SAND2007-3644C

κ-compatible Tessellations

Philippe P. Pébay[1] and David Thompson[2]

[1] Sandia National Laboratories*
P.O. Box 969, MS 9051, Livermore CA 94551, U.S.A.
pppebay@sandia.gov
[2] Sandia National Laboratories*
P.O. Box 969, MS 9152, Livermore CA 94551, U.S.A.
dcthomp@sandia.gov

1 Introduction

Finite element methods have advanced significantly since their conception several hundred years ago. 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. Finite element solvers that incorporate *hp*-adaptivity are becoming popular since they often converge to a solution with fewer total degrees of freedom than hierarchical adaptivity alone.

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 characterize the behavior of their geometric and scalar field curvatures during post-processing. We call these partitions curvature- or κ-compatible tessellations. While a past paper [6] has presented our software framework for creating these partitions, this paper contains a rigorous proof of the conditions under which the algorithm will work and terminate.

Currently, visualization techniques for quadratic, cubic, and higher order finite element solutions are very limited in scope [1, 3, 6]. This prevents analysts from exploiting such simulations. For instance, although some tools can currently render boundary surfaces of higher-order elements (see [6] for surface rendering examples), they do not always do so correctly. As an example,

*These authors were 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.

consider the following scalar field:

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

interpolated over a single Q^2 Lagrange element (hexahedron with degree 2 Lagrange tensor-product interpolation over 20 nodes), with linear geometry, where one is interested in the isocontours $\Phi_1 = 1$ and $\Phi_1 = 2$.

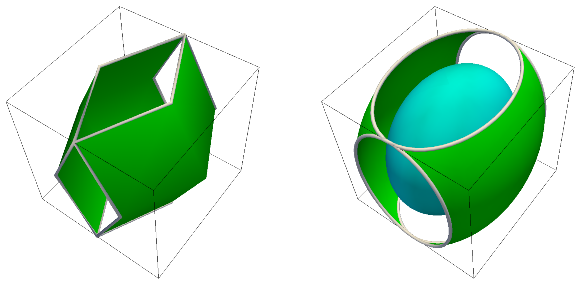
Fig. 1. Isocontours of Φ_1 for the isovalues 1 (cyan) and 2 (green): linear isocontouring approach (left), and our new topology-based approach (right).

As shown in Figure 1, left, the linear isocontouring approach (implemented here in ParaView) completely misses the $\Phi_1 = 1$ isocontour, because it is entirely contained within the cell. Meanwhile, the topology of the $\Phi_1 = 2$ isocontour 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 the $\Phi_1 = 2$ isocontour than linear isocontouring does.

1.1 Higher Order Finite Elements

Here we briefly review the finite element method to develop notation used throughout the paper. Recall that the finite element method approximates the solution, $f : \Omega \mapsto \mathbb{R}$, of some differential equation 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 parameterized with a map (usally 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 to the differential equation may be written in terms of parametric coordinates (as Φ_e) or in terms of geometric coordinates:

$$f_e(\mathbf{x}) = \Phi_e \circ \Xi_e^{-1},$$

where

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

Φ_e 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 finite element 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 finite element 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 $|E'| > |E|$[3]. Similarly, p-adaptation is the technique of increasing the order of polynomials Φ and/or Ξ rather than the number of finite elements. hp-adaptation is then some combination of h- and p-adaptation over Ω. In the end, the finite element 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 charactize the maps Φ and Ξ 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 that have guarantees on their derivatives and the restrictions of these derivatives to the boundaries of the regions. $\diamond$

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

2 Partitioning Finite Elements

Now that we've developed some notation for higher order elements, let's examine the application driving the development of 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 minima and maxima (*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. As the paper proceeds we'll show how these tessellations can be applied to adapt linear tetrahedral isocontouring to higher order isocontouring. The only requirement we impose is that the higher order interpolant have only isolated critical points (or that it has been perturbed slightly to meet this condition).

The linear tetrahedral isosurfacing algorithm assumes:

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

Let's examine how changing to a higher-order interpolant affects these assumptions.

