Finite Elements in Analysis and Design 158 (2019) 17–30

Contents lists available at ScienceDirect

Finite Elements in Analysis and Design

journal homepage: www.elsevier.com/locate/finel

A general mesh smoothing method for finite elements

R. Durand [a,*], B.G. Pantoja-Rosero [b], V. Oliveira [a]

[a] Department of Civil Engineering, University of Brasilia, Campus Darcy Ribeiro, Brasilia, DF, CEP 70910-900, Brazil
[b] Faculty of Civil Engineering, University Santo Tomás, Campus Aguas Claras, Villavicencio, CEP 500003, Colombia

ARTICLE INFO

Keywords:
Finite elements
Mesh smoothing
Least-squares fitting

ABSTRACT

This paper presents a general mesh smoothing method for finite elements. The method deals with two and three-dimensional meshes with virtually any type of element. To evaluate the quality of different element types, the paper also introduces a broad quality function. The proposed method works by solving a standard deformation analysis where nodal forces aim to deform each element into an optimally placed reference element. A detailed algorithm of the proposed smoothing method is presented which is suitable for a straightforward computer implementation. Several application examples in two and three-dimensions are presented and analyzed in order to demonstrate the capabilities of the proposed algorithm. In all cases very good quality improvements were obtained with very few iterations. Also, the proposed method provided greater improvements when compared to conventional Laplacian-based methods.

1. Introduction

The accuracy of finite element analyses is highly dependent on the mesh quality. In fact, very thin or skewed elements may lead to poor approximations results. Several authors pointed the negative effect on finite element analyses due to the presence of low quality elements [1–5]. Mesh improving techniques aim to enhance the shape of individual elements according to some quality metric. Different mesh improvement methods are available in the literature (see e.g. Refs. [6–18,18–27]). Park & Shontz [23] classified the methods in three groups: adaptivity, smoothing and swapping. In particular, smoothing methods aim to enhance the quality of elements by moving individual nodes while preserving the domain topology. Canann et al. [9] described three main approaches for mesh smoothing: Laplacian, optimization-based and physics-based.

In the Laplacian method [6] nodal positions are iteratively updated by the arithmetic mean of connected nodes. Although this method is simple and computationally cheap, it does not guarantee improvement in the element quality since it is possible to produce an invalid mesh containing elements that are inverted or have negative area [12]. To address this problem, smart variants have been proposed, which incorporate a quality metric and conditional updating [12,14,28]. In the other hand, new nodal positions can also be derived by solving local or global optimization problems with different types of objective functions [7,13,19,20,22]. This requires additional computational effort, but usually improves the resulting mesh if compared to Laplacian-based smoothing. Another alternative is the use of physics-based methods where, for example, the mesh is considered as a deformable system and forces are applied in order to improve the shape of each element [10,17,18].

This paper proposes a physics-based smoothing method where the mesh is considered as a deformable media. The method is applicable to two and three-dimensional finite element meshes composed by virtually any type of element, including less conventional ones such as wedges and pyramids. The whole procedure can be divided in two stages: best fitting of reference elements and global enhancement. The first stage is performed at element level where the best position and orientation of a reference element (with ideal shape) is found using a least squares fitting technique. The second stage aims the global enhancement by performing an elastic deformation analysis in the whole mesh. For each element, a set of nodal forces is calculated in order to deform the element itself into the reference one. Displacement constrains at the domain boundary are properly chosen in order to preserve the global topology. Significant enhancement was obtained by applying the proposed method on meshes composed by triangular, quadrilateral, tetrahedral, pentahedral (wedges and pyramids) and hexahedral elements. To assess the quality of different element types, the paper also introduces a broad quality function. For meshes with limited conditions of enhancement of low quality elements an extended approach is presented. This aims to prioritize the improvement of elements with the lowest quality while

* Corresponding author.
E-mail addresses: durand@unb.br (R. Durand), bryanpantoja@usantotomas.edu.co (B.G. Pantoja-Rosero), vicente@aluno.unb.br (V. Oliveira).

https://doi.org/10.1016/j.finel.2019.01.010
Received 22 October 2018; Received in revised form 24 January 2019; Accepted 28 January 2019
Available online XXX
0168-874X/

slightly reducing the quality of the best ones. The proposed method is novel and represents a contribution to the mesh smoothing literature since most methods available today are designed to work with specific types of elements (e.g. only triangles, quadrilaterals, tetrahedra, etc.) with particular quality metrics. Also, the method uses a single quality metric and a novel approach based on the singular value decomposition to best fit reference (ideal) elements over each element in the mesh.

The paper presents a throughout description of all required steps in the proposed method including a detailed algorithm aiming the computer implementation. Several application examples in two and three-dimensions are presented and analyzed in order to demonstrate the capabilities of the proposed algorithm. In all cases very good quality meshes were obtained with very few iterations and the computation time was fairly small. Also, the proposed method show greater improvements when compared to some Laplacian-based smoothing methods.

2. Smoothing procedure

In this proposed procedure, classified as a physics-based smoothing, the mesh is considered as a deformable media. The main idea is to find an optimal geometry for each element in the mesh. For this purpose, for each element a reference (ideal) element version is found in order to find an optimal shape and position. Later, all the displacements required to move the nodes of each element to its ideal version are balanced by means of a mechanical analysis.

The whole algorithm can be divided in two stages. The first stage deals with finding the optimal shape and position for each element. For a particular element, this is performed by defining a reference element (a regular shape with same area or volume) and applying a least-squares fitting of their nodal coordinates over the original element geometry. After fitting, the nodal distances between reference and original elements are stored and considered as the displacements required to transform the element into its best possible version. In the second stage, the nodal displacements found for each element are converted into forces by using an elastic stiffness matrix, assuming the mesh as an elastic and isotropic media. Later, all forces are applied in a finite element deformation analysis. This aims to promote the displacement of the mesh nodes and to improve the shape of all elements at once. The essential boundary conditions are set according to the boundary geometry. For example, boundary nodes may be fixed in all directions in order to keep the domain topology. Also, boundary nodes may be allowed to move along flat surfaces and straight edges without disturbing the boundary geometry. At the end of the elastic analysis, the change in shape quality for each element can be assessed by using available quality metrics. Since this work aims to deal with different types of elements a broad quality definition is introduced.

2.1. A general quality metric

According to Knupp [29], "*an element quality metric is a scalar function of node positions that measures some geometric property of the element*". There are several metrics proposed to represent the quality of elements in a finite element mesh, see for example [30–33]. Those metrics are frequently given in terms of internal angles and/or diagonal lengths ratios providing highest values to elements with shapes close to reference geometries. Also, in most cases those metrics are specific to certain type of elements. Since this paper aims to deal with different types of elements, a broad and robust definition is proposed. This definition provides values from 0 to 1, where 0 represents a flat element and 1 an element with ideal shape (e.g. regular polygon or polyhedron). For a two-dimensional element the quality function is initially defined as the ratio between the perimeter p_r of a reference element (e.g. regular polygon) with the same area and the perimeter p of the element itself. Later, this definition is extended to:

$$q_{2d}(\boldsymbol{e}) = \left(\frac{p_r}{p}\right)^{\beta}, \tag{1}$$

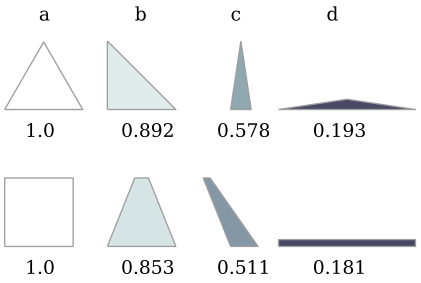


Fig. 1. Quality values for 2D elements with different distortion levels and $\beta = 2$.

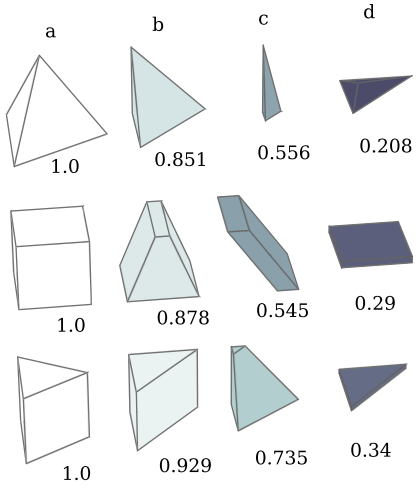


Fig. 2. Quality values for 3D elements with different distortion levels and $\beta = 2$.

where $\boldsymbol{e}$ refers to the element and $\beta > 1$ can be chosen to regulate the quality distribution in the range $[0, 1]$. For a three-dimensional element an analogous definition is given by:

$$q_{3d}(\boldsymbol{e}) = \left(\frac{s_r}{s}\right)^{\beta}, \tag{2}$$

where s is the surface area of the element and s_r is the surface area of a regular polyhedron with the same volume. In this work $\beta = 2$ is used for all cases since this produces a reasonable distribution of quality values for different distortion levels. Figs. 1 and 2 present the quality values for common finite elements under different levels of distortion. According to Osserman [34], the isoperimetric inequality for regular polygons guarantees that the terms p_r/p and s_r/s from Eqs. (1) and (2) are in the interval $(0, 1]$.

Given the area A of a 2D element, the reference element perimeter is calculated as:

$$p_r = \begin{cases} 6\sqrt{\dfrac{A}{\sqrt{3}}} & \text{for triangles} \\ 4\sqrt{A} & \text{for quadrilaterals} \end{cases} \tag{3}$$

In turn, given the volume V of a 3D element, the surface area of the reference element is calculated as:

$$s_r = \begin{cases} \sqrt{3}\left(6\sqrt{2}V\right)^{\frac{2}{3}} & \text{for tetrahedra} \\ \left(1+\sqrt{3}\right)\left(3\sqrt{2}V\right)^{\frac{2}{3}} & \text{for pyramids} \\ 6V^{\frac{2}{3}} & \text{for hexahedrons} \\ \left(3+\dfrac{\sqrt{3}}{2}\right)\left(\dfrac{4}{\sqrt{3}}V\right)^{\frac{2}{3}} & \text{for wedges} \end{cases} \tag{4}$$

Table 1
Quality measures for 2D elements with different distortion levels.

