4.4

κ-Compatible Tessellations*

Philippe P. Pébay and David Thompson

Sandia National Laboratories
P.O. Box 969, Livermore CA 94551, U.S.A.
{pppebay,dcthomp}@sandia.gov

Summary. The vast majority of visualization algorithms for finite element (FE) simulations assume that linear constitutive relationships are used to interpolate values over an element, because the polynomial order of the FE basis functions used in practice has traditionally been low – linear or quadratic. However, higher order FE solvers, which become increasingly popular, pose a significant challenge to visualization systems as the assumptions of the visualization algorithms are violated by higher order solutions. This paper presents a method for adapting linear visualization algorithms to higher order data through a careful examination of a linear algorithm's properties and the assumptions it makes. This method subdivides higher order finite elements into regions where these assumptions hold (κ-compatibility). Because it is arguably one of the most useful visualization tools, isosurfacing is used as an example to illustrate our methodology.

1 Introduction

People have been approximating solutions to partial differential equations (PDEs) ever since PDEs were conceived. Much more recently, a class of techniques known as *hp*-adaptive methods have been developed in an effort to converge to a solution faster than previously possible, or to provide more accurate approximations than traditional finite element simulation within the same amount of computational time. These new techniques can increase both the hierarchical (h) and polynomial (p) levels of detail – or degrees of freedom – during a simulation.

Once these solution approximations have been computed, they must be characterized in some way so that humans can understand and use the results. This paper develops a technique for partitioning higher-order cells in order to

*This work was supported by the United States Department of Energy, Office of Defense Programs. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed-Martin Company, for the United States Department of Energy under contract DE-AC04-94-AL85000.

characterize the behavior of their geometric and scalar field curvatures during post-processing. We say that such partitions are κ-*compatible*. While a past paper [8] has presented our software framework for creating these partitions, this paper presents a full description of the algorithm and a rigorous proof of the conditions under which it will work and terminate.

Currently, visualization techniques for quadratic and higher order FE solutions are very limited in scope and/or cannot guarantee that all topological features are captured [1, 3, 7, 8, 6]. These techniques are also limited in their applicability to a subset of visualization techniques. Moreover, although some production-level tools currently offer support for quadratic elements, they do not always do so correctly (*cf.* [8] for a discussion); in the case of isocontouring, consider for example the following scalar field:

$$\begin{aligned} \Phi_1 : \ [-1,1]^3 &\longrightarrow \mathbb{R} \\ (x,y,z)^T &\mapsto x^2+y^2+2z^2, \end{aligned}$$

interpolated over a single Q2 Lagrange element (hexahedron with degree 2 Lagrange tensor-product interpolation over 27 nodes), with linear geometry, where one is interested in the isocontours $\Phi_1^{-1}(1)$ and $\Phi_1^{-1}(2)$.

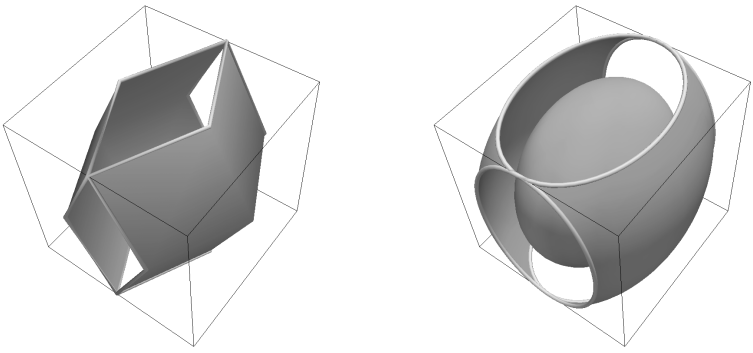

Fig. 1. Isocontours of Φ_1 for the isovalues 1 (cyan) and 2 (green): linear isocontouring approach (left), and our new topology-based approach (right).

As shown in Figure 1, left, the linear isocontouring approach (implemented here in ParaView) completely misses $\Phi_1^{-1}(1)$, because it is entirely contained within the cell. Meanwhile, the topology of $\Phi_1^{-1}(2)$ is correct, but its geometry is poorly captured. On the other hand, our new method (Figure 1, right) captures the correct topologies of both isocontours, and provides a much better geometric approximation of $\Phi_1^{-1}(2)$ than linear isocontouring does.

The lack of tools that are applicable to most visualizations of higher order element simulations, and that can guarantee correctness of the results, prevents analysts from exploiting such simulations. In this paper, we propose a solution to this problem.

1.1 Higher Order Finite Elements

Here we briefly review the FE method to develop notation used throughout the paper. Recall that the FE method approximates the solution, $f : \Omega \mapsto \mathbb{R}$, of some PDE as a set of piecewise functions over the problem domain, $\Omega \subset \mathbb{R}^{\mathrm{d}}$. Although Ω may be any general d-dimensional domain, we'll assume it is 3-dimensional. The fact that we have a piecewise approximant divides Ω into subdomains $\Omega_e \subseteq \Omega$, $e \in E_\Omega$ that form a closed cover of Ω. Each Ω_e is itself a closed set that is parametrized with a map (usually polynomial in form) from parametric coordinates $\mathbf{r} = (r, s, t) \in R \subset \mathbb{R}^3$ to geometric coordinates $\mathbf{x} = (x, y, z) \in \Omega_e$:

$$\Xi_e(\mathbf{r}) = B_{0,0,0} + B_{1,0,0}r + B_{0,1,0}s + B_{0,0,1}t + B_{1,1,1}rst + \cdots$$

where $B_{i,j,k} \in \mathbb{R}^3$ such that Ξ_e is invertible for all points of Ω_e. Therefore, the approximate solution may be written in terms of parametric coordinates (as Φ_e) or in terms of geometric coordinates: $f_e(\mathbf{x}) = \Phi_e \circ \Xi_e^{-1}(\mathbf{x})$, where

$$\begin{array}{rcl} \Phi_e : R \subset \mathbb{R}^3 & \longrightarrow & \mathbb{R} \\ (r, s, t)^T & \mapsto & \Phi_e(r, s, t) \end{array}$$

is the approximating function over Ω_e expressed in terms of parametric coordinates. Furthermore, we require that each Φ_e is polynomial in the parameters:

$$\Phi_e(\mathbf{r}) = A_{0,0,0} + A_{1,0,0}r + A_{0,1,0}s + A_{0,0,1}t + A_{1,1,1}rst + \cdots$$

These coefficients, $A_{i,j,k} \in \mathbb{R}$, are known as *degrees of freedom* (DOFs). Each one corresponds to a particular modal shape, and sets of modal shapes can be grouped together into nodes by the regions of parameter-space over which they have an effect (a given corner, edge, face, or the interior volume).