Remark 1. (C4^0) only regards Ξ_e, and is satisfied by hypothesis. Note that it can be restated as:

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

Note that the example provided in § 1 and illustrated in Figure 1 violates (C3^0) in the case of the $\Phi_1 = 1$ isocontour. This is precisely why the corresponding isosurface is entirely missed by the linear isocontouring technique.

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

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

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

$$a_x \frac{\partial \Phi_e}{\partial r} + a_y \frac{\partial \Phi_e}{\partial s} + a_z \frac{\partial \Phi_e}{\partial t} \neq 0$$

over the interior of E.

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

$$\begin{cases} a_x \dfrac{\partial \Phi_e}{\partial r} + a_y \dfrac{\partial \Phi_e}{\partial s} + a_z \dfrac{\partial \Phi_e}{\partial t} \neq 0 \\ b_x \dfrac{\partial \Phi_e}{\partial r} + b_y \dfrac{\partial \Phi_e}{\partial s} + b_z \dfrac{\partial \Phi_e}{\partial t} \neq 0 \end{cases}$$

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) can be equivalently restated as follows:

(C1^0) No restriction of Φ_e to a finite element edge has extrema interior to the edge unless the restriction is constant on the edge.

(C2^0) No restriction of Φ_e to a finite 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 finite element unless the interpolant is constant over the entire element.

As Φ_e is a polynomial function, it is C^1 over R. 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 parameterized with one variable as follows:

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

from which we obtain the restriction of Φ_e to E:

$$\Phi_{e|E} = \Phi_e \circ \eta : I \longrightarrow \mathbb{R},$$

and its derivative $\mathrm{d}\Phi_{e|E} \in \mathcal{L}(\mathbb{R})$ can be calculated at any arbitrary point $t_0 \in I$ with

$$\mathrm{d}\Phi_{e|E}(t_0) = \mathrm{d}\Phi_e(\eta(t_0)) \circ \mathrm{d}\eta(t_0), \tag{1}$$

or, in expanded form,

$$\mathrm{d}\Phi_{e|E}(t_0) = \left. \begin{pmatrix} \dfrac{\partial \Phi_e}{\partial r} & \dfrac{\partial \Phi_e}{\partial s} & \dfrac{\partial \Phi_e}{\partial t} \end{pmatrix} \right|_{\eta(t_0)} \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} \mathrm{d}t. \tag{2}$$

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 parameterized with two variables as follows:

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

and the restriction of $\Phi_{e|F}$ to F,

$$\Phi_{e|F} = \Phi_e \circ \eta : U \longrightarrow \mathbb{R},$$

has a derivative $\mathrm{d}\Phi_{e|F} \in \mathcal{L}(\mathbb{R}^2, \mathbb{R})$ that we find with

$$\mathrm{d}\Phi_{e|F}\left(u_0, v_0\right) = \mathrm{d}\Phi_e\left(\eta\left(u_0, v_0\right)\right) \circ \mathrm{d}\eta\left(u_0, v_0\right), \tag{3}$$

or, in expanded form,

$$= \left(\frac{\partial \Phi_e}{\partial r}\frac{\partial \Phi_e}{\partial s}\frac{\partial \Phi_e}{\partial t}\right)\Bigg|_{\eta(u_0,v_0)} \begin{pmatrix} a_x & b_x \\ a_y & b_y \\ a_z & b_z \end{pmatrix} \begin{pmatrix} du \\ dv \end{pmatrix}. \tag{4}$$

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 also be expressed as

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

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 extremal point is a critical point. However, the converse is not true and thus each (Ci) condition is stronger than the corresponding (Ci^0). This means that using (Ci) rather than (Ci^0) can yield critical points that are not extremal – 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 critical points may occur as well. Therefore, rather than degrading computational performance, the tradeoff 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 critical points that are not extremal can be the locus of important geometric features (*e.g.*, saddle points). In short, the (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 Section 2, we presented the requirements a tetrahedral element must meet for isocontouring algorithm to work. As we noted earlier, higher order finite 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 on the class of finite elements; according to Proposition 1, satisying (C1) through (C3) is achieved if no edge, triangle, or tetrahedron of the tessellation contains in its interior a critical point of Φ_e or one of Φ_e's restrictions to triangles and edges of the partition. Therefore the partition is subdivided until it meets all criteria: (C1), (C2), and (C3); once this is achieved, we say that the final partition is *Φ-compatible*.