Author	Equation	Quality values			
		Triangular elements			
		a	b	c	d
Proposed	$(p_r/p)^2$	1.000	0.892	0.578	0.193
Lo [36]	$4\sqrt{3}A/\sum l_i^2$	1.000	0.866	0.487	0.172
Baker [37]	$l_{\min}/l_{\max}$	1.000	0.707	0.297	0.506
Berzins [38]	$(1/(4q_b)+\sqrt{3}/(16q_w))^{-1}$	1.000	0.885	0.552	0.187
Serrate [37]	$3\min(\phi_i)/\pi$	1.000	0.750	0.284	0.142
Pebay and Baker [39]	$(l_{\max}\sum l_i/(4\sqrt{3}A))^{-1}$	1.000	0.717	0.443	0.129
		Quadrilaterals elements			
		a	b	c	d
Proposed	$(p_r/p)^2$	1.000	0.853	0.511	0.181
Lo [36]	$q_3q_4/(q_1q_2)$	1.000	0.076	0.071	1.000
Robinson [40]	$\max(X_1/X_2, X_2/X_1)$	1.000	0.600	0.219	0.050
Frey and George [41]	$((\sqrt{2}/8)\,h_{\max}h_s/A_{\min})^{-1}$	1.000	0.265	0.132	0.100
Knup [33]	$8/\left(\sum_{k=1}^{4}((\lambda_{11}^k+\lambda_{22}^k)/\alpha_k)\right)$	1.000	0.490	0.234	0.100
Pebay [42]	$(l_{\max}\sum l_i/(4A))^{-1}$	1.000	0.664	0.293	0.095
Knupp et al. [43]	$\sqrt{2}\,l_{\min}/d_{\max}$	1.000	0.243	0.110	0.071

For elements that cannot be mapped to a regular polyhedron (e.g. pyramids and wedges), a polyhedron with regular faces and equal edges is considered as reference element.

Since the introduced quality measure is relative to the geometry of a regular polygon/polyhedron (ideal geometry), it takes into account features as skew and elongation of elements. The presented quality functions can also be applied to higher order elements. However, since middle-side nodes may produce curved edges, the shape of a higher order element may differ from a polygon/polyhedron and consequently the proposed quality metric can yield, although in rare cases, values greater that one. Thus, to extend previous definitions to also consider higher order elements the following equation is proposed:

$$q(\boldsymbol{e}) = 1 - \left|1 - \left(\frac{m_r}{m}\right)^{\beta}\right|, \tag{5}$$

where m is the perimeter or surface area according to the element dimension.

According to the definitions presented by Knupp [29] the proposed quality function is dimension-free, element-free, versatile, scale-free, unitless and referenced. This simple quality definition can be easily implemented in computer codes. Required quantities as perimeters or surface areas from elements can be computed by analytical formulas or approximated via numerical integration.

2.2. Comparison with other quality metrics

Tables 1 and 2 show the quality values obtained by the proposed and other metrics applied to the same elements presented in Figs. 1 and 2. The tables include the author and the corresponding metric equa-

Table 2
Quality measures for 3D elements with different distortion levels.

Author	Equation	Quality values			
		Tetrahedrons			
		a	b	c	d
Proposed	$(s_r/s)^2$	1.000	0.851	0.556	0.208
Shephard et al. [44]	$2187\ \mathrm{V}^4/(\sum f_i^2)^3$	1.000	0.500	0.096	0.004
Liu and Joe [45]	$12(3\ \mathrm{V})^{2/3}/\sum l_i^2$	1.000	0.840	0.353	0.300
Parthasarathy [46]	$(R_i^3\sqrt{2}/(12\ \mathrm{V}))^{-1}$	1.000	0.770	0.210	0.165
Weatherill et al. [38]	$(1/(8.48528\ \mathrm{V})(\sum l_i/6)^3)^{-1}$	1.000	0.804	0.318	0.196
Knupp et al. [43]	$\min(l_i)/\max(l_i)$	1.000	0.707	0.243	0.265
		Hexahedrons			
		a	b	c	d
Proposed	$(s_r/s)^2$	1.000	0.878	0.545	0.290
Taylor [35]	$\max(\overline{A}_1, \overline{A}_2, \overline{A}_3)$	1.000	0.600	0.246	0.100
Kwok [47]	$(1 - W_{\max})J_{\min}/J_{\max}$	1.000	0.444	1.000	1.000
Knupp [33]	$24/\left(\sum_{k=1}^{8}(\lambda_{11}^k+\lambda_{22}^k+\lambda_{33}^k)/\alpha_k^{2/3}\right)$	1.000	0.625	0.521	0.322
Menendez et al. [18]	$q_4q_5q_6/(q_1q_2q_3)$	1.000	0.006	0.343	1.000
Knupp et al. [43]	$\sqrt{3}\min(l_i)/\max(d_i)$	1.000	0.225	0.300	0.122
		Wedges			
		a	b	c	d
Proposed	$(s_r/s)^2$	1.000	0.929	0.735	0.34
Kwok [47]	$(1 - W_{\max})J_{\min}/J_{\max}$	1.000	1.000	0.127	1.000
Knupp et al. [43]	$\min(l_i)/\max(l_i)$	1.000	0.707	0.156	0.049
Knupp et al. [43]	$\min\{J_i\}/V$	1.000	1.000	0.245	1.000
Knupp et al. [43]	$\min(\overline{q}_1, \overline{q}_2, \overline{q}_3)$	1.000	0.816	0.200	0.070

tion. For some metrics, the inverse was taken in order to get values in the interval $[0,1]$. Regarding the metrics in Table 1, l is the length of an edge, ϕ is an internal angle, $q_b = 4\sqrt{3}\,a/\sum l_i^2$, $q_w = (1/(3a))(\sum l_i)^2$, $A_{\min}$ is the minimum area from four possible triangles in a quadrilateral element, q_1 to q_4 are the Lo's quality values for each possible triangle in a quadrilateral element sorted in descending order, $h_s = \sqrt{\sum l_i^2}$, $h_{\max} = \max(d_1, d_2, \min(l_i))$, $X_1 = |\mathbf{P}_2 - \mathbf{P}_1 + \mathbf{P}_3 - \mathbf{P}_4|$, $X_2 = |\mathbf{P}_3 - \mathbf{P}_2 + \mathbf{P}_4 - \mathbf{P}_1|$ and $\mathbf{P}$ is a vector with the coordinates of a node. Quantities λ_{ij}^k and α_k are detailed in Ref. [33]. As for the metrics in Table 2, f is the area of a face, l is the length of an edge, d is a diagonal length, J is the Jacobian at an integration point, $W_{\max}$ is considered zero for flat faces, q_1 to q_6 are the Lo's quality values for each quadrilateral face, $\overline{q}_1$ to $\overline{q}_3$ are the quality values of each quadrilateral face in a wedge calculated using the formula $\sqrt{2}\,l_{\min}/d_{\max}$, Quantities $\overline{A}_1$ to $\overline{A}_3$ are detailed in Ref. [35] and quantities λ_{ij}^k and α_k in Ref. [33].

As can be seen in Tables 1 and 2, the proposed quality metric provides decreasing values according to the element distortions presented in Figs. 1 and 2. This is compatible with most of the other quality metrics presented. Also, in these cases, the obtained values are relatively close. This suggest that the proposed metric can be useful when comparing the elements quality of different types of meshes including hybrid meshes. In the other hand, it is worth to note that some quality metrics return values of 1.0 at some considerably distorted elements, although with internal right angles, as the quadrilateral, hexahedral and wedge shapes presented at column d in Figs. 1 and 2. For these elements, the proposed metric provided small values as expected.

2.3. Least-squares fitting of two point sets

This procedure is used to best fit a reference element over a distorted one based on the nodal coordinates. In general, given two point sets, $\mathbf{P}_i$ and $\mathbf{P}'_i$ with $i = 1, 2, 3, \ldots, n$, this procedure is used to fit (by translation and rotation, maintaining their relative positions) $\mathbf{P}_i$ as close as possible to $\mathbf{P}'_i$. If the corresponding rotation and translation matrices ($\mathbf{R}$ and $\mathbf{T}$, respectively) are known, the following equality can be written:

$$\mathbf{P}'_i = \mathbf{R}\mathbf{P}_i + \mathbf{T} + \mathbf{N}_i, \tag{6}$$

where $\mathbf{N}_i$ is a set of noise vectors. Therefore, to find $\mathbf{R}$ and $\mathbf{T}$ the following norm needs to be minimized:

$$S = \sum_{i=1}^{n} \left\| \mathbf{P}'_i - (\mathbf{R}\mathbf{P}_i + \mathbf{T}) \right\|^2. \tag{7}$$

Arun et al. [48] presented a procedure for least-squares fitting of two point sets that provides $\mathbf{R}$ and $\mathbf{T}$. The process starts by calculating a 3×3 matrix $\mathbf{H}$ using the coordinates from both sets, $\mathbf{P}_i$ and $\mathbf{P}'_i$:

$$\mathbf{H} = \sum_{i=1}^{n} \mathbf{d}_i \mathbf{d}'^{T}_i, \tag{8}$$

where:

$$\mathbf{d}_i = \mathbf{P}_i - \overline{\mathbf{P}}. \tag{9}$$

$$\mathbf{d}'_i = \mathbf{P}'_i - \overline{\mathbf{P}}'. \tag{10}$$

$$\overline{\mathbf{P}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{P}_i. \tag{11}$$

$$\overline{\mathbf{P}}' = \frac{1}{n}\sum_{i=1}^{n} \mathbf{P}'_i. \tag{12}$$

The singular value decomposition of $\mathbf{H}$ provides:

$$\mathbf{H} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T \tag{13}$$

from where the rotation and translation matrices are found as:

$$\mathbf{R} = \mathbf{V}\mathbf{U}^T \quad \text{and} \tag{14}$$

$$\mathbf{T} = \overline{\mathbf{P}}' - \overline{\mathbf{P}}\,\mathbf{R}^T. \tag{15}$$

2.4. Best fitting of reference elements

For a given element $\boldsymbol{e}$ in the mesh, this process aims to find a reference element with the same area (or volume) and fit it over the original element coordinates $\mathbf{C}_e$. In short, we want to find the ideal shape and orientation for $\boldsymbol{e}$. We start by considering a reference element $\boldsymbol{e}_r$ with regular geometry and local coordinates $\mathbf{C}_l$ that provide an area (or volume) equal to one. For example, for triangular and quadrilateral elements, we may consider:

$$\mathbf{C}_l^{tri} = \begin{bmatrix} 0 & 0 \\ l & 0 \\ \frac{l}{2} & \frac{\sqrt{3}}{2}l \end{bmatrix} \quad \text{with} \quad l = \frac{2}{\sqrt[4]{3}} \tag{16}$$

$$\mathbf{C}_l^{qua} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{bmatrix}. \tag{17}$$

