Solid and Physical Modeling with Chain Complexes

Antonio DiCarlo *
Dipartimento di Strutture
Università "Roma Tre"
Via Segre, 6
00146 Rome, Italy

Franco Milicchio †
Dip. Informatica e Autom.
Università "Roma Tre"
Via Vasca Navale, 79
00146 Rome, Italy

Alberto Paoluzzi ‡
Dip. Informatica e Autom.
Università "Roma Tre"
Via Vasca Navale, 79
00146 Rome, Italy

Vadim Shapiro§
Dept. of Mech. Eng. & Comp. Science
University of Wisconsin
1513 University Avenue
Madison, WI 53706, USA

Abstract

In this paper we show that the (co)chain complex associated with a decomposition of the computational domain, commonly called a mesh in computational science and engineering, can be represented by a block-bidiagonal matrix that we call the Hasse matrix. Moreover, we show that topology-preserving mesh refinements, produced by the action of (the simplest) Euler operators, can be reduced to multi-linear transformations of the Hasse matrix representing the complex.

Our main result is a new representation of the (co)chain complex underlying field computations, a representation that provides new insights into the transformations induced by local mesh refinements. This paper is a further contribution towards bridging the subject of computer representations for solid and physical modeling—which flourished border-line between computer graphics, engineering mechanics and computer science with its own methods and data structures—under the general cover of linear algebra and algebraic topology. The main advantage of such an approach is that topology, geometry and physics may coexist in one and the same formalized framework, concurring together to define, represent and simulate the behavior of the model.

Our approach is based on first principles and is general in that it applies to most representational domains that can be characterized as cell complexes, without any restrictions on their type, dimension, codimension, orientability, manifoldness, connectedness. Contrary to what might appear at first sight, the theoretical complexity of the present approach is not greater than that of current methods, provided that sparse-matrix techniques with double element access (by rows and by columns) are employed. Last but not least, our tensor-based approach is a significant step forward in achieving close integration of geometrical representations and physics-based simulations, i.e., in the concurrent modeling of shape and behavior.

CR Categories: K.6.1 [Management of Computing and Information Systems]: Project and People Management—Life Cycle; K.7.m [The Computing Profession]: Miscellaneous—Ethics

Keywords:

*adicarlo@mac.com
†milicchio@mac.com
‡paoluzzi@dia.uniroma3.it
§vshapiro@engr.wisc.edu

1 Introduction

1.1 Motivation

The boundary representation has become the representation of choice in many academic and virtually all commercial solid modeling systems. As a consequence, most geometric, scientific and engineering applications have to be formulated in terms of boundary representations, often leading to nontrivial representation conversion problems. Well known examples of such problems include Boolean set operations, finite element meshing, and subdivision algorithms.

Formally, all boundary representations are widely recognized as graph-based data structures [Baumgart 1972; Guibas and Stolfi 1985; Mäntylä 1988; Brisson 1993] representing one of several possible cells complexes [Requicha 1977; Requicha 1980; Silva 1981; O'Connor and Rossignac 1990]. Space requirements and computational efficiency of such data structures have been studied in the literature (see, e.g., [Woo 1985]). Historically, such cell complexes have been restricted to (unions of) two-dimensional orientable manifolds, but a number of extensions to more general orientable cellular spaces have been proposed (see, e.g., [Masuda 1993; Yamaguchi and Kimura 1995; O'Connor and Rossignac 1990]). Depending on a particular choice of data structures, boundary representations are constructed, edited, and updated using a small set of basic operators on the graph representation, while preserving and/or updating the basic topological invariants of the cell complex. Such operators are commonly called Euler operators [Eastman and Weiler 1979; Mäntylä 1988; Masuda 1993], because they enforce the Euler-Poincaré formula. All higher-level algorithms and applications of boundary representations are implemented in terms of such operators.

This evolutionary development of boundary representations also led to several fundamental difficulties:

- Variety of assumptions about the cell complexes and graph representations make standardization difficult. This in turns complicates the issues of data exchange and transfer, and leads to proliferation of incompatible algorithms.

- Boundary representation algorithms are dominated by graph searching algorithms (boundary traversals) that tend to force serial processing. Nor is it clear how to combine such graph representations with multi-resolution representations and algorithms.
- Extending boundary representations to more general cellular spaces has proved challenging. Despite many proposals, most commercial systems are still restricted to two-dimensional orientable surfaces.
- Last, but not least, solid modeling has developed into a highly specialized discipline that is largely disconnected from many standard computational techniques. In particular, boundary representations do not appear to be directly related to the methods for physical analysis and simulation such as finite differences, finite elements, and finite volumes.

In this paper, we claim that *all* representations of cell complexes are properly represented by a (co)chain complex [Munkres 1984; Hatcher 2002]. It captures all combinatorial relationship of interest in solid and physical modeling formally and unambiguously, using standard algebraic topological operators of boundary ∂ and coboundary δ. We show that the (co)chain complex can be represented by a sparse block-bidiagonal matrix that we call the Hasse matrix. We also show that topology-preserving refinements of such cell complexes correspond to simple Euler operators and are easily formulated as multi-linear transformations of the Hasse matrix. There are at least three important consequence of our proposal. First, the proposed approach applies to all cell complexes, without restrictions on type, dimension, codimension, orientability, manifoldness, and so on. Secondly, as we will see in Section 7, this formalism unifies geometric and physical computations within a common formal computational structure. And finally, our formulation explicitly shows that many geometric computations (and computations with boundary representations in particular) can be formulated and implemented in terms of standard sparse matrix computational techniques, opening a possibility for a wide range of computational breakthroughs and opportunities.

1.2 Related Work

Algebraic-topological properties of boundary representations are well understood—see [Requicha 1977; Hoffmann 1989; Mäntylä 1988; O'Connor and Rossignac 1990] for details. In particular, Branin [Branin 1966] and Tonti [Tonti 1975] advocated a unified discrete view of all physical theories using concepts from algebraic topology and the De Rham cohomology. More recently, this early research led to new efforts in developing unified computational models and languages for analysis, simulation, and engineering design. Notably, Palmer and Shapiro [Palmer and Shapiro 1993] proposed a unified computational model of engineering systems that relies on concepts from algebraic topology. A number of researchers went beyond the use of chains and cochains as general-purpose data types, considering that a sound numerical method should reflect the algebraic-topological structure of the underlying physical theory in a faithful way. Notably, Strang [Strang 1988] observed that the FEM encodes a pervasive balance pattern, which is at the center of the classification in [Tonti 1975]. Mattiussi [Mattiussi 1997] provided interpretations of FEM, FVM, and FDM in terms of the topological properties of the corresponding field theory. Tonti [Tonti 2001] presented his cell method as a direct discrete method, bypassing the underlying continuum model. In [Hyman and Shashkov 1997] FDMs that satisfy desired topological properties are discussed. In our previous work [Milicchio et al. 2006], physical information is attached to an adaptive, full-dimensional decomposition of the computational domain. Giving preeminence to the cells of highest dimension allows to generate the geometry and to simulate the physics simultaneously. Such a formulation removes artificial constraints on the shape of discrete elements and unifies commonly unrelated finite methods into a single computational framework [Milicchio 2007]. Our goal is to graft this approach to field modeling onto an already established computational framework for geometric modeling with cell complexes [Paoluzzi et al. 1995]. This framework has been recently extended to provide parallel and progressive generation of very large datasets using streams of continuous approximations of the domain with convex cells [Scorzelli et al. 2006]. The approach also supports progressive Boolean operations [Paoluzzi et al. 2004], providing continuous streaming of geometrical features and adaptive refinement of their details.

1.3 Overview

In Section 2 we review some standard concepts from algebraic topology and their representations using matrices and Hasse diagram. Section 3 introduces our block-matrix representation of a chain complex. In Section 4 we use algebraic-topological notions to define a minimal set of operators as transformations between cell complexes that preserve the Euler characteristics. These operators are shown to correspond to multi-linear transformations of the Hasse matrix in Section 5. Section 6 demonstrates how common algorithms for splitting a cell complex may be formulated in algebraic topological terms. Section 7 explains how the proposed representation may unify geometric and physical modeling in a common computational framework. The Appendix shows how easily local adjacency information, including the discrete Jacobians, can be computed using only some standard linear algebra.

2 Background

We take for granted the elementary notions of *simplex*, *cell*, *orientation*, *cell complex*, *face*, and refer the reader to [Requicha 1977; Munkres 1984]. Apart from 0-cells, we take all cells as *open*.