A global approximant $f(\mathbf{x})$ can then be constructed from the piecewise elemental approximants. This leaves only the matter of what to do where Ω_e and $\Omega_{j,j\neq e}$ intersect. Usually, these subdomains intersect over $(d-1)$-dimensional or lower regions (2-dimensional faces, 1-dimensional edges, and "0"-dimensional vertices in our case). In these regions, Φ is not well-defined since Φ_e and Φ_j may disagree. Usually, the FE method constrains Φ_e and Φ_j to be identical in these regions, however some methods such as the discontinuous Galerkin method do not require this and subsequently have no valid approximant in these regions.

For a given decomposition of Ω, the FE method may not converge to the correct (or indeed, any) solution. When Φ_e and Ξ_e are trilinear polynomials for all e, a technique called h-adaptation is often used to force convergence and/or increase solution accuracy. In this technique, some subdomains $E \subseteq E_\Omega$ are replaced with a finer subdivision E' such that $\bigcup_{e\in E} \Omega_e = \bigcup_{e\in E'} \Omega_e$ but $|E'| > |E|$[2]. Similarly, p-adaptation is the technique of increasing the

[2]In temporal simulations, we do allow $|E'| < |E|$ in regions where Ω has been adapted to some time-transient phenomenon.

order of polynomials Φ_e and/or Ξ_e rather than the number of finite elements. *hp*-adaptation is then some combination of *h*- and *p*-adaptation over Ω. In the end, the FE method provides an approximation to $f = \Phi \circ \Xi^{-1}$ by solving a collection of equations for coefficients (A and B in the examples above). Our task is then to characterize the maps Φ_e and Ξ_e in a way that aids human understanding of the solution.

The rest of this paper presents a method for partitioning higher-order finite elements into regions were visualization assumptions are satisfied.

2 Partitioning Finite Elements

Using these notation, we now examine what requirements must be satisfied by κ-compatible tessellations. A common assumption made by linear visualization algorithms is that critical points (points where all partial derivatives of f_e vanish) may only occur at vertices. This one assumption can show up in many different ways. Algorithms that iterate over an element's corner vertices to identify extrema (*e.g.*, thresholding) all make this assumption. Other examples include (but are not limited to) isosurfacing, cutting, and clipping.

This paper is concerned with tessellating finite elements into regions where the critical-points-at-vertices assumption holds as part of an overall strategy to adapt existing techniques to work with higher order elements. Because we use it as a running example of an important application driving the development of κ-compatible tessellation, we will focus on the prerequisites of the linear tetrahedral isosurfacing algorithm, keeping in mind that other visualization techniques result in the same set of constraints. The linear tetrahedral isosurfacing algorithm assumes:

($\mathrm{C1}^0$) each tetrahedron edge intersects a particular isocontour at most once,
($\mathrm{C2}^0$) no isocontour intersects a tetrahedron face without intersecting at least two edges of the face,
($\mathrm{C3}^0$) no isocontour is completely contained within a single tetrahedron, and
($\mathrm{C4}^0$) the map from parametric to geometric coordinates must be bijective.

Remark 1. Note that ($\mathrm{C4}^0$) only regards Ξ_e and can be restated as:

(C4) $\forall \mathbf{x} \in \Omega_e$, $\exists! \, \mathbf{r} \in R$ such that $\Xi_e(\mathbf{r}) = \mathbf{x}$.

In this paper, it is satisfied by hypothesis as we focus on the Φ_e map. However, in the context of higher order element mesh generation and modification, our methodology can be applied to Ξ_e to verify the correctness of each element.

Let's examine how changing to a higher-order interpolant affects these assumptions. For instance, the $\Phi_1^{-1}(1)$ isocontour in the example of §1 violates ($\mathrm{C3}^0$), which explains why it is entirely missed by linear isocontouring.

We now translate the criteria into requirements on Φ_e, that are slightly stronger for reasons that are discussed later on.

Proposition 1. *($C1^0$), ($C2^0$) and ($C3^0$) are respectively implied by:*

(C1) for each edge E of R, with direction vector (a_x, a_y, a_z),

$$a_x \frac{\partial \Phi_e}{\partial r} + a_y \frac{\partial \Phi_e}{\partial s} + a_z \frac{\partial \Phi_e}{\partial t} \neq 0 \quad \textit{over the interior of } E.$$

(C2) for each face F of R, with vector basis $((a_x, a_y, a_z), (b_x, b_y, b_z))$,

$$\begin{cases} a_x \dfrac{\partial \Phi_e}{\partial r} + a_y \dfrac{\partial \Phi_e}{\partial s} + a_z \dfrac{\partial \Phi_e}{\partial t} \neq 0 \\ b_x \dfrac{\partial \Phi_e}{\partial r} + b_y \dfrac{\partial \Phi_e}{\partial s} + b_z \dfrac{\partial \Phi_e}{\partial t} \neq 0 \end{cases} \quad \textit{over the interior of } F.$$

(C3) $\nabla \Phi_e \neq 0$ over the interior of R.

Proof. By definition, ($C1^0$), ($C2^0$) and ($C3^0$) are equivalently stated as:

($C1^0$) No restriction of Φ_e to an element edge has extrema interior to the edge unless the restriction is constant on the edge.
($C2^0$) No restriction of Φ_e to an element face has extrema interior to the face unless the restriction is constant on the face.
($C3^0$) Φ_e has no extrema inside the interior of the element unless the interpolant is constant over the entire element.

