_2} = \mathbf{H}_{K_4} = \begin{bmatrix} 1 & 0 \\ 0 & -3 \end{bmatrix} \quad \text{and} \quad \mathbf{H}_{K_1} = \mathbf{H}_{K_3} = 10^{-2}\mathbf{H}_{K_2}. \tag{8}$$

The isolines are not convex.

Convexity of the objective function is fundamental in optimization. If the optimization algorithm converges to a stationary point of a convex objective function, then the algorithm has converged to a global minimizer. When the matrix $\mathbf{H}_K$ has positive and negative eigenvalues, the function μ is not always convex. So we prefer to replace the matrix $\mathbf{H}_K$ with its absolute value and introduce the function

$$\tilde{\mu}(K) = \frac{l_1^2 + l_2^2 + l_3^2}{192\,|K|^2}[(l_1^2\mathbf{t}_1^T\,|\mathbf{H}_K|\,\mathbf{t}_1)^2 + (l_2^2\mathbf{t}_2^T\,|\mathbf{H}_K|\,\mathbf{t}_2)^2 + (l_3^2\mathbf{t}_3^T\,|\mathbf{H}_K|\,\mathbf{t}_3)^2]. \tag{9}$$

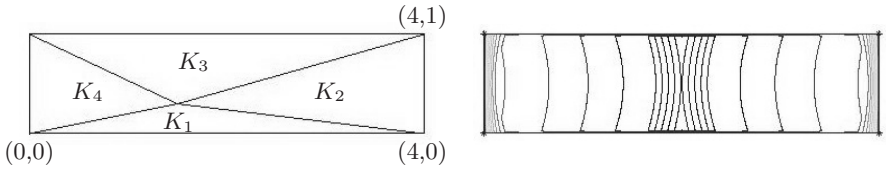

Fig. 2. Contour lines on a patch of four elements

$|\mathbf{H}_K|$ is the absolute value of the matrix $\mathbf{H}_K$, defined as $|\mathbf{H}_K| = \mathbf{S}\,|\boldsymbol{\Lambda}|\,\mathbf{S}^T$ where $(\mathbf{S}, \boldsymbol{\Lambda})$ are the eigenpairs for the matrix $\mathbf{H}_K$ and $|\boldsymbol{\Lambda}|$ is a diagonal matrix with the absolute values of eigenvalues as diagonal entries. For every vector $\mathbf{t}$, we have

$$\left(\mathbf{t}^T \mathbf{H}_K \mathbf{t}\right)^2 \leq \left(\mathbf{t}^T \left|\mathbf{H}_K\right| \mathbf{t}\right)^2 .$$

So the function $\tilde{\mu}$ is an upper bound for the local quality metric μ. Asymptotically, the function $\tilde{\mu}$ is also an upper bound for the max-norm of the gradient of the interpolation error. We were not able to prove that the function $\tilde{\mu}$ is convex. But our experiments gave convex level curves for this function. For example, with the matrices $\mathbf{H}_K$ defined in (8), the isolines for the functional

$$\max_{K_1, K_2, K_3, K_4} \tilde{\mu}(K), \tag{10}$$

are convex (see Figure 3).

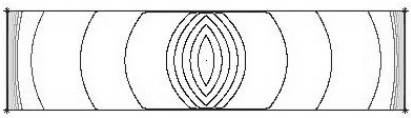

Fig. 3. Contour lines for functional (10) on a patch of four elements

Even though $\tilde{\mu}$ formally has a "barrier", the "barrier" term can become very weak when the matrix $|\mathbf{H}_K|$ approaches zero. The function $\tilde{\mu}$ satisfies

$$0 \leq \tilde{\mu}(K) \leq \|\mathbf{H}_K\|_2^2 \, \frac{l_1^2 + l_2^2 + l_3^2}{192\,|K|^2} \left(l_1^4 + l_2^4 + l_3^4\right).$$

When the matrix $\mathbf{H}_K$ is exactly zero in a mesh region, the "barrier" term is canceled for the elements in that region. Then degenerate elements or elements of infinite size may appear where the function u is linear. When the matrix $\mathbf{H}_K$ has a small norm, *i.e.* when the function u is almost linear, the "barrier" term will prevent inverted elements. But a small norm for $\mathbf{H}_K$ can postpone the impact of the barrier function, *i.e.* the function $\tilde{\mu}$ can remain small even when the element K becomes flat.

To minimize the interpolation error, the aspect ratio of a triangle aligned with a function u must depend on the condition number of the Hessian matrix, *i.e.*

the ratio of largest over smallest eigenvalues (see [9, 13, 26]). A condition number of 1 will result in an equilateral triangle. A large condition number will result in a stretched triangle. To control the level of anisotropy in the optimized mesh, we modify $\tilde{\mu}$ and introduce the function $\tilde{\mu}_\varepsilon$,

$$\tilde{\mu}_\varepsilon(K) = \frac{l_1^2 + l_2^2 + l_3^2}{192\,|K|^2}\; s(|\mathbf{H}_K| + \varepsilon\mathbf{I}, K). \tag{11}$$

This regularization controls the level of anisotropy in the optimized mesh because it sets an upper bound on the condition number of $|\mathbf{H}_K| + \varepsilon\mathbf{I}$,

$$\frac{\lambda_{\max}(|\mathbf{H}_K| + \varepsilon\mathbf{I})}{\lambda_{\min}(|\mathbf{H}_K| + \varepsilon\mathbf{I})} = \frac{\lambda_{\max}(|\mathbf{H}_K|) + \varepsilon}{\lambda_{\min}(|\mathbf{H}_K|) + \varepsilon} \le 1 + \frac{\lambda_{\max}(|\mathbf{H}_K|)}{\varepsilon}. \tag{12}$$

This upper bound on the condition number will limit the aspect ratios for triangles in the optimized mesh.