Also, for tetrahedra, pyramids and wedges we have:

$$\mathbf{C}_l^{tet} = \begin{bmatrix} 0 & 0 & 0 \\ l & 0 & 0 \\ \frac{l}{2} & \frac{\sqrt{3}}{2}l & 0 \\ \frac{l}{2} & \frac{\sqrt{3}}{6}l & \frac{\sqrt{6}}{3}l \end{bmatrix} \quad \text{with} \quad l = (6\sqrt{2})^{\frac{1}{3}} \tag{18}$$

$$\mathbf{C}_l^{pyr} = \begin{bmatrix} 0 & 0 & 0 \\ l & 0 & 0 \\ l & l & 0 \\ 0 & l & 0 \\ \frac{l}{2} & \frac{l}{2}l & \frac{\sqrt{2}}{2}l \end{bmatrix} \quad \text{with} \quad l = (3\sqrt{2})^{\frac{1}{3}} \tag{19}$$

$$\mathbf{C}_l^{wed} = \begin{bmatrix} 0 & 0 & 0 \\ l & 0 & 0 \\ \frac{l}{2} & \frac{\sqrt{3}}{2}l & 0 \\ 0 & 0 & l \\ l & 0 & l \\ \frac{l}{2} & \frac{\sqrt{3}}{2}l & l \end{bmatrix} \quad \text{with} \quad l = \left(\frac{4}{\sqrt{3}}\right)^{\frac{1}{3}}. \tag{20}$$

The next step is to scale the reference element $\boldsymbol{e}_r$ so it gets the same area (or volume) as the original element. This is carried out by the use of a scaling factor s, thus the coordinates matrix of $\boldsymbol{e}_r$ is given by:

$$\mathbf{C}_r = s\,\mathbf{C}_l. \tag{21}$$

The value of s is found as:

$$s = \begin{cases} \sqrt{A} & \text{for 2D} \\ \sqrt[3]{V} & \text{for 3D,} \end{cases} \tag{22}$$

where A is the original element area of a bi-dimensional element and V the volume of a three-dimensional one.

The final step is the best fitting adjustment of the reference element over the original one. Using the least squares method, as described in

section 2.3, the required rotation matrix **R** and translation vector **T** are found from the singular value decomposition of:

$$\mathbf{H} = \mathbf{D}^T \mathbf{D}', \tag{23}$$

where:

$$\mathbf{D}'_{ij} = (\mathbf{C}_e)_{ij} - (\overline{\mathbf{C}}_e)_j \tag{24}$$

$$\mathbf{D}_{ij} = (\mathbf{C}_r)_{ij} - (\overline{\mathbf{C}}_r)_j \tag{25}$$

$$(\overline{\mathbf{C}}_e)_j = \frac{1}{n}\sum_{i=1}^{n} (\mathbf{C}_e)_{ij} \tag{26}$$

$$(\overline{\mathbf{C}}_r)_j = \frac{1}{n}\sum_{i=1}^{n} (\mathbf{C}_r)_{ij}. \tag{27}$$

In the equations above, $\mathbf{C}_e$ is a matrix containing the nodal coordinates of the original element; also, $\overline{\mathbf{C}}_e$ and $\overline{\mathbf{C}}_r$ are vectors containing the centroid coordinates for the original element and its reference counterpart respectively. Subindices i and j vary as $i = 1 \dots n$ and $j = 1 \dots d$ with d being the element dimension. Later, the rotation matrix **R** is calculated from the decomposition $\mathbf{H} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T$ using Eq. (14) while the translation vector is found as:

$$\mathbf{T} = \overline{\mathbf{C}}_e - \mathbf{R}\,\overline{\mathbf{C}}_r. \tag{28}$$

Finally, the coordinates matrix of the best fit reference element is given by:

$$\mathbf{C} = \mathbf{C}_r\mathbf{R}^T + \mathbf{T}_m, \tag{29}$$

where $\mathbf{T}_m$ is a matrix with components:

$$(\mathbf{T}_m)_{ij} = \mathbf{T}_j. \tag{30}$$

Fig. 3 illustrates the whole adjustment process of an element including the scaling and best fitting steps. Fig. 4 shows the nodal displacements that would be necessary to convert the original element into the reference one once the adjustment process is accomplished.

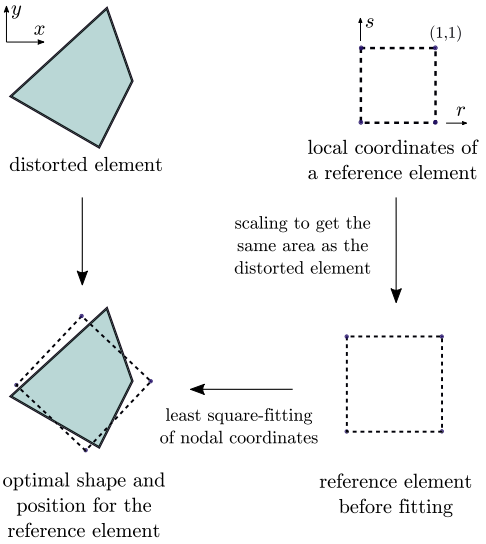


Fig. 3. Adjustment process of a reference geometry over a distorted element.

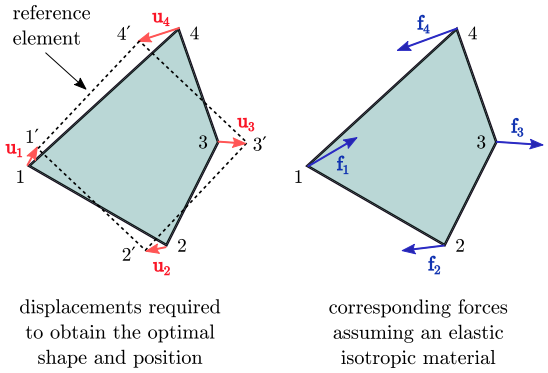


Fig. 4. Displacements (red arrows) and corresponding forces (blue arrows) required to convert a distorted element into its reference counterpart. The forces are found assuming an elastic isotropic material.

2.5. Global enhancement

Once the ideal geometry and orientation for a given element are found, the distances between corresponding nodes are considered as required displacements and are compiled into a vector $\mathbf{U}_e$. Considering the element as deformable (elastic isotropic material), a set of forces (Fig. 4) that reproduce the displacement field given by $\mathbf{U}_e$ is found by:

$$\mathbf{F}_e = \mathbf{K}_e\,\mathbf{U}_e, \tag{31}$$

where $\mathbf{K}_e$ is an elastic stiffness matrix. As for the elastic parameters used in $\mathbf{K}_e$, the Young's modulus value is chosen as $E = 1.0$ while the Poisson ratio is set to $\nu = 0$. Note that the Young's modulus value is not determinant since the forces will be proportional to the stiffness. In turn, the Poisson ratio equal to zero leads to uncoupled strains and, consequently, to greater deformability. This aims to provide faster mesh improvements to the proposed method.

In the sequence, a deformation analysis is carried out in order to find the deformations that improve the shape of all elements at once. For this purpose, a linear system is defined by assembling the forces and the stiffness matrix for each element.

$$\mathbf{K}\,\mathbf{U} = \mathbf{F}. \tag{32}$$

The essential boundary conditions are set at boundary nodes in order to keep the domain geometry unaltered. For example, we may fix all boundary nodes in all directions. However, allowing the displacement of nodes located at flat surfaces and straight edges lead to extra quality enhancements while still keeping the domain topology. Since flat surfaces and straight edges in a mesh may be inclined, the use of inclined supports is required in the elastic analysis. Thus, to improve the smoothing of meshes with these conditions, we include inclined supports by the use of Lagrange multipliers (see e.g. Ref. [49]). Thus,

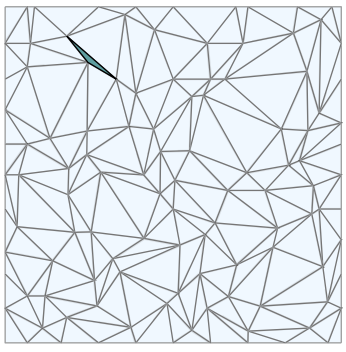


Fig. 5. Initial mesh for Example 1. One element was highlighted for further analysis.

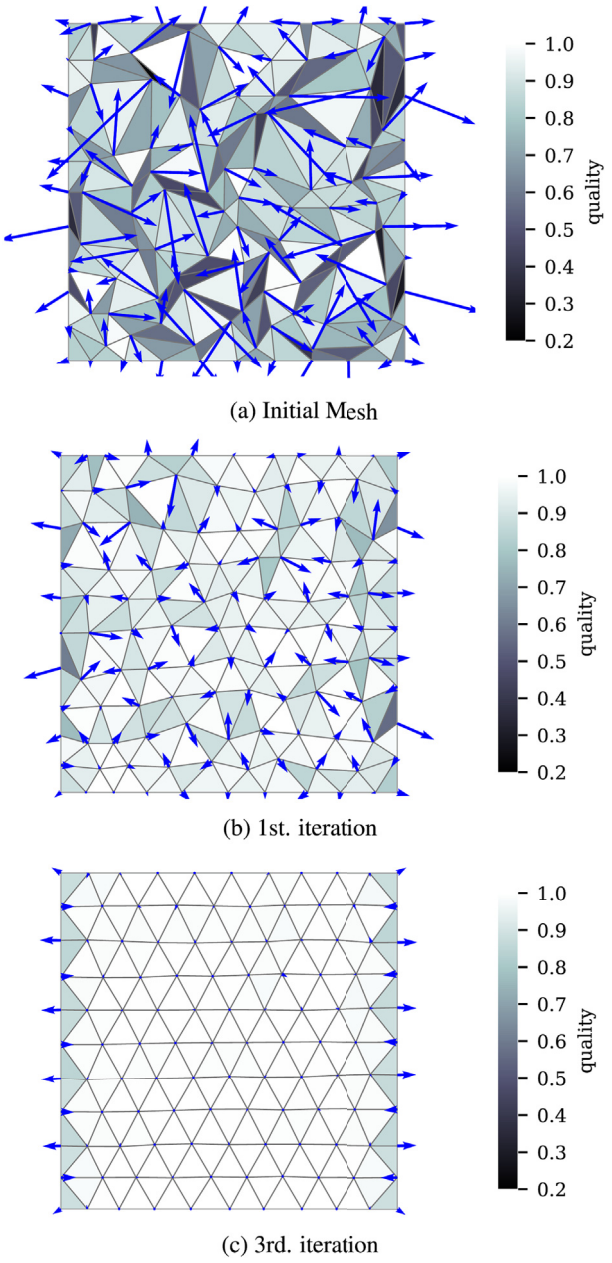


Fig. 6. Mesh smoothing iterations for Example 1 featuring the force vectors required to improve the mesh quality.