Remark 2. From now onwards, **it is assumed that all critical points are isolated**. This additional 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. The implications of this additional requirement will be 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($M, \Phi$)
1  $C \leftarrow$ DOF-Criticalities($M, \Phi$)
2  $(T_0, S) \leftarrow$ Triangulate-Boundaries($M, C$)
3  Correct-Triangle-Topology($M, \Phi, S, T_0$)
4  $(T_1, S) \leftarrow$ Tetrahedralize-Interior($M, T_0, C$)
5  Correct-Tetrahedral-Topology($M, \Phi, S, T_1$)
6  return $T_1$
```

The output of the scheme is a tetrahedral subdivision T_1 of M. In the following paragraphs, we provide a detailed discussion of each step, along with theoretical results that justify our approach.

DOF-Criticalities

The first step in the process is the location of the critical points of Φ_e inside each element, as well as of the restrictions of Φ_e to all element faces and edges. Because finding critical points is a time-consuming process, we do not wish to process same shared edge or face multiple times. This extra work can be 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 algorithm DOF-Criticalities takes the mesh M and a scalar field Φ as its input and yields a set C of critical points: critical points of Φ, and of the restrictions of Φ to all faces and edges of M. Given an element e in M, let $\textsc{Bdy}^1(R)$ and $\textsc{Bdy}^2(R)$ respectively denote the set of 1-dimensional and 2-dimensional boundaries of the parametric domain R of e.

```
DOF-Criticalities($M$)
1  for $e \leftarrow |M|$
2      do Find critical points of $\Phi_e$ in $R$
3         Store critical points indexed by volumetric DOF node.
4         for $f_i^2 \in \textsc{Bdy}^2(R)$
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 $f_i^1 \in \textsc{Bdy}^1(R)$
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 $\left(C = \cup_i C_{f_i^1}\right) \cup \left(\cup_i C_{f_i^2}\right)$
```

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 makes the assumption that the polynomial system solver always terminates in finite time, then DOF-Criticalities does as well. Note that, as 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 3. 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 of triangulating the two-dimensional boundaries of all elements. This ensures that any volumetric elements that reference a particular face use the same triangulation – otherwise our model could have cracks along element boundaries[4]. In order to satisfy (C2) on a given element e, the restriction of Φ_e to the face of an element 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 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 $\{Q_1, \ldots, Q_n\}$. With these conventions, the algorithm is as shown in Table 1.

All the sets involved in Triangulate-Boundaries are finite, and no recursion is involved. Therefore, Triangulate-Boundaries terminates in finite time. It is important to acknowledge that, upon completion, there is no guarantee that (C1) is satisfied. Nonetheless, upon completion of Triangulate-Boundaries, (C1) is satisfied on all the edges of T_0 that either belong to the initial mesh M or are subdivisions of edges or faces of M. Moreover, at this stage (C2) is also satisfied as all critical points of restrictions of the field to element faces have been inserted as vertices of T_0, and no new such critical point may have been created.

Example 1. Consider a face f_i, illustrated in Figure 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 Figure 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 Figure 2(c), has eliminated the remaining interior critical points b and c by making them nodes of the triangulation; however, as illustrated in Figure 2(d), new critical points have appeared:, on the restrictions of Φ_e to edges ab, ag, am, and cg.

[4]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.

Table 1. An algorithm to create a triangulation of the boundaries of a higher-order element that contains all critical points of the restriction of Φ to that boundary.

