Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 194 (2004) 611–631
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

A finite element method for unstructured grid smoothing

Glen Hansen [a,*], Andrew Zardecki [a], Doran Greening [b], Randy Bos [b]

[a] *Computational Science Methods Group, Applied Physics Division, MS F645, Los Alamos National Laboratory, Los Alamos, NM 87545, USA*

[b] *Materials Science Group, Applied Physics Division, MS F699, Los Alamos National Laboratory, Los Alamos, NM 87545, USA*

Received 12 December 2002; received in revised form 22 September 2003; accepted 22 September 2003

Abstract

The finite element method is applied to grid smoothing for two-dimensional planar geometry. The coordinates of the grid nodes satisfy two quasi-linear elliptic equations in the form of Laplace equations in a Riemann space. By forming a Dirichlet boundary value problem, the proposed method is applicable to both structured and unstructured grids. The Riemannian metric, acting as a driving force in the grid smoothing, is computed iteratively beginning with the metric of the unsmoothed grid. Smoothing is achieved by computing the metric tensor on the dual mesh elements, which incorporates the influence of neighbor elements. Numerical examples of this smoothing methodology, demonstrating the efficiency of the proposed approach, are presented.

PACS: 1991 MSC: 65M50; 51P05; 65N30

Keywords: Finite elements; Galerkin methods; Mesh generation; Elliptic smoothing

1. Introduction

Grid generation methods continue to advance to meet the needs of models involving more complex phenomena on intricate computational domains. There are two primary areas of interest when considering approaches to grid generation and optimization. The first area concentrates on the generation of the "initial" mesh. This mesh is designed to meet the geometric requirements of the domain and problem for the first cycle(s) of the simulation process. For example, it is necessary to "stretch" the mesh uniformly across the domain without excessive concentration of mesh in any particular area. Furthermore, the mesh must "fit" the internal and external boundaries of the domain, and provide sufficient accuracy in the initial stages of the computation. In this phase, there are significant interactions between mesh geometric qualities and early time solution criteria, with the computational robustness of the initial mesh being of paramount

* Corresponding author. Tel.: +1-505-667-0655.
E-mail addresses: ghansen@lanl.gov (G. Hansen), azz@lanl.gov (A. Zardecki), dgreening@lanl.gov (D. Greening), rbos@lanl.gov (R. Bos).

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2003.09.014

importance. The second area of mesh generation is concerned with "optimizing" an existing mesh (likely obtained from a previous solution step) to meet the requirements of the simulation in progress. The geometric fidelity of the previous mesh is assumed to be adequate, thus solution criteria become a larger consideration as the simulation progresses. In application, there is not a distinct boundary between the two approaches, but a blend where the influence of geometry decreases and solution criteria increases. The proposed finite element method is currently targeted only at the area of initial mesh generation and is limited in function to capturing the domain geometry details.

There is often a diverse set of requirements that the initial mesh must satisfy. One area of interest concerns detailed simulations of high speed flow regimes involving the formation of strong shocks in a multi-dimensional computational domain. Problems of this nature are quite challenging as they are often described by large values of geometric curvature which mandate a hybrid unstructured mesh topology. Furthermore, there are constraints imposed by the solution on mesh features in the neighborhood of domain boundaries and shock fronts and accuracy concerns require bounding mesh anisotropy within geometric regions. It is also necessary to insure a repeatable initial state across the problem set of interest. To provide a predictable initial state on identical problems, it is necessary to fully converge the appropriate grid generation method. Furthermore, there is also merit to beginning the simulation from a converged mesh state even if the problems are distinct.

Elliptic grid generation methods have shown themselves to be quite effective for robust initial mesh generation tasks on structured grids. Elliptic methods, when control functions are included, effectively equidistribute the mesh across complex domains. Current approaches do not lend themselves well to application on unstructured meshes, due to the assumption of a global parameter space for the evaluation of the elliptic operator and the metric terms contained in the control functions. To address these issues while maintaining the strengths of the elliptic grid generation method, the proposed approach is based on developing a finite element analog of the method. The finite element approach allows expressing the elliptic operator without assuming a global mapping between a logical space and the physical mesh coordinate system. However, it is not clear that the space metric may be approximated by using only information local to an element, as the metric tensor is explicitly defined using a global mapping. Results indeed show that when an approximation using only the local element metric is employed, the mesh is stationary. Thus, the proposed approach globalizes the metric approximation by considering information obtained from the location of the centroid of neighbor elements. This approach also allows the freedom to impose an external quality metric which adheres to a particular global strategy (e.g. to impose orthogonality metrics).

2. Mathematical basis

Unstructured grid generation was developed to support the use of the finite element method in structural modeling [1]. Formally, the distinction between structured and unstructured grids may be formulated by describing the connectivity type; i.e. by defining the connectivity between grid vertices. Following Frey and George [2], a grid is structured if its connectivity is of finite difference type; otherwise, the grid is unstructured. Stated differently, a planar domain on which a structured grid has been established has a global coordinate system (u, v) mapping a logical square in the (u, v)-plane into a region in the (x, y)-plane.

Both the theory of Riemann surfaces [3] and modern differential geometry emphasize the local properties of a differential manifold, which is a topological space resembling Euclidean space locally [4]. In this approach, for each point p of the manifold M, there exists a local homeomorphism, called a chart, from an open set in Euclidean space to an open neighborhood $U(p)$ of p. In many cases, such as geometrically complex domains discretized with an unstructured mesh, it may be intractable to define a global coordinate system as a basis for a mapping between logical and physical space. It is, however, generally possible to introduce local coordinates (u, v) in the neighborhood of a point. A collection of local coordinate systems,

such that their associated neighborhoods cover M, form an atlas. A differentiable manifold affords a definition of differentiable functions and of a local metric structure. A Riemannian manifold is a differentiable manifold on which a metric form is defined.

This development is limited to manifolds that are Riemann spaces, which may locally be described by a metric tensor $g_{\alpha\beta}(u,v)$. The metric tensor, through the underlying differential equations, controls the character of the grid. In the context of the finite element method, the local structure of the manifold was discussed in the article of Lautersztajn and Samuelsson [5]. The goal of this paper is to apply the finite element method to grid smoothing. The initial grid allows the estimation of the initial metric properties of the mesh, and provides a structure to compute the effect of neighboring elements on the metric tensor of the element under consideration.

For unstructured grid generation Knupp [6] applied the Winslow elliptic smoother using the finite difference approach. The finite element method was applied earlier by Allievi and Calisal [7] and Tipton [8], and additionally noted by Knupp. The results of Allievi and Calisal are restricted to orthogonal grids, in which the off-diagonal components of the metric tensor are neglected; furthermore, their shape factor, defined as the ratio of two remaining components of the metric tensor, is not related to the geometry of the neighboring elements. Unlike Tipton, who modifies the stiffness matrix by setting to zero the terms responsible for interaction between physical and unphysical modes, the algorithm proposed here is based on a geometric approach. The actual components of the metric tensor are replaced by the target components, derived from the interaction between the current element and its neighbors. Furthermore, the Winslow grid smoother is extended by including the derivative (curvature) terms of the metric tensor. As discussed in the next section, this is equivalent to employing the Thompson–Thames–Mastin grid generator.

3. Two-dimensional boundary value problem

Consider a planar domain D in Euclidean space (x^1,x^2), where (x,y) may be written as (x,y) and (u^1,u^2) expressed as (u,v). The domain D is defined locally in terms of parametric equations

$$\vec{x} = \vec{x}(u,v), \tag{1}$$

with a length element given as

$$\mathrm{d}s^2 = \mathrm{d}\vec{x}\cdot\mathrm{d}\vec{x} = \left(\vec{x}_u\,\mathrm{d}u + \vec{x}_v\,\mathrm{d}v\right)\cdot\left(\vec{x}_u\,\mathrm{d}u + \vec{x}_v\,\mathrm{d}v\right). \tag{2}$$

Eq. (2) may be written as

$$\mathrm{d}s^2 = E\,\mathrm{d}u^2 + 2F\,\mathrm{d}u\,\mathrm{d}v + G\,\mathrm{d}v^2, \tag{3}$$

where $E = (\vec{x}_u\cdot\vec{x}_u)$, $F = (\vec{x}_u\cdot\vec{x}_v)$, and $G = (\vec{x}_v\cdot\vec{x}_v)$. The coefficients E, F, and G of the first fundamental form are identified with the covariant components g_{11}, g_{12}, and g_{22} of the metric tensor [9]. The contravariant components $g^{\alpha\beta}$ are obtained by inverting the matrix $g_{\alpha\beta}$

$$g_{\alpha\gamma}g^{\gamma\beta} = \delta^{\beta}_{\alpha}, \tag{4}$$

where δ^{β}_{α} denotes the Kronecker delta symbol. Throughout, Einstein's summation convention is used, where summation over repeated indices is implied. Since $\mathrm{d}s^2$ is always positive, the determinant

$$g = g_{11}g_{22} - g_{12}g_{12} \tag{5}$$

is also positive.