Eq. (32) is extended to:

$$\begin{bmatrix} \mathbf{K} & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \mathbf{U} \\ \lambda \end{pmatrix} = \begin{pmatrix} \mathbf{F} \\ \mathbf{b} \end{pmatrix}. \tag{33}$$

where λ is a vector of Lagrange multipliers and $\mathbf{A}$ and $\mathbf{b}$ are related to displacements constraints. The solution of Eq. (33) yields global displacements $\mathbf{U}$ that are used to update the mesh coordinates. This provides great quality enhancement to individual elements and the whole mesh. The mesh quality improvement is assessed by the increase of the average and minimum quality values, q_{avg} and $q_{\min}$, respectively, taken from all elements. If needed, the process can be repeated iteratively to get further improvements. The iterative process can be stopped when a maximum number of iterations is reached. Another criterion is to stop when the changes in $q_{\min}$ or q_{avg} are under a given tolerances, $\epsilon_{\min}$ and ϵ_{avg}, respectively. Yet, usually just one to three iterations are needed to highly improve meshes with low quality elements.

Table 3
Elements quality evolution for Example 1.

Iteration	$q_{\min}$	$q_{\max}$	q_{avg}	σ
0	0.233	0.998	0.750	0.17859
1	0.561	1.000	0.932	0.07129
2	0.794	1.000	0.975	0.04108
3	0.849	1.000	0.982	0.03779
4	0.868	1.000	0.983	0.03735
5	0.871	1.000	0.983	0.03719

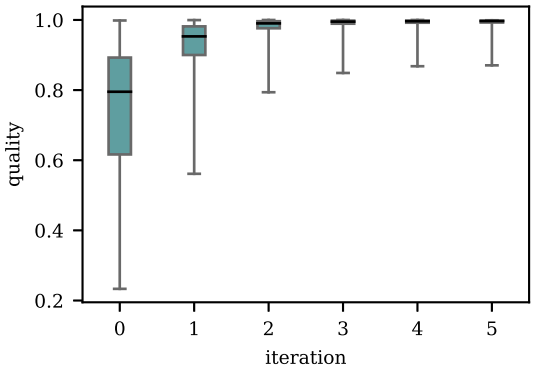


Fig. 7. Box plot of quality values along iterations for Example 1.

In meshes with elements with very low quality values, the proposed method may lead, although in rare cases, to some elements with negative Jacobian values. To overcome this situation, and similar to the smart Laplacian approach (see Ref. [12]), nodal positions are updated only if the quality values of attached elements (patch) attend the following condition:

$$(q_{\min}^{patch})_{after} > \gamma\,(q_{\min}^{patch})_{before} \tag{34}$$

where $(q_{\min}^{patch})_{before}$ is the minimum quality value of a patch before updating, $(q_{\min}^{patch})_{after}$ is the minimum quality value of the same patch if the corresponding nodal position were updated and γ is an adjustable parameter. In this work we use $\gamma = 0.8$ for all cases.

2.6. Extended approach

There are some particular meshes that include low quality elements where the application of the proposed method (as explained in section 2.5) improves q_{avg} as expected but may not provide substantial improvement to $q_{\min}$. To address this problem, an extended approach is presented where elements with lower quality are prioritized over elements with higher quality. This leads to the reduction of low quality elements although some good quality ones may be slightly worsen. This may also provide a small reduction in the quality average q_{avg}; however the reduction of the number of low quality elements minimizes the possibility of approximation errors in finite element analyses.

In this extended approach, element forces are calculated in terms of its own quality q and the mesh lowest quality value $q_{\min}$. Thus, a new version of Eq. (31) is given by:

$$\mathbf{F}_e = \frac{1-q}{1-q_{\min}} \mathbf{K}_e \mathbf{U}_e \tag{35}$$

In addition, in order to avoid possible large deformations, a reduction factor α is introduced to modify Eq. (21), thus it is rewritten as:

$$\mathbf{C}_r = \alpha s\, \mathbf{C}_l. \tag{36}$$

This extension, given by Eqs. (35) and (36), can be used in some cases to provide extra improvements to the lowest quality elements in the mesh.

Algorithm 1 General smoothing.

Input: A finite element mesh, maximum number of iterations $maxit$, tolerance ϵ_{avg} for q_{avg}, tolerance ϵ_{min} for q_{min}, reduction factor α (optional, default $\alpha = 1.0$)

Output: A smoothed version of the mesh

! Initial quality calculations
Compute the quality q of each element
Compute the lowest quality q_{min} among elements

! Smoothing algorithm iterative process
for $i = 1$ **to** $maxit$ **do**
  ! Compute local stiffness matrix and forces vector
  foreach element in the mesh **do**
    Get the element coordinates $\mathbf{C}_0$
    Get the element volume (or area) V
    Get the local coordinates $\mathbf{C}_l$
    $s = V^{\frac{1}{dim}}$ *! scale factor Eq. 22*
    $\mathbf{C}_r = \alpha\, s\, \mathbf{C}_l$ *! see Eqs. 21 and 36*
    Compute best fit rotation and translation matrices: $\mathbf{R}$ and $\mathbf{T}$ based on $\mathbf{C}_0$ and $\mathbf{C}_r$
    $\mathbf{C} = \mathbf{C}_r\, \mathbf{R}^T + \mathbf{T}$ *! translate and rotate Eq. 29*
    $\mathbf{U}_e = \mathbf{C} - \mathbf{C}_0$ *! required displacements*
    Compute element stiffness $\mathbf{K}_e$
    if extended approach **then**
      $\mathbf{F}_e = \frac{1-q}{1-q_{min}} \mathbf{K}_e\, \mathbf{U}_e$ *! Eq. 35*
    else
      $\mathbf{F}_e = \mathbf{K}_e\, \mathbf{U}_e$
    Assemble $\mathbf{K}_e$ into global $\mathbf{K}$ matrix
    Assemble $\mathbf{F}_e$ into global $\mathbf{F}$ vector
  ! Find the displacements global vector and update
  Mount global system using Lagrange multipliers
  Solve the system and find displacements $\mathbf{U}$
  Use $\mathbf{U}$ to update the mesh coords. at nodes that attend Eq. 34
  Update the quality q of each element
  Compute the lowest quality q^i_{min} in the mesh
  Compute the quality average q^i_{avg} in the mesh
  ! Check for convergence
  $\Delta q_{min} = |q^i_{min} - q^{i-1}_{min}|$
  $\Delta q_{avg} = |q^i_{avg} - q^{i-1}_{avg}|$
  if $\Delta q_{avg} < \epsilon_{avg}$ and $\Delta q_{min} < \epsilon_{min}$ **then**
    break
Return the smoothed mesh

Finally, a summary of the whole smoothing procedure is presented in Algorithm 1 including the case for the extended approach.

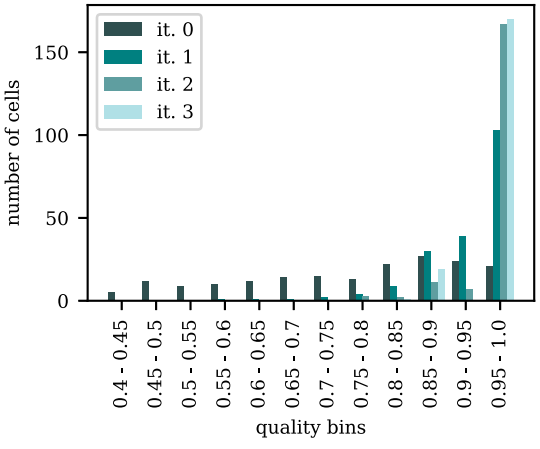


Fig. 8. Elements quality distribution along iterations for Example 1.

3. Application examples

This section presents six application examples of the proposed smoothing method in two and three dimensions. The first three examples use the original approach as presented in section 2.5. The fourth example uses the original approach, whose results are further improved by the application of the extended version. The last two examples present two 3D unstructured meshes where the original and the extended version of the proposed algorithm are used. Examples 4 to 6 also present comparisons with results from the application of Laplacian-based smoothing methods. The unstructured meshes (2D and 3D) in this section were elaborated using a Delaunay mesh generator (see Ref. [50]). All examples were run using a 3 GHz machine and an implementation in Julia [51] (a dynamic language) with no memory optimizations on linear algebra operations. To asses the level of mesh quality improvement, the following quantities are observed along iterations: the mesh average quality q_{avg}, the maximum and minimum quality values, $q_{\min}$ and $q_{\max}$, the quality range $q_{\max} - q_{\min}$ and the standard deviation σ. Also, some examples evaluate the number of elements with very poor quality. In this case we chose elements with $q < 0.2$.

Regarding the Laplacian-based methods used for comparative purposes, the following formulations were applied. For the classical Laplacian method, the nodal coordinate of an interior point $\mathbf{P}_i$ in the mesh was set to the arithmetic mean of all n points $\mathbf{P}_j$ (nodes) connected to $\mathbf{P}_i$, thus

$$\mathbf{P}_i = \frac{1}{n}\sum_{j=1}^{m}\mathbf{P}_j \tag{37}$$

In the smart Laplacian version [12], Eq. (37) is also used but each new node position is only set if this leads to an increment of the arithmetic mean of the quality values of all attached elements. Since this method led to invalid elements in some of the presented examples, the implementation used in this work considers that each new nodal position is only set if it leads to an increment in the minimum quality value of all attached elements. Finally, for the weighted Laplacian smoothing method [28], a displacement vector $\Delta\mathbf{P}$ is calculated from the weighted mean of centroidal coordinates $\mathbf{C}_j$ and volumes V_j from all m elements attached to node $\mathbf{P}_i$, thus:

$$\Delta\mathbf{P}_i = \frac{\sum_{j=1}^{m} V_j\,(\mathbf{C}_j - \mathbf{P}_i)}{\sum_{j=1}^{m} V_j} \tag{38}$$

Since there were examples where the weighted Laplacian method also provided invalid elements, we used a "smart weighted Laplacian" version which combines the smart and the weighted versions.

3.1. Example 1: simple mesh with triangular elements

This example shows the application of the proposed method to a mesh with highly distorted triangular elements (Fig. 5). This mesh was obtained by applying random displacements to the nodes of a regular mesh with good quality elements. In the smoothing process, all boundary nodes were restricted to move only along the boundary edges in order to preserve the mesh topology. Fig. 6 shows the initial mesh and the results from the first and third iteration colored according to the quality level. As can be seen, very good results were obtained within few iterations. In fact, at the end of the third iteration the regular mesh was almost completely recovered. Fig. 6 also shows the directions and magnitudes of the forces computed to improve the mesh at each iteration (see Eq. (31)). As seen, greater forces are required in the first iteration. For nodes at the domain boundary, only tangential components are taken into account due to the presence of normal displacement constraints.