```
TRIANGULATE-BOUNDARIES($M, C$)
 1  for $f_i^2 \leftarrow$ each 2-boundary of every 3-D finite element
 2        do if $|C_{f_i^2}| > 0$
 3              then $c \leftarrow C_{f_i^2,0}$
 4              else  $c \leftarrow$ FACE-CENTER(BDY$_i^2$($\mathbb{R}$))
 5           $T_i \leftarrow \emptyset$
 6           $Q \leftarrow$ corner points of face $i \bigcup$ isolated critical points
             of all bounding edges of face $f_i^2$, ordered in a
             counterclockwise loop around $f_i^2$.
 7           for $j \in \{0, \ldots, |Q| - 1\}$
 8                do Insert STAR$^2$($c, Q_j, Q_{(j+1) \bmod |Q|}$) into $T_i$
 9           $C_{f_i^2}{}' \leftarrow \{C_{f_i^2} \setminus C_{i,0}\}$
10           for $c \in C_{f_i^2}{}'$
11                do Find $t \in T_i$ such that $c \in t$
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 $T_0 = \cup_i T_i$
```

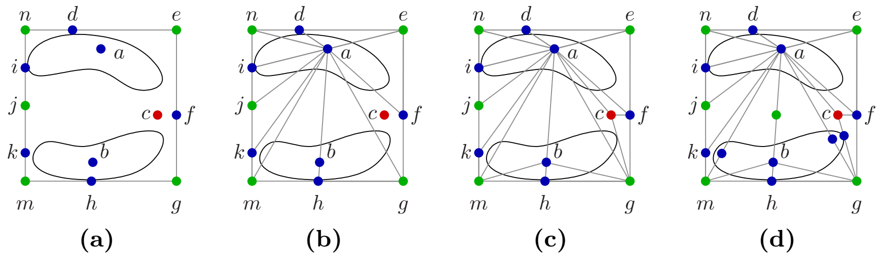

Fig. 2. An example face f_i^2 with critical points shown as blue dots (maxima), red dots (saddles), and green dots (minima). Far left: The input to TRIANGULATE-BOUNDARIES. Middle left: Resulting triangulation of f_i^2 after all STAR^2 procedures of TRIANGULATE-BOUNDARIES have been performed. Middle right: The triangulation at the completion of TRIANGULATE-BOUNDARIES. Far right: The new critical points introduced by the first stage of CORRECT-TRIANGLE-TOPOLOGY.

Remark 4. Note that 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 Figure 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 Figure 2(c), critical 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.

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

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

TETRAHEDRALIZE-INTERIOR

In the third step, each element interior is tetrahedralized, and although we treat the whole mesh, there is here no issue of interelement consistency, as this part of the scheme only regards element interiors. Tetrahedralizing the mesh elements could be done using only element corners along with the face and edge points that have been added in TRIANGULATE-BOUNDARIES; however, in order to satisfy (C3) within a given hexahedral element e, no tetrahedron of the partition of e may have a critical point of Φ_e in its interior. Therefore, 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 initial tetrahedralization of each element e is constrained by the triangulations of the faces of e that have been computed by TRIANGULATE-BOUNDARIES, 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($M, T_0, C$)
$S \leftarrow \emptyset$
$T_1 \leftarrow \emptyset$
for $e \leftarrow |M|$
    do Let $T \subseteq T_0$ be all triangles on Bdy$^2(R)$
       if $|C_e| > 0$
          then $c \leftarrow C_{e,0}$
          else $c \leftarrow$ Element-Center($R$)
       $V \leftarrow$ Star$^3(c, T)$
       for $t \leftarrow V$
           do if Mark-Tetrahedron($t$)
                 then Push $t$ onto $S$
       for $c \in \{C_e \setminus C_{e,0}\}$
           do Find $t \in V$ s. t. $c \in t$
              Remove $t$ from $V$ and $S$
              $U \leftarrow$ Star$^3(c, t)$
              for $t' \leftarrow U$
                  do if Mark-Tetrahedron($t'$)
                        then Push $t'$ onto $S$
                     Insert $t'$ into $V$
       Insert $V$ into $T_1$
Return $(T_1, S)$
```

All the sets involved in Tetrahedralize-Interior being finite, and as there is no recursion, this procedure terminates in finite time. In addition,

Proposition 2. *(C3) is satisfied across the tetrahedralization T_1 upon completion of* Tetrahedralize-Interior. *Moreover, any subtetrahedralization 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 (2) (if p is on a face) or (4) (if p is on an edge) vanishes as well. Therefore, p belongs to C_{f_i} for a face f_i^2 or an edge f_i^1 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 hypotheis; *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) is satisfied on all the edges of T_1 that either belong to M or are subdivisions of edges or faces of M.