Being a polynomial function, Φ_e is C^1, and so are its restrictions to the edges and faces of R. Let E be an edge of R that passes through the point (p_x, p_y, p_z), with direction vector (a_x, a_y, a_z). This edge can be parametrized as follows:

$$\begin{aligned} \eta : I \subset \mathbb{R} &\longrightarrow \mathbb{R}^3 \\ t &\longmapsto (a_x t + p_x, a_y t + p_y, a_z t + p_z)^T , \end{aligned}$$

from which we obtain the restriction of Φ_e to E, $\Phi_{e|E} = \Phi_e \circ \eta : I \longrightarrow \mathbb{R}$. Its derivative $\mathrm{d}\Phi_{e|E} \in \mathcal{L}(\mathbb{R})$ at any arbitrary point $t_0 \in I$ is thus

$$\mathrm{d}\Phi_{e|E}(t_0) = \mathrm{d}\Phi_e\left(\eta(t_0)\right) \circ \mathrm{d}\eta(t_0) = \left(\frac{\partial \Phi_e}{\partial r}, \frac{\partial \Phi_e}{\partial s}, \frac{\partial \Phi_e}{\partial t} \right)\Bigg|_{\eta(t_0)} (a_x, a_y, a_z)^T \mathrm{d}t. \quad (1)$$

As $\Phi_{e|E}$ is C^1 over I, a necessary condition for $\Phi_{e|E}$ to have an extremum in t_0, interior to I, is that $\mathrm{d}\Phi_{e|E}(t_0) = 0$, *i.e.*,

$$a_x \frac{\partial \Phi_e}{\partial r}(t_0) + a_y \frac{\partial \Phi_e}{\partial s}(t_0) + a_z \frac{\partial \Phi_e}{\partial t}(t_0) = 0.$$

Similarly, let F be a face of R that passes through the point (p_x, p_y, p_z) and has basis $((a_x, a_y, a_z), (b_x, b_y, b_z))$. This face can be parametrized with two variables as follows:

$$\begin{aligned} \eta : U \subset \mathbb{R}^2 &\longrightarrow \mathbb{R}^3 \\ (u, v)^T &\longmapsto (a_x u + b_x v + p_x, a_y u + b_y v + p_y, a_z u + b_z v + p_z)^T . \end{aligned}$$

The derivative $\mathrm{d}\Phi_{e|F} \in \mathcal{L}(\mathbb{R}^2, \mathbb{R})$ of $\Phi_{e|F} = \Phi_e \circ \eta : U \longrightarrow \mathbb{R}$ is

$$\mathrm{d}\Phi_{e|F}(u_0, v_0) = \mathrm{d}\Phi_e(\eta(u_0, v_0)) \circ \mathrm{d}\eta(u_0, v_0) \tag{2}$$

$$= \left(\frac{\partial \Phi_e}{\partial r}, \frac{\partial \Phi_e}{\partial s}, \frac{\partial \Phi_e}{\partial t} \right)\Bigg|_{\eta(u_0,v_0)} \begin{pmatrix} a_x & b_x \\ a_y & b_y \\ a_z & b_z \end{pmatrix} \begin{pmatrix} du \\ dv \end{pmatrix}. \tag{3}$$

As $\Phi_{e|F}$ is C^1 over U, a necessary condition for $\Phi_{e|F}$ to have an extremum in (u_0, v_0), interior to U, is that $\mathrm{d}\Phi_{e|F}(u_0, v_0) = 0$, which may be restated as

$$a_x \frac{\partial \Phi_e}{\partial r}(\eta(u_0, v_0)) + a_y \frac{\partial \Phi_e}{\partial s}(\eta(u_0, v_0)) + a_z \frac{\partial \Phi_e}{\partial t}(\eta(u_0, v_0)) = 0$$
$$b_x \frac{\partial \Phi_e}{\partial r}(\eta(u_0, v_0)) + b_y \frac{\partial \Phi_e}{\partial s}(\eta(u_0, v_0)) + b_z \frac{\partial \Phi_e}{\partial t}(\eta(u_0, v_0)) = 0.$$

Finally, as Φ_e is C^1 over R, a necessary condition for Φ_e to have extremum in (r_0, s_0, t_0), interior to R, is that $\mathrm{d}\Phi_e(r_0, s_0, t_0) = 0$, *i.e.*, $\nabla\Phi_e(r_0, s_0, t_0) = 0$. □

Note that the proof mainly relies on the fact that, for a C^1 function over an open domain, an extremum is a critical point. However, the converse is not true and thus each (Ci) is stronger than the corresponding (Ci^0). Therefore, using (Ci) rather than (Ci^0) can yield non-extremal critical points – just consider $r \mapsto r^3$ in 0. In order to eliminate such "false positives", we would need to evaluate second- and higher-order derivatives, which would incur further computational costs, and even this would not always suffice as degenerate cases may occur. Therefore, rather than degrading computational performance, the trade-off we make is to accept the stronger (Ci) conditions and extra points that are not required by the (Ci^0) conditions. This may even be beneficial in terms of geometric approximation, as non-extremal critical points can be the locus of important geometric features (*e.g.*, saddle points). In short, (Ci) conditions mean that all of the differences between the linear and higher order isocontouring implementations can be attributed to critical points of Φ_e.

2.1 Creating the Partition

In §2, we presented the requirements a tetrahedral element must meet for isocontouring algorithm to work. As we noted earlier, higher order elements that have non-simplicial domains (such as hexahedra, pyramids, etc.) will have to be decomposed into tetrahedra $\mathcal{T}$. However, these tetrahedra must additionally meet the requirements (C1) to (C4). As explained in Remark 1, (C4) is satisfied by hypothesis; therefore the partition is subdivided until it meets criteria (C1), (C2), and (C3). Because not only a single scalar field, but a set κ of such fields may be of interest, these criteria must be satisfied for all fields in κ. Once this is achieved, the final partition is said to be κ-*compatible*. For the sake of legibility, we only discuss here the case where $\kappa = \{\Phi\}$, but the method remains the same when κ is not a singleton.