2.1 Chains and Cochains

Let K be a cell complex representing a finite partition of a compact set $D \in E^d$. We call *p-skeleton* $K_p \subset K$ the subset of oriented p-cells of K, and denote with k_p the number of p-cells: $k_p := \#K_p$, hence

$$\#K = k_0 + k_1 + \cdots + k_d.$$

The p-skeleton K_p will be ordered by labeling each p-cell σ_p with a positive integer: $K_p = (\sigma_p^1, \ldots, \sigma_p^{k_p})$. In the following, the ordinal and/or the dimension of cells will be dropped from notation whenever convenient. In its turn, the complex K will be identified with the tuple of its ordered p-skeletons ($p = 0, \ldots, d$): $K \cong (K_p)$.

2.1.1 Chain groups

Let $(G, +)$ be an *abelian* (i.e., commutative) *group*. A *p-chain* of K with coefficients in G is a mapping $c_p : K_p \to G$ such that, for each $\sigma \in K_p$, reversing a cell orientation changes the sign of the chain value:

$$c_p(-\sigma) = -c_p(\sigma).$$

Chain addition is defined by addition of chain values: if c_p, d_p are p-chains, then $(c_p + d_p)(\sigma) = c_p(\sigma) + d_p(\sigma)$, for each $\sigma \in K_p$. The resulting group is denoted $C_p(K; G)$. In the following the group G will often left implied, writing $C_p(K)$.

Let σ be an oriented cell in K and $g \in G$. The *elementary chain* whose value is g on σ, $-g$ on $-\sigma$ and 0 on any other cell in K is denoted $g\sigma$. Each chain can then be written in a unique way as a finite sum of elementary chains:

$$c_p = \sum_{k=1}^{k_p} g_k \sigma_p^k \ .$$

Chains may be thought of as attaching *multiplicity* to cells; if coefficients are taken from the smallest nontrivial group, i.e. $G = \{-1, 0, 1\}$, then cells can only be *discarded* or *selected*, possibly *inverting their orientation*.

2.1.2 Cochain groups

By definition, the set of *p-cochains* of K, with coefficients in G, is the *group of all homomorphisms* of $C_p(K)$ into G:

$$C^p(K) := Hom(C_p(K), G)$$

Cochains may be thought of as measuring the *content* of G-valued *additive* quantities in chains. If γ^p is a p-cochain, its content in the p-chain c_p is often denoted as a pairing:

$$\langle \gamma^p, c_p \rangle := \gamma^p(c_p).$$

2.2 Boundary and Coboundary

2.2.1 Boundary operator

The *boundary operator* $\partial_p : C_p(K) \to C_{p-1}(K)$ is first defined on *simplices*. If σ_p is a p-simplex, then

$$\partial_p \sigma_p := \sum_{k=0}^{p} (-1)^k \sigma_{p-1,\,k}$$

where $\sigma_{p-1,\,k}$ denotes the k-th face of σ_p. The next move is to extend ∂_p to *cells*, by partitioning them into simplices and assuming ∂_p to be *additive*. This is then extended to *elementary chains*, by taking

$$\partial_p(g\,\sigma) := g(\partial_p \sigma)$$

and finally to *all chains* by additivity.

2.2.2 Coboundary operator

The *coboundary operator* δ^p is defined as the *dual* of the boundary operator $\partial_{p+1} : C_{p+1}(K) \to C_p(K)$, so that

$$\delta^p : C^p(K) \ \to C^{p+1}(K)$$

in such a way that, for all $\gamma \in C^p$ and $c \in C_{p+1}$,

$$\langle \delta^p \gamma, c \rangle = \langle \gamma, \partial_{p+1}\, c \rangle \, .$$

The pairing notation makes transparent that this defining property is a combinatorial prototype of the *Stokes theorem*.

2.2.3 Matrix representation of (co)chains

A very simple and powerful abstraction consists in representing p-chains and p-cochains as matrices *indexed* on the cells of K and *parameterized* in the underlying G group.

Let K be a d-complex, with $k_p = \#K_p$, $0 \le p \le d$. We may represent a p-chain $c_p \in C_p(K)$ as a *column matrix* $\mathbf{x}_p \in G^{k_p}$, and we write $\mathbf{x}_p = [c_p]$, or $\mathrm{x}_p^i = [c_p]^i$. Analogously, we may represent a p-cochain $\gamma^p \in C^p(K)$ as a *row matrix* $\mathbf{y}^p \in G^{k_p}$, and we write $\mathbf{y}^p = [\gamma^p]^\top$, or $\mathrm{y}_i^p = [\gamma^p]_i$. The *content* of the p-cochain γ^p in the p-chain c_p is given by the *matrix product*

$$\mathbf{y}^p\, \mathbf{x}_p = \langle \gamma^p, c_p \rangle.$$

2.2.4 Incidence matrices

The intersection between p-cells and $(p+1)$-cells may be characterized by the *p-incidence matrix* (a_p^{ij}) defined by:

$a_p^{ij} = 0$ if $\sigma_p^i \cap \overline{\sigma}_{p+1}^j = \emptyset$ ($\overline{\sigma}$ being the *closure* of σ);

$a_p^{ij} = \pm 1$ otherwise, with $+1$ (-1) if the orientation of σ_p^i is equal (opposite) to that of the corresponding face of σ_{p+1}^j.

Of course, the transpose of (a_p^{ij}) describes how $(p+1)$-cells intersect with p-cells.

It is easy to check that (a_p^{ij}) represents through matrix multiplication the action of the boundary operator $\partial_{p+1} : C_{p+1} \to C_p$, while its transpose represents the action of the coboundary operator $\delta^p : C^p \to C^{p+1}$:

$$\begin{aligned} \sum_{j=1}^{k_{p+1}} a_p^{ij}\, [c_{p+1}]^j &= [\partial_{p+1}\, c_{p+1}]^i, \\ \sum_{i=1}^{k_p} a_p^{ij}\, [\gamma^p]_i &= [\delta^p\, \gamma^p]_j \, . \end{aligned}$$

Example 1 (Boundary and Coboundary) *Let the 2-chain $c \in C_2(K)$ be defined by*

$$c(\sigma_1) = 1, \quad c(\sigma_2) = 1, \quad c(\sigma_3) = 1, \quad c(\sigma_4) = 1,$$

where K is the 2*-complex given in Figure 1. The* boundary 1-chain

$$\partial_2 c = \partial_2(\sigma_1 + \sigma_2 + \sigma_3 + \sigma_4) = \tau_1 + \tau_3 + \tau_4 + \tau_8 + \tau_9$$

is represented by

$$[\partial_2] \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix},$$

where the incidence matrix $[\partial_2] = [\delta^1]^\top$, and

$$[\delta^1] = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1 & 0 \end{pmatrix}.$$

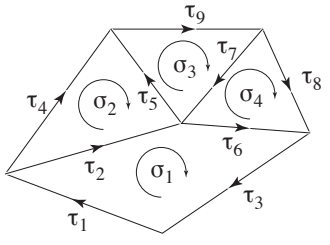


Figure 1: A 2-complex K, whose 2-cells are coherently oriented.

2.3 Hasse Diagram of a Chain Complex

Hasse diagrams, named after the German mathematician Helmut Hasse (1898–1979), illustrate the cover relation of a partial order and are commonly used for representing lattices.

In order theory, a *Hasse diagram* is a graph $\mathcal{H} = (N, E)$, where N is a finite poset, such that for any $x, y \in N$, there exists $(x, y) \in E$ if and only if $x < y$, and there is no $z \in N$ such that $x < z < y$.

If, given a d-complex K, the sets N and E are defined as follows, then the graph $\mathcal{H}(K) = (N, E)$ provides a complete representation of the topology of K:

1. $N := K_0 \cup K_1 \cup \cdots \cup K_d$,
2. $E := E_1 \cup \cdots \cup E_d$, with
3. $E_p := \{(\sigma_p, \sigma_{p-1}) \mid \sigma_{p-1} \in \partial\sigma_p\}$, $1 \le p \le d$.

Attaching a label from $\{-1, 1\}$ to the arc $(x, y) \in E_p$, denoted $\mathrm{sgn}(x, y)$, sufficies to specify the relative orientation between the p-cell represented by the node x and the $(p-1)$-cell represented by the node y.

Given a Hasse graph $\mathcal{H}(K) = (N, E)$, with $N = \cup_p K_p$, for each node $x \in N$ define:

1. $E^x := \{(x, y)\} \mid y \in N, (x, y) \in E\ \}$,
2. $N^x := \{y \mid y \in N, (x, y) \in E^x\}$.