The following employs the homogeneous Thompson–Thames–Mastin elliptic grid generator [10], which may be introduced in a general framework of the harmonic coordinates requirement. For $i = 1, 2$, x^i is a function of u^α, $\alpha = 1, 2$. The harmonic coordinates x^i are defined by the condition

$$\Delta x^i = 0, \tag{6}$$

i.e. as two solutions to the Laplace equation. In Eq. (6), the Laplacian operator acting on scalar variables x^i may be represented as [11]

$$\Delta x^i = \frac{1}{\sqrt{g}} \frac{\partial}{\partial u^\alpha} \left(\sqrt{g} g^{\alpha\beta} \frac{\partial x^i}{\partial u^\beta} \right). \tag{7}$$

A development based on harmonic coordinates generalizes naturally to three or more dimensions [12]. Additionally, this form of the governing system, in conjunction with the concept of local maps, leads directly to an implementation applicable to unstructured meshes.

When the right-hand side of Eq. (7) is expanded, using Eq. (6), one obtains

$$g^{\alpha\beta} \frac{\partial^2 x^i}{\partial u^\alpha \partial u^\beta} + \Delta u^\alpha \frac{\partial x^i}{\partial u^\alpha} = 0. \tag{8}$$

Here

$$\Delta u^\alpha = \frac{1}{\sqrt{g}} \frac{\partial}{\partial u^\beta} \left(\sqrt{g} g^{\alpha\beta} \right), \tag{9}$$

denotes the Laplacian operator acting on u^α.

Reverting to the notation (u, v), using subscripted variables x and y to express partial derivatives with respect to u and v, and further defining $P = \Delta u / g_{22}$ and $Q = \Delta v / g_{11}$ results in

$$g_{22}(x_{11} + P x_1) - 2 g_{12} x_{12} + g_{11}(x_{22} + Q x_2) = 0, \tag{10}$$

$$g_{22}(y_{11} + P y_1) - 2 g_{12} y_{12} + g_{11}(y_{22} + Q y_2) = 0. \tag{11}$$

Eqs. (10) and (11) define the Thompson–Thames–Mastin grid generator, with convective terms P and Q [13].

The system of Eq. (8) or, equivalently Eqs. (10) and (11), are approximated given the boundary conditions that prescribe the grid at the boundary ∂D of the domain D. In other words, it is necessary to solve Eq. (8) for x^i subject to

$$x^i = \hat{x}^i \tag{12}$$

on ∂D. The finite element method is based on a variational formulation of this boundary value problem. A detailed derivation of the general method is given by Becker et al. [14]. As noted by Reddy [15], one may write a weighted-integral form of a differential equation, even if it is not possible to construct a functional whose variation is equal to the variational form.

The method of weighted residuals is based on multiplying the left-hand side of Eq. (6) by a sufficiently smooth test function w, integrating this result over D, and then setting the weighted average equal to zero. The integration will be invariant with respect to coordinate transformations if an invariant area element $\sqrt{g}\,\mathrm{d}^2 u$ (where g has been introduced in Eq. (5)) is used [16]. With the aid of Eq. (7), this procedure gives the weak form of a boundary value problem

$$\int_D w \frac{\partial}{\partial u^\alpha} \left(\sqrt{g} g^{\alpha\beta} \frac{\partial x^i}{\partial u^\beta} \right) \mathrm{d}^2 u = 0. \tag{13}$$

On integrating by parts and using the Gauss divergence theorem, one obtains

$$\int_D \frac{\partial w}{\partial u^\alpha}\sqrt{g}g^{\alpha\beta}\frac{\partial x^i}{\partial u^\beta}\,\mathrm{d}^2u - \int_{\partial D} w\sqrt{g}g^{\alpha\beta}\frac{\partial x^i}{\partial u^\beta}\,\mathrm{d}s_\alpha = 0, \tag{14}$$

where $\mathrm{d}s_\alpha = n_\alpha\,\mathrm{d}s$ is a covariant oriented line element, and n^α are the components of the unit outward normal.

The essential boundary conditions, given by Eq. (12), enter the problem through the definition of the classes of admissible functions. In this development, the choice of test functions is limited to functions w that satisfy $w = 0$ on ∂D. The solutions of the variational boundary value problem are therefore the functions x^i satisfying

$$\int_D \frac{\partial w}{\partial u^\alpha}\sqrt{g}g^{\alpha\beta}\frac{\partial x^i}{\partial u^\beta}\,\mathrm{d}^2u = 0, \tag{15}$$

such that $x^i = \hat{x}^i$ on ∂D. These functions form the problem state vector.

4. Finite element approximation

In the Galerkin procedure [15], both the weight function w and the state vector x^i are represented as finite linear combinations of basis functions $\psi_n(u)$:

$$w(u) = \sum_{n=1}^{N} b_n\psi_n(u), \tag{16}$$

$$x^i(u) = \sum_{n=1}^{N} a_n^i\psi_n(u), \tag{17}$$

where b_n and a_n^i are the expansion coefficients. Since the b_ns are arbitrary, Eq. (15) implies an algebraic equation

$$\sum_{n=1}^{N} K_{mn}a_n^i = 0, \quad i = 1,2, \tag{18}$$

where the stiffness matrix K_{mn} is

$$K_{mn} = \int_D \frac{\partial\psi_m}{\partial u^\alpha}\sqrt{g}g^{\alpha\beta}\frac{\partial\psi_n}{\partial u^\beta}\,\mathrm{d}^2u. \tag{19}$$

When the region D is partitioned into finite elements $D = \{D_e\}$, one obtains the stiffness matrix associated with the element D_e

$$K_{mn}^e = \int_{D_e} \frac{\partial\psi_m^e}{\partial u^\alpha}\sqrt{g}g^{\alpha\beta}\frac{\partial\psi_n^e}{\partial u^\beta}\,\mathrm{d}^2u. \tag{20}$$

Here ψ_m^e and ψ_n^e represent restrictions of ψ_m and ψ_n to D_e. In the following, each cell D_e^x in the physical (x,y)-plane is viewed as a local image of a unit cell D_e^u in the (u,v)-plane generated by the mapping $x^i = x^i(u,v)$. For the quadrilateral grid, the cells in the (u,v)-plane are obtained as linear or quadratic maps of the square master element D_e^ξ in the (ξ,η)-plane satisfying $-1 \leqslant \xi \leqslant 1$, $-1 \leqslant \eta \leqslant 1$. An explicit form of this map may be

derived with the use of N_e shape functions for a master element with M nodes. For example, the bilinear shape functions are:

$$\begin{aligned}
\varphi_1(\xi,\eta) &= \frac{1}{4}(1-\xi)(1-\eta),\\
\varphi_2(\xi,\eta) &= \frac{1}{4}(1+\xi)(1-\eta),\\
\varphi_3(\xi,\eta) &= \frac{1}{4}(1+\xi)(1+\eta),\\
\varphi_4(\xi,\eta) &= \frac{1}{4}(1-\xi)(1+\eta).
\end{aligned} \tag{21}$$

If (u_m^e, v_m^e) and $m = 1,\ldots,M$ are the coordinates of the M vertices of D_e^u, the mapping of the master element onto D_e^u is given as

$$\begin{aligned}
u^e &= \sum_{m=1}^{M} u_m^e \varphi_m(\xi,\eta),\\
v^e &= \sum_{m=1}^{M} v_m^e \varphi_m(\xi,\eta).
\end{aligned} \tag{22}$$

In the case of an isoparametric map [14], the basis functions used in calculating the local approximate solution are the same as the shape functions used in defining the coordinate map, Eq. (22). Therefore, in this case, the basis functions in Eq. (20) may be identified with the shape functions introduced in Eq. (21), resulting in $M = N_e$. Due to tensorial character of Eq. (20), this form remains invariant when the variables of integration are changed from (u,v) to $(\xi^\alpha) = (\xi,\eta)$. This gives

$$K_{mn}^e = \int_{D_e^\xi} \frac{\partial\varphi_m}{\partial\xi^\alpha}\sqrt{g(\xi)}g^{\alpha\beta}(\xi)\frac{\partial\varphi_n}{\partial\xi^\beta}\,\mathrm{d}^2\xi. \tag{23}$$