The goal of our mesh optimization algorithm is to compute approximately

$$\min_{\mathcal{T}\in\mathcal{F}}\ \max_{K\in\mathcal{T}} \tilde{\mu}_\varepsilon(K) \tag{13}$$

via edge swapping and node movement in order to reduce the interpolation error. The max-norm for the gradient of interpolation error and the function $\tilde{\mu}_\varepsilon$ satisfy

$$\max_{\mathbf{x}\in K}\ |\nabla u(\mathbf{x}) - \nabla u_L(\mathbf{x})|^2 \approx \mu(K) \le \tilde{\mu}(K) \le \tilde{\mu}_\varepsilon(K). \tag{14}$$

2.3 Edge Swapping and Node Movement Algorithms

In this section, we describe our algorithm to solve approximately (13). Problem (13) is a non-smooth optimization problem because the max-value function is non-differentiable. Such a problem requires special-purpose algorithms (see, for example, Fletcher [17]). Here we prefer to look for a cheap approximate solution to the optimization problem and we replace problem (13) by the following optimization problem

$$\min_{\mathcal{T}\in\mathcal{F}} \sum_{K\in\mathcal{T}} \tilde{\mu}_\varepsilon(K). \tag{15}$$

The goal of our algorithm is now to compute an approximate solution to (15).

First, we describe our edge swapping algorithm. Figure 4 illustrates the swapping of edge e into edge γ. Note that when edge e is flipped, the surrounding elements $K_{e,1}$ and $K_{e,2}$ are replaced with the elements $K_{\gamma,1}$ and $K_{\gamma,2}$. Let Ω_e denote the patch surrounding the edge e such that

$$\overline{\Omega_e} = \overline{K_{e,1}} \cup \overline{K_{e,2}} = \overline{K_{\gamma,1}} \cup \overline{K_{\gamma,2}}.$$

Algorithm 1 describes our edge swapping algorithm for problem (15). The first step is to check that the patch Ω_e is convex. When Ω_e is not convex, the flip would generate inverted elements, so the algorithm skips the edge. Then we

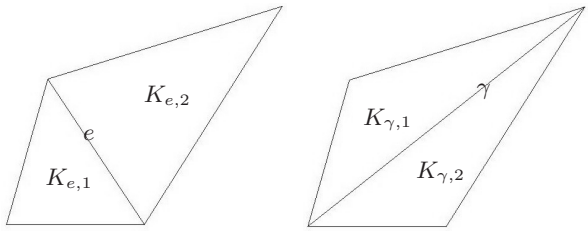

Fig. 4. Illustration of swapping the edge e into the edge γ

Algorithm 1. Edge Swapping Algorithm:

```
repeat
  for any edge e in the mesh do
    if Ω_e is not convex then
      Skip the edge
    end if
    Compute μ̃_ε(K_{e,1}) and μ̃_ε(K_{e,2}).
    Let γ denote the candidate flipped edge.
    Compute μ̃_ε(K_{γ,1}) and μ̃_ε(K_{γ,2}).
    if max(μ̃_ε(K_{γ,1}), μ̃_ε(K_{γ,1})) ≥ max(μ̃_ε(K_{e,1}), μ̃_ε(K_{e,1})) then
      Skip the edge.
    end if
    if μ̃_ε(K_{γ,1}) + μ̃_ε(K_{γ,1}) ≥ μ̃_ε(K_{e,1}) + μ̃_ε(K_{e,1}) then
      Skip the edge.
    end if
    Perform the edge swapping.
  end for
until no edge is flipped.
```

compare the value of the function $\tilde{\mu}_\varepsilon$ on all the four elements. If the maximum and the sum of $\tilde{\mu}_\varepsilon$ over the patch both decrease, then we perform the flipping of edge e into edge γ. When swapping an edge, the value of the objective function, $\sum_{K\in\mathcal{T}} \tilde{\mu}_\varepsilon(K)$, varies only for the elements in Ω_e. So we perform checks on the maximum and the sum only over the patch Ω_e. These checks arise from the fact that the algorithm looks for an approximate solution to problem (15) and to the original problem (13).