A total of five iterations were performed for illustrative purposes. Table 3 presents a summary of results where iteration 0 corresponds to the initial mesh. By observing the evolution of $q_{\min}$, $q_{\max}$ and σ it is possible to see a considerable improvement in the first iteration. Fig. 7 shows the box plot of quality values along iterations depicting the median q_{med}, quartiles and quality limits. Note that $q_{\min}$ and q_{med} increase monotonically and from the third iteration almost no changes are observed in the median value. Also the quality range $(q_{\max} - q_{\min})$ reduces monotonically suggesting a stable behaviour. In order to visualize the elements distribution according to their quality, Fig. 8 presents a histogram with results up to the third iteration. It is seen that most elements with low quality in the initial mesh are quickly moved to higher quality bins in the first iterations. As a matter of fact, after two iterations, most elements were placed in the interval $(0.95, 1.0]$.

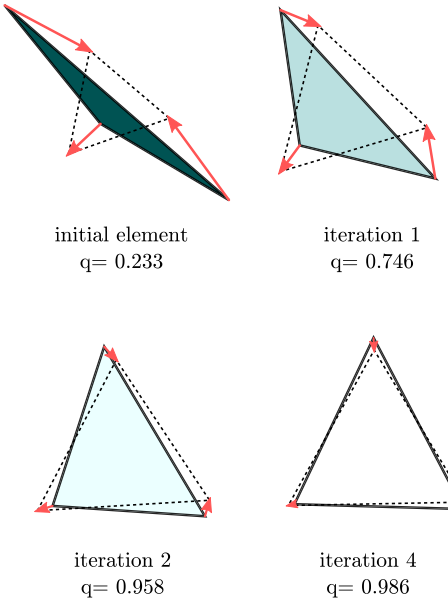


Fig. 9. Enhancement of element 175 along iterations. Red arrows represent the displacements required to deform the element into its reference counterpart.

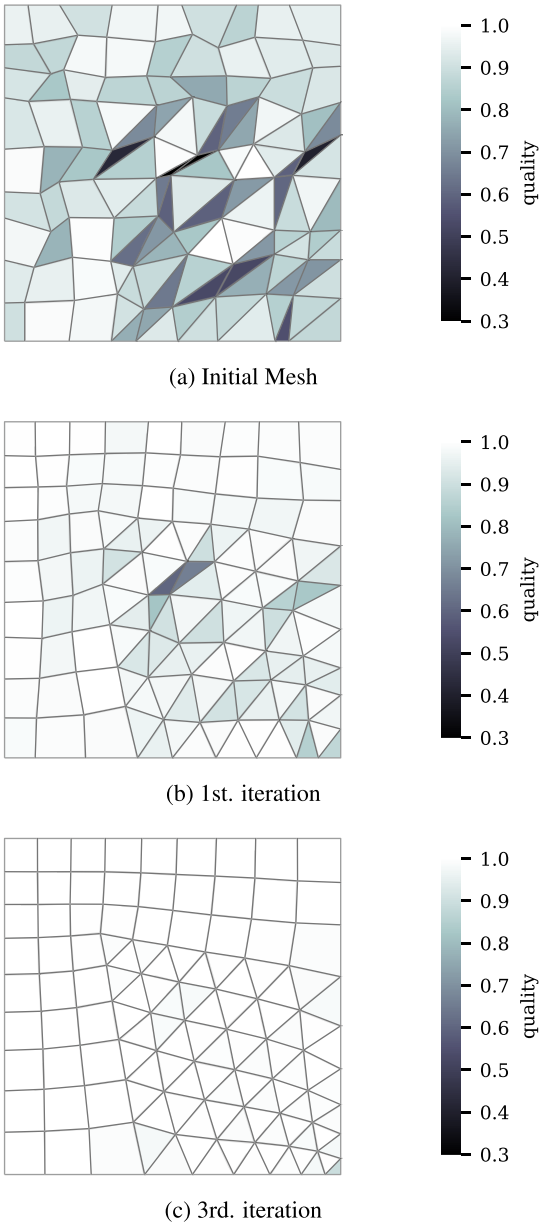


Fig. 10. Mesh smoothing results for Example 2.

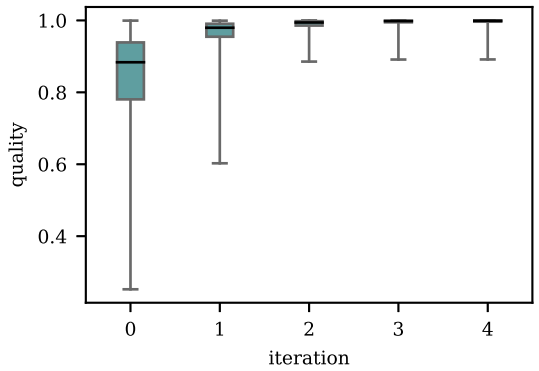


Fig. 11. Box plot of quality values along iterations for Example 2.

Table 4
Elements quality evolution for Example 2.

Iteration	$q_{\min}$	$q_{\max}$	q_{avg}	σ
0	0.252	0.999	0.838	0.15122
1	0.603	0.999	0.961	0.05705
2	0.886	1.000	0.989	0.01701
3	0.891	1.000	0.995	0.01078
4	0.892	1.000	0.997	0.01049

To illustrate the gradual quality enhancement, an element from the mesh is analyzed. This element was highlighted in Fig. 5 and the results are presented in Fig. 9. In this figure, doted triangles represent the corresponding reference elements that were optimally placed according to the best fitting procedure presented in section 2.4. Also, the red arrows represent the displacements field required to deform the element and obtain the reference geometry. For the observed element we can see that after few iterations the quality value tends to 1.0, which is the highest possible value.

3.2. Example 2 - hybrid mesh

To illustrate the application of the proposed algorithm to meshes with different types of elements this example uses a mesh of distorted triangular and quadrilateral elements. During the smoothing process all boundary nodes were restricted to move along the border edges. Fig. 10 shows the initial mesh including the results from the first and third iterations. In total, four iterations were performed for illustrative purposes. Notice that triangular and quadrilateral elements were evenly improved.

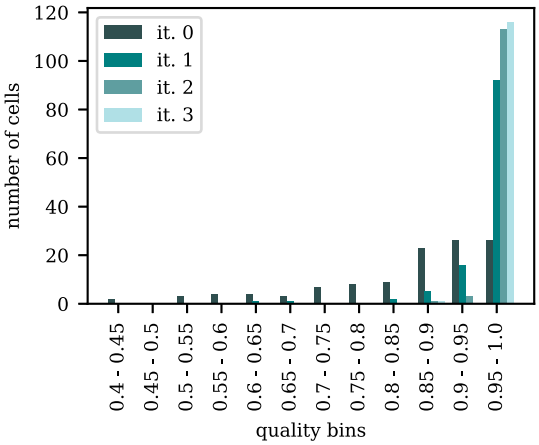


Fig. 12. Elements quality distribution along iterations for Example 2.

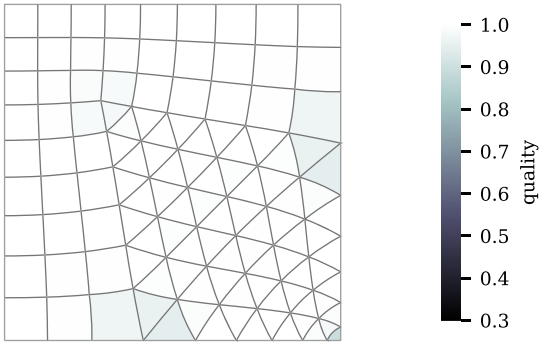


Fig. 13. Results using quadratic elements in Example 2.

Table 4 presents a summary of the four smoothing iterations. We can see that $q_{\min}$ and q_{avg} improve quickly and monotonically from the first iteration. In fact, the lowest quality value jumps from 0.252 to 0.603 in the first iteration and to 0.886 in the second one. Also we can observe that the standard deviation reduces along iterations and tends to converge. Fig. 11 shows the box plot of the quality values along iterations. Due to the rapid increase in $q_{\min}$ the quality range reduces substantially in the first two iterations. Also the median converges towards 1.0. By analyzing the first quartile (0.99) in the third iteration, we can say that at least 75% of elements have quality greater than 0.99. On the sequence, Fig. 12 present the quality distribution up to the third iteration. Note that most elements were moved to bin $[0.95, 1]$ within the first iteration.

As an extension for this example, Fig. 13 shows the results obtained from the same mesh but composed by six and eight-node quadratic elements. In this case, the results presents some elements with slightly

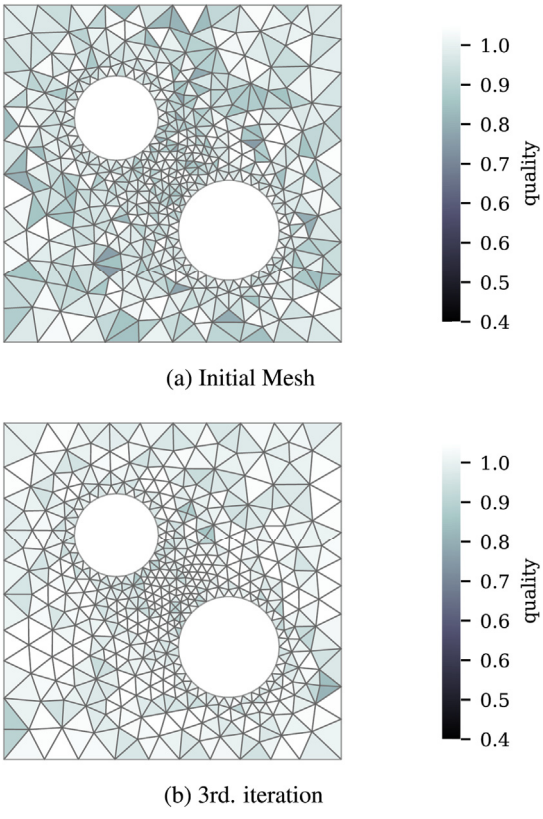


Fig. 14. Initial mesh and results for Example 3.

Table 5
Elements quality evolution for Example 3.