In Eq. (23), $g^{\alpha\beta}(\xi)$ are the contravariant components of the metric tensor associated with the element D_e. From the mapping of the master element to the element D_e^x in the physical plane, one may obtain $g_{\alpha\beta}(\xi)$ [5]. In Fig. 1, D_e^x may be viewed schematically as an image of D_e^u or, alternatively, of D_e^ξ. Denoting the coordinates of the vertices defining D_e^x by (x_m^e, y_m^e), one obtains

$$\begin{aligned}
x^e &= \sum_{m=1}^{M} x_m^e \varphi_m(\xi,\eta),\\
y^e &= \sum_{m=1}^{M} y_m^e \varphi_m(\xi,\eta).
\end{aligned} \tag{24}$$

With the notation $\vec{r}_m^e = (x_m^e, y_m^e)$, the covariant components of the metric tensor associated with D_e become

$$g_{\alpha\beta}^e = \sum_{m=1}^{M}\sum_{n=1}^{M} \vec{r}_m^e \cdot \vec{r}_n^e \frac{\partial\varphi_m}{\partial\xi^\alpha}\frac{\partial\varphi_n}{\partial\xi^\beta}, \tag{25}$$

which allows the evaluation of the stiffness matrix for element D_e.

This section provides the explicit formulas for the stiffness matrix assuming a quadrilateral grid. When the mesh elements are triangular or of mixed type, a similar derivation may be performed by using the

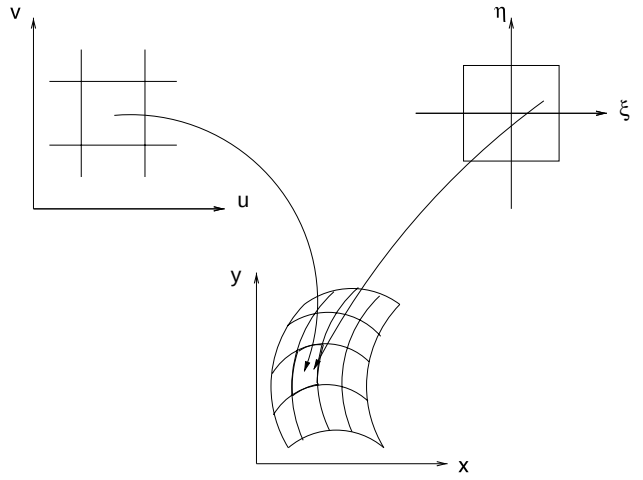

Fig. 1. An element D_e in the physical plane (x, y) viewed as an image under linear mapping of an element D_e^u in the (u, v)-plane or, alternatively, of the master element D_e^ξ in the (ξ, η)-plane.

isoparametric basis functions relevant to triangular elements. For the sake of completeness, one may express linear shape functions as:

$$\begin{aligned} \chi_1(\xi, \eta) &= 1 - \xi - \eta, \\ \chi_2(\xi, \eta) &= \xi, \\ \chi_3(\xi, \eta) &= \eta, \end{aligned} \tag{26}$$

which replace Eq. (21). In numerical computations, both linear and quadratic shape functions were employed. In the quadratic approximation, nine-node shape functions were used for quadrilaterals and six-node shape functions were used for triangular elements. Their explicit form may be found in [14,15].

The global stiffness matrix K_{mn} is obtained from the local stiffness matrices K_{mn}^e corresponding to element D_e through the assembly process. Algorithmically, this is implemented by defining a connectivity array C_{ek}, which lists the vertices having coordinates (x_k^e, y_k^e) associated with element indexed as e. Given the nodes m and n, the element K_{mn} of the stiffness matrix is then a sum of contributions arising from the elements comprising m and n. Schematically, this is written as

$$K_{mn} = \sum_e K_{mn}^e, \tag{27}$$

where the sum extends over the elements that contain m and n. The approximate grid coordinates are obtained by solving Eq. (18), subject to a boundary node Dirichlet condition (all nodes on domain boundaries are fixed at the given initial coordinates).

5. Metric tensor

The evaluation of the element stiffness matrix as given by Eq. (23), involves the evaluation of a local metric tensor on the manifold. As implied by Eq. (25), $g_{\alpha\beta}^e$ is known only within an element. Given an initial grid, Eq. (25) allows one to compute $g_{\alpha\beta}^e$ consistent with the grid. The grid thus obtained by solving Eq. (18) does not differ from the initial grid. Stated differently, with the metric tensor determined by the initial mesh,

no grid smoothing is accomplished. There are at least two ways out of this predicament. First, one may distinguish between the current and a target metric. Whereas the current metric is computed from the unsmoothed initial grid, the target metric may involve the consideration of ancillary information, such as the orthogonality condition $g^e_{\xi\eta} = 0$. Second, by analogy with various structured grid generators, one may employ a metric prescription resulting from a predictive procedure involving several elements. This approach is conceptually similar to the reference grid method employed by Castillo et al. [17].

Essentially, this method computes the components of the metric tensor on a dual mesh. Given an arbitrary quadrilateral element D_e, consider the line segments connecting the center O of D_e with the four centers of its neighbors, as shown in Fig. 2. The four nodes denoted by 1, 2, 3, and 4 correspond to the points $(-1,-1)$, $(1,-1)$, $(1,1)$, and $(-1,1)$ of the master element in the (ξ,η) space. The original definition of the metric tensor components, Eq. (25), identifies (for bilinear shape functions) the component g_{11} at the center of D_e with the square of the arithmetic average of vectors $\overline{1,2}$ and $\overline{4,3}$. This observation suggests a modified averaging procedure of the form

$$g_{11} = \left(\frac{1}{2}\overline{P_4P_2}\right)\cdot\left(\frac{1}{2}\overline{P_4P_2}\right). \tag{28}$$

The notation in Eq. (28) implies that one half of the vector $\overline{P_4P_2}$ is used to form a scalar product with itself. Similarly

$$g_{12} = \left(\frac{1}{2}\overline{P_4P_2}\right)\cdot\left(\frac{1}{2}\overline{P_1P_3}\right), \tag{29}$$

$$g_{22} = \left(\frac{1}{2}\overline{P_1P_3}\right)\cdot\left(\frac{1}{2}\overline{P_1P_3}\right). \tag{30}$$

The topology of a triangular element requires a slight modification of the outlined procedure. Again, the nodes 1, 2, and 3 (see Fig. 3) correspond to the points $(0,0)$, $(1,0)$, and $(0,1)$ of the master element in the (ξ,η) space. The components of the metric tensor at the center of D_e now become

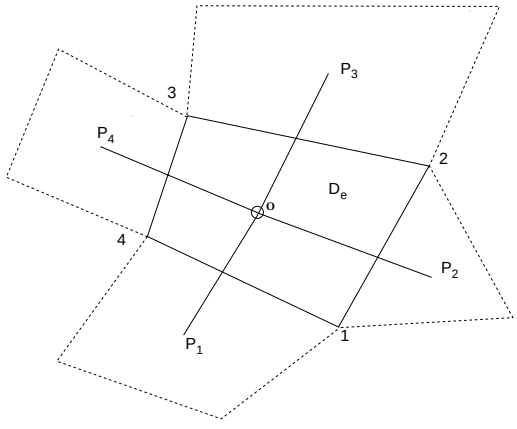

Fig. 2. The vertices 1, 2, 3, and 4 of a quadrilateral correspond to the local labeling (coordinate system) of element D_e. Given the locations P_1, P_2, P_3, and P_4 of the centers of the neighbors of D_e, drawn schematically using dashed lines, the components of the metric tensor are obtained by forming appropriate scalar products of the vectors $\overline{P_4P_2}$ and $\overline{P_1P_3}$.