However, there is no guarantee that (C1) is satisfied on all edges 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 Q^2 element with linear geometry and coordinates in $[-1, 1]^3$, as illustrated in Figure 3(a): 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, that (C1) is not satisfied across T_1, which requires further modifications.

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.

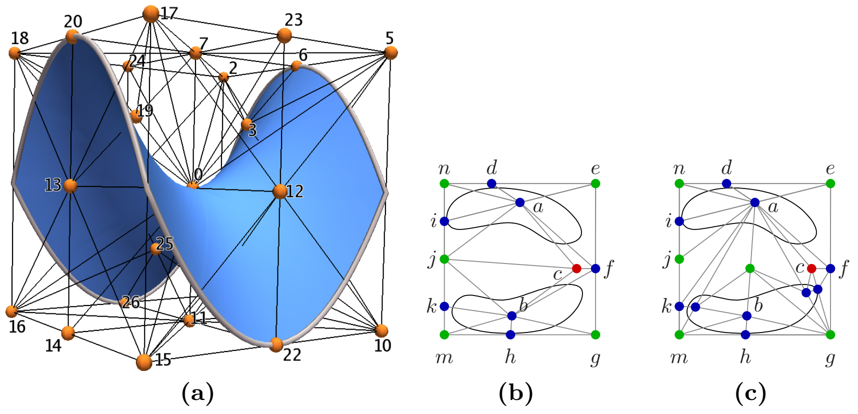

Fig. 3. **(a)** Topology-based tetrahedralization of a single Lagrange Q^2 element with linear geometry for the field $\Phi_2(x, y, z) = x^2 - y^2 + z$. **(b)** A triangulation satisfying (C1) with edge flips from Figure 2(c). **(c)** Another triangulation satisfying (C1) using the proposed algorithm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

Example 3. For instance, given the triangulation of a face in Figure 2(c) for which additional critical points have appeared (critical points of the restrictions of the field to some of the new edges), it is evidently possible to perform a series of edge flips on the so that the final connectivity satisfies (C1), as shown in the resulting tessellation in Figure 3(b).

Nevertheless, it is unclear whether performing only edge flips allows *in general* to find a face tessellation that satisfies (C1). Although this may be the case, we have not further explored this path, because the matter is more complicated here, because 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 this result does **not** extend to 3-D tetrahedralizations[2], which is one reason we are skeptical of the flipping approach.

Therefore, we decided to take a different approach, guided by Proposition 2, as we not only have a tetrahedralization that satisfies (C3), but we also know that subsequent tetrahedral subdivisions of T_1 will not alter this fact. Rather than attempting to identify a set of "problem" edges and prove that they may always be flipped into a satisfactory configuration, we simply introduce a new vertex along each internal edge that has a critical point on the restriction of Φ to its domain. The vertex is then connected to each node in its star. Any new edges and faces created by this operation must be examined for critical points of Φ restricted to their respective domains. However, it is only be necessary to focus on critical points of the restrictions of Φ to the previously marked edges and faces, as unmarked ones already satisfy (C1) and (C2), thanks to TRIANGULATE-BOUNDARIES. Figure 3(c) shows this procedure applied to Example 3. Here is pseudocode for the fourth step of our scheme:

```
CORRECT-TETRAHEDRAL-TOPOLOGY($M, \Phi, S, T_1$)
 1  while $S$ not empty
 2      do Pop $t$ from $S$
 3         $C \leftarrow \emptyset$
 4         for $e \leftarrow$ marked edges of $t$
 5             do Insert critical points of $e$ into $C$
 6         for $f \leftarrow$ marked faces of $t$
 7             do Insert critical points of $f$ into $C$
 8         for $c \leftarrow C$
 9             do Find $t \in T_1$ s. t. $c \in t$
10                Remove $t$ from $T_1$ and $S$
11                $U \leftarrow$ STAR$^3(c, t)$
12                for $t' \leftarrow U$
13                    do if MARK-TETRAHEDRON($t'$)
14                          then Push $t'$ onto $S$
15                       Insert $t'$ into $T_1$
```