Iteration	$q_{\min}$	$q_{\max}$	q_{avg}	σ
0	0.765	1.000	0.935	0.04708
1	0.796	1.000	0.963	0.03180
2	0.802	1.000	0.966	0.02969
3	0.812	1.000	0.967	0.02923
4	0.817	1.000	0.967	0.02913

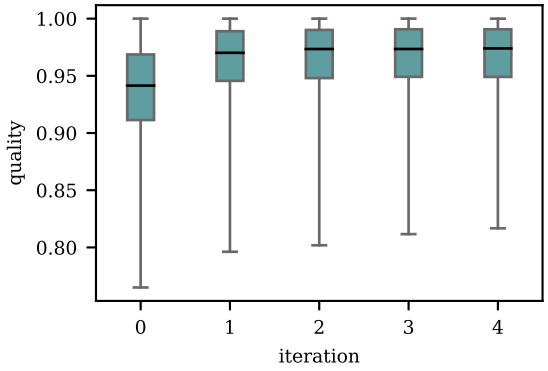


Fig. 15. Box plot of quality values for Example 3.

curved faces, although still with very good quality. However, as it is well known, elements with straight sides provide better approximations in finite element analyses, thus for the use of higher order elements it is preferable to smooth a linear version of the mesh first. Nonetheless, this example illustrates the strength and versatility of the proposed algorithm to deal not only with different types of elements but also with elements of higher order.

3.3. Example 3 - unstructured mesh with two holes

This example shows the case of an unstructured mesh composed by triangular elements featuring two holes (Fig. 14a). This mesh was elaborated in a Delaunay mesh generator (see Ref. [50]). Currently Delaunay mesh generators are particularly efficient at producing good quality triangulations. In fact, the minimum quality value in the initial mesh is equal to 0.765. However, we test the proposed algorithm seeking for further improvements. Again, it was considered that all boundary nodes were restricted to move along the border edges. The nodes that define the holes were completely constrained in order to keep the domain topology. A visual comparison of the initial mesh and the results from the third iteration step of the proposed algorithm is presented in Fig. 14.

Table 5 shows a summary of the quality evolution for four iterations. Note that $q_{\min}$ increased to 0.812 at the end of the third iteration and still tends to increase. Regarding q_{avg}, it increased monotonically and tends to converge.

Fig. 15 shows the box plot of quality values along iterations. Considering the quartiles and the median value, it demonstrates that most elements were actually improved. Fig. 16 presents the elements quality distributions showing that in fact most elements were moved to the highest quality bin within the first iteration.

As an extension for this example, the algorithm was tested against a three-dimensions version of the initial mesh. This 3D version was obtained from an automatic mesh generator that produced six-node wedge elements by extrusion. During the smoothing process, boundary nodes were restricted to move only along flat faces and straight edges thanks to the use of Lagrange multipliers for essential conditions. Fig. 17 shows the initial 3D mesh and the results from the second iteration of the smoothing algorithm. Fig. 18 shows the corresponding quality distribution along iterations. We can observe, relevant improvements as in the 2D case. In fact, in two iterations, $q_{\min}$ improved from 0.525 to 0.634 and q_{avg} improved from 0.884 to 0.915. This example extension demonstrates that the proposed algorithm can be applied straight away to 3D meshes even if composed by less conventional wedge elements.

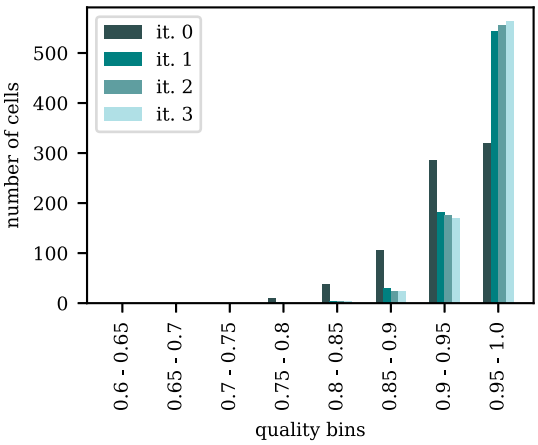


Fig. 16. Elements quality distribution for Example 3.

3.4. Example 4 - structured mesh of a concave solid

This example presents a general case of a 3D mesh composed by 560 hexahedral elements (Fig. 19a). The mesh was generated by coordinate mapping. In this case the domain was initially divided in 6 quadratic hexahedrons that were later subdivided. The nodal coordinates of resulting elements were found by interpolation taking advantage of quadratic shape functions. Yet, the obtained mesh presented

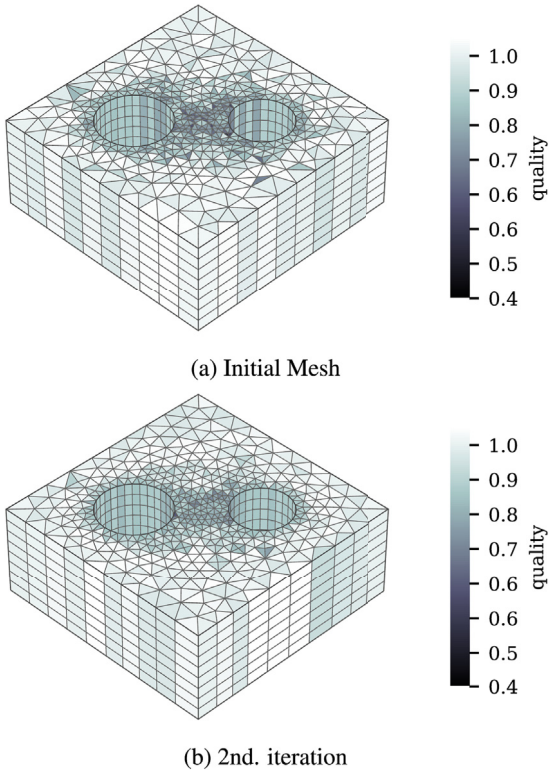


Fig. 17. Initial mesh and results for the extended version of Example 3.

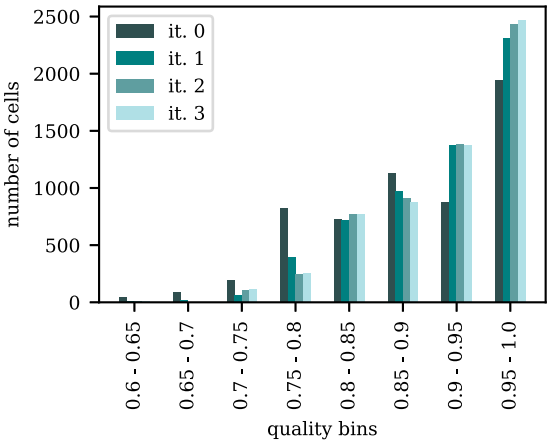


Fig. 18. Elements quality distribution for the extended version of Example 3.

some bad-shaped elements with $q_{\min} = 0.519$. During the smoothing process, border nodes were restricted to move along flat surfaces and straight edges. Fig. 19b shows the results after two iterations of the non-extended approach. Note that elements with lowest qualities were greatly enhanced and the mesh was turned quite suitable for finite element analyses. Figs. 20 and 21 show the results after an extra iteration using the extended approach with $\alpha = 1$ (notice whiter colors). This provided further enhancements; for instance, it increased $q_{\min}$ and reduced the quality range as shown in the results for iteration 3 in Fig. 22. Fig. 23 presents the elements quality distribution and Table 6 shows the quality evolution along iterations. Note that at the end all elements got quality greater than 0.8.

To analyze the efficacy of the proposed method, the results were compared with the ones obtained from Laplacian-based methods. Fig. 24 compares the obtained quality distributions with the classical Laplacian, smart Laplacian and the smart weighted Laplacian versions. All methods were limited to three iterations. It is clear that the proposed method moved a greater quantity of elements to highest quality bins.

3.5. Example 5 - unstructured mesh of a rocker arm

This example presents the smoothing process of an unstructured mesh and the comparison of results with Laplacian based methods. The initial mesh is composed by 12865 tetrahedral elements and represents a rocker arm (Fig. 25). The geometry was taken from Ref. [52]. This mesh was generated using a three-dimensional Delaunay generator (see Ref. [50]). Although Fig. 25 apparently does not show elements of poor quality, most low quality elements are inside. Even though 3D mesh generators aim to provide good quality elements frequently they may

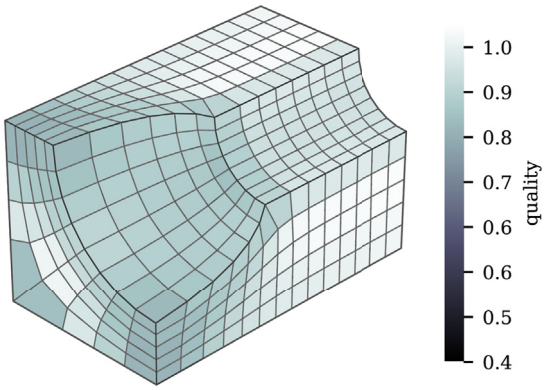


Fig. 20. Results after one extra iteration using the extended approach.

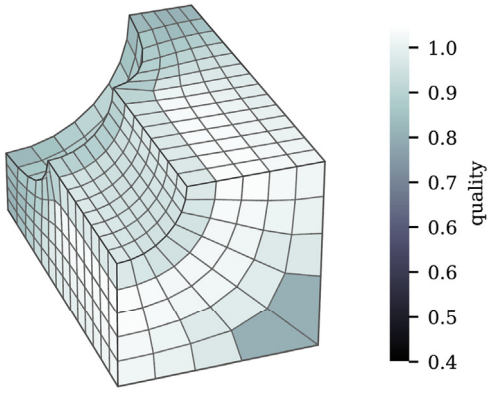
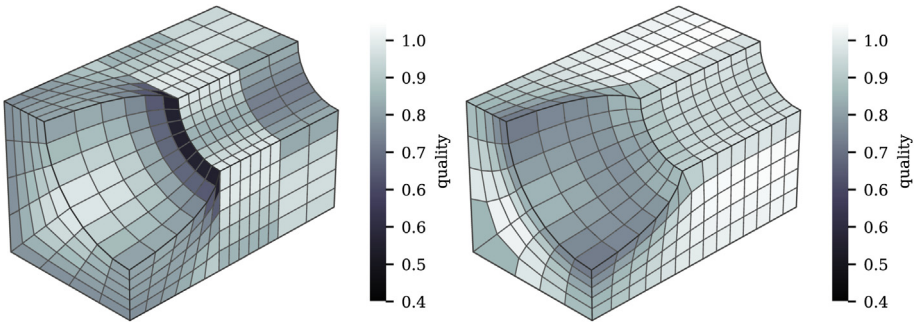


Fig. 21. Rear view of the final results from Example 4.

generate poor ones in the process to fill a constrained region. Thus, it is not rare to find unstructured meshes containing almost flat or needle-shaped elements which provide quality values (according to our metric) around 0.2 or below. It is worth mentioning that the presence of elements with this level of quality is unfavorable to finite element analyses and may invalidate their results, therefore reducing the number of these poor quality elements is quite desirable.