A key component of the edge swapping algorithm is the evaluation of $\tilde{\mu}_\varepsilon$ on the four elements. This evaluation requires a matrix $|\mathbf{H}_K|$ on each element. The inputs for our mesh optimization algorithm are a non-inverted initial mesh $\mathcal{T}^0$ and, for every vertex V of $\mathcal{T}^0$, a matrix $\tilde{\mathbf{H}}_V$. The initial mesh will be stored as a background mesh for the piecewise linear interpolation of the absolute value of these nodal matrices, $\boldsymbol{\Pi}^0(\{|\tilde{\mathbf{H}}_V|\}_{V\in\mathcal{T}^0})$. For any point $\mathbf{x}$ in element K^0 of the background mesh $\mathcal{T}^0$, we choose

$$\boldsymbol{\Pi}^0(\{|\tilde{\mathbf{H}}_V|\}_{V\in\mathcal{T}^0})(\mathbf{x}) = |\tilde{\mathbf{H}}_{V_1}|\lambda^0_{V_1}(\mathbf{x}) + |\tilde{\mathbf{H}}_{V_2}|\lambda^0_{V_2}(\mathbf{x}) + |\tilde{\mathbf{H}}_{V_3}|\lambda^0_{V_3}(\mathbf{x}), \qquad (16)$$

where the vertices V_1, V_2, and V_3 form the element K^0 and $\lambda_V^0(\mathbf{x})$ denotes the barycentric coordinate of $\mathbf{x}$ with respect to vertex V in element K^0. To evaluate $\tilde{\mu}_\varepsilon$, we set the matrix $|\mathbf{H}_K|$ to be

$$|\mathbf{H}_K| = \boldsymbol{\Pi}^0(\{|\tilde{\mathbf{H}}_V|\}_{V\in\mathcal{T}^0})(\mathbf{G}_K), \tag{17}$$

where $\mathbf{G}_K$ is the centroid of element K. Requiring the matrices $\tilde{\mathbf{H}}_V$ as input of an adaptive scheme is reasonable. Indeed, $\mathbf{H}_K$ is the Hessian matrix for the quadratic interpolant, u_Q, of u in element K. As u_Q is usually unknown, the discrete Hessian matrix at vertex V is recovered from a discrete solution. Several nodal-based recovery schemes of Hessian matrix from a discrete solution are available. Discussing such recovery schemes is outside the scope of this paper. For further details, we refer to [30] and the references therein.

For the node movement algorithm, an iterative Gauss-Seidel-like method is used where we sweep through the vertices, locally optimizing the position of a single vertex while holding all others vertices fixed. Given a vertex V, let Ω_V be the patch of elements sharing V as a vertex. Algorithm 2 describes our node movement algorithm for problem (15). The first step is to compute the matrices $|\mathbf{H}_K|$ for all the elements K. To reduce computational expenses, these matrices are kept constant during one whole sweep through the mesh vertices. To optimize the position of vertex V, we use the nonlinear conjugate gradient algorithm CG_DESCENT [19] on the objective function, $\sum_{K\in\mathcal{T}} \tilde{\mu}_\varepsilon(K)$, viewed as a function of (x_V, y_V). Algorithm CG_DESCENT is stopped when the norm of the gradient of the objective function has been reduced by a factor 10^{-6}, when the relative change in objective function is smaller than 10^{-12}, or when 100 iterations have been performed. The barrier term in $\tilde{\mu}_\varepsilon$ will keep vertex V in Ω_V or in the subregion of Ω_V visible from all points on the boundary of Ω_V (when Ω_V is not convex). Moving a vertex V can affect the values of $\tilde{\mu}_\varepsilon$ only for the elements having V as vertex, *i.e.* the elements in Ω_V. Algorithm CG_DESCENT will reduce

Algorithm 2. Node Movement Algorithm:

```
for iter = 1, ··· , it_max do
  Get all the matrices |H_K| with formula (17).
  Compute the objective function μ_init = Σ_{K∈T} μ̃_ε(K).
  for any vertex V in the mesh do
    Compute the maximum value of μ̃_ε on Ω_V, μ_max,init.
    Apply the nonlinear conjugate gradient to move vertex V in Ω_V.
    Compute the maximum value of μ̃_ε on Ω_V, μ_max,new.
    if μ_max,new ≥ μ_max,init then
      Reset vertex V to the latest position.
    end if
  end for
  Compute the objective function μ_new = Σ_{K∈T} μ̃_ε(K).
  if |μ_new − μ_init| ≤ tol · μ_init then
    Stop the iterations.
  end if
end for
```

the value $\sum_{K \in \Omega_V} \tilde{\mu}_\varepsilon(K)$. As the original optimization problem (13) looks at the maximum value of $\tilde{\mu}_\varepsilon$, we check also the maximum value of $\tilde{\mu}_\varepsilon$ on Ω_V. When the movement of vertex V creates an increase, we do not perform the movement and set vertex V to its latest position. Finally, we stop sweeping through vertices when the relative change in objective function is smaller than a tolerance, *tol*, or when a maximal number of iterations has been reached. For boundary vertices, we apply the same algorithm, but with appropriate constraints to keep the vertex on the boundary.

Several choices are available to combine the edge swapping and the node smoothing algorithms. In the following section, we will study on model problems how to combine these two algorithms.