Proposition 3. *If, for all e in M, all critical points of the restriction of Φ_e to any arbitrary face are isolated, then Algorithm* TRIANGULATE-BOUNDARIES *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 TRIANGULATE-BOUNDARIES($\{f_i\}, C_{|f_i}$) 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 this means that 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 Algorithm TRIANGULATE-BOUNDARIES will insert these n points, and yield a triangulation of f_i in $N \in \mathbb{N}^*$ triangles $t_{i,k}$, such that

$$\begin{cases} \bigcup_{k=1}^{N} \overset{\circ}{t}_{i,k} = \overset{\circ}{f}_i \\ (\forall\, 1 \leq k, k' \leq N) \quad k \neq k' \iff \overset{\circ}{t}_{i,k} \cap \overset{\circ}{t}_{i,k'} = \emptyset \end{cases} \tag{5}$$

where all the points of C are vertices of some of $t_{i,k}$ ($\overset{\circ}{p}$ denotes the interior of the polygon p, in the sense of the natural topology induced on p 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[5]. 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 Algorithm Triangulate-Boundaries recursively proceeds on 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 2).

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 Figure 3(a), labelled 3, 19, and 25) in a subtetrahedralization 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 5. Note that the hypothesis of Proposition 3 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 Results

In the absence of other isosurfacing algorithms that work on higher-order finite elements, we choose to compare our technique to a brute-force sampling and degree-reduction algorithm that approximates the higher order interpolant by

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

resampling it onto a mesh of linear cells. This is a difficult comparison to make because it is sensitive to the size of the input mesh and there are very few large, higher-order meshes in existence.

A simple analytical example that illustrates how critical critical points are is provided by he following scalar field:

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

interpolated over a single Q^3 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.

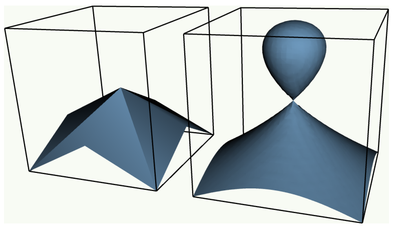

Fig. 4. Tricubic isocontouring of Φ_3 for the isovalue 0, using a linear isocontouring technique (left), and using our approach (right).

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 the isocontour $\Phi^{-1}(0)$ 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 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 functions 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 disjoined 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 technque to fail (by missing this connected component).

In the current example, Φ_3 has 2 critical points: $(0,0,0)$ and $(0,0,\frac{2}{3})$, respectively a saddle point and a local minimum, and both are interior to the unique element of the mesh. These points are indeed inserted in the tessellation by Tetrahedralize-Interior, and this takes care of (C1) through

(C3). Additionally, $(0, 0, 0)$ belongs to $\Phi^{-1}(0)$. And in fact, this not only an anecdoctal effect valid for this example only, but we can see that this is always the case: by "eliminating" critical points from the final tesselation, the method ensures that (thanks to the Implicit Function Theorem), the isocontour is locally a 2-dimensional 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 into regions useful for characterizing some scalar field Φ. By forcing critical points of Φ – and the restrictions of Φ to the boundaries of the partition – to be vertices of the partition, we can easily adapt visualization and other post-processing operations to higher order elements. This is illustrated by an adaptation of linear isocontouring to higher order elements.

An estimation of the computational complexity of Correct-Tetrahedral-Topology as a function of input parameters such as the order of the interpolant would be of great theoretical – and probably practical – interest but beyond the scope of this paper. Improved multivariate polynomial system solvers are another area of interest for future work.

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. 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.