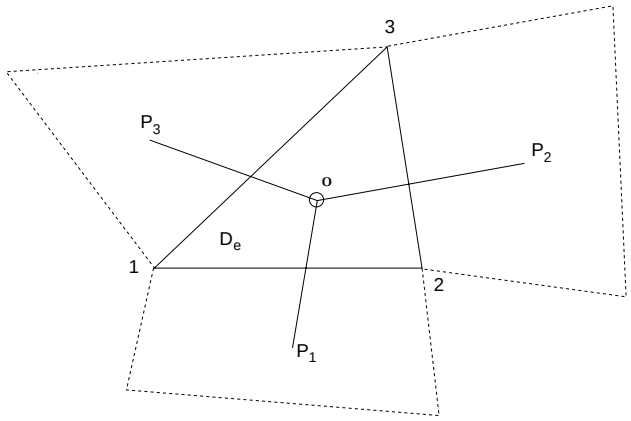

Fig. 3. The vertices 1, 2, and 3 of a triangle correspond to the local labeling (coordinate system) of element D_e. The centers of the neighbors of D_e, drawn schematically using dashed lines, are denoted by P_1, P_2, and P_3.

$$\begin{aligned} g_{11} &= \overline{OP_1} \cdot \overline{OP_1}, \\ g_{12} &= \overline{OP_1} \cdot \overline{OP_3}, \\ g_{22} &= \overline{OP_3} \cdot \overline{OP_3}. \end{aligned} \tag{31}$$

In a sense, the metric tensor computed from the smoothing procedure may be considered as a target metric tensor, in contrast to the current metric tensor computed from the actual grid. In some situations, when dealing with triangular elements, a smoother grid may be achieved by identifying the target components of the metric tensor with the components corresponding to an equilateral triangle.

As outlined previously, the finite element system (Eq. (18)) is solved, given the above metric prescription. This system also requires boundary conditions for the state vector, x and y. In this paper, the domain boundary nodal coordinates are fixed at their initial locations (Dirichlet condition). When prescribing the metric, it is also necessary to provide a boundary condition on the metric tensor at the domain boundaries. Three types of behavior at the boundary are potentially useful:

1. Fixed boundary spacing to some input δ (may be a function of the parametric location on the boundary). This is a Dirichlet condition on the metric along the boundary.
2. Specification of coordinate line slope at the boundary (i.e. the grid should meet the boundary normal to the bounding geometry). This is a Neumann condition on the metric at the boundary.
3. Continuous metric at the boundary. This condition states that the metric curvature slightly "outside" the boundary should be continuous with the curvature just inside the boundary.

Combinations of these possibilities might also be useful, for example specifying δ to indicate the thickness of the boundary row of cells (a Dirichlet value for g_{11}), specifying that the grid should be orthogonal at the boundary (a Dirichlet $g_{12} = 0$), and specifying that the cell height should "float" (a Neumann g_{22}).

6. Solution procedure

If the metric tensor $g_{\alpha\beta}$ is expressed in terms of the basis functions as in Eq. (25), then Eqs. (6) and (7) may be expressed as a linear system of the form

$$K\vec{u} = \vec{f}, \tag{32}$$

where $\vec{u} = (x_1, y_1, \ldots, x_N, y_N)$ denotes the nodal state vector, and $\vec{f}$ arises from the imposed Dirichlet boundary conditions. This linear system may be solved for the coordinates x and y in the most convenient manner. As was discussed in the previous section, this solution method does not change the mesh state, as the target metric is simply the current metric. This result verifies that the descriptive metric is quite effective in capturing the coordinate transformation [8]; the solution $\vec{u}$ is always precisely the current mesh state.

To accomplish smoothing of the mesh, it is necessary to prescribe the target element metric. In this case, the metric tensor $g_{\alpha\beta}$ is dependent on the mesh geometry as the current state of the mesh is used to formulate the equidistribution strategy. As such, the global problem is a quasi-linear elliptic system, with the non-linearity arising from the coupling between the target element metric and the solution state vector. In this case

$$K_{mn} = \sum_e K^e_{mn} = \sum_e \sqrt{\tilde{g}_e(\vec{u})}\tilde{g}_e^{\alpha\beta}(\vec{u}) \int_{D_e^\xi} \frac{\partial \varphi_m}{\partial \xi^\alpha} \frac{\partial \varphi_n}{\partial \xi^\beta} \, \mathrm{d}^2\xi, \tag{33}$$

where $\tilde{g}_e^{\alpha\beta}$ are the components of the target contravariant metric tensor for element e, and $\tilde{g}_e$ is the determinant of the covariant target metric tensor. These target tensor components (Eqs. (28)–(30)) are removed from the element integral as they are not a function of ξ within the element.

Given this form of the stiffness matrix, one approach is to linearize the system by evaluating the target metric on the current mesh, then solving for x and y. This method is iterative, as $\tilde{g}_e$ is only a rough approximation for the ultimate target metric. In this case, the assembly matrix is evaluated on the current mesh for the current iteration i

$$K^i_{mn} = \sum_e \left(\sqrt{\tilde{g}_e}\tilde{g}_e^{\alpha\beta} \right)^i \int_{D_e^\xi} \frac{\partial \varphi_m}{\partial \xi^\alpha} \frac{\partial \varphi_n}{\partial \xi^\beta} \, \mathrm{d}^2\xi. \tag{34}$$

This approach solves the linear system

$$K^i \vec{u}^{i+1} = \vec{f}^{\,i}, \quad i = 0, \ldots, M-1, \tag{35}$$

where M is the number of iterations required to converge the system, and $\vec{u}^0$ is the initial mesh state. In practice, however, the convergence rate of this approach is generally unacceptable and it tends to further degrade with increasing problem size.

A more effective method to solve this system is through the use of a Newton iterative method on the non-linear finite element problem [18], where the target metric is computed implicitly within the solution procedure. This approach is based on solving a system of non-linear algebraic equations

$$\vec{F}(\vec{u}) = \begin{pmatrix} F_1(x_1, y_1, \ldots, x_N, y_N) \\ \vdots \\ F_N(x_1, y_1, \ldots, x_N, y_N) \end{pmatrix} = 0, \tag{36}$$

where

$$F_m(\vec{u}) = \sum_e \sqrt{\tilde{g}_e(\vec{u})}\tilde{g}_e^{\alpha\beta}(\vec{u}) \sum_n \int_{D_e^\xi} \frac{\partial \varphi_m}{\partial \xi^\alpha} \frac{\partial \varphi_n}{\partial \xi^\beta} \, \mathrm{d}^2\xi u_n - f_m. \tag{37}$$

Given this form, the Newton–Krylov solution procedure is based on using a Krylov subspace method to solve the linear system

$$J^i \delta \vec{u}^i = -\vec{F}(\vec{u})^i, \tag{38}$$

where

$$J_{kl} = \frac{\partial F_k(\vec{u})}{\partial u_l} \tag{39}$$

is the Jacobian matrix, and $\vec{F}(\vec{u})^i$ is the residual vector for the ith iterate. The solution may be advanced to the next iteration using the expression

$$\vec{u}^{i+1} = \vec{u}^i + \delta\vec{u}^i. \tag{40}$$

Recognizing that the implementation of the Jacobian operator is simplified in this application if the order of the finite element assembly process and the differentiation operator are interchanged [19], it is possible to rewrite Eq. (39) as

$$J_{kl} = \sum_e J^e_{kl} = \sum_e \frac{\partial F^e_k(\vec{u})}{\partial u_l}, \tag{41}$$

where

$$F^e_m(\vec{u}) = \sqrt{\tilde{g}_e(\vec{u})}\tilde{g}_e^{\alpha\beta}(\vec{u}) \sum_n \int_{D^\xi_e} \frac{\partial\varphi_m}{\partial\xi^\alpha}\frac{\partial\varphi_n}{\partial\xi^\beta}\,\mathrm{d}^2\xi u_n - f_m. \tag{42}$$

As a stopping criterion, the non-linear iteration is halted when the residual vector is sufficiently small. In this implementation, the convergence criteria is $\|\vec{F}^i(\vec{u})\|_2/\|\vec{F}^0(\vec{u})\|_2 \leqslant 10^{-6}$, where $\vec{F}^0(\vec{u})$ is the residual vector on the initial "guess". A block preconditioner toolkit, BPKIT [20], was used to solve the linear system (38). For all the results presented here, the toolkit's incomplete LU factorization option ILU(1) was used as the local preconditioner, with 2 passes of symmetric successive over-relaxation (SSOR) used as a global preconditioner. There were 16 blocks used in the preconditioner, and the toolkit's flexible general minimal residual linear solver (FGMRES) was used as the Krylov solver.

7. Sample results

This section examines the application of the finite element procedure to mesh smoothing. As indicated in the previous section, the finite element method, based on the local metric tensor introduced by Eq. (25), essentially reproduces the initial mesh. When a target metric is constructed for element D_e that uses information from neighboring cells, motion of the mesh nodes will result. The approach proposed above essentially equidistributes the metric values g_{11}, g_{12}, and g_{22} between the element being considered and its neighbors.