3 Numerical Experiments

In this section, numerical experiments on model problems are presented to illustrate the mesh optimization algorithm. The goal is to assess whether the algorithm can reduce the interpolation error or not. The meshes are adapted to three given analytical functions with different anisotropic features, so that we can compute exactly the interpolation error. We do not use a realistic engineering problem because the exact solution would not be known.

The inputs for the mesh optimization algorithm are a non-inverted initial mesh and Hessian matrices at the vertices of the initial mesh. The input matrices at the vertices are computed with the analytical expression of the function to separate the potential error reduction, due to the mesh algorithm, from the effect of a Hessian recovery operator.

3.1 Experiment on the Regularization Parameter

For the first test case, only the node movement algorithm is used. Based on a scaling argument via (12), the regularization parameter ε is set to $\sigma \ \max_{V \in \mathcal{T}^0} \|\tilde{\mathbf{H}}_V\|_2$, where σ is a parameter in $[0, 1]$. We compare numerically the effect of σ on the optimization algorithm.

The first function u is given by

$$u(x, y) = \exp\left[-100(x - \frac{1}{2})^2 - 100(y - \frac{1}{2})^2\right] \tag{18}$$

in the domain $\Omega = (0, 1) \times (0, 1)$. The Gaussian function u is symmetric, decays rapidly to zero, and exhibits a large region where it is flat and where the norm $\left\|D^2 u(\mathbf{x})\right\|_F$ is small. The function has no anisotropy in it. Figure 5 illustrates the initial uniform structured mesh with 32 elements per direction and the contour of function (18).

Table 1 lists the norms of interpolation error on the initial mesh and after optimization via node movement, when σ varies. The interpolation error norms decrease after optimization. Except for the L^2-norm, the decrease is almost monotone with σ. For the $W^{1,\infty}$ norm, the reduction of interpolation error

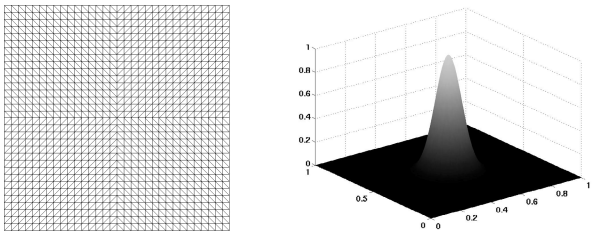

Fig. 5. Initial structured mesh (left) — Contour for function (18) (right)

Table 1. Evolution of interpolation error for function (18) as a function of σ

	$\|u-u_L\|_{L^2}$	$\|u-u_L\|_{H^1}$	$\|u-u_L\|_{L^\infty}$	$\|u-u_L\|_{W^{1,\infty}}$	$\|\mathbf{A}_K\|_F \left\|\mathbf{A}_K^{-1}\right\|_F$
Initial	3.28×10^{-3}	3.52×10^{-1}	3.80×10^{-2}	2.40	[2, 3]
$\sigma = 10^{-2}$	0.80×10^{-3}	1.57×10^{-1}	1.24×10^{-2}	1.67	[2, 10]
$\sigma = 10^{-4}$	1.63×10^{-3}	1.43×10^{-1}	1.01×10^{-2}	1.53	[2, 69]
$\sigma = 10^{-8}$	1.52×10^{-3}	1.42×10^{-1}	0.98×10^{-2}	1.51	[2, 560]
$\sigma = 0.0$	1.52×10^{-3}	1.43×10^{-1}	1.00×10^{-2}	1.52	[2, 567]

is between 30% and 37%. The other norms are reduced by at least 50%. Table 1 also lists the range for condition number of $\mathbf{A}_K$. We recall that $\mathbf{A}_K$ is the Jacobian matrix for mapping the unit right-angled triangle to the physical triangle K. The range of the condition numbers increase when σ gets smaller, indicating that some elements become more anisotropic.

Figure 6 describes the optimized meshes for $\sigma = 10^{-2}$ and $\sigma = 10^{-4}$. When $\left\|D^2u(\mathbf{x})\right\|_F$ is large, small isotropic elements are created. In the region where $\left\|D^2u(\mathbf{x})\right\|_F$ is small, the mesh optimization allows flat or stretched elements as these elements do not increase the interpolation error.

Function (18) has no anisotropic feature. Therefore, anisotropic elements are not needed to represent this function. The optimized meshes contain anisotropic elements because of the flatness of u, the boundary constraints, and the fixed

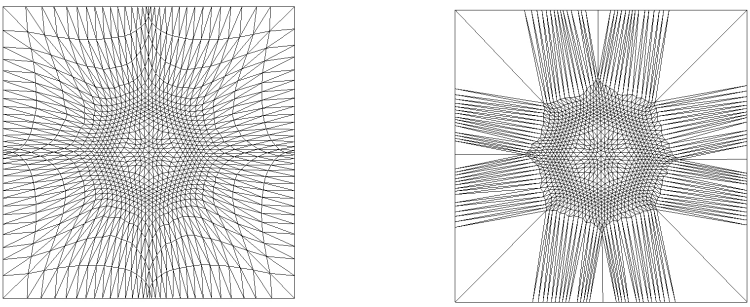

Fig. 6. Optimized meshes for (18) with $\sigma = 10^{-2}$ (left) and $\sigma = 10^{-4}$ (right)

topology. In order to reduce the error, the optimization gathers nodes in the regions that matter. In those regions, the mesh is notably isotropic. Elsewhere, the function is almost linear and anisotropic elements do not cause the error to increase. The regularization controls here the degree of anisotropy at the expense of a slight increase in the error.