Let $\sigma \in K$ be the cell represented by the node x. Then, the boundary of the elementary chain $g\sigma$ is obtained by transferring the (properly signed) coefficient from the node x to its "children" in $\mathcal{H}(K)$:

$$\partial(g\sigma) = g(\partial\sigma) = g \sum_{y \in N^x} \mathrm{sgn}(x, y)\, \tau(y) = \sum_{y \in N^x} \mathrm{sgn}(x, y)\, g\, \tau(y)$$

Where $\tau(y)$ denotes the cell represented by the node y. The computation of the boundary $\partial\, c$ of a general chain c follows by linearity.

2.3.1 Chain and cochain complexes

A Hasse diagram, together with the above representation of the boundary operator ∂, is a convenient representation of a *chain complex*, whose formal definition is as follows.

A *chain complex* $\mathcal{C} = (C_p, \partial_p)$ is a sequence

$$\cdots \longrightarrow C_{p+1} \xrightarrow{\partial_{p+1}} C_p \xrightarrow{\partial_p} C_{p-1} \longrightarrow \cdots \longrightarrow C_1 \xrightarrow{\partial_1} C_0$$

of abelian groups C_p, paired with homomorphisms ∂_p, $p \ge 1$, that satisfies the relation $\partial_p \circ \partial_{p+1} = 0$, for each $p \ge 1$.

The dual *cochain complex* $\mathcal{C}' = (C^p, \delta^p)$ is the sequence

$$\cdots \longleftarrow C^{p+1} \xleftarrow{\delta^p} C^p \xleftarrow{\delta^{p-1}} C^{p-1} \longleftarrow \cdots \longleftarrow C^1 \xleftarrow{\delta^0} C_0$$

The relations $\delta^p \circ \delta^{p-1} = 0$ $(p \ge 1)$ are satisfied by duality.

2.3.2 Chain maps

Let $\mathcal{C}(C_p, \partial_p)$ and $\widetilde{\mathcal{C}}(\widetilde{C}_p, \widetilde{\partial}_p)$ be two chain complexes. A *chain map* $\phi : \mathcal{C} \to \widetilde{\mathcal{C}}$ is a p-family of homomorphisms

$$\phi_p : C_p \longrightarrow \widetilde{C}_p$$

such that $\widetilde{\partial}_p \circ \phi_p = \phi_{p-1} \circ \partial_p$, i.e., the following diagram is *commutative*:

$$\begin{array}{ccc} C_p & \xrightarrow{\phi_p} & \widetilde{C}_p \\ \partial_p \downarrow & & \downarrow \widetilde{\partial}_p \\ C_{p-1} & \xrightarrow{\phi_{p-1}} & \widetilde{C}_{p-1} \end{array}$$

3 Matrix representation

In this section we introduce a block-matrix representation of the topology of the chain complex associated to a decomposition of the computational domain, and call it *Hasse matrix*. Later we show that, since all blocks transform by the one and the same pattern of transformations, so also the Hasse matrix transforms by the same pattern.

3.1 Block-Matrix Decomposition

A chain complex $\mathcal{C}(C_p, \partial_p)$ and its dual $\mathcal{C}'(C^p, \partial^p)$ can be represented by a block-bidiagonal matrix. Since the boundary operators ∂_p $(p \ge 1)$ are well represented by incidence matrices and the coboundary operators δ^{p-1} by their transposes, we may represent the p-families of homomorphisms ∂_p, δ^{p-1} $(p \ge 1)$ by a block-structured matrix. Notice that, from now on, we shall often write δ_p instead of δ^p.

Let K be a d-complex and $\mathcal{H}(K)$ its Hasse graph. The *Hasse matrix* will have the block structure shown in Figure 2:

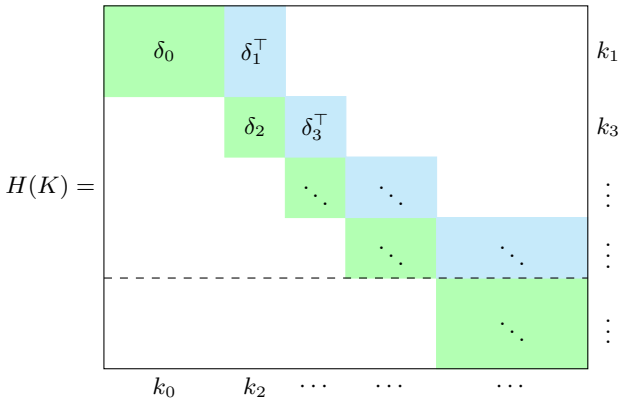


Figure 2: The whole scheme holds for d odd; for d even, the last block-row should be discarded.

The transposed Hasse matrix $H^\top(K)$ represents the dual complex K^*, whose Hasse graph $\mathcal{H}(K^*)$ is isomorphic to $\mathcal{H}(K)$, with $K_p^* \cong K_{d-p}$ $(0 \le p \le d)$, where the boundary and coboundary operators are swapped by duality.

Example 2 (Hasse graph (3D)) *Below we give a picture of the graph $\mathcal{H}(K)$ of a 3-complex K (a cube), representing its 6 boundary and coboundary operators as topological mappings between its sub-complexes K_p.*

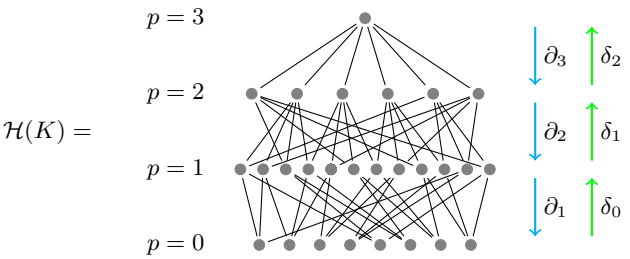


Figure 3: The Hasse diagram of the chain complex representing the topology of a 3-cube.

Example 3 (Hasse matrix (3D)) *Operators* $\delta_0, \delta_1^\top = \partial_2$*, and* δ_2 *may be represented as a single block-matrix (the Hasse matrix):*

$$H \in \mathcal{M}_{k_0+k_2}^{k_1+k_3}(G), \qquad G = \{-1, 0, 1\},$$

defined as below:

$$H = \begin{array}{|c|c|l} \hline \delta_0 & \delta_1^\top & k_1 \\ \hline 0 & \delta_2 & k_3 \\ \hline k_0 & k_2 & \end{array}$$

According to their definition, the operators $\partial_1, \partial_2^\top = \delta_1$*, and* ∂_3 *are collected in the transpose matrix* $H^\top$*:*

$$H^\top = \begin{array}{|c|c|l} \hline \partial_1 & 0 & k_0 \\ \hline \partial_2^\top & \partial_3 & k_2 \\ \hline k_1 & k_3 & \end{array}$$

Example 4 (Linear graph) *If* K *is a 1-complex, i.e. a linear graph, then* $H(K)$ *and the* incidence matrix *of vertices on edges coincide.* $H(K)$ *and its transpose represent the two topological operators available, i.e.,* δ_0 *and* ∂_1*.*

4 Euler operators

In solid modeling it is common to refer to Euler operators as an independent set of operators [Mäntylä 1988; Hoffmann 1989] that transform a boundary representation of a solid into a different one, satisfying the Euler-Poincaré formula. They may be allowed to change the Euler characteristic, whose definition is recalled below.

4.1 Euler characteristic

A well-known invariant of a finite d-dimensional cell complex is its *Euler characteristic*, that can be defined as the alternating sum

$$\chi = k_0 - k_1 + k_2 - k_3 + \cdots + (-1)^d k_d \,.$$

For polyhedra homeomorphic to the 3-sphere, the Euler characteristic is $V - E + F = 2$. According to the above, the *simplest* set of independent refining (coarsening) operators for a d-space that do not change its Euler characteristic has to increase (decrease) by one both k_{p-1} and k_p, for $p \in \{1, \ldots, d\}$. There are therefore d elementary refining operators and the same number of elementary coarsening operators.

In order to change the Euler characteristic, i.e. to change the shape of a space, it is appropriate to use some Boolean operator, according to the properties [Alexandroff 1998; Baez 2003] recalled below.

4.1.1 Properties of the Euler characteristic

Let $\chi(M)$ and $\chi(N)$ be the Euler characteristics of any two topological spaces M and N. Then, their sum is the Euler characteristic of the disjoint union of M and N:

$$\chi(M \sqcup N) = \chi(M) + \chi(N).$$

More generally, if M and N are subspaces of a larger space X, then so are their union and intersection, and the Euler characteristic obeys the rule:

$$\chi(M \cup N) = \chi(M) + \chi(N) - \chi(M \cap N).$$

Moreover, the Euler characteristic of any product space is

$$\chi(M \times N) = \chi(M)\,\chi(N).$$

4.2 Make and Kill operations

The simplest Euler operators that transform a cell complex K into a new complex $\widetilde{K}$ *without changing its Euler characteristic* χ, add (remove) just *two* cells to (from) the complex, with dimensions p and $(p+1)$. They will be denoted as β and κ, from the Greek words "blastos" and "klastos", referring to construction and destruction, respectively.