As an illustration of these considerations, this smoothing method was applied to some non-trivial grids from the Rogue's gallery of Knupp and Steinberg [9]. In all of these experiments, the initial grid was a transfinite interpolation (TFI) grid, which may be readily obtained for plane geometries. Fig. 4 illustrates the TFI mesh for a rectangular horseshoe domain. The grid on the boundary curves provides the boundary condition $\hat{x}^i$ for the solution.

To smooth the quadrilateral grids in this example set, the method constructs a set of ghost elements outside the domain by extending lines perpendicular to the boundary at each boundary node point. These extensions are of length δ, a measure of the desired element width at the boundary. For this case, an average element "width" was computed by examining all the boundary elements. In general, δ may be a function of the parametric location on the boundary curve. These ghost elements are completed by interconnecting the endpoints of the δ-length segments, forming quadrilateral ghost cells. The smoothing algorithm then proceeds by calculating the target metric for each of the participating elements, and solving the above quasi-linear system. This procedure results in the mesh shown in Fig. 5.

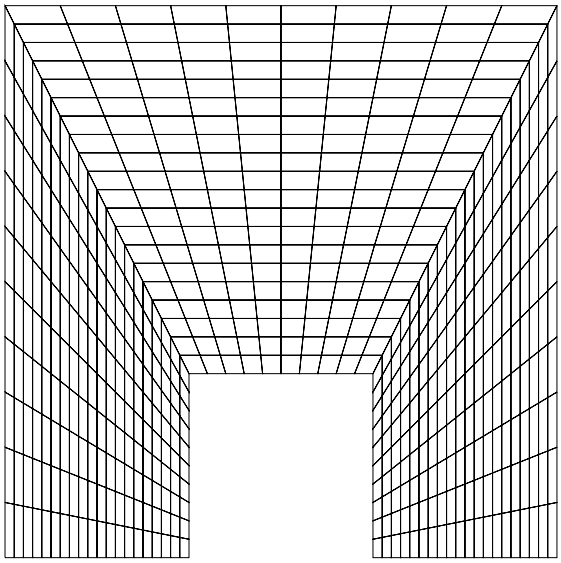

Fig. 4. The TFI grid for a rectangular horseshoe.

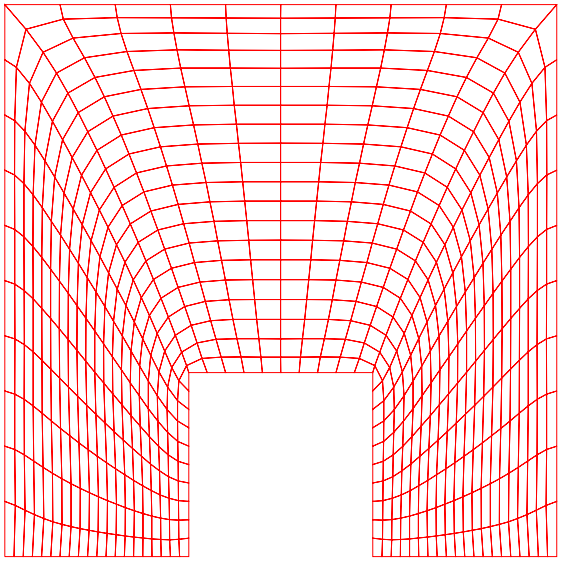

Fig. 5. The smoothed grid for a rectangular horseshoe obtained from the finite element method.

The difficulty in smoothing a mesh like this horseshoe geometry lies in the large curvature at the upper-right and upper-left external corners of the domain. Many popular methods (including Winslow) do not capture this curvature properly, and "pull" the mesh away from these corners (cf. Knupp and Steinberg [9, pp. 201–202]). Furthermore, several contrasting methods fold the mesh over the internal obstacle.

Additional experiments involving a chevron grid are shown in Figs. 6 and 7. The non-convex nature of the grid domain, apparent in the middle-right portion of Fig. 6, is again a source of difficulty in generating

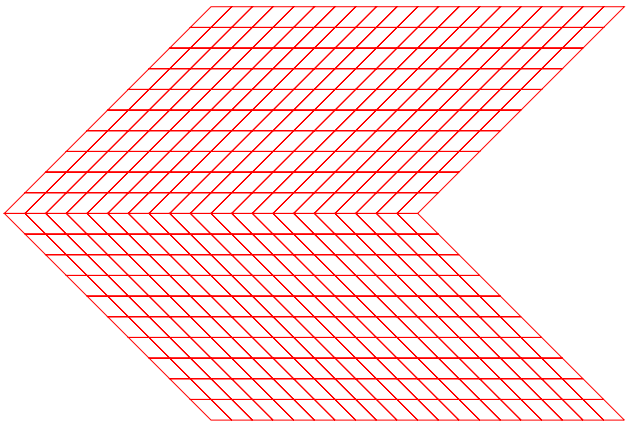

Fig. 6. TFI generation of a chevron grid from the Rogue's gallery of Knupp and Steinberg.

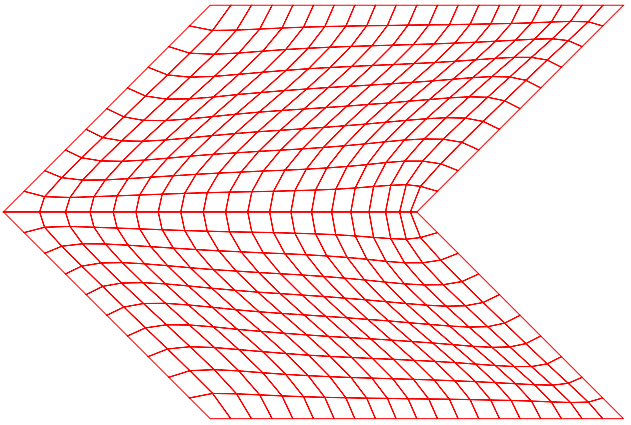

Fig. 7. The smoothed chevron grid obtained from the finite element method.

the grid on this problem. As an aside, the results of the finite element method are remarkably similar to the results of the area and length method [9]. Again, many contrasting methods were not successful on this problem, as shown in [9, pp. 271–273].

The results on a swan grid are depicted in Figs. 8 and 9. The swan is challenging due to the presence of a concave boundary (along the top of the domain) and a convex boundary (along the right of the domain) in close proximity. Again, most contrasting methods "pull" the mesh away from the convex boundary, and "pack" the mesh towards the concave boundary (or fold it over the top), as can be seen in [9, pp. 269–270].

These examples serve to indicate that the finite element method provides qualitative results of comparable or superior quality to popular structured methods detailed in [9].

Figs. 10 and 11 show the performance of the finite element smoother on a more complex domain, a cross section of a single cylinder overhead valve engine. This example is notable due to its domain complexity, both in the number of components within the domain and that it contains both convex and concave parts. Additionally, this problem has a large number of elements with a non-trivial internal boundary structure. Fig. 10 is the initial TFI mesh, which is of reasonable quality. The lower figure shows the smoothed mesh.

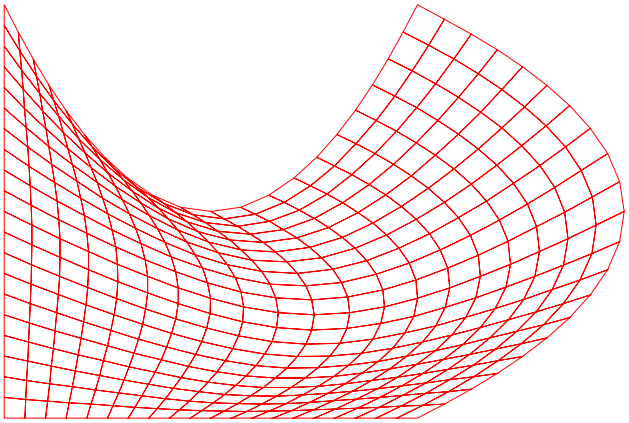

Fig. 8. TFI of a swan grid from the Rogue's gallery of Knupp and Steinberg.