3.2 Experiments on Combining Swapping and Smoothing

For the second series of tests, we combine edge swapping and node smoothing and compare numerically different combinations for the mesh optimization algorithm. In the following, the letter E denotes one call to Algorithm 1 and the letter N one call to Algorithm 2.

Table 2. Evolution of interpolation error for function (18) with $\sigma = 10^{-2}$

	$\|u - u_L\|_{L^2}$	$\|u - u_L\|_{H^1}$	$\|u - u_L\|_{L^\infty}$	$\|u - u_L\|_{W^{1,\infty}}$	$\|\mathbf{A}_K\|_F \left\|\mathbf{A}_K^{-1}\right\|_F$
Initial	3.28×10^{-3}	3.52×10^{-1}	3.80×10^{-2}	2.40	[2, 3]
N	0.804×10^{-3}	1.57×10^{-1}	1.24×10^{-2}	1.67	[2, 10]
(N, E, N)	0.702×10^{-4}	1.36×10^{-1}	0.986×10^{-2}	1.51	[2, 8.1]
E	2.71×10^{-3}	2.98×10^{-1}	3.79×10^{-2}	2.40	[2, 3]
(E, N)	1.57×10^{-3}	2.37×10^{-1}	2.23×10^{-2}	2.16	[2, 5]

σ is set at 10^{-2}. Table 2 lists the interpolation error norms with different combinations of edge swapping and node movement for function (18). All the interpolation error norms decrease after optimization. The combination (N, E, N) gives the smallest error norms. The $W^{1,\infty}$ norm is reduced by 37%, the H^1 norm by 55%, the L^2 norm by 78%, and the L^∞ norm by 67%. The $W^{1,\infty}$ error in Table 1 corresponding to Figure 6 (right) is nearly the same as the error in Table 2 corresponding to Figure 7 (left). Therefore, one can achieve the same error level with a lot less mesh anisotropy by adding the ability to swap edges to the adaptive algorithm.

Figure 7 plots the optimized meshes for the Gaussian function (18) with the combinations (N, E, N) and (E, N). The meshes are visually different from the optimized mesh with node movement only (see Figure 6). For this function and this initial mesh, starting with edge swapping creates a mesh topology that prevents outer vertices to gather in the middle of the mesh. This behavior limits the error reduction in comparison to combinations starting with node movement.

In the subsequent experiments, we use a heuristic formula to set the regularization parameter ε. We choose ε to be

$$\varepsilon = \sigma^0 \max_{V \in \mathcal{T}^0} \|\tilde{\mathbf{H}}_V\|_2 \quad \text{and} \quad \sigma^0 = \frac{1}{100 \max_{K^0 \in \mathcal{T}^0} \|\mathbf{A}_{K^0}\|_F \left\|\mathbf{A}_{K^0}^{-1}\right\|_F}. \tag{19}$$

In this formula, the constant σ^0 depends only on the initial mesh $\mathcal{T}^0$.

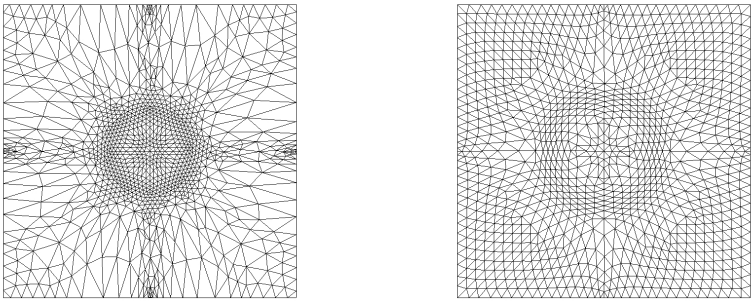

Fig. 7. Optimized meshes for (18) with (N, E, N) (left) and (E, N) (right)

The next function, from Huang [21], is given by

$$u(x, y) = \tanh(24y) - \tanh\left[24\left(x - y - \frac{1}{2}\right)\right] \tag{20}$$

in the domain $\Omega = (0, 1) \times (0, 1)$. This function simulates the interaction of a boundary layer (along the line $y = 0$) with an oblique shock wave (along the line $y = x - 1/2$). Figure 8 illustrates the initial unstructured mesh with 1002 vertices and 2006 elements and the surface defined by function (20). Based on (19), this initial mesh results in σ^0 to be $1/420$.