From now onwards, **it is assumed that all critical points are isolated**. This requirement is necessary so that the set of all critical points is finite, since a polynomial function can only have a finite number of isolated critical points; its implications are discussed at the end of this section. In this context, the general scheme of our method applied to the input parameters M (initial mesh) and Φ (field interpolated over M) is

PARTITION-MESH(M, Φ)
1 $C \leftarrow$ DOF-CRITICALITIES(M, Φ)
2 $(T_0, S) \leftarrow$ TRIANGULATE-BOUNDARIES(M, C)
3 CORRECT-TRIANGLE-TOPOLOGY(M, Φ, S, T_0)
4 $(T_1, S) \leftarrow$ TETRAHEDRALIZE-INTERIOR(M, T_0, C)
5 CORRECT-TETRAHEDRAL-TOPOLOGY(M, Φ, S, T_1)
6 **return** T_1

The output of the scheme is a tetrahedral subdivision T_1 of M. We now discuss each step in details,and theoretically establish the validity of the approach.

DOF-CRITICALITIES

This algorithm locates the critical points of Φ inside each element and of the restrictions of Φ to all element faces and edges. It takes M and Φ as inputs and yields a set C of critical points. BDY$^1(R)$ and BDY$^2(R)$ respectively denote the set of 1- and 2-dimensional boundaries of the parametric domain R.

DOF-CRITICALITIES(M)
1 **for** $e \leftarrow |M|$
2 **do** Find critical points of Φ_e in R
3 Store critical points indexed by volumetric DOF node.
4 **for** $f_i^2 \in$ BDY$^2(R)$
5 **do if** $\nabla\Phi_{e|f_i^2} = 0$ not marked,
6 **then** Find critical points of $\Phi_{e|f_i^2}$
7 Store critical points of $\Phi_{e|f_i^2}$ in $C_{f_i^2}$
8 Mark $\Phi_{e|f_i^2}$ as done
9 **for** $f_i^1 \in$ BDY$^1(R)$
10 **do if** $\nabla\Phi_{e|f_i^1} = 0$ not marked,
11 **then** Find critical points of $\Phi_{e|f_i^1}$
12 Store critical points of $\Phi_{e|f_i^1}$ in $C_{f_i^1}$
13 Mark $\Phi_{e|f_i^1}$ as done
14 **return** $C = \left(\cup_i C_{f_i^1}\right) \cup \left(\cup_i C_{f_i^2}\right)$

Note that, because finding critical points is a time-consuming process, we do not wish to process shared edges or faces multiple times. This extra work is avoided by storing critical points indexed by the DOF with which they are associated – critical points on a face are stored with the index used to

retrieve the coefficients for that face's degrees of freedom, and likewise for edges. Therefore, DOF-Criticalities operates on the mesh a whole, and not independently on each element. The issue of how to actually find the critical points is a complex and challenging problem of its own. We have not specifically researched this issue, and we handle it as follows:

- for edge critical points, where the problem amounts to finding all roots of a polynomial in a bounded interval, we have implemented exact solvers for up to quartic equations (and hence, quintic interpolants), and a Lin-Bairstow solver for higher order equations;
- for face and body critical points, where the problem amounts to finding all roots of a polynomial system within a bounded domain, we solve exactly if the system is linear, and otherwise make use of the PSS package [4, 5]. However, we think that this aspect deserves much further investigation.

If one assumes that the polynomial system solver always terminates in finite time, then DOF-Criticalities does as well. As mentioned earlier, because restrictions of the field to edges and faces are marked as they are done, each is processed only once even when it is shared by several elements.

Remark 2. The methodology we present requires the ability to detect all critical points on arbitrary line segments and triangular faces in the domain of an element. Most polynomial system solvers require a power-basis representation of a system to be solved and that is not usually how finite elements are represented. Given that we wish to perform this change of basis as infrequently as possible, it behooves us to find a way to derive the restriction of Φ_e to a line or face from the full representation of Φ_e.

Triangulate-Boundaries

Once the critical points have been located, the second step of our scheme consists in triangulating the two-dimensional boundaries of all elements. This ensures that all volumetric elements that reference a particular face use the same triangulation – otherwise our model could have cracks along element boundaries[3]. In order to satisfy (C2) on a given element e, a restriction of Φ_e to a face of the tessellation of e is not permitted to have a critical point; therefore, we "eliminate" the critical points of the restriction of Φ_e to the faces of e by inserting them in the list of points to be triangulated. The algorithm Triangulate-Boundaries thus takes the mesh M and its related set of critical points C as inputs, and returns a triangulation T_0 of the set of faces in M. In this algorithm, the method Face-Center takes a face as its input and returns its parametric center, and $\text{Star}^2(c, Q_1, \ldots, Q_n)$ creates a

[3] Discontinuous Galerkin elements can be accommodated by using different indices (as opposed to a shared index i) for edges and faces of adjoining elements. Cracks would occur, but they would be faithful representations of the interpolant discontinuity.

triangulation composed of triangles cQ_1Q_2,..., cQ_nQ_1 (with the requirement that c is contained in the interior of the convex hull of $\{Q_1, \ldots, Q_n\}$).

TRIANGULATE-BOUNDARIES(M, C)

```
1  for f_i^2 ← each 2-boundary of every 3-D finite element
2      do if |C_{f_i^2}| > 0
3           then c ← C_{f_i^2,0}
4           else c ← FACE-CENTER(BDY_i^2(IR))
5         T_i ← ∅
6         Q ← corner points of face i ⋃ isolated critical points
          of all bounding edges of face f_i^2, ordered in a
          counterclockwise loop around f_i^2.
7         for j ∈ {0, ..., |Q| − 1}
8             do Insert STAR^2(c, Q_j, Q_{(j+1) mod |Q|}) into T_i
9         C_{f_i^2}' ← {C_{f_i^2} \ C_{i,0}}
10        for c ∈ C_{f_i^2}'
11            do Find t ∈ T_i such that c ∈ t
12               Remove c from C_{f_i^2}'
13               Remove t from T_i
14               Subdivide t into 2 or 3 triangles t_k
15               Insert t_k into T_i
16 return T_0 = ∪_i T_i
```

All sets involved in TRIANGULATE-BOUNDARIES are finite, and no recursion is involved. Therefore, this procedure terminates in finite time. In addition,