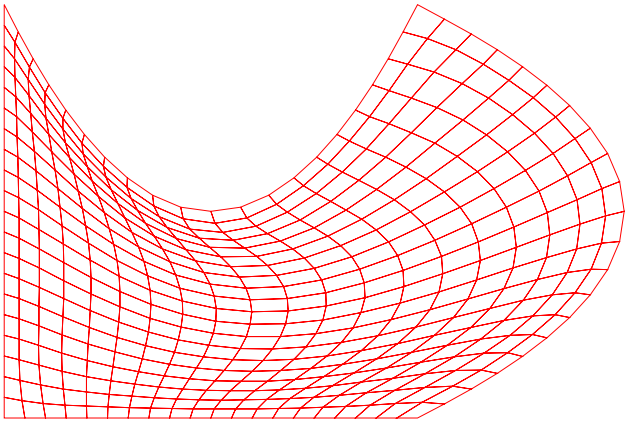

Fig. 9. The smoothed swan grid obtained from the finite element method.

In this illustration, the quality improvements are particularly evident near the large and small ends of the connecting rod, and near the intake and exhaust ports.

The above examples are all structured, quadrilateral grids; effective smoothing could have been achieved without any reliance on the finite element method and its underlying complexity. It is important to show that the method is effective on structured meshes; however, this method's development was motivated by the desire to effectively smooth meshes without a convenient global structure. Towards this goal, an inherently unstructured grid was generated by randomly selecting 100 points in a square and applying Delaunay triangulation to obtain the mesh. This approach used the triangular mesh generator *Triangle*, advanced by Shewchuk [21]. Fig. 12 shows the mesh generated by the Triangle algorithm; the smoothed version is illustrated in Fig. 13.

The triangle meshes were smoothed by specifying $g_{12} = 0.5\sqrt{g_{11}g_{22}}$. Furthermore, Neumann values of g_{11} and g_{22} were used by the creation of a ghost element outside the boundary that reflects the values of the neighbor element within the domain.

Again using *Triangle*, a planar grid in a region enclosed by an outline of Lake Superior was generated (Fig. 14). In this example, the Triangle mesh was of rather high quality, so certain nodes of this mesh were

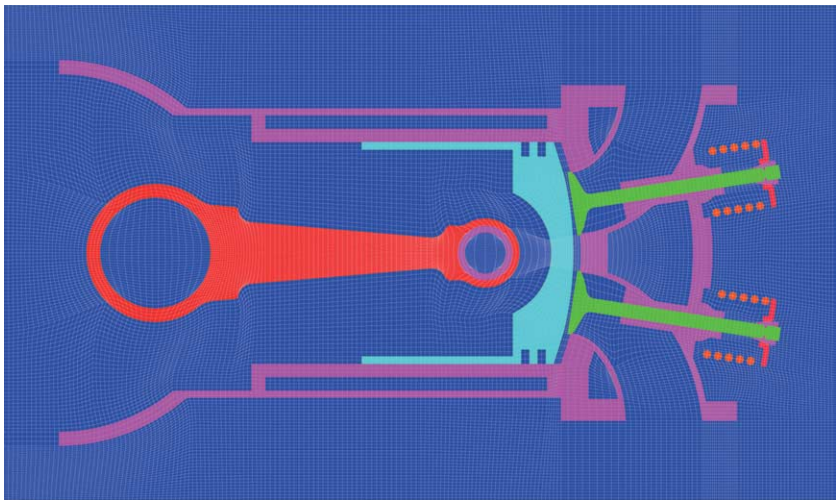

Fig. 10. The TFI grid for an engine prototype.

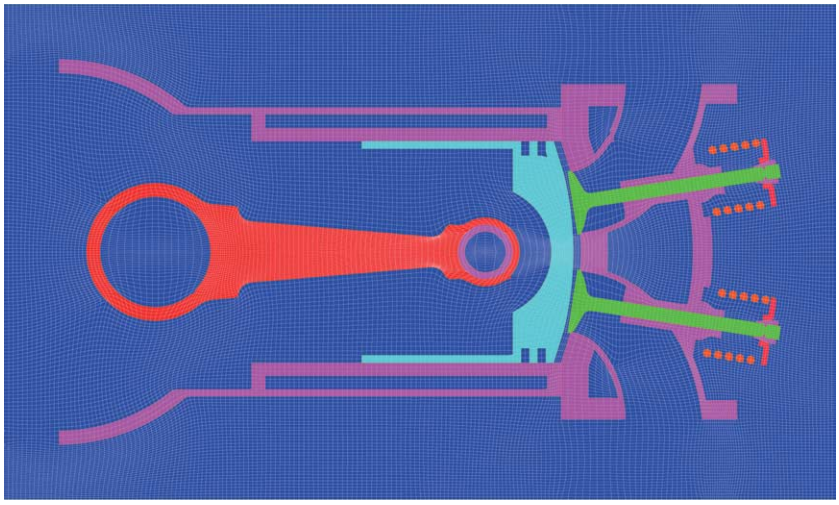

Fig. 11. Smoothed grid for the engine prototype.

relocated to lower the quality of the initial mesh for the smoother. One of these points may be seen to the left of Isle Royale (the large island near the center of the lake, 1/3 from the top). The smoothed version of this grid is shown in Fig. 15.

As a measure of mesh quality, the smoothness functional discussed by Knupp et al. [22] is adopted. Using the notation of Section 2, the smoothness functional reads

$$I[\xi,\eta] = \frac{1}{2}\int_D g^{\alpha\beta}(u)\left(\frac{\partial\xi}{\partial u^\alpha}\frac{\partial\xi}{\partial u^\beta}+\frac{\partial\eta}{\partial u^\alpha}\frac{\partial\eta}{\partial u^\beta}\right)\sqrt{g(u)}\,\mathrm{d}^2u. \tag{43}$$

It is customary to invert the variables in the smoothness functional by regarding the variables (ξ^1,ξ^2) as independent. In the framework of the tensor calculus, this inversion can be accomplished by employing the

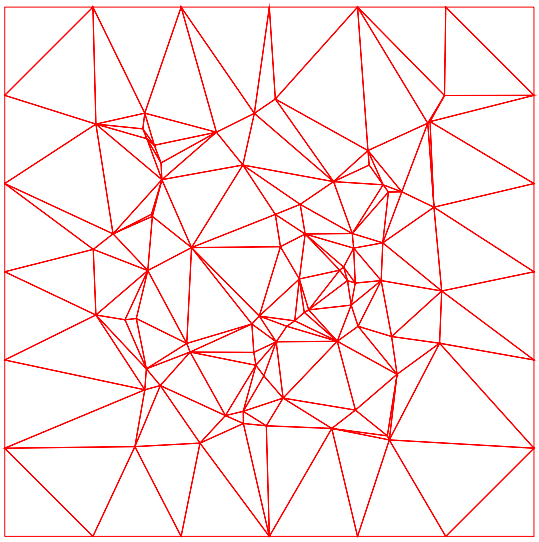

Fig. 12. A triangular grid generated by Delaunay triangulation using 100 random points.

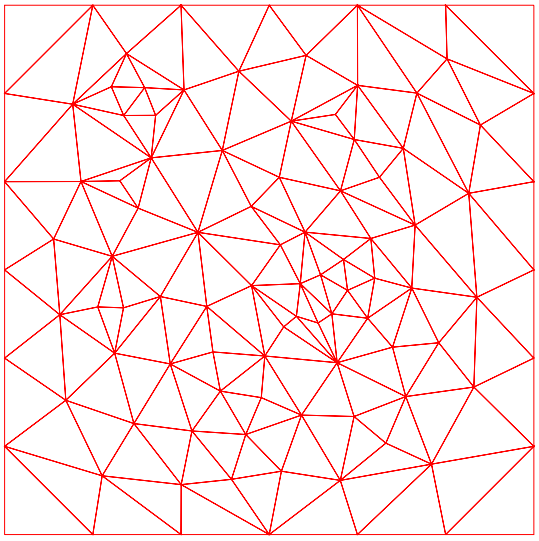

Fig. 13. The smoothed triangular grid from the finite element method.

relations for the transformation of the contravariant metric tensor $g^{\alpha\beta}$. Using the same symbol for the components of the metric tensor in two coordinate systems, one can write

$$g^{\alpha\beta}(\xi) = \frac{\partial \xi^{\alpha}}{\partial u^{\mu}} \frac{\partial \xi^{\beta}}{\partial u^{\nu}} g^{\mu\nu}(u). \tag{44}$$