By definition, the operator β^p adds a p-cell and a $(p+1)$-cell to K, thus transforming it into $\widetilde{K}$. The reverse operator κ^p deletes a p-cell and a $(p-1)$-cell.

In this section we discuss how the coboundary operators transform under the action of a refinement operation β^q:

$$\delta_p \circ \beta^q : \delta_p(K) \mapsto \delta_p(\beta^q(K)), \qquad p = 0, \ldots, n-1.$$

It is easily seen that β^q affects in a nontrivial way only the coboundary operators whose domain and/or codomain change under its action, namely:

1. $\delta_{q+1} \mapsto \widetilde{\delta}_{q+1}$,
2. $\delta_{q-1} \mapsto \widetilde{\delta}_{q-1}$,
3. $\delta_q \mapsto \widetilde{\delta}_q$.

as shown by the commutative diagram:

$$\begin{array}{ccccccccc} \widetilde{C}^{q+2} = C^{q+2} & \xleftarrow{\delta_{q+1}} & C^{q+1} & \xleftarrow{\delta_q} & C^q & \xleftarrow{\delta_{q-1}} & C^{q-1} & = \widetilde{C}^{q-1} \\ & \swarrow_{\widetilde{\delta}_{q+1}} & \downarrow{\scriptstyle \beta^q} & & \downarrow{\scriptstyle \beta^q} & \swarrow_{\widetilde{\delta}_{q-1}} & & \\ & & \widetilde{C}^{q+1} & \xleftarrow[\widetilde{\delta}_q]{} & \widetilde{C}^q & & & \end{array}$$

Three different computations have to be performed, depending on whether only the domain changes (case 1), or only the codomain (case 2), or both change (case 3).

4.2.1 Addition of a column ($\delta_{q+1} \mapsto \widetilde{\delta}_{q+1}$)

Let the matrix $[\delta_{q+1}]$ be $m \times n$; then, the matrix $[\widetilde{\delta}_{q+1}]$ will be $m \times (n+1)$. The column to be added to $[\delta_{q+1}]$ represents the cochain in $\beta^q(C^{q+2})$ incident on the new cell $\widetilde{\sigma}_{q+1}$. It is a linear combination of the columns of $[\delta_{q+1}]$, i.e., of the preexistent cochains in C^{q+2}. We have:

$$[\widetilde{\delta}]_{m\times(n+1)} = [\delta]_{m\times n} \left(\begin{array}{c|c} \mathbf{I}_{n\times n} & \begin{matrix} c_1 \\ \vdots \\ c_n \end{matrix} \end{array} \right) = [\delta]_{m\times n}\, \mathbf{C}$$

4.2.2 Addition of a row ($\delta_{q-1} \mapsto \widetilde{\delta}_{q-1}$)

The row to be added to $[\delta_{q-1}]$ represents the chain of $\beta^q(C_{q-1})$ incident on the new cell $\widetilde{\sigma}_q$. It is a linear combination of the rows of $[\delta_{q-1}]$. We have:

$$[\widetilde{\delta}]_{(m+1)\times n} = \left(\frac{\mathbf{I}_{m\times m}}{r_1 \ \cdots \ r_m}\right)[\delta]_{m\times n} = \mathbf{R}\,[\delta]_{m\times n}$$

4.2.3 Addition of a column and a row ($\delta_q \mapsto \widetilde{\delta}_q$)

One of the rows of $[\delta_q]$ (one chain in C_q) is substituted by *two* rows (two chains in $\beta^q(C_q)$), whose components on the added cell $\widetilde{\sigma}_q$ sum up to zero. The matrix $[\widetilde{\delta}_q]$ is obtained as the sum

$$[\widetilde{\delta}_q]_{(m+1)\times(n+1)} = \sum_{i=1}^{3} \mathbf{S}_i\,[\delta_q]_{m\times n}\mathbf{T}_i\,,$$

where the first term ($i = 1$) provides the contribution of the split cell σ_{q+1}, the second one ($i = 2$) the contribution of the added cell $\widetilde{\sigma}_{q+1}$, and the third one ($i = 3$) the contribution of all of the other cells in K_{q+1}.

4.3 Examples

In Figs. 4–6 we show a very simple 2-complex K and its refinement $\widetilde{\widetilde{K}}$, obtained by applying first the operator β^0 to split the 1-cell σ_1^1, then the operator β^1 to split the 2-cell σ_2^1.

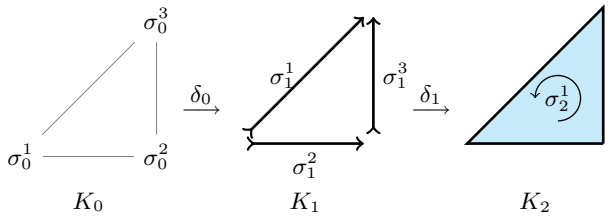


Figure 4: Coarse complex $K = (K_0, K_1, K_2)$.

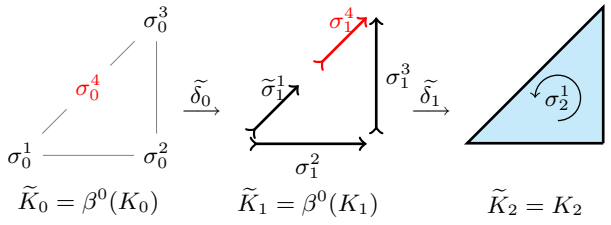


Figure 5: First refinement step: $\widetilde{K} = \beta^0(K) = (K_0 \cup \{\sigma_0^4\}, K_1 \cup \{\sigma_1^4\}, K_2)$

Let us compute the matrix representation of the coboundary operators δ_0, δ_1, on K and on their refinements $\widetilde{K} = \beta^0(K)$ and $\widetilde{\widetilde{K}} = \beta^1(\widetilde{K})$. The boundary operators ∂_1, ∂_2, as well as their refinements, are obtained by transposition.

Example 5 (Coboundary $\delta_0 : C^0(K) \to C^1(K)$) *Both domain and codomain have dimension 3. From Figure 4 it is seen that the matrix representation of δ_0 is*

$$[\delta_0] = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 1 & 0 \\ 0 & -1 & 1 \end{pmatrix}.$$

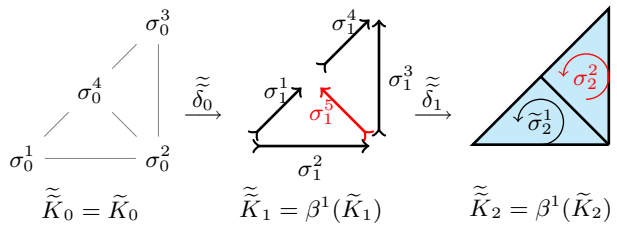


Figure 6: Second refinement step: $\widetilde{\widetilde{K}} = \beta^1(\widetilde{K}) = (\widetilde{K}_0, \widetilde{K}_1 \cup \{\sigma_1^5\}, \widetilde{K}_2 \cup \{\sigma_2^2\})$

Example 6 (Coboundary $\delta_1 : C^1(K) \to C^2(K)$) *In this case we have $k_1 = 3$ and $k_2 = 1$, so that*

$$[\delta_1] = (\ -1 \quad 1 \quad 1\ ).$$

Example 7 (Coboundary $\widetilde{\delta}_0 : C^0(\widetilde{K}) \to C^1(\widetilde{K})$) *We have $k_0 = k_1 = 3 + 1$. In Figure 5 the new 0-cell and 1-cell are displayed in red. Since both domain and codomain dimensions increase, the new operator has to be computed as the sum of three contributions (see Section 4.2.3).*

$$[\widetilde{\delta}_0] \ = \ (\ \mathbf{S}_1 \quad \mathbf{S}_2 \quad \mathbf{S}_3\ )\,[\delta_0] \begin{pmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \mathbf{T}_3 \end{pmatrix}$$

where $(\ \mathbf{S}_1 \quad \mathbf{S}_2 \quad \mathbf{S}_3\ )$ *and* $(\ \mathbf{T}_1 \quad \mathbf{T}_2 \quad \mathbf{T}_3\ )^\top$ *are block-matrices, and*

$$\mathbf{S}_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \qquad \mathbf{S}_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix},$$

$$\mathbf{S}_3 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}.$$

Matrices $\mathbf{S}_1$, $\mathbf{S}_2$ extract the row of $[\delta_0]$ that corresponds to the 1-cell σ_1^1 to be split (recall that a row of $[\delta_0]$ equals a column of $[\partial_1]$); $\mathbf{S}_1$ associates that row to $\widetilde{\sigma}_1^1$, while $\mathbf{S}_2$ associates it to the added cell $\widetilde{\sigma}_1^4$; matrix $\mathbf{S}_3$ keeps all other rows of $[\delta_0]$ unchanged. The actions of $\mathbf{S}_1$, $\mathbf{S}_2$, and $\mathbf{S}_3$ on $[\delta_0]$ are explicitly given below:

$$\mathbf{S}_1\,[\delta_0] = \begin{pmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \qquad \mathbf{S}_2\,[\delta_0] = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{pmatrix},$$

$$\mathbf{S}_3\,[\delta_0] = \begin{pmatrix} 0 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 0 \end{pmatrix}.$$