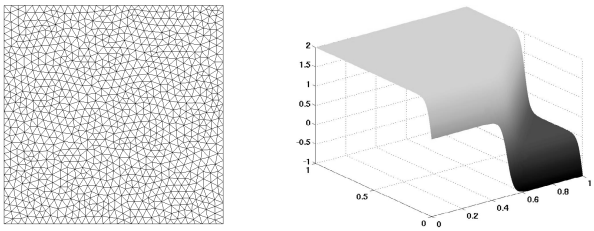

Fig. 8. Initial unstructured mesh (left) – Contour surface for function (20) (right)

Table 3 lists the interpolation error norms on the optimized mesh with different combinations of edge swapping and node smoothing. All the interpolation error norms decrease with more levels of optimization. Combining edge swapping and node movement results in smaller error norms than using only the edge swapping or only the node movement. Combinations starting with node movement produce larger reduction in error norms than the ones starting with edge swapping. After the combinations (N, E, N) and (E, N), adding more cycles of edge swapping and node smoothing does not reduce significantly the interpolation error. The combination (N, E, N) reduces the $W^{1,\infty}$ norm by 37% and the other norms by at least 63%. It reduces the error more than the combination (E, N) and exhibits a larger condition number range, indicating that anisotropic

Table 3. Evolution of interpolation error for function (20) for $\sigma^0 = 1/420$

	$\|u - u_L\|_{L^2}$	$\|u - u_L\|_{H^1}$	$\|u - u_L\|_{L^\infty}$	$\|u - u_L\|_{W^{1,\infty}}$	$\|\mathbf{A}_K\|_F \left\|\mathbf{A}_K^{-1}\right\|_F$
Initial	1.72×10^{-2}	1.74	1.75×10^{-1}	15.17	[2, 4.2]
N	0.655×10^{-2}	0.755	0.575×10^{-1}	9.56	[2, 28.0]
(N, E, N)	0.471×10^{-2}	0.628	0.575×10^{-1}	9.56	[2, 28.5]
$(N, (E, N)^2)$	0.467×10^{-2}	0.620	0.575×10^{-1}	9.56	[2, 28.4]
$(N, (E, N)^3)$	0.459×10^{-2}	0.619	0.575×10^{-1}	9.56	[2, 28.1]
E	0.933×10^{-2}	1.11	1.48×10^{-1}	15.17	[2, 23.4]
(E, N)	0.538×10^{-2}	0.78	0.902×10^{-1}	11.37	[2, 18.0]
$(E, N)^2$	0.525×10^{-2}	0.762	0.902×10^{-1}	11.37	[2, 23.2]
$(E, N)^3$	0.524×10^{-2}	0.759	0.902×10^{-1}	11.37	[2, 24.7]

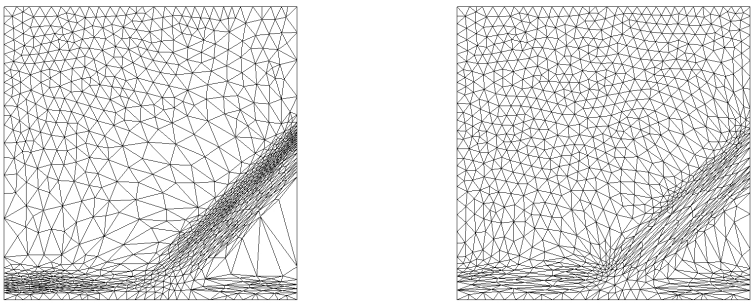

Fig. 9. Optimized meshes for (20) with (N, E, N) (left) and (E, N) (right)

elements help reducing the error. Figure 9 illustrates the optimized meshes for the combinations (N, E, N) and (E, N). Starting with edge swapping creates a mesh topology that prevents vertices to gather along the anisotropic features of function (20).

Finally, we study the function from [2]

$$u(x, y) = \sin\left[5(2x - 1.2)^3(4y^2 - 6y + 3)\right]. \tag{21}$$

The initial mesh was generated by the code BAMG of Hecht [5] and it is pre-adapted to function (21) so that the relative L^∞-norm of the interpolation error is smaller than 5 %. The mesh contains 3173 vertices and 5962 elements. Based on (19), this initial mesh results in σ^0 to be 1/73520. The value of σ^0 is smaller because the pre-adapted initial mesh contains stretched elements. Figure 10 illustrates the initial pre-adapted mesh and the contour for function (21). Table 4 lists the interpolation error norms on the optimized mesh with different combinations of edge swapping and node smoothing. All the interpolation error norms decrease after optimization. For this function and this initial mesh, the H^1 and $W^{1,\infty}$ norms are more reduced when the mesh optimization starts with node movement. For the L^2 and L^∞ norms, it is better to start with edge swapping. The combination (N, E, N) decreases the $W^{1,\infty}$ norm by 53% and the other

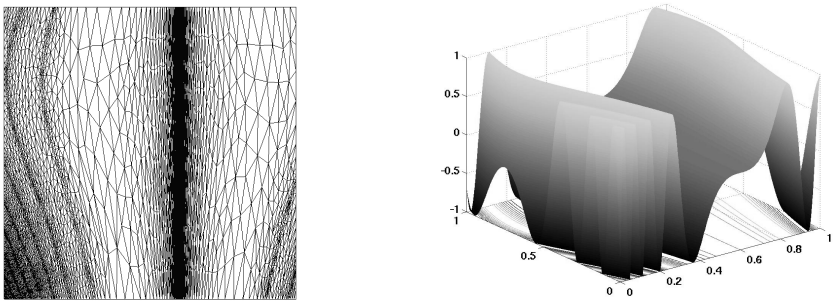

Fig. 10. Initial mesh pre-adapted for (21) (left) – Contour surface for (21) (right)