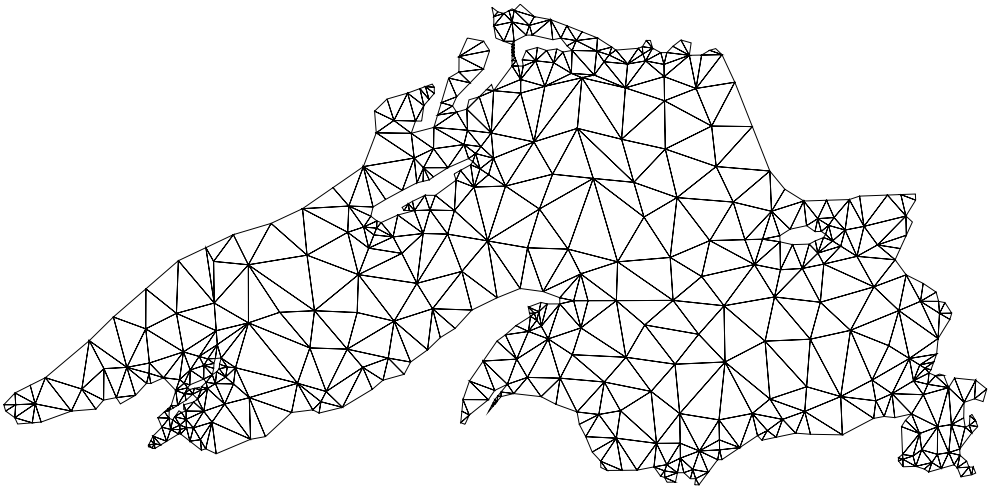

Fig. 14. A triangular grid of Lake Superior generated with *Triangle*.

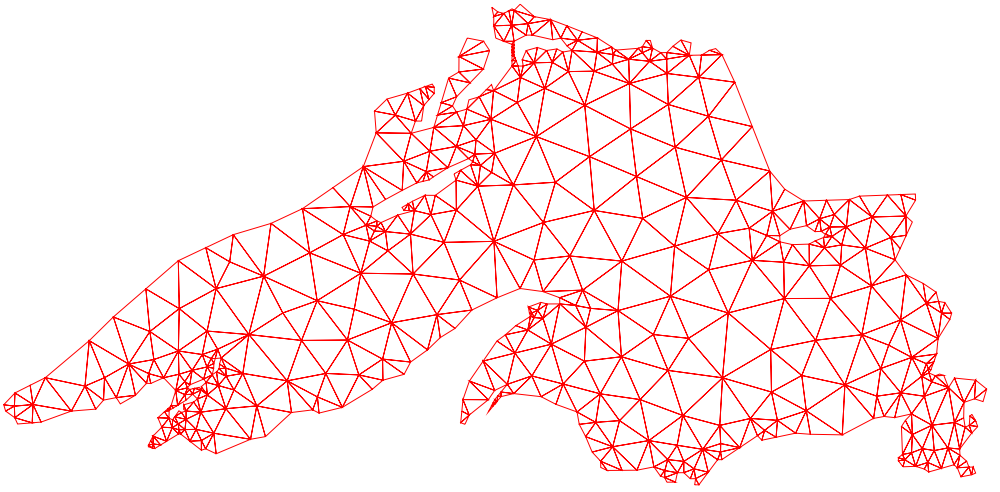

Fig. 15. The smoothed triangular grid of Lake Superior.

Eq. (43) now becomes

$$I[\vec{x}] = \frac{1}{2}\int_D \left(g^{11}(\xi) + g^{22}(\xi)\right)\sqrt{g(\xi)}\,\mathrm{d}^2\xi, \tag{45}$$

where the covariant components of the metric tensor are

$$g_{\alpha\beta}(\xi) = \frac{\partial\vec{x}}{\partial\xi^\alpha}\cdot\frac{\partial\vec{x}}{\partial\xi^\beta}. \tag{46}$$

The values of the smoothness functional, as a function of the iteration number, have been normalized to unity by dividing the successive values by the values corresponding to the initial mesh. Table 1 lists the normalized values for the cases considered in this section.

Table 1
Value of the normalized smoothness functional (NSF) vs. Newton iteration number for the examples considered

Grid	Iteration number					
	1	2	3	4	5	6
Horseshoe	1.0000	0.9370	0.8591	0.8514	0.8507	0.8506
Chevron	1.0000	0.9993	0.9299	0.9272	0.9257	0.9257
Swan	1.0000	0.9063	0.8766	0.8760	0.8762	0.8762
Triangular	1.0000	0.5738	0.5738	0.5738	0.5738	0.5738
Lake Superior	1.0000	0.9117	0.9116	0.9116	0.9116	0.9116
Engine	1.0000	0.9928	0.9946	0.9950	0.9950	0.9950

In all of the examples, the finite element smoother increased the quality of the input mesh, based on the smoothness functional metric. The magnitude of the quality increase appears to be a function of element type (triangle vs. quadrilateral) and the quality of the initial mesh input to the finite element procedure. As such, it is difficult to quantify the amount and rate of quality improvement in general; i.e. one must carefully compare methods given a complete description of the specific input mesh. Given these example results, the "fastest" approach to a smooth grid is achieved for spatially isotropic configurations, such as the random collection of triangles.

The "quality" of a particular mesh is generally a strong function of the particular application that the mesh is intended for. It is often a requirement that a particular mesh be free of various "computational issues". For example, on the horseshoe domain, it is possible to obtain a rather "smooth" mesh using the Winslow method. However, the mesh may retract from the concave sections of the geometry to the extent that the application using the mesh could not compute on the result. Certainly, an unacceptable computational issue would be the presence of one or more elements with a negative cell Jacobian (inverted cell). It is therefore quite important that the smoothing method prevents the formation of inverted cells if possible. In fact, this requirement generally overwhelms the need for a mesh to rigorously satisfy some global geometric or equidistribution principle.

The existence of an extremum principle is a sufficient condition for a mesh optimization system to exhibit a one-to-one mapping in the final mesh [23]. The proposed finite element method reduces to a Poisson system with mesh control functions (see Eqs. (10) and (11)). It is therefore unlikely that this method meets this condition. Certainly, it is possible to select a target element metric prescription that will invert one or more mesh cells. The suggested metric prescription is effective on the class of examples provided in this section.

Fig. 16 illustrates the non-linear convergence behavior of the Newton solver on the example set. The dashed curve without data markers is a reference indicating the line of logarithmic residual reduction. The solver appears to converge somewhat better on the unstructured triangular meshes. However, the results on the quadrilateral meshes are certainly acceptable. Given this limited data, the method appears to be insensitive to mesh size (the engine mesh is 150 by 150 nodes, the chevron is 20 by 20 nodes). This illustration shows the residual history out to nine Newton iterations for completeness. However, the stopping criterion is generally a residual norm below 10^{-6}. For this criterion, a worst-case of six Newton iterations were necessary to achieve convergence of the method (this, on the engine example).

Table 2 compares the proposed smoothing method to classic Laplacian smoothing [6]. The Laplace method uses point Jacobi relaxation to solve the unstructured analogue of the system

$$\begin{aligned} x_{\xi\xi} + x_{\eta\eta} &= 0, \\ y_{\xi\xi} + y_{\eta\eta} &= 0. \end{aligned} \tag{47}$$

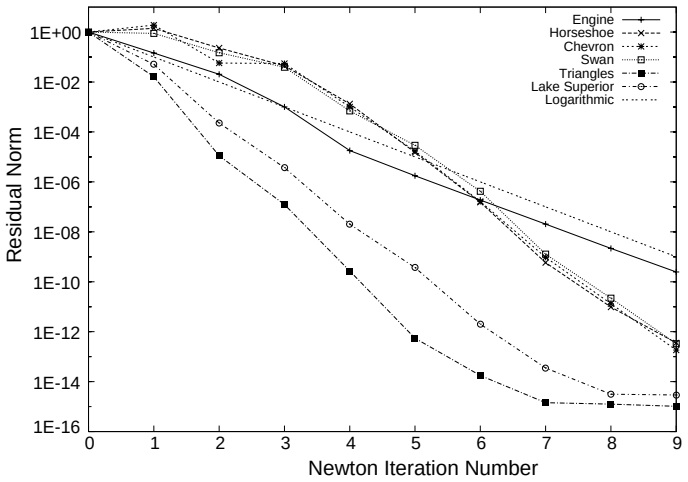

Fig. 16. Newton's method convergence plot of the example set.

Table 2
Comparison of the finite element-based method and standard Laplace smoothing for the example set

Grid	CPU (ms)		NSF metric	
	Laplace	FEM	Laplace	FEM
Horseshoe	–	5265	–	0.8507
Chevron	–	4297	–	0.9257
Swan	–	4687	–	0.8762
Triangular	15	156	0.9512	0.5738
Lake Superior	31	469	0.9949	0.9117
Engine	2173	147,100	1.0270	0.9950

Entries marked with "–" indicate failure of the method (see text).