In addition to the proposed method, the mesh was smoothed by applying the smart Laplacian and smart weighted Laplacian methods. The application of the classical Laplacian method returned invalid elements, thus results from this method are not included. Fig. 26 presents a comparison of the corresponding quality distributions after three iterations. Although all methods provide some level of enhancement, the proposed method provided a smaller number of elements with quality under 0.2 and a greater number of elements with quality above



(a) Initial Mesh (b) 2nd. iteration

Fig. 19. Mesh smoothing results for Example 4.

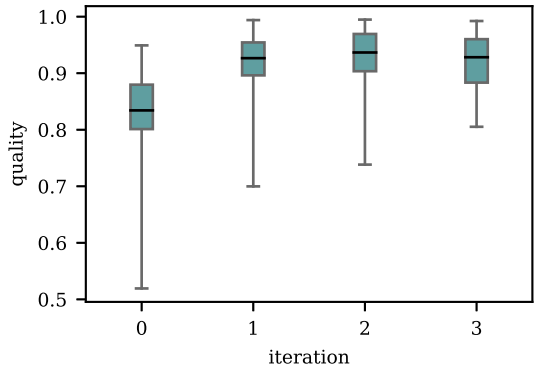


Fig. 22. Box plot of quality values for Example 4.

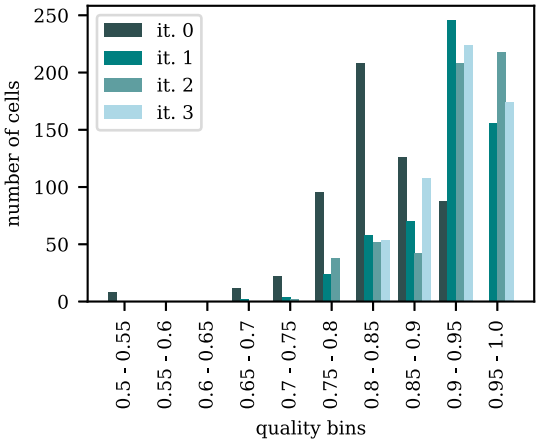


Fig. 23. Elements quality distribution for Example 4.

Table 6
Elements quality evolution for Example 4.

Iteration	q_{min}	q_{max}	q_{avg}	σ
0	0.519	0.949	0.834	0.06839
1	0.700	0.994	0.914	0.05660
2	0.738	0.995	0.922	0.06019
3	0.805	0.992	0.920	0.04773

0.8. Table 7 shows a summary of the quality average and the number of elements with $q < 0.2$. It can be seen that from 273 elements with $q < 0.2$ in the initial mesh only 75 remained. This number is well below the numbers provided by the Laplacian-based methods. The application of the extended version of the proposed algorithm, with $\alpha = 1$, was able to reduce this number to 65. Table 7 also provides the average time spent per iteration. Although the proposed method and the extended version spend more time they provided greater reductions of very low quality elements.

Since the initial mesh is composed by thousands of elements, after the application of the smoothing methods, the resulting meshes provided few noticeable changes in the quality field when observed. For these reason no figures with the resulting meshes were presented in this example.

3.6. Example 6 - unstructured mesh of a swing frame

This example presents the smoothing process of another unstructured mesh and the comparison with Laplacian-based methods. The initial mesh is composed by 32383 tetrahedral elements and represents

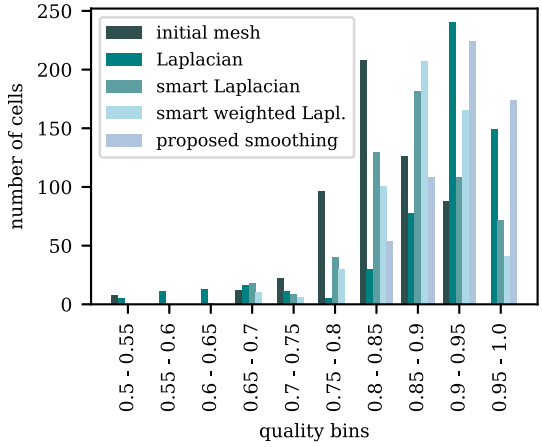


Fig. 24. Comparison of quality distributions after three iterations for Example 4.

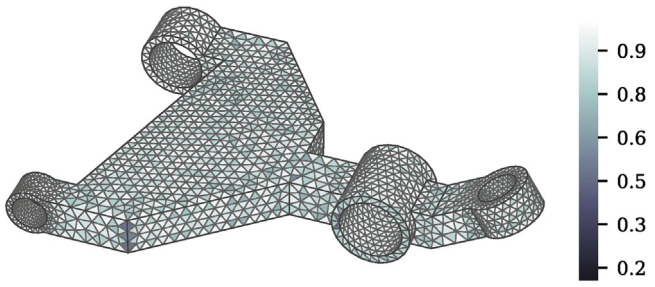


Fig. 25. Initial mesh showing the quality field for Example 5.

a swing frame (Fig. 27). As in the previous example, the mesh was obtained by using a Delaunay mesh generator. Although the mean quality value in the mesh is 0.724, the mesh includes several element with very low quality. For instance, it presents 938 elements with quality under 0.2.

Fig. 28 shows the quality distributions obtained after three iterations of the proposed method and two Laplacian-based methods. Again, the application of the classical version of the classical Laplacian method provided invalid elements and was not considered. As can be seen, once again, the proposed method provided a smaller number of elements with quality lower than 0.2. Also it provided a greater number of elements with quality above 0.8. Table 8 presents a summary of the quality average and the number of elements where $q < 0.2$ for each

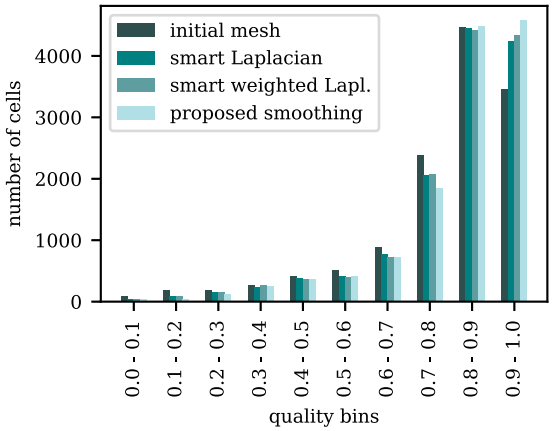


Fig. 26. Comparison of quality distributions after three iterations for Example 5.

Table 7
Comparison of quality values after three iterations for Example 5.

Methods	q_{avg}	$q < 0.2$	time per inc.
initial mesh	0.785	273	–
smart Laplacian	0.809	134	0.42 s
smart weighted Laplacian	0.811	131	0.63 s
proposed smoothing	0.819	75	1.05 s
proposed smoothing (extended)	0.808	65	1.07 s

Table 8
Comparison of quality values after three iterations for Example 6.

Method	q_{avg}	$q < 0.2$	time per inc.
initial mesh	0.724	938	–
smart Laplacian	0.739	503	1.32 s
smart weighted Laplacian	0.740	481	1.79 s
proposed smoothing	0.747	296	2.52 s
proposed smoothing (extended)	0.742	213	2.58 s

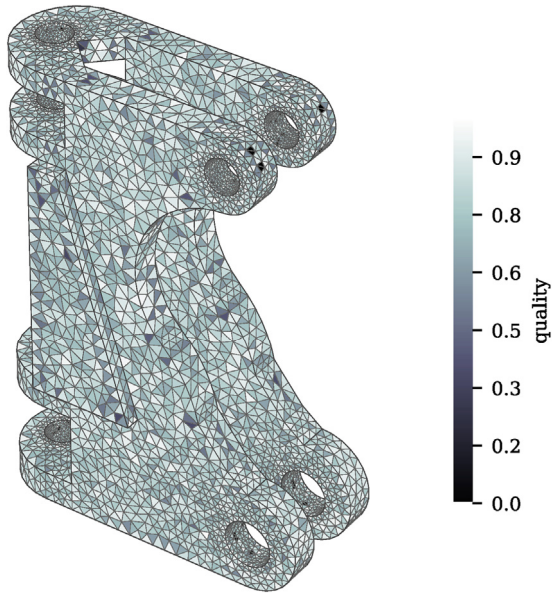


Fig. 27. Initial mesh for Example 6.

method. It is shown that from 938 elements with $q < 0.2$ only 293 remained after 3 iterations of the proposed method. The application of the extended version (with $\alpha = 1$) managed to reduced this number to 213. These numbers are well below the results from the Laplacian-based methods employed in this example. Table 8 also presents the average time per increment. It is noted that even for this case, the time spent by the proposed algorithm is not much higher than the other methods.

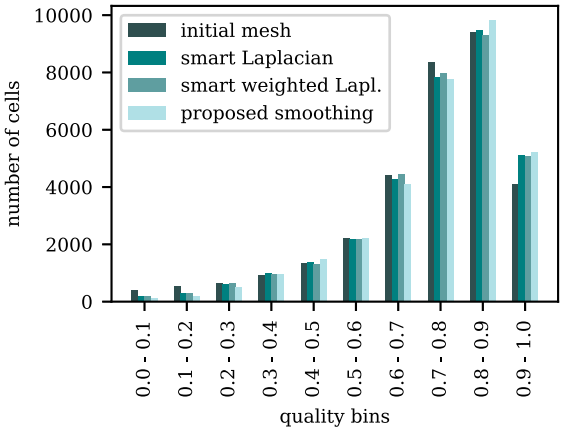


Fig. 28. Comparison of quality distributions after three iterations for Example 6.

4. Conclusions

This paper presented a general mesh smoothing method for finite elements. The method deals with two and three-dimensional meshes with virtually any element type and even meshes with mixed element types. To evaluate the quality of different element types, the paper also introduces a broad quality function. The method works by solving a deformation analysis where force boundary conditions are calculated aiming to deform each element into an optimally placed reference shape. An approach based on the singular value decomposition is used to find the optimal position and orientation for reference elements. For meshes with very low quality elements an extended approach is presented. This aims to prioritize the enhancement of elements with the lowest quality while slightly reducing the quality of best ones.

Several application examples in two and three-dimensions were presented and analyzed in order to demonstrate the capabilities of the proposed algorithm. In all cases good improvements were obtained with very few iterations (two or three); yet in some cases just one iteration may be needed. The improvements were even more remarkable in structured meshes. Also, for the same number of iterations in comparative tests, the proposed method provided greater improvements when compared to conventional Laplacian-based methods, especially by reducing the number of poor-shaped elements.

Regarding the computation time, it was fairly small for a physics-based method; especially if we consider the smoothing process as a pre-processing step in a finite element analysis. In addition, the authors envision the extension of the proposed algorithm to the use of parallel computing.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.finel.2019.01.010.