Each column of matrix $\mathbf{T}_i$ ($i = 1, 2, 3$) corresponds to a 1-cell in $\widetilde{K}_1$. Each $\mathbf{T}_i$ matrix represents the linear trasformation that maps one or more chains of K_0 elements into the cor-

responding chains of $\widetilde{K}_0$ elements:

$$\mathbf{S}_1\,[\delta_0]\,\mathbf{T}_1 = \begin{pmatrix} -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

$$\mathbf{S}_2\,[\delta_0]\,\mathbf{T}_2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix},$$

$$\mathbf{S}_3\,[\delta_0]\mathbf{T}_3 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

In conclusion, we get:

$$[\widetilde{\delta_0}] = \begin{pmatrix} -1 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix}.$$

The reader may check this result looking at Figure 5.

Example 8 (Coboundary $\widetilde{\delta_1} : C^1(\widetilde{K}) \to C^2(\widetilde{K})$)
In this case, $\widetilde{k}_1 = 3+1$ and $\widetilde{k}_2 = 1$; one gets:

$$[\widetilde{\delta_1}] = [\delta_1]\,\mathbf{C} = [\delta_1] \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} -1 & 1 & 1 & -1 \end{pmatrix}.$$

Example 9 (Coboundary $\widetilde{\widetilde{\delta}}_0 : C^0(\widetilde{\widetilde{K}}) \to C^1(\widetilde{\widetilde{K}})$)
We have: $\widetilde{\widetilde{k}}_0 = \widetilde{k}_0 = 4$, $\widetilde{\widetilde{k}}_1 = \widetilde{k}_1 + 1 = 5$, and we get:

$$[\widetilde{\widetilde{\delta_0}}] = \mathbf{R}\,[\widetilde{\delta_0}] = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} -1 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix}$$

$$= \begin{pmatrix} -1 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & -1 & 0 & 1 \end{pmatrix}.$$

Example 10 (Coboundary $\widetilde{\widetilde{\delta}}_1 : C^1(\widetilde{\widetilde{K}}) \to C^2(\widetilde{\widetilde{K}})$)
Now we have $\widetilde{\widetilde{k}}_1 = \widetilde{k}_1 + 1 = 5$ and $\widetilde{\widetilde{k}}_2 = \widetilde{k}_2 + 1 = 2$. Since both domain and codomain dimensions increase, by performing the same operations as in Example 7, we get:

$$[\widetilde{\widetilde{\delta}}_1] = \begin{pmatrix} \mathbf{S}_1 & \mathbf{S}_2 & \mathbf{S}_3 \end{pmatrix} [\widetilde{\delta}_1] \begin{pmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \mathbf{T}_3 \end{pmatrix}$$

$$= \begin{pmatrix} -1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & -1 & -1 \end{pmatrix}.$$

5 Hasse transformations

Let K be a d-complex and $H(K)$ be its $n \times m$ Hasse matrix, where $\chi(K) = m - n$. In this section we introduce the Hasse transformations

$$\eta_p^h(K) : \mathcal{M}_m^n \to \mathcal{M}_{m+1}^{n+1},$$

such that

$$H(K) \mapsto H(\widetilde{K}),$$

where the $(p+1)$-cell σ_{p+1}^h is split by the *blastos* (or "make") β^p operator into two cells:

$$\widetilde{\sigma}_{p+1}^h \quad \text{and} \quad \widetilde{\sigma}_{p+1}^{(k_{p+1})+1},$$

and a new p-cell $\widetilde{\sigma}_p^{k_p+1}$ is added to the complex. Notice that, while m and n increase under topology-preserving refinements, their difference does not. Let us distinguish between even and odd values of d, and assume, without loss of generality, that $d = 3$. In this case there are two diagonal blocks $[\delta_0]$ and $[\delta_2]$, and one upper-diagonal block $[\delta_1]^\top$ in H (see Section 3).

Remark 1 (Make operators β^0, β^1 and β^2)
Different but similar computational patterns arise, depending on the order of the make *operator:*

$$\beta^0(H) = \left(\begin{array}{c|c} \begin{pmatrix} \mathbf{S}_1 & \mathbf{S}_2 & \mathbf{S}_3 \end{pmatrix}[\delta_0] \begin{pmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \mathbf{T}_3 \end{pmatrix} & \mathbf{R}\,[\delta_1]^\top \\ \hline \mathbf{0} & [\delta_2] \end{array}\right),$$

$$\beta^1(H) = \left(\begin{array}{c|c} \mathbf{R}[\delta_0] & \begin{pmatrix} \mathbf{S}_1 & \mathbf{S}_2 & \mathbf{S}_3 \end{pmatrix}[\delta_1]^\top \begin{pmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \mathbf{T}_3 \end{pmatrix} \\ \hline \mathbf{0} & [\delta_2]\mathbf{C} \end{array}\right),$$

$$\beta^2(H) = \left(\begin{array}{c|c} [\delta_0] & [\delta_1]^\top \mathbf{C} \\ \hline \mathbf{0} & \begin{pmatrix} \mathbf{S}_1 & \mathbf{S}_2 & \mathbf{S}_3 \end{pmatrix}[\delta_2] \begin{pmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \mathbf{T}_3 \end{pmatrix} \end{array}\right).$$

In 3D the only *make* operators are $\beta^0, \beta^1, \beta^2$. Each β^p inserts two new cells $\widetilde{\sigma}_p$ and $\widetilde{\sigma}_{p+1}$ into $\widetilde{K}$. In order to specify the corresponding Hasse transformation, we need to extract the diagonal and upper-diagonal blocks of H:

$$H = \begin{pmatrix} [\delta_0] & [\delta_1]^\top \\ \mathbf{0} & [\delta_2] \end{pmatrix} = \underbrace{\begin{pmatrix} [\delta_0] & \mathbf{0} \\ \mathbf{0} & [\delta_2] \end{pmatrix}}_{H_1} + \underbrace{\begin{pmatrix} \mathbf{0} & [\delta_1]^\top \\ \mathbf{0} & \mathbf{0} \end{pmatrix}}_{H_2}$$

Then, we need only to apply the elementary transformations already given for a single operator, and to add the resulting matrices:

$$\beta^p(H) = \beta^p(H_1) + \beta^p(H_2).$$

6 Hyperplane splitting

In this section we discuss a subdivision algorithm (SPLIT) developed by Bajaj and Pascucci in [Bajaj and Pascucci 1996], rephrasing it in terms of the algebraic machinery developed in the previous sections. This algorithm works efficiently on a single d-cell of a d-complex. Our algebraic formulation is general and easy to implement using standard packages for sparse-matrix computation [Davis 2006].

The SPLIT algorithm is a useful tool for refining cell complexes, providing the ability to compute Boolean operations when combined with BSP trees in a progressive way [Paoluzzi et al. 2004]. The SPLIT algorithm is also useful to approximate continuous maps between cell complexes. A formal definition of *subdivision* of a complex goes this way [Munkres 1984]:

Definition 1 *Let K be a cell complex. Then, a complex $\widetilde{K}$ is a* subdivision *of K if:*

1. *for each $\widetilde{\sigma} \in \widetilde{K}$ there exists $\sigma \in K$ such that $\widetilde{\sigma} \subseteq \sigma$;*

2. *for each* $\sigma \in K$*, there exists a finite subset* $\{\widetilde{\sigma}_i\} \subseteq \widetilde{K}$*, such that* $\sigma = \cup_i \widetilde{\sigma}_i$*.*

The SPLIT algorithm—to be detailed in the following—generates a subdivision, since for every cell $\widetilde{\sigma} \in \widetilde{K}$ we have by construction $\widetilde{\sigma} \subseteq \sigma \in K$. Property 2 is also satisfied, since every cell in $\widetilde{K}$ is mapped into the union of at most two halves $\widetilde{\sigma}^-$ and $\widetilde{\sigma}^+$, produced by the operation SPLIT K.