Proposition 2. *Upon completion of* TRIANGULATE-BOUNDARIES*, (C1) is satisfied on the edges of the triangulation T_0 that either belong to M or are subdivisions of edges or faces of M, and (C2) is satisfied across T_0.*

Proof. By construction, TRIANGULATE-BOUNDARIES inserts in T_0 all critical points of restrictions of Φ to edges of M. Since this process cannot create novel critical points on these edges, the first part of the proposition ensues. Compliance with (C2) across T_0 is ensured because all critical points of restrictions of Φ to faces of M have been inserted as vertices of T_0, and no new such critical point may have been created. □

Note that that there is no guarantee that (C1) is satisfied across T_0 upon completion of TRIANGULATE-BOUNDARIES, as shown in the following example:

Example 1. Consider a face f_i, illustrated in Fig. 2(a), such that the restriction of the field to the interior of f_i has 3 critical points (denoted a, b, and c), and each of the restrictions of the field to the edges of f_i has at least one critical point (denoted d, f, h, i, j, and k). The triangulation displayed in Fig. 2(b) is obtained once all STAR^2 procedures have performed by taking a as the first internal critical point to be inserted. The final tessellation, shown in Fig. 2(c), has eliminated the remaining interior critical points b and c by making them

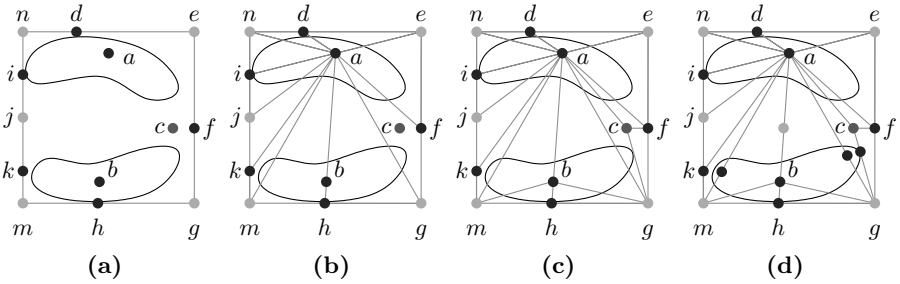

Fig. 2. An example face f_i^2 with critical points shown as blue dots (maxima), red dots (saddles), and green dots (minima). **(a)** The input to Triangulate-Boundaries. **(b)** Resulting triangulation of f_i^2 after all Star^2 procedures of Triangulate-Boundaries have been performed. **(c)** The triangulation at the completion of Triangulate-Boundaries. **(d)** The new critical points introduced by the first stage of Correct-Triangle-Topology.

nodes of the triangulation; however, as illustrated in Fig. 2(d), new critical points have appeared, on the restrictions of Φ_e to edges ab, ag, am, and cg.

Remark 3. Line 14 of Triangulate-Boundaries allows for 2 different ways to subdivide a triangle, depending on whether the face critical point lies within or on the boundary of the triangle; for instance, in Fig. 2(c), triangles ahm and afg are split in, respectively, 2 and 3 triangles. In practice, to avoid unnecessary creation of quasi-degenerate triangles, the implementation of Triangulate-Boundaries uses a predefined threshold (that can be related to the distance of the critical point to the closest triangle edge, or to a triangle quality estimate of the subdivided triangles) below which a critical point is moved to the appropriate edge; for instance, in the example illustrated in Fig. 2(c), point b is considered as belonging to ah, even if it slightly off.

Correct-Triangle-Topology

This algorithm searches for critical points in the restriction of Φ_e to each unmarked edge of T_0. When points are found, they are inserted into T_0, and iteratively repeats the procedure until (C1) and (C2) are satisfied throughout the entire triangulation of the faces of M. Fig. 3(b) shows this procedure applied to Example 1.

CORRECT-TRIANGLE-TOPOLOGY(M, Φ, S, T_0)

```
1  while S not empty
2      do Pop t from S
3         C ← ∅
4         for e ← marked edges of t
5             do Insert critical points of e into C
6         for c ← C
7             do Find t ∈ T_1 s. t. c ∈ t
8                Remove t from T_1 and S
9                U ← STAR^2(c, t)
10               for t' ← U
11                   do if MARK-TRIANGLE(t')
12                         then Push t' onto S
13                              Insert t' into T_0
```

Proposition 3. *If, for all e in M, all critical points of the restriction of Φ_e to any face of e are isolated, then* CORRECT-TRIANGLE-TOPOLOGY *terminates. In addition, upon termination, (C1) and (C2) are satisfied across T_0.*

Proof. For brevity, the proof is not provided here: it can readily be obtained by reducing the proof of Proposition 5 to the 2-dimensional case.□

Remark 4. Because there is no need to refine below a certain size for visualization purposes, our implementation uses a tolerance $\varepsilon \in \mathbb{R}_+$ so that, in line 5, a new critical point p of the restriction of Φ to a marked edge is inserted only if there is no p' in C such that $|p - p'| < \varepsilon$. Therefore, the procedure may terminate before (C1) and (C2) are fully satisfied, but are satisfied up to ε.

TETRAHEDRALIZE-INTERIOR

Each element interior is now tetrahedralized, and although we treat the whole mesh, there is here no issue of inter-element consistency, as this part of the scheme only regards element interiors. As in TRIANGULATE-BOUNDARIES, we "eliminate" critical points of Φ_e that are interior to e by adding them to the set of points to be tetrahedralized. Therefore, the tetrahedralization of each element e is constrained by the triangulations of the faces of e that result from CORRECT-TRIANGLE-TOPOLOGY, and by the critical points of Φ_e that are interior to e. Additionally, when the finite element is starred into a set of tetrahedra, we know that the triangular base of each tetrahedron and its 3 bounding edges will not have any critical points since those have already been identified and inserted into the triangulation of the two-dimensional boundary of the element. However, the remaining 3 faces and 3 edges must be marked because Φ_e restricted to their domain may contain critical points. This is accomplished by MARK-TETRAHEDRON, which sets a bit code for each edge and face not on the base of the given tetrahedron (which must be properly oriented when passed to the subroutine). The algorithm is then as follows:

```
TETRAHEDRALIZE-INTERIOR(M, T_0, C)
 1  S ← ∅
 2  T_1 ← ∅
 3  for e ← |M|
 4       do Let T ⊆ T_0 be all triangles on BDY^2(R)
 5          if |C_e| > 0
 6             then c ← C_{e,0}
 7             else  c ← ELEMENT-CENTER(R)
 8          V ← STAR^3(c, T)
 9          for t ← V
10               do if MARK-TETRAHEDRON(t)
11                     then Push t onto S
12          for c ∈ {C_e \ C_{e,0}}
13               do Find t ∈ V s. t. c ∈ t
14                  Remove t from V and S
15                  U ← STAR^3(c, t)
16                  for t' ← U
17                       do if MARK-TETRAHEDRON(t')
18                             then Push t' onto S
19                          Insert t' into V
20          Insert V into T_1
21  Return (T_1, S)
```