References

[1] I. Babuška, A.K. Aziz, On the angle condition in the finite element method, SIAM J. Numer. Anal. 13 (2) (1976) 214–226.
[2] H. Oh, J. Lim, A simple error estimator for size and distortion of 2d isoparametric finite elements, Comput. Struct. 59 (6) (1996) 989–999.
[3] N. Lee, K. Bathe, Effects of element distortions on the performance of isoparametric elements, Int. J. Numer. Methods Eng. 36 (20) (1993) 3553–3576.
[4] J. Shewchuk, What is a good linear finite element? interpolation, conditioning, anisotropy, and quality measures, in: 11th International Meshing Roundtable, Sandia National Laboratories, Ithaca, New York, 2002, pp. 115–126.
[5] J. Kim, S.P. Sastry, S.M. Shontz, A numerical investigation on the interplay amongst geometry, meshes, and linear algebra in the finite element solution of elliptic pdes, Eng. Comput. 28 (4) (2012) 431–450.
[6] D.A. Field, Laplacian smoothing and delaunay triangulations, Commun. Appl. Numer. Methods 4 (6) (1988) 709–712.
[7] V. Parthasarathy, S. Kodiyalam, A constrained optimization approach to finite element mesh smoothing, Finite Elem. Anal. Des. 9 (1991) 309–320, https://doi.org/10.1016/0168-874x(91)90004-i.
[8] S.A. Canann, S. Muthukrishnan, R. Phillips, Topological improvement procedures for quadrilateral finite element meshes, Eng. Comput. 14 (2) (1998) 168–177.
[9] S.A. Canann, J.R. Tristano, M.L. Staten, An approach to combined laplacian and optimization-based smoothing for triangular, quadrilateral, and quad-dominant meshes, in: IMR, Citeseer, 1998, pp. 479–494.

[10] K. Shimada, Physically-based Mesh Generation: Automated Triangulation of Surfaces and Volumes via Bubble Packing, Ph.D. thesis, ME Dept., ME Dept., Massachusetts Institute of Technology, Cambridge, MA, 1993.
[11] R.E. Bank, R.K. Smith, Mesh smoothing using a posteriori error estimates, SIAM J. Numer. Anal. 34 (3) (1997) 979–997.
[12] L.A. Freitag, On combining laplacian and optimization-based mesh smoothing techniques, in: AMD Trends in Unstructured Mesh Generation, vol. 220, ASME, 1997, pp. 37–43.
[13] N. Amenta, M. Bern, D. Eppstein, Optimal point placement for mesh smoothing, J. Algorithm 30 (1999) 302–322, https://doi.org/10.1006/jagm.1998.0984.
[14] Z. Tian, K. Shimada, An angle-based approach to two-dimensional mesh smoothing, in: Proceedings, 9th International Meshing Roundtable, Sandia National Laboratories, Sandia National Laboratories, New Orleans, Louisiana, 2000, pp. 373–384.
[15] P.M. Knupp, Achieving finite element mesh quality via optimization of the jacobian matrix norm and associated quantities. part ia framework for surface mesh optimization, Int. J. Numer. Methods Eng. 48 (3) (2000) 401–420.
[16] L.A. Freitag, P.M. Knupp, Tetrahedral mesh improvement via optimization of the element condition number, Int. J. Numer. Methods Eng. 53 (6) (2002) 1377–1391.
[17] A. Menéndez-Díaz, C. González-Nicieza, A.E. Álvarez-Vigil, Improvement of quadrilateral meshes for discretization of tunnels, Comput. Geotech. 31 (2004) 47–56, https://doi.org/10.1016/j.compgeo.2003.10.001.
[18] A. Menéndez-Díaz, C. González-Nicieza, A.E. Álvarez-Vigil, Hexahedral mesh smoothing using a direct method, Comput. Geosci. 31 (2005) 453–463, https://doi.org/10.1016/j.cageo.2004.11.002.
[19] L.F. Diachin, P. Knupp, T. Munson, S. Shontz, A comparison of two optimization methods for mesh quality improvement, Eng. Comput. 22 (2006) 61–74, https://doi.org/10.1007/s00366-006-0015-0.
[20] H. Xu, T.S. Newman, An angle-based optimization approach for 2d finite element mesh smoothing, Finite Elem. Anal. Des. 42 (2006) 1150–1164, https://doi.org/10.1016/j.finel.2006.01.016.
[21] A.E. Yilmaz, M. Kuzuoglu, A particle swarm optimization approach for hexahedral mesh smoothing, Int. J. Numer. Methods Fluids 60 (1) (2009) 55–78.
[22] X. Jiao, D. Wang, H. Zha, Simple and effective variational optimization of surface and volume triangulations, Eng. Comput. 27 (2011) 81–94, https://doi.org/10.1007/s00366-010-0180-z.
[23] J. Park, S.M. Shontz, An alternating mesh quality metric scheme for efficient mesh quality improvement, Procedia Comput. Sci. 4 (2011) 292–301.
[24] L. Sun, G. Zhao, X. Ma, Quality improvement methods for hexahedral element meshes adaptively generated using grid-based algorithm, Int. J. Numer. Methods Eng. 89 (6) (2012) 726–761.
[25] D. Vartziotis, J. Wipper, M. Papadrakakis, Improving mesh quality and finite element solution accuracy by getme smoothing in solving the Poisson equation, Finite Elem. Anal. Des. 66 (2013) 36–52.
[26] R.J. Renka, Mesh improvement by minimizing a weighted sum of squared element volumes, Int. J. Numer. Methods Eng. 101 (11) (2015) 870–886.
[27] P. Wei, D. Lu, T. Huang, L. Wang, Hexahedral mesh smoothing via local element regularization and global mesh optimization, Comput. Aided Des. 59 (2015) 85–97.
[28] R.E. Jones, A self-organizing mesh generation program, J. Pressure Vessel Technol. 96 (3) (1974) 193–199, https://doi.org/10.1115/1.3454166.
[29] P.M. Knupp, Algebraic mesh quality metrics, SIAM J. Sci. Comput. 23 (1) (2001) 193–218.
[30] J. Robinson, Some new distortion measures for quadrilaterals, Finite Elem. Anal. Des. 3 (3) (1987) 183–197.
[31] A. Oddy, J. Goldak, M. McDill, M. Bibby, A distortion metric for isoparametric elements, Trans. Can. Soc. Mech. Eng. 12 (1988) 213–217.
[32] D.A. Field, Qualitative measures for initial meshes, Int. J. Numer. Methods Eng. 47 (4) (2000) 887–906.
[33] P.M. Knupp, Algebraic mesh quality metrics for unstructured initial meshes, Finite Elem. Anal. Des. 39 (3) (2003) 217–241, https://doi.org/10.1016/S0168-874X(02)00070-7 arXiv:arXiv:1011.1669v3.
[34] R. Osserman, The isoperimetric inequality, Bull. Am. Math. Soc. 84 (6) (1978) 1182–1238.
[35] L. Taylor, D. Flanagan, PRONTO3D: A Three-Dimensional Transient Solid Dynamics Program, 1989.
[36] S.H. Lo, A New mesh generation scheme for arbitrary planar domains, Int. J. Numer. Methods Eng. 21 (1985) 1403–1426 July 1984.
[37] J. Sarrate, J. Palau, A. Huerta, Numerical representation of the quality measures of triangles and triangular meshes, Commun. Numer. Methods Eng. 19 (7) (2003) 551–561, https://doi.org/10.1002/cnm.585.
[38] M. Berzins, A solution-based triangular and tetrahedral mesh quality indicator, SIAM J. Sci. Comput. 19 (6) (1998) 2051–2060, https://doi.org/10.1137/S1064827596305222, http://epubs.siam.org/doi/10.1137/S1064827596305222.
[39] P.P. Pébay, T.J. Baker, Analysis of triangle quality measures, Math. Comput. 72 (244) (2003) 1817–1840, https://doi.org/10.1090/S0025-5718-03-01485-6, http://www.ams.org/journal-getitem?pii=S0025-5718-03-01485-6.
[40] J. Robinson, CRE method of element testing and the Jacobian shape parameters∗, Eng. Comput. 4 (2) (1987) 113–118, https://doi.org/10.1108/eb023689 arXiv:NIHMS150003.
[41] F. Pascal Jean, P.L. George, Mesh Generation - Application to Finite Elements, Science, Hermès, 2000.
[42] P.P. Pebay, Planar quadrilateral quality measures, Eng. Comput. 20 (2) (2004) 157–173, https://doi.org/10.1007/s00366-004-0280-8.
[43] P.M. Knupp, C. Ernst, D.C. Thompson, C. Stimpson, P.P. Pebay, The Verdict Geometric Quality Library, Tech. rep.. Sandia National Laboratories, mar 2006https://doi.org/10.2172/901967https://doi.org/10.2172/901967.
[44] M.S. Shephard, M.K. Georges, Automatic three-dimensional generation by finite octree technique, Int. J. Numer. Methods Eng. 32 (1991) 709–749 December 1990.
[45] A. Liu, B. Joe, Relationship between tetrahedron quality measures, BIT Numer. Math. 34 (1994) 268–287 July 1993.
[46] V.N. Parthasarathy, C.M. Graichen, A.F. Hathaway, A comparison of tetrahedron quality measures, Finite Elem. Anal. Des. 15 (3) (1994) 255–261, https://doi.org/10.1016/0168-874X(94)90033-7.
[47] W. Kwok, Z. Chen, A simple and effective mesh quality metric for hexahedral and wedge elements, in: Proceedings of 9th International Meshing Roundtable, 2000, pp. 325–333. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.31.9186&rep=rep1&type=pdf.
[48] K.S. Arun, T.S. Huang, S.D. Blostein, Least-squares fitting of two 3-d point sets, IEEE Trans. Pattern Anal. Mach. Intell. (5) (1987) 698–700.
[49] K.J. Bathe, Finite Element Procedures, Prentice Hall, 2006.
[50] C. Geuzaine, J.-F. Remacle, Gmsh: a 3-D finite element mesh generator with built-in pre- and post-processing facilities, Int. J. Numer. Methods Eng. 79 (11) (2009) 1309–1331, https://doi.org/10.1002/nme.2579.
[51] J. Bezanson, A. Edelman, S. Karpinski, V.B. Shah, Julia, A fresh approach to numerical computing, Soc. Ind. Appl. Math. 59 (1) (2017) 65–98.
[52] A.C.O. Miranda, W.W.M. Lira, J.B. Cavalcante-Neto, R.A. Sousa, L.F. Martha, A three-dimensional adaptive mesh generation approach using geometric modeling with multi-regions and parametric surfaces, J. Comput. Inf. Sci. Eng. 13 (2) (2013) 021002, https://doi.org/10.1115/1.4024106, https://doi.org/10.1115/1.4024106.