6.1 The split algorithm

Let us first introduce two auxiliary operators, to be used for the matrix formulation of the SPLIT algorithm.

Definition 2 (Sign function)
The operator $\mathrm{sgn}_\varepsilon : R^d \to \{-1, 0, 1\}^d$ *returns the matrix listing the signs of the elements* v_i *of a d-tuple* $\mathbf{v} = (\mathrm{v}_i)$*, taking into consideration the numerical tolerance* $\varepsilon > 0$*:*

$$(\mathrm{sgn}_\varepsilon \mathbf{v})_j = \begin{cases} -1, & \mathrm{v}_j < -\varepsilon \\ 0, & -\varepsilon \le \mathrm{v}_j \le \varepsilon \\ 1, & \mathrm{v}_j > \varepsilon \end{cases}$$

Definition 3 (Absolute value function)
The function abs *operates on a matrix* $M = (m_{ij})$ *returning the matrix of the absolute values of its elements:*

$$\mathrm{abs}\, M = (|m_{ij}|)$$

Consider the splitting hyperplane h characterized by the equation $\sum_p h_p x_p = b$ as a linear (affine homogeneous) form $E^{d+1} \to R$, represented by the row-matrix

$$\mathbf{h} = (\ h_1 \quad h_2 \quad \dots \quad h_d \quad -b\ ).$$

Let $\mathbf{v}$ be the column-matrix representation formed by the homogeneous coordinates of the 0-cell σ_0:

$$\mathbf{v} = (\ x_1 \quad x_2 \quad \dots \quad x_d \quad 1\ )^\top$$

Clearly, σ_0 belongs to the *above* subspace h^+ if and only if $h(\sigma_0) > 0$, while it belongs to the *below* subspace h^- if and only if $h(\sigma_0) < 0$. The sign of the scalar product $\mathbf{h}\,\mathbf{v}$ solves the *point location problem.*

Introducing the matrix

$$\mathbf{V} = (\ \mathbf{v}_1 \quad \mathbf{v}_2 \quad \cdots \quad \mathbf{v}_{k_0}\ )$$

that collects the homogeneous coordinates of all the 0-cells in K_0, their classification with respect to the h splitting hyperplane is codified by the 0-chain $c : K_0 \to \{-1, 0, 1\}$, represented by the matrix:

$$\mathbf{c}_0 = \mathrm{sgn}_\varepsilon(\mathbf{h}\,\mathbf{V}).$$

The SPLIT algorithm proceeds hierarchically from 0-cells up to d-cells by (a) classifying the cells with respect to the splitting hyperplane, and (b) updating the cell complex accordingly, including the new elements in the skeletons of all orders. The algorithm is sketched in Figure 7.

Remark 2 *For each dimension p, the absolute value* $|c_p^i|$ *of* $c_p(\sigma_p^i)$ *is compared with the value* $f_p^i = f_p(\sigma_p^i)$ *(step 5). In fact, the only p-cells that intersect the splitting hyperplane h are characterized by the inequality* $|c_p^i| \neq f_p^i$*.*

---

algoritm SPLIT (**input**: $K, \mathbf{V}, \mathbf{h}$; **output**: $\widetilde{K}, \widetilde{\mathbf{V}}$);

---

1. $p := 0$
2. Classify the 0-cells: $\mathbf{c}_0 := \mathrm{sgn}_\varepsilon(\mathbf{h}\mathbf{V})$
3. $p := p + 1$
4. Classify the p-cells and find their "face" class:

   $\mathbf{c}_p := (\mathrm{abs}\,[\delta_{p-1}])\,\mathbf{c}_{p-1}$

   $\mathbf{f}_p := (\mathrm{abs}\,[\delta_{p-1}])\,(\mathrm{abs}\,\mathbf{c}_{p-1})$
5. **foreach** $|c_p^i| \neq f_p^i$ **do**: Update the cell complex:

   Split the i-th p-cell: $K := \beta^{p-1}(K)$;

   Set the new element value: $c_{p-1}^{k_{p-1}} := 0$
6. Re-classify the p-cells of the updated cell complex:

   $\mathbf{c}_p := \mathrm{sgn}_\varepsilon\,((\mathrm{abs}\,[\delta_{p-1}])\,\mathbf{c}_{p-1})$
7. **if** $p < d$ **then** GOTO step 3, **else** STOP.

Figure 7: The SPLIT algorithm, implemented by using a classification chain and the coboundary operator.

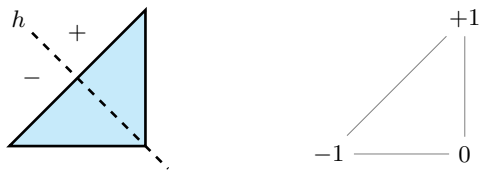


Figure 8: (a) The splitting hyperplane h, and (b) the classification of vertices.

6.2 Split example

Let us go back to the splitting example already discussed in Section 4.3 and refine the 2-complex with the hyperplane specified in Fig. 8a. The reader should recall Figs. 4, 5, and 6 and refer to them to locate by name the cells of the complex.

The SPLIT algorithm is initialized by setting $p = 0$ and by classifying the vertices through the 0-chain

$$\mathbf{c}_0 = \mathrm{sgn}_\varepsilon\,(\mathbf{h}\ (\ \mathbf{v}_1\quad \mathbf{v}_2\quad \mathbf{v}_3\ )) = (\ -1\quad 0\quad 1\ ),$$

as shown in Figure 8b. Then p is increased to 1 and 1-cells are classified by computing the 1-chains:

$$\begin{aligned}\mathbf{c}_1 &= (\mathrm{abs}\,[\delta_0])\,\mathbf{c}_0 = (\ 0\quad -1\quad 1\ ),\\ \mathbf{f}_1 &= (\mathrm{abs}\,[\delta_0])\,(\mathrm{abs}\,\mathbf{c}_0) = (\ 2\quad 1\quad 1\ ).\end{aligned}$$

Results are illustrated in Fig. 9: we see that σ^1_1 should be split, since $|c^1_1| \neq f^1_1$.

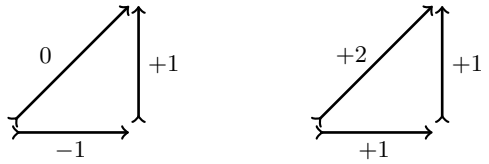


Figure 9: The 1-chains $\mathbf{c}_1$ and $\mathbf{f}_1$ used to detect the 1-cells that intersect the splitting hyperplane.

The application of the β^0 operator adds a new 0-cell (classified to 0) and a new 1-cell (see Figs. 10). The two 1-cells resulting from the split one, as shown in Fig. 10b, are reclassified.

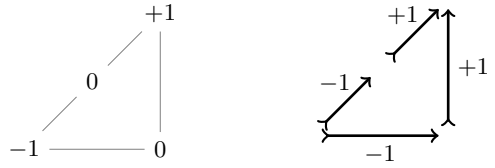


Figure 10: The updated cell complex, with 1-cells reclassified.

Then, p is increased to 2 and 2-cells are classified:

$$\mathbf{c}_2 = \mathrm{abs}\,[\delta_1]\,\mathbf{c}_1 = (\ 1\quad 1\quad 1\quad 1\ )(\ -1\quad -1\quad 1\quad 1\ )^\top = 0$$
$$\mathbf{f}_2 = \mathrm{abs}\,[\delta_1]\,\mathrm{abs}\,\mathbf{c}_1 = (\ 1\quad 1\quad 1\quad 1\ )(\ 1\quad 1\quad 1\quad 1\ )^\top = 4$$

(see Fig. 11a). Hence, the σ^1_2 cell gets split, the splitting being executed by the β^1 operator that creates one 1-cell and one 2-cell, as shown in Figs 11. Finally the algorithm re-classifies the 2-cells and terminates, since $p = d$. The result is illustrated in Fig. 11c, where the 2-chain generated on the refined complex $\widetilde{K}$ is illustrated.

6.3 Subdivision of a complex

Let us denote the support space of the complex K as $[\![K]\!]$. Since SPLIT is a subdivision generator, the process can be iterated by performing a second split, i.e. $\text{SPLIT}_2(\text{SPLIT}_1 K)$, and, more in general $\text{SPLIT}^N K$.