All sets involved in TETRAHEDRALIZE-INTERIOR are finite, and no recursion is involved. Therefore, this procedure terminates in finite time. In addition,

Proposition 4. *(C3) is satisfied across the tetrahedralization T_1 upon completion of* TETRAHEDRALIZE-INTERIOR. *Moreover, any sub-tetrahedralization of T_1 satisfies (C3) as well.*

Proof. Let p be a critical point of the field Φ; then, 2 cases may occur:

1. p is contained in the interior of an element $e \in M$. In this case, p belongs to C_e (and to no other $C_{e'}$) and thus, thanks to STAR^3, p is a tetrahedron vertex in T_1.
2. p is contained on the boundary of an element $e \in M$. In this case, it is also a lower-dimensional critical point, *i.e.*, a critical point for the restriction of Φ_e to one of its faces or edges, because the fact $\mathrm{d}\Phi_e$ vanishes in p ensures that the left hand side of (1) (if p is on an edge) or (2) (if p is on a face) vanishes as well. Therefore, p belongs to C_{f_i} for an edge f_i^1 or a face f_i^2 and hence has been made a triangle vertex of T_0 by TRIANGULATE-BOUNDARIES. Since all triangles of T_0 become tetrahedral faces in T_1, then p is a tetrahedron vertex in T_1.

In both cases, upon completion of TETRAHEDRALIZE-INTERIOR, p is a tetrahedron vertex in T_1. Therefore, for all $t \in T_1$, p is not a critical point of Φ_t interior to t. As this is true for any critical point p of the field Φ, it follows

that T_1 satisfies (C3). Finally, as (C3) is satisfied on any tetrahedron $t \in T_1$, it is also satisfied on any tetrahedral subdivision of t: indeed, if one could find a tetrahedron $t' \subset t$ and a critical point p of Φ contained in the interior of t', then p would also be in the interior of t, which would contradict the hypothesis; *ad absurdum*, the result ensues.□

Correct-Tetrahedral-Topology

A this point, a brief summary of what has been obtained through the 3 first stages of our scheme will most likely be useful to the reader. A tetrahedralization T_1 of the initial mesh M has been obtained, such that

- (C3), and (C4) by hypothesis, are satisfied;
- (C1) and (C2) are satisfied on all edges and faces of T_1 that either belong to M or are subdivisions of edges or faces of M.

However, there is no guarantee that (C1) and (C2) are satisfied on all edges and faces of T_1 that are not included (*stricto* or *lato sensu*) in M.

Example 2. Consider the scalar field defined by $\Phi_2(x, y, z) = x^2 - y^2 + z$ over a single Lagrange Q2 element with linear geometry and coordinates in $[-1, 1]^3$, as illustrated in Fig. 3(c): each restriction of Φ_2 to the faces perpendicular to the z-axis has a critical point at the corresponding face center (labeled 7 and 11), and each restriction of Φ_2 to the edges perpendicular to the z-axis has a critical point at the corresponding edge midpoint (labeled 6, 23, 20, 24, 22, 14, and 26 for those that are visible), whereas Φ_2 proper does not have any critical point. Upon completion of Tetrahedralize-Interior, all of these points have been inserted in T_1 which contains, among others, edges from the element center to points 6, 20, 22, and 26. It is easy to check that the restrictions of Φ_2 to each of these edges have a critical point at the edge midpoint, and thus (C1) is not satisfied across T_1, that therefore requires further modification.

We must therefore examine how, in general, T_1 can be modified so in order to satisfy (C1) and (C2). A natural question is to wonder whether it is possible to perform a series of edge-face flips on the T_1 so that the final tessellation satisfies the desired properties.

Example 3. Given the triangulation of a face in Fig. 2(c) for which additional critical points (of the restrictions of the field to some of the new edges) have appeared, one can easily perform a series of edge flips so that the final connectivity satisfies (C1), as shown in the resulting tessellation in Fig. 3(a).

Nevertheless, it is unclear whether it is always possible to retrieve a tessellation that satisfies (C1) using only edge flips. Although this may be the case, we have not further explored this path, because the matter is more complicated as both (C1) and (C2) must be satisfied in a problem that is intrinsically three-dimensional: for instance, although any 2-D triangulation can always be converted to a Delaunay triangulation by a finite sequence of edge flips, but

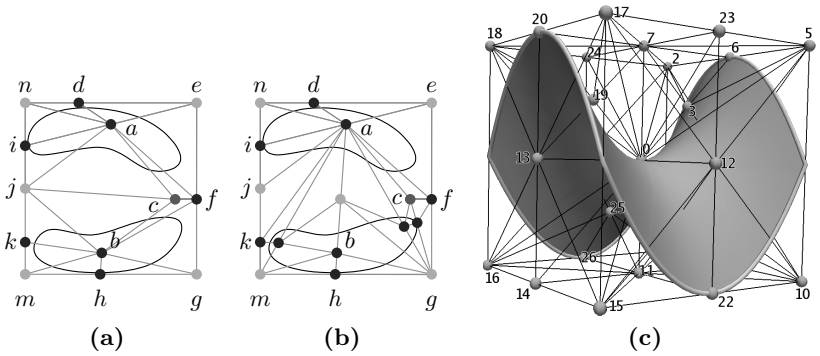

Fig. 3. (a) A triangulation satisfying (C1) with edge flips from Fig. 2(c). **(b)** Another triangulation satisfying (C1) using Correct-Triangle-Topology and the same initial tessellation. **(c)** Topology-based tetrahedralization of a single Lagrange Q2 element with linear geometry for the field $\Phi_2(x, y, z) = x^2 - y^2 + z$.