In the table, the first column compares CPU time in milliseconds on a Pentium 2.2 GHz PC to achieve a "converged" normalized smoothness functional. Convergence, in this context, means that the normalized smoothness functional has converged to four significant figures. The second column compares the normalized smoothness functional for the two approaches on the example set. The table entries marked by "–" signify failure of the Laplace method exhibited by folding of the grid near highly curved boundaries. As folding creates negative area elements, the smoothness metric is undefined for these examples (thus the time to convergence is also undefined).

Laplacian smoothing is known for both its computational efficiency and low quality results. When Laplacian smoothing did not result in a folded mesh, it was from ten to nearly 100 times faster than the finite element approach. However, in every case, it did not provide the same level of mesh quality as defined by the normalized smoothness functional. In the case of the engine test problem, Laplacian smoothing actually *lowered* the quality of the input mesh (generated using block transfinite interpolation).

Fig. 17 illustrates the folded mesh obtained from Laplacian smoothing on the horseshoe example problem. The behavior on the chevron and swan problems were similar. Clearly, this would not be a useful approach on problems of this type. Negative area elements are created where the mesh "spills" outside the domain at the corners of the internal obstacle. Furthermore, the mesh "pulls" away from the convex

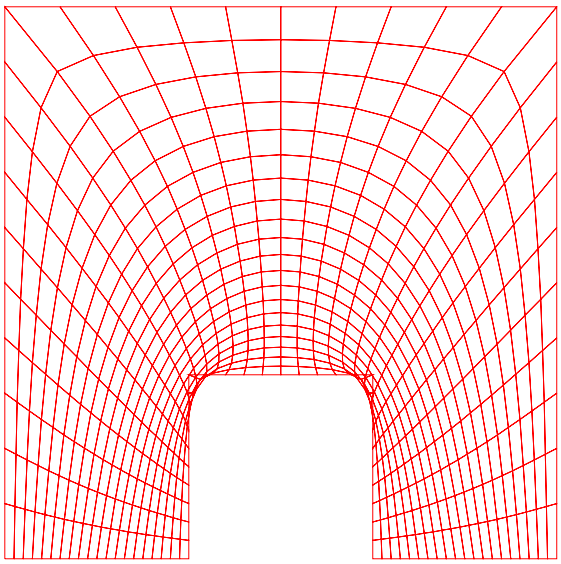

Fig. 17. The smoothed grid for the rectangular horseshoe obtained from simple Laplacian smoothing.

corners at the upper right and left of the problem. Qualitatively comparing this result with the horseshoe generated by the finite element approach (Fig. 5) clearly illustrates the superiority of the finite element approach for this problem. To be fair, however, the structured mesh examples were specifically chosen by the authors to challenge popular smoothing methods.

For the two unstructured examples (Triangular and Lake Superior), Laplacian smoothing was more competitive. The quality was not as high as the finite element approach but the CPU speed advantage may offset this consideration, depending on the application.

8. Conclusions

This paper proposes a finite-element approximation to solve an elliptic system of equations for generating grid coordinates. This methodology, based on finding a solution to a system of elliptic partial differential equations, appears to be fully effective for both structured and unstructured grids. Furthermore, the approach is general in nature. It is not necessary to include a distinction within the method for each mesh type. As developed, it was not necessary to specify domain parameters beyond the problem geometry for the examples considered. Further work remains in developing more rigor in the estimation of the target element metric. It is clear that there is considerable flexibility in defining the target metric, perhaps enough to extend the methodology to anticipate evolving solution features and to incorporate other optimization criteria important to the solution application.

To place this development in proper context, it may be viewed as supplementing optimization methods, discussed by Carey [24], in which the mesh redistribution is achieved through an optimization problem with objective functions based on Dirichlet integrals. Future activities would also likely include the use of objective functions to prescribe (or refine) the local element metric to further enhance final mesh quality measures.

A generalization of this method to three-dimensional grids and surface mesh generation is also indicated. These extensions, albeit computationally involved, are conceptually straightforward. They also will be a subject of future investigation.

Acknowledgements

This work was performed under the auspices of the US Department of Energy by Los Alamos National Laboratory under Contract W-7405-ENG-36 (LA-UR-02-7232).

References

[1] J.F. Thompson, N.P. Weatherill, Fundamental concepts and approaches, in: J.F. Thompson, B.K. Soni, N.P. Weatherill (Eds.), Handbook of Grid Generation, CRC Press, Boca Raton, FL, 1999, pp. 1–30 (Chapter 1).
[2] P.J. Frey, P.-L. George, Mesh Generation: Application to Finite Elements, Hermes Science, Oxford, 2000.
[3] G. Springer, Introduction to Riemann Surfaces, Chelsea, New York, 1981.
[4] T. Frankel, The Geometry of Physics: An Introduction, Cambridge University Press, Cambridge, 1997.
[5] N. Lautersztajn, A. Samuelsson, On application of differential geometry to computational mechanics, Comput. Methods Appl. Eng. 150 (1-4) (1997) 25–38.
[6] P.M. Knupp, Winslow smoothing on two-dimensional unstructured meshes, in: Seventh International Meshing Roundtable, Sandia National Laboratory, 1998, pp. 449–457.
[7] A. Allievi, S. Calisal, Application of Bubnov–Galerkin formulation to orthogonal grid generation, J. Comput. Phys. 98 (1) (1992) 163–173.
[8] R.E. Tipton, Grid optimization by equipotential relaxation, Tech. Rep., Lawrence Livermore National Laboratory, July 1992.
[9] P. Knupp, S. Steinberg, Fundamentals of Grid Generation, CRC Press, Boca Raton, FL, 1994.
[10] J.F. Thompson, F.C. Thames, C.W. Mastin, Automatic numerical generation of body-fitted curvilinear coordinate system for field containing any number of arbitrary two-dimensional bodies, J. Comput. Phys. 15 (1974) 299–319.
[11] D.J. Struik, Lectures in Classical Differential Geometry, Dover, New York, 1961.
[12] V. Fock, The Theory of Space Time and Gravitation, Pergamon Press, New York, 1959.
[13] J.F. Thompson, F.C. Thames, C.W. Mastin, TOMCAT – a code for numerical generation of boundary-fitted curvilinear coordinate systems on fields containing any number of arbitrary two-dimensional bodies, J. Comput. Phys. 24 (1977) 274–302.
[14] E.B. Becker, G.F. Carey, J.T. Oden, in: Finite Elements: An Introduction, vol. 1, Prentice-Hall, Englewood Cliffs, NJ, 1981.
[15] J.N. Reddy, An Introduction to the Finite Element Method, second ed., McGraw-Hill, Boston, 1993.
[16] L.D. Landau, E.M. Lifshitz, The Classical Theory of Fields, Pergamon Press, New York, 1975.
[17] J.E. Castillo, S. Steinberg, P. Roache, On the folding of numerically generated grids: use of a reference grid, Comm. Appl. Num. Methods 4 (1988) 471–481.
[18] X.-C. Cai, W.D. Gropp, D.E. Keyes, R.G. Melvin, D.P. Young, Parallel Newton–Krylov–Schwarz algorithms for the transonic full potential equation, SIAM J. Sci. Comput. 19 (1) (1998) 246–265.
[19] A. Stagg, G. Hansen, A. Zardecki, Comments on applying Newton's method to nonlinear finite element problems, Tech. Rep. LAUR-03-6657, Los Alamos National Laboratory, September 2003.
[20] E. Chow, M.A. Heroux, An object-oriented framework for block preconditioning, ACM Trans. Math. Soft. 24 (1998) 159–183.
[21] J.R. Shewchuk, Delaunay refinement algorithms for triangular mesh generation, Comput. Geometry Theory Appl. 22 (1–3) (2002) 21–74.
[22] P. Knupp, L. Margolin, M. Shashkov, Reference Jacobian optimization-based rezone strategies for arbitrary Lagrangian Eulerian methods, J. Comput. Phys. 176 (1) (2002) 93–128.
[23] J.F. Thompson, Z.U.A. Warsi, C.W. Mastin, Numerical Grid Generation: Foundations and Applications, North Holland, New York, NY, 1985.
[24] G.F. Carey, Computational Grids: Generation, Adaptation, and Solution Strategies, Taylor & Francis, Washington, DC, 1997.