From the *finite approximation theorem* [Munkres 1984], we have that for any continuous map $\phi : [\![K]\!] \to [\![L]\!]$ between two cell complexes K and L, with K finite, there exists $N \in N$ such that ϕ may be approximated by a map $\psi : \text{SPLIT}^N K \to L$.

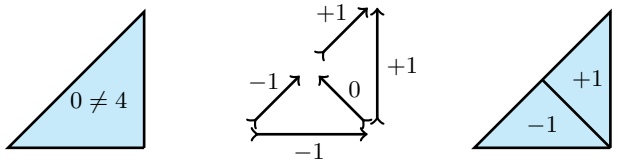


Figure 11: (a) Classification of the 2-cells, (b) the classification 1-chain on the refined 1-skeleton, and (c) the refined 2-skeleton with the classification 2-chain.

Another property of the SPLIT subdivision is guaranteed by the *algebraic subdivision theorem* [Munkres 1984]. The splitting induces a unique *chain map* ζ such that

$$\zeta : \mathcal{C}(K) \longrightarrow \mathcal{C}(\widetilde{K}).$$

Therefore, the induced chain map can be applied either to the subdivided chain or to the original chain complex, since

$$\partial \circ \zeta = \zeta \circ \partial.$$

The chain map can be summarized in the following *commutative* diagram:

$$\begin{array}{ccccc} \cdots \longrightarrow & C_p & \xrightarrow{\partial} & C_{p-1} & \longrightarrow \cdots \\ & \zeta\downarrow & & \downarrow\zeta & \\ \cdots \longrightarrow & \widetilde{C}_p & \xrightarrow{\partial} & \widetilde{C}_{p-1} & \longrightarrow \cdots \end{array}$$

As a result, boundaries in the *refined* cell complex $\widetilde{K}$ may be computed by applying the chain map ζ to boundaries evaluated in the *coarse* cell complex K.

7 Geometry & physics modeling

The (co)chain-complex formalism and the Hasse-matrix representation generalize in a natural and straightforward way to physical modeling. Chains assign *measures* to cells, measures that may be tuned to represent the physical properties of interest (mass, charge, conductivity, stiffness, and so on). Cochains, on the other side, may be used to represent all physical quantities associated to cells through *integration* with respect to a measure. The coboundary operator stays behind the basic structural laws (balance and compatibility) involving physically meaningful cochains [34, 25, 28]. It is also well known that k-cochains are the coarse-grained analogue of differential k-forms [36, 10]. Correspondingly, the cochain complex introduced in Section 2.3.1 is a discrete version of the De Rham complex [7, 23, 2], naturally represented by the Hasse matrix (or its transpose).

This view on physical modeling has been increasingly advocated [6, 2, 16] as a way to increase numerical stability and accuracy of various numerical methods. Even more important is the that a proper use of the Hasse matrix has the potential to bring both geometric and physical modeling within a unified computational framework. According to its definition (see Section 3.1), $H(K)$ provides a compact representation of purely *topological* operators, boundary ∂ and coboundary δ, acting on chains or cochains defined on K. Such a representation is mediated by a *metric* structure which embodies far more information than the topology of the cell complex K plus the measure-like properties imparted to it by the introduction of chains. This additional structure is brought in by the seemingly innocuous identification between elementary chains and elementary cochains.

However, the "obvious" cell-wise identification we have performed in Section 2.2.3, is associated with a conventional metric structure, easy to use on K, but totally unrelated—in general—to the metric properties relevant to the physics under consideration. Of course, the underlying topology stays untouched. Therefore, as long as one is only interested in having an easy-to-use *metrical representation* of topological operators, the metric involved is instrumental and one is allowed to use whichever is found convenient. Nevertheless, when the object itself, not only its representation, *does* depend on the metric—as when introducing the notion of *adjacency* between cells and the related notion of *Laplacean* (see the Appendix)—then it is essential to import into the model the relevant, physics-based metric structure, through a well-tuned identification of chains with cochains. As a consequence, the elementary chain $1\,\sigma$ will *not* be identified—in general—with the elementary cochain $1\,\sigma$. Approaching these issues is basic to gain the possibility of transferring information from K to its refinement $\widetilde{K}$.

A deeper discussion on metric issues is out of scope; we simply stress here that the same data structures and algorithms may be used both for solid modeling and physics-based simulations. From our vantage point, boundary representations and finite element meshes appear as two different aspects of the same Hasse representation. Furthermore, there is no fundamental distinction between different types of approximation methods: in [21, 20], by telling apart the metrical and topological properties embodied in the Hasse representation, we showed that all linear problem formulated by all finite methods are basically equivalent. Within our framework, the split algorithm described in Section 6 becomes a powerful method for progressive refinement not only of *shapes*, but also of the representation of *fields* living on those shapes.

8 Conclusions

Historically, the development of boundary representation schemes in solid modeling was driven by limited computational resources, and the usual space-time trade-offs [Woo 1985]. A typical boundary representation was chosen (a) to save memory, when RAM was small and expensive, and (b) to spare disk access times, by giving efficient answers to topological queries. Contrary to what might appear at first sight, the present approach does not imply higher theoretical complexity, since the number of non-zero elements in the Hasse matrix $H(K)$ is essentially of the same order as the number of adjacency pointers in a typical graph-based representation of the cell complex K. Furthermore, the Hasse matrix serves as a unifying standard for all boundary representations; the difference between different graph structures amount to different methods [Davis 2006] for encoding a subset of the sparse matrix $H(K)$.

We also note that the chain complex is a standard tool for representing and analyzing topological properties of arbitrary cellular spaces. It follows that the proposed Hasse matrix and transformations may codify much more general models, without restrictions on orientability, (co)dimension, manifoldness, connectivity, homology, and so on. The resulting framework, centered on a matrix representation of the domain of interest, unifies several geometric and physical finite formulations, and supports local progressive refinement and coarsening. This approach is inspired by the applications to be developed within the next generation of computational sciences. In particular, the new "big science" of life need simulation models of field problems where geometric and physical properties are generated, detailed, and refined *simultaneously* and *progressively*.

References

Alexandroff, P. S. 1998. *Combinatorial Topology*. Dover, New York.

Baez, J. 2003. Euler characteristic versus homotopy cardinality. Lecture at the Program on Applied Homotopy Theory, Fields Institute, Toronto, September.

Bajaj, C. L., and Pascucci, V. 1996. Splitting a complex of convex polytopes in any dimension. In *SCG '96: Proceedings of the twelfth annual symposium on Computational geometry*, ACM Press, Philadelphia, PA, 88–97.

Baumgart, B. G. 1972. *Winged-Edge Polyhedron Representation*. Tech. Rep. Stan-CS-320, Artificial Intelligence Laboratory, Stanford University, CA.

Branin, F. H. 1966. The algebraic-topological basis for network analogies and the vector calculus. In *Proceeding of the Symposium on Generalized Networks*, vol. 16, Polytechnic Institute of Brooklyn, 453–491.

Brisson, E. 1993. Representing geometric structures ind dimensions: Topology and order. *Discrete and Computational Geometry 9*, 1, 387–426.

Davis, T. A. 2006. Direct methods for sparse linear systems. In *Fundamentals of Algorithms*. Siam.

Eastman, C., and Weiler, K. 1979. *Geometric Modeling Using the Euler Operators*. Institute of Physical Planning, Carnegie-Mellon University.

Guibas, L., and Stolfi, J. 1985. Primitives for the manipulation of general subdivisions and the computation of Voronoi. *ACM Transactions on Graphics (TOG) 4*, 2, 74–123.

Hatcher, A. 2002. *Algebraic topology*. Cambridge University Press.

Hoffmann, C. 1989. *Geometric and solid modeling: an introduction*. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA.

Hyman, J., and Shashkov, M. 1997. Natural discretizations for the divergence, gradient and curl on logically rectangular grids. *International Journal of Computers and Mathematics with Applications 33*, 4, 81–104.

Mäntylä, M. 1988. *Introduction to Solid Modeling*. WH Freeman & Co. New York, NY, USA.

Masuda, H. 1993. Topological operators and Boolean operations for complex-based nonmanifold geometric models. *Computer Aided Design 25*, 2, 119–29.

Mattiussi, C. 1997. An analysis of finite volume, finite element, and finite difference methods using some concepts from algebraic topology. *Journal of Computational Physics 133*, 289–309.

Milicchio, F., DiCarlo, A., Paoluzzi, A., and Shapiro, V. 2006. A codimension-zero approach to discretizing and solving field problems. *Advanced Engineering Informatics*. Submitted for publication.