Table 4. Evolution of interpolation error for function (21) for $\sigma^0 = 1/73520$

	$\|u - u_L\|_{L^2}$	$\|u - u_L\|_{H^1}$	$\|u - u_L\|_{L^\infty}$	$\|u - u_L\|_{W^{1,\infty}}$	$\|\mathbf{A}_K\|_F \left\|\mathbf{A}_K^{-1}\right\|_F$
Initial	6.15×10^{-3}	1.67	3.85×10^{-2}	31.73	[2, 735.2]
N	5.19×10^{-3}	1.29	3.05×10^{-2}	17.49	[2, 430]
(N, E, N)	4.98×10^{-3}	1.04	3.05×10^{-2}	14.85	[2, 67.5]
$(N, (E, N)^2)$	4.94×10^{-3}	0.99	3.05×10^{-2}	14.24	[2, 54.65]
E	5.57×10^{-3}	1.31	2.92×10^{-2}	22.58	[2, 168]
(E, N)	4.83×10^{-3}	1.07	2.22×10^{-2}	17.24	[2, 52.8]
$(E, N)^2$	4.67×10^{-3}	1.01	2.20×10^{-2}	17.24	[2, 61.3]

norms by at least 19%. The combination (E, N) reduces the $W^{1,\infty}$ norm by 45% and the other norms by at least 21%. In comparison with the previous experiments, it is notable that even when the initial mesh is pre-adapted, our algorithm can still make worthwhile reductions in the error. Figure 11 plots the optimized meshes for the combinations (N, E, N) and (E, N).

Based on these experiments, we can draw the following conclusions. Combining edge swapping and node movement results in smaller error norms than using only the edge swapping or only the node movement. The edge swapping should

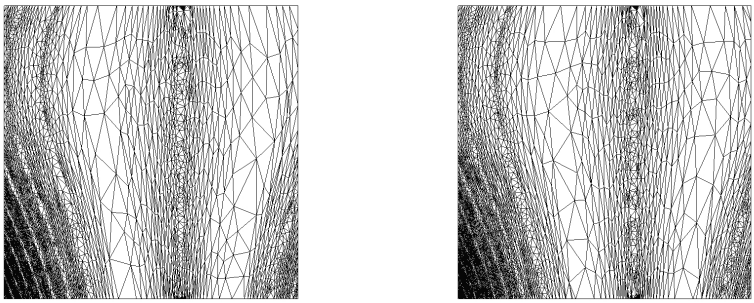

Fig. 11. Optimized meshes for (21) with (N, E, N) (left) and (E, N) (right)

be followed by node smoothing to benefit from the topology changes. When the initial mesh is not pre-adapted, it seems important to move the nodes before swapping any edge. The combination (N, E, N) is usually a good combination.

4 Conclusion

An optimization-based mesh adaptation algorithm has been presented. It combines edge swapping and node movement to minimize an objective function. The mesh adaptation algorithm exploits information from a discrete Hessian matrix. The objective function is based on the Bank and Smith [6] formula for the H^1 semi-norm of the linear interpolation error. In this formula, we replace the Hessian matrix for the quadratic interpolant of the fonction of interest u with a symmetric definite positive approximation. The objective function contains naturally a "barrier" term to ensure that starting from a mesh without inverted elements, the resulting mesh will not contain any inverted elements.

On model problems, we illustrated numerically that the algorithm can diminish the $W^{1,\infty}$ semi-norm of the interpolation error. This decrease is gratifying because our objective function is only asymptotically an upper bound to the $W^{1,\infty}$ semi-norm of the interpolation error. In our numerical experiments, the reduction in the $W^{1,\infty}$ semi-norm is accompanied by a decrease in the L^∞, L^2, and H^1 norms. The amount of decrease is problem-dependent. However, when the initial mesh is already appropriate to represent the function of interest, the decrease can be limited. When the initial mesh is not aligned with the function, the algorithm computes a better mesh topology and node distribution to represent the function. Then the reduction of the error norm can be significant.

A limitation of the algorithm is the lack of guarantee that all the error norms will always decrease. Report [20] contains a numerical experiment where some norms increase after optimization. Without more knowledge about the function of interest (like a governing partial differential equation for engineering applications), we are not aware of any algorithm with such a guarantee. The focus of this paper is on triangular two-dimensional meshes. For other situations (quadrilateral element, three-dimensional meshes), we expect that our approach will apply as well. However, important details remain to be worked out. Future directions for this work include the coupling with a recovery operator for the Hessian matrix, an efficient implementation into the Mesquite library [24], the extension to multiple dependent variables, and the use of quantities of interest for defining the objective function.