this result does **not** extend to 3-D tetrahedralizations [2], making us skeptical of the flipping approach. We therefore took a different route, guided by Proposition 4, as not only T_1 satisfies (C3), but we also know that will not be altered by subsequent subdivisions of T_1. Rather than attempting to identify a set of "problem" entities and prove that they may always be flipped into a satisfactory configuration, we introduce each critical point of a restriction of Φ to a new entity into the tessellation. Any edges and faces created by this operation must then be examined as well for critical points of Φ restricted to their respective domains. However, it is only necessary to focus on critical points of the restrictions of Φ to the previously marked edges and faces, as others already satisfy (C1) and (C2), thanks to Triangulate-Boundaries.

Correct-Tetrahedral-Topology(M, Φ, S, T_1)

```
 1  while S not empty
 2      do Pop t from S
 3         C ← ∅
 4         for e ← marked edges of t
 5             do Insert critical points of e into C
 6         for f ← marked faces of t
 7             do Insert critical points of f into C
 8         for c ← C
 9             do Find t ∈ T1 s. t. c ∈ t
10                Remove t from T1 and S
11                U ← Star^3(c, t)
12                for t' ← U
13                    do if Mark-Tetrahedron(t')
14                          then Push t' onto S
15                       Insert t' into T1
```

Remark 5. The same approach as that explained in Remark 4 is used here, using a tolerance to avoid refining below a certain size. Therefore, and for the same reason, this allows the algorithm to possibly terminate faster than normally possible if all critical points created on new edges and faces and the refinement proceeds had to be inserted without regard to the distance to an already inserted vertex.

Proposition 5. *If, for all e in M, all critical points of the restriction of Φ_e to any arbitrary face are isolated, then* Correct-Tetrahedral-Topology *terminates. In addition, upon termination, (C1), (C2) and (C3) are satisfied.*

Proof. To establish this result, it is sufficient to make sure the algorithm terminates, starting from any arbitrary face of an arbitrary element in M. So, let f_i be one of the faces of an arbitrary $e \in M$, and we then shall prove that Correct-Tetrahedral-Topology terminates.

First, remark that if the restriction $\Phi_{e|f_{ij}}$ of Φ_e to one edge f_{ij} of f_i has a non-isolated critical point, then the derivative of $\Phi_{e|f_{ij}}$ vanishes along a nonempty open segment of f_{ij}, and therefore has an infinity of zeros. Because this derivative is itself a univariate polynomial function, it can thus only be zero everywhere, and thus $\Phi_{e|f_{ij}}$ is constant along the edge. Therefore, the only case when non-isolated critical points along a bounding edge of f_i arises is when the interpolant is constant along that edge, and therefore no other points than its endpoints are contained in P. P is indeed a finite set, as polynomials can only have a finite number of isolated critical points.

Now assume the restriction $\Phi_{e|f_i}$ of Φ_e to the interior of f_i has $n \in \mathbb{N}^*$ critical points. The innermost loop of Correct-Tetrahedral-Topology will insert these n points, and yield a triangulation of f_i in $N \in \mathbb{N}^*$ triangles $t_{i,k}$, such that $\cup_{k=1}^{N} t_{i,k} = f_i$ and

$$(\forall\, 1 \leq k, k' \leq N) \quad k \neq k' \iff \overset{\circ}{t}_{i,k} \cap \overset{\circ}{t}_{i,k'} = \emptyset,$$

where all the points of C are vertices of some of $t_{i,k}$ ($\overset{\circ}{t}$ is the interior of t in the sense of the natural topology induced on t by embedding it in $\mathbb{R}^2$). Therefore, none of the restrictions of Φ_e to $t_{i,k}$ has an internal critical point (otherwise this point would belong to C, which is impossible because all points of C are vertices of some of $t_{i,k}$).

However, the restriction of Φ_e to some edges of this triangulation of f_i may have critical points[4]. Denote η such an edge. If the restriction of Φ to η has any non-isolated critical point, then the same argument as above holds and thus the corresponding edges do not need to be further refined. On the contrary, if such an edge critical point is isolated (in this case, the edge must be internal to f_i, as all isolated critical points along the edges of f_i have been inserted priorly), then Correct-Tetrahedral-Topology recursively proceeds on

[4] In other words, the subdivision of f_i cannot create new face critical points, but it can create new edge critical points.

η. However, the process terminates because all face critical points are supposed to be isolated according to the hypothesis. Therefore, for each critical point p_i of the restriction of Φ_e to f_i, there exists a neighborhood of p_i in which all directional derivatives of Φ_e are nonzero and thus, there exists a finite number of triangle subdivisions after which no edge critical points are left (because such a critical point implies one directional derivative is equal to 0).

Finally, upon completion of the algorithm, for the same reasons as for Tetrahedralize-Interior, (C3) is satisfied for each tetrahedron of the final partition (the final tetrahedra are subdivisions of initial tetrahedra that all satisfied (C3), according to Proposition 4).□

Example 4. Consider the same case as Example 2: the execution of Correct-Tetrahedral-Topology results in the insertion of the 4 previously mentioned edge midpoints (3 of which are visible in Fig. 3(c), labeled 3, 19, and 25) in a sub-tetrahedralization of T_1 that becomes the final tessellation, across which conditions (C1) through (C4) are satisfied. In this case, only one level of refinement was necessary, as none of the edges and faces created upon insertion of the 4 aforementioned edge midpoints violates (C1) or (C2).

Remark 6. Note that the hypothesis of Proposition 5 is very stringent, as it is not limited to faces of the mesh, but extends to all possible faces. In fact, it is sufficient that the restrictions of Φ_e to any face of the successive partitions of e only has isolated critical points. However, as this condition depends on the particular subdivision path, it is, albeit weaker, more difficult to prove.