Milicchio, F. 2007. *Towards topological unification of finite computational methods*. PhD thesis, Dept. "Informatica e Automazione", University "Roma Tre". In preparation.

MUNKRES, J. R. 1984. *Elements Of Algebraic Topology.* Addison Wesley, Reading MA.

O'CONNOR, M. A., AND ROSSIGNAC, J. R. 1990. SGC: A dimension independent model for pointsets with internal structures and incomplete boundaries. In *IFIP/NSF Workshop on Geometric Modeling, Rensselaerville, NY, 1988*, North-Holland.

PALMER, R., AND SHAPIRO, V. 1993. Chain models of physical behavior for engineering analysis and design. *Research in Engineering Design 5*, 3, 161–184.

PAOLUZZI, A., PASCUCCI, V., AND VICENTINO, M. 1995. Geometric programming: a programming approach to geometric design. *ACM Trans. Graph. 14*, 3, 266–306.

PAOLUZZI, A., PASCUCCI, V., AND SCORZELLI, G. 2004. Progressive dimension-independent boolean operations. In *Proceedings of the 9th ACM Symposium on Solid Modeling and Applications*, G. Elber, N. Patrikalakis, and P. Brunet, Eds., ACM, 203–212.

REQUICHA, A. 1977. Mathematical models of rigid solid objects. Technical Memo 28, Production Automation Project, University of Rochester, Rochester, NY, November.

REQUICHA, A. 1980. Representations for rigid solids: Theory, methods, and systems. *Computing Surveys 12*, 4, 437–464.

SCORZELLI, G., PAOLUZZI, A., AND PASCUCCI, V. 2006. Parallel solid modeling using BSP dataflow. *Journal of Computational Geometry and Applications 16*. Accepted for publication.

SILVA, C. 1981. Alternative definitions of faces in boundary representatives of solid objects. Tech. rep., Production Automation Project, University of Rochester.

STRANG, G. 1988. A framework for equilibrium equations. *SIAM Review*, 30, 283–297.

TONTI, E. 1975. On the formal structure of physical theories. Tech. rep., Istituto di Matematica del Politecnico di Milano.

TONTI, E. 2001. A direct discrete formulation of field laws: The cell method. *Computer Modeling in Engineering & Sciences 2*, 2, 237–258.

WOO, T. 1985. Combinatorial analysis of boundary data structure schemata. *IEEE Computer Graphics and Applications 5*, 3, 19–27.

YAMAGUCHI, Y., AND KIMURA, F. 1995. Nonmanifold topology based on coupling entities. *Computer Graphics and Applications, IEEE 15*, 1, 42–50.

A Adjacency matrices

In graph theory, the adjacency matrix of vertices is one of the possible representations of a graph $G = (N, E)$ which is, by definition, a 1-complex $K = (K_0, K_1)$.

The well-known relation between the incidence matrix of a graph, its transpose and the adjacency matrix of its vertices can be generalized to boundary and coboundary operators of every order, and to the adjacency of p-cells in K_p, for any dimension p.

The topology of the 3-complex K depicted in Fig. 12 is represented by the matrices $[\delta_0] = [\partial_1]^\top$, $[\delta_1] = [\partial_2]^\top$, and $[\delta_2] = [\partial_3]^\top$.

Definition 4 *The symmetric matrices*

$$[\partial_{p+1}]\,[\delta_p] \quad \textit{and} \quad [\delta_{p-1}]\,[\partial_p]$$

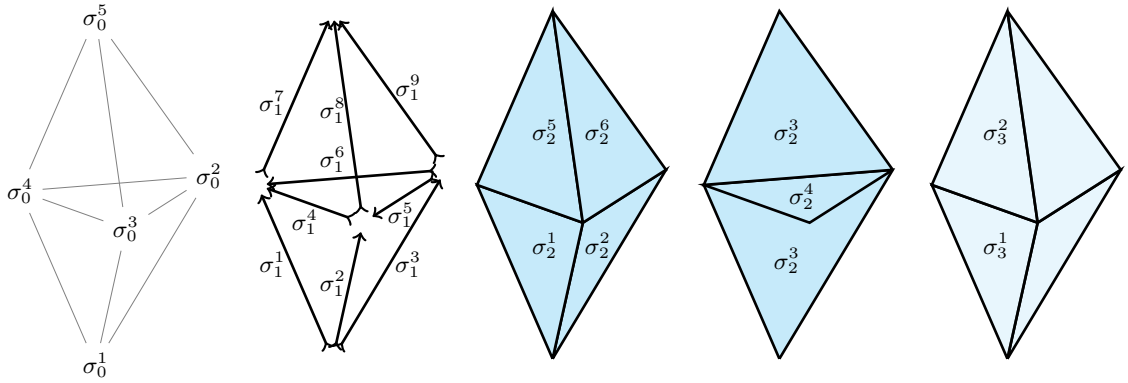
define the adjacency between p-cells through $(p+1)$-cells and $(p-1)$-cells, respectively.

While leaving to the reader the straightforward construction of this matrices, we stress here that such a representation makes use of the *standard* metric on K, by which each elementary chain 1σ is identified with the elementary cochain 1σ. The metric information introduced in this way becomes important when introducing and computing *adjacency matrices*, which imply the successive application of the boundary and coboundary operators (or viceversa).

It is worth mentioning that the discrete *Laplace-De Rham operators*

$$[\partial_{p+1}][\delta_p] + [\delta_{p-1}][\partial_p]$$

are just sums of adjacency matrices. They depend essentially on the metric carried by the matrix representation of boundary and coboundary operators.



K_0 K_1 K_2 K_3

$$[\partial_1]\,[\delta_0] = \begin{pmatrix} 3 & -1 & -1 & -1 & 0 \\ -1 & 4 & -1 & -1 & -1 \\ -1 & -1 & 4 & -1 & -1 \\ -1 & -1 & -1 & 4 & -1 \\ 0 & -1 & -1 & -1 & 3 \end{pmatrix}, \qquad [\partial_2]\,[\delta_1] = \begin{pmatrix} 2 & -1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & -1 & 2 & 0 & 1 & 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 3 & 1 & -1 & 1 & -1 & 0 \\ 0 & -1 & 1 & 1 & 3 & -1 & 0 & 1 & -1 \\ -1 & 0 & 1 & -1 & -1 & 3 & 1 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 2 & -1 & -1 \\ 0 & 0 & 0 & -1 & 1 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & -1 & -1 & -1 & -1 & 2 \end{pmatrix},$$

$$[\delta_0]\,[\partial_1] = \begin{pmatrix} 2 & 1 & 1 & 1 & 0 & 1 & -1 & 0 & 0 \\ 1 & 2 & 1 & 0 & 1 & -1 & 0 & -1 & 0 \\ 1 & 1 & 2 & -1 & -1 & 0 & 0 & 0 & -1 \\ 1 & 0 & -1 & 2 & 1 & 1 & -1 & 0 & 1 \\ 0 & 1 & -1 & 1 & 2 & -1 & 0 & -1 & 1 \\ 1 & -1 & 0 & 1 & -1 & 2 & -1 & 1 & 0 \\ -1 & 0 & 0 & -1 & 0 & -1 & 2 & 1 & 1 \\ 0 & -1 & 0 & 0 & -1 & 1 & 1 & 2 & 1 \\ 0 & 0 & -1 & 1 & 1 & 0 & 1 & 1 & 2 \end{pmatrix}, \qquad [\partial_3]\,[\delta_2] = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 2 & -1 & -1 & -1 \\ 0 & 0 & 0 & -1 & 1 & 1 & 1 \\ 0 & 0 & 0 & -1 & 1 & 1 & 1 \\ 0 & 0 & 0 & -1 & 1 & 1 & 1 \end{pmatrix},$$

$$[\delta_1]\,[\partial_2] = \begin{pmatrix} 3 & -1 & -1 & -1 & -1 & 0 & 0 \\ -1 & 3 & -1 & -1 & 0 & -1 & 0 \\ -1 & -1 & 3 & -1 & 0 & 0 & -1 \\ -1 & -1 & -1 & 3 & 1 & 1 & 1 \\ -1 & 0 & 0 & 1 & 3 & -1 & -1 \\ 0 & -1 & 0 & 1 & -1 & 3 & -1 \\ 0 & 0 & -1 & 1 & -1 & -1 & 3 \end{pmatrix}, \qquad [\delta_2]\,[\partial_3] = \begin{pmatrix} 4 & -1 \\ -1 & 4 \end{pmatrix}.$$

Figure 12: A 3-complex $K := (K_0, K_1, K_2, K_3)$ and its adjacency matrices.