References

1. Agouzal, A., Lipnikov, K., Vassilevski, Y.: Adaptive generation of quasi-optimal tetrahedral meshes. East-West J. Numer. Math. 7(4), 223–244 (1999)
2. Alauzet, F., Frey, P.: Estimation d'erreur géométrique et métriques anisotropes pour l'adaptation de maillage. Partie II: exemples d'applications. Technical Report 4789, INRIA (2003)

3. Apel, T., Berzins, M., Jimack, P., Kunert, G., Plaks, A., Tsukerman, I., Walkley, M.: Mesh shape and anisotropic elements: Theory and practice. In: The Mathematics of Finite Elements and Applications X, pp. 367–376. Elsevier, Amsterdam (2000)
4. Babuška, I., Aziz, A.K.: On the angle condition in the finite element method. SIAM J. Numer. Anal. 13, 214–226 (1976)
5. BAMG, Bidimensional anisotropic mesh generator, `http://pauillac.inria.fr/cdrom/www/bamg/eng.htm`
6. Bank, R., Smith, R.: Mesh smoothing using a posteriori error estimates. SIAM J. Numer. Anal. 34, 979–997 (1997)
7. Bank, R., Xu, J.: Asymptotically exact a posteriori error estimators, part II: general unstructured grids. SIAM J. Numer. Anal. 41, 2313–2332 (2003)
8. Buscaglia, G., Dari, E.: Anisotropic mesh optimization and its application in adaptivity. Int. J. Numer. Meth. Engng. 40, 4119–4136 (1997)
9. Cao, W.: On the error of linear interpolation and the orientation, aspect ratio, and internal angles of a triangle. SIAM J. Numer. Anal. 43, 19–40 (2005)
10. Chen, L.: Mesh smoothing schemes based on optimal Delaunay triangulations. In: Proceedings of 13th International Meshing Roundtable (2004)
11. Chen, L., Sun, P., Xu, J.: Optimal anisotropic meshes for minimizing interpolation errors in L^p-norm. Math. Comp. 76, 179–204 (2007)
12. Courty, F., Leservoisier, D., George, P.L., Dervieux, A.: Continuous metric and mesh adaptation. Appl. Numer. Math. 56, 117–145 (2006)
13. D'Azevedo, E., Simpson, R.: On optimal interpolation triangle incidences. SIAM J. Sci. Comput. 10, 1063–1075 (1989)
14. Dompierre, J., Vallet, M.G., Bourgault, Y., Fortin, M., Habashi, W.: Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver independent CFD. Part III: unstructured meshes 39, 675–702 (2002)
15. Felippa, C.: Optimization of finite element grids by direct energy search. Appl. Math. Modelling 1, 93–96 (1976)
16. Field, D.: Qualitative measures for initial meshes. Int. J. Numer. Meth. Engng. 47, 887–906 (2000)
17. Fletcher, R.: Practical methods of optimization. Wiley, Chichester (1987)
18. Frey, P., Alauzet, F.: Anisotropic mesh adaptation for CFD computations. Comput. Methods Appl. Mech. Engrg. 194, 5068–5082 (2005)
19. Hager, W., Zhang, H.: Algorithm 851: CG_DESCENT, a conjugate gradient method with guaranteed descent. ACM Trans. Math. Softw. 32, 113–137 (2006)
20. Hetmaniuk, U., Knupp, P.: Mesh optimization algorithms to decrease the linear interpolation error. SAND Report (2008)
21. Huang, W.: Mathematical principles of anisotropic mesh adaptation. Commun. Comput. Phys. 1, 276–310 (2006)
22. Jimack, P., Mahmood, R., Walkley, M., Berzins, M.: A multilevel approach for obtaining locally optimal finite element meshes. Advances in Engineering Software 33, 403–415 (2002)
23. Lagüe, J.F.: Optimisation de maillage basée sur une erreur locale d'interpolation. PhD thesis, Université de Paris VI (2006)
24. Mesquite, Mesh quality improvement toolkit, `http://www.cs.sandia.gov/optimization/knupp/Mesquite.html`
25. Remacle, J.F., Li, X., Shephard, M., Flaherty, J.: Anisotropic adaptive simulation of transient flows using discontinuous Galerkin methods. Int. J. Numer. Meth. Engng. 62, 899–923 (2005)

26. Rippa, S.: Long and thin triangles can be good for linear interpolation. SIAM J. Numer. Anal. 29, 257–270 (1992)
27. Roe, P., Nishikawa, H.: Adaptive grid generation by minimizing residuals. Int. J. Numer. Meth. Fluids 40, 121–136 (2002)
28. Tourigny, Y., Hülsemann, F.: A new moving mesh algorithm for the finite element solution of variational problems. SIAM J. Numer. Anal. 35, 1416–1438 (1998)
29. Tourigny, Y.: The optimisation of the mesh in first-order systems least-squares methods. J. Sci. Comput. 24, 219–245 (2005)
30. Vallet, M.G., Manole, C., Dompierre, J., Dufour, S., Guibault, F.: Numerical comparison of some Hessian recovery techniques. Int. J. Numer. Meth. Engng. 72, 987–1007 (2007)
31. Walkley, M., Jimack, P., Berzins, M.: Anisotropic adaptivity for the finite element solutions of three-dimensional convection-dominated problems. Int. J. Numer. Meth. Fluids 40, 551–559 (2002)