3 Application to Isocontouring

We now illustrate the application of our method to isocontouring, and the discuss an additional benefit of κ-tessellation that comes as a side-effect. Consider the following scalar field:

$$\begin{array}{rcl} \Phi_3 : \ [-1,1]^3 & \longrightarrow & \mathbb{R} \\ (x,y,z)^T & \mapsto & x^2+y^2+z^2(z-1), \end{array}$$

interpolated over a single Q3 Lagrange element with linear geometry, i.e., a tricubic (hence with total degree 9) cell obtained by tensorization of cubic Lagrange interpolants. The test consists of representing the 0-isocontour of Φ_3, which is tricky because an entire lobe of the resulting isosurface is contained within the element.

Figure 4, left, shows that if a linear isocontouring marching cubes technique is applied (after uniform subdivision of the hexahedron into 48 tetrahedra), then a substantial part of $\Phi^{-1}(0)$ is missing. This example is interesting because the missing part of $\Phi^{-1}(0)$ is not a disconnected component and, therefore, (C3) is *not* violated for this particular isovalue. While intuition may suggest that a linear isocontouring technique should retrieve the correct

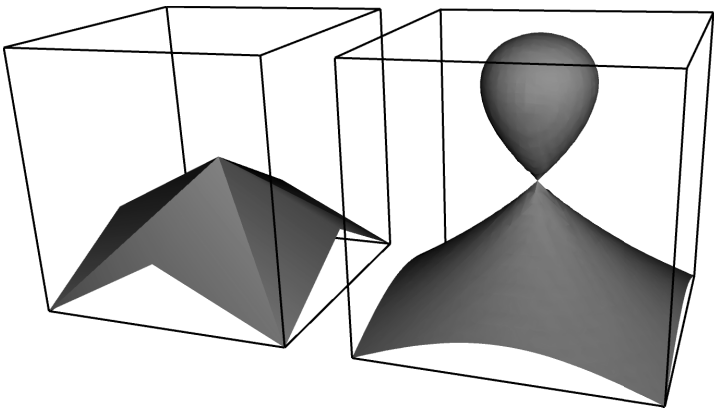

Fig. 4. Tricubic isocontouring of Φ_3 for the isovalue 0, using a linear isocontouring technique (left), and using our approach (right).

topology of $\Phi^{-1}(0)$ in the interior of the hexahedral element, in fact $\Phi^{-1}(0)$ is not a 2-dimensional submanifold of $\mathbb{R}^3$ at (and only at) point $(0, 0, 0)$ (where $\Phi^{-1}(0)$ is not simply connected). Therefore, $\Phi^{-1}(0)$ is not a surface and this causes the isocontouroing algorithm to fail, as illustrated. Moreover, the Implicit Function Theorem shows that for any value of α in $]-2, 0[$, $\Phi^{-1}(\alpha)$ is a surface, but can also easily check that for such values of α, it is not connected; in fact, it has 2 disjoint connected components, one of which is entirely contained in the interior of the element. Hence, in this case, (C3) is indeed violated; which also causes the linear isocontouring technique to fail (by missing this connected component).

From what we already know of our topology-based technique, we can expect it to properly retrieve $\Phi^{-1}(\alpha)$ for these problematic values of α in $]-2, 0[$, and it does indeed. However, what was not initially expected, is that it could fix the case when $\alpha = 0$, as we do not expect the technique to handle non-manifold isocontours. Nonetheless, as shown in Figure 4, right, our topology-based method is able to construct the correct topology of the non-simply connected $\Phi^{-1}(0)$. And in fact, this not an anecdotal effect valid for this example only, but we can see that this is always the case: by "eliminating" critical points from the final tessellation, the method ensures that (thanks to the Implicit Function Theorem), the isocontour is locally a 2-dimensional manifold within the interior of each element. In other words, the scheme produces a tessellation that not only satisfies criteria (C1) through (C4), but additionally ensures that the isocontour is indeed a surface inside each element. Note that this extends to the lower-dimensional case, for the same reason: the isocontours inside the faces of the final tessellation are simply connected curves.

4 Conclusions

We have outlined an algorithm for partitioning finite elements to which is associated a scalar field into a κ-compatible tessellation, and proved that it works and terminates under a limited set of assumptions. This technique allows to easily adapt visualization and other post-processing tools to higher order elements. We have illustrated this methodology with the isocontouring operation.

Future work will include estimating the computational complexity of the algorithm as a function of input parameters such as the order of the interpolant, and improving multivariate polynomial system solution strategies.

References

1. Michael Brasher and Robert Haimes. Rendering planar cuts through quadratic and cubic finite elements. In *Proceedings of IEEE Visualization*, pages 409–416, October 2004.
2. Barry Joe. Three dimensional triangulations from local transformations. *SIAM Journal on Scientific and Statistical Computing*, 10:718–741, 1989.
3. Rahul Khardekar and David Thompson. Rendering higher order finite element surfaces in hardware. In *Proceedings of the first international conference on computer graphics and interactive techniques in Australasia and South East Asia*, pages 211–ff, February 2003.
4. Gregorio Malajovich. PSS 3.0.5: Polynomial system solver, 2003. URL `http://www.labma.ufrj.br:80/~gregorio`.
5. Gregorio Malajovich and Maurice Rojas. Polynomial systems and the momentum map. In *Proceedings of FoCM 2000, special meeting in honor of Steve Smale's 70th birthday*, pages 251–266. World Scientific, July 2002.
6. M. Meyer, B. Nelson, R.M. Kirby, and R. Whitaker. Particle systems for efficient and accurate finite element subdivision. *IEEE Trans. Visualization and Computer Graphics*, 13, 2007.
7. J.-F Remacle, N. Chevaugeon, E. Marchandise, and C. Geuzaine. Efficient visualization of high-order finite elements. *Intl. J. Numerical Methods in Engineering*, 69:750–771, 2006.
8. W. J. Schroeder, F. Bertel, M. Malaterre, D. C. Thompson, P. P. Pébay, R. O'Bara, and S. Tendulkar. Framework and methods for visualizing higher-order finite elements. *IEEE Trans. on Visualization and Computer Graphics, Special Issue Visualization 2005*, 12(4):446–460, 2006.