Linear Algebraic Representation of Big Geometric Data

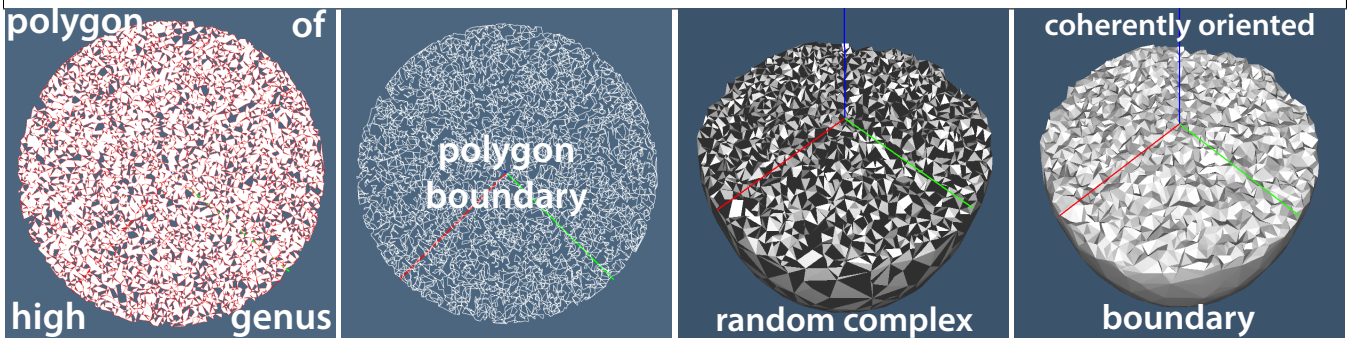
Alberto Paoluzzi*
Università "Roma Tre", Italy

Antonio DiCarlo†
Università "Roma Tre", Italy

Vadim Shapiro‡
University of Wisconsin, Madison, USA

If there is anything like a unifying aesthetic principle in mathematics, it is this: simple is beautiful.
"A Mathematician's Lament", **Paul Lockhart**

Simplicity is the ultimate sophistication. **Leonardo Da Vinci**
Numquam ponenda est pluralitas sine necessitate.
(paraphrased as Occam's razor) **William of Occam**



Abstract

A central notion in solid modeling is the concept of representation scheme, a mapping from mathematical solid models to actual computer representations. The aim of this paper is to introduce a novel representation scheme, usable in computer graphics, geometric design and solid modeling, as well as in computational science. Off-the-shelf representations and methods are needed to support geometrical and physical computing on meshed domains. However, unified representations of mesh connectivity in general applications and simulations do not yet exist. In this paper we introduce a linear algebraic method to represent and process mesh connectivity, in which only the vertices (or control points) of each cell of a model decomposition are listed. This approach is based on elementary homological methods from algebraic topology, takes advantage of sparse incidence matrices between vertices (or control points) and the generators of chains of cells, and applies to a superset of finite CW-complexes.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations; Geometric algorithms, languages, and systems;

Keywords: solid modeling, representation scheme, sparse matrix

Links: DL PDF

*e-mail:apaoluzzi@me.com
†e-mail:adicarlo@me.com
‡e-mail:vadim.shapiro@gmail.com

1 Introduction

Big data are commonly defined as datasets that grow so large that it becomes too hard working on them with standard database management tools [Hellerstein 2008]. Difficulties include storage, searching, sharing, sharding, doing computations and visualizing. Present-day problems in science and technology require simulation of big geometric data. Big geometric models, called meshes in computational science and engineering, are of primary importance in advanced application areas, such as materials science[1] and biomedicine[2]. Novel applications require the convergence of shape synthesis and analysis from computer graphics and computer-aided geometric design with discrete meshing of domains used for physical simulations. Pursuing such an objective calls for most methods underlying solid and physical modeling to be rethought from scratch to make them distributed and parallel.

In this paper we demonstrate that the topology of complex models is computable from simple mathematical data structures. In particular, we show that topological queries can be answered by straightforward sparse-matrix computations based on linear operators and elementary algebra, namely, by multiplying and transposing sparse matrices. Furthermore, our algebraic models are very compact, and there is no overhead in running time to answer topological queries in comparison to the traditional graph-based methods. Advanced implementations on modern hardware may use open standards for parallel programming of heterogeneous systems. Our approach works plainly with convex cell decompositions. With some more care, it applies even to non-manifold models, non-connected cells, d-cells non homeomorphic to d-balls.

The literature on *representation schemes* in solid modeling started with the foundational paper by [Requicha 1980], where a mathematical framework for characterising the important aspects of *computer representations of solids* was introduced. The groundbreaking paper by [Baumgart 1972] had supplied the first but already efficient *boundary representation* scheme. He also introduced *Euler operators*, elementary operations for step-wise build-

[1] Think of engineered surfaces, phase transitions or metamaterials.
[2] Consider, e.g., modeling the docking of biomolecules, the inter-neural space, or personalized inserts in orthopaedic surgery.

ing well-formed polyhedra, using atomic software actions satisfying the Euler-Poincaré formula at each stage. The *Quad-Edge* data structure, providing efficient primitives for planar subdivisions and Voronoi diagrams, was proposed in [Guibas and Stolfi 1985], and is still largely used in computational geometry algorithms and in geometric libraries. Variations of the *radial-edge* non-manifold representation by [Weiler 1988] have been embodied in almost every commercial CAD system that uses a non-manifold boundary representation. The *Cell-Tuple* structure [Brisson 1989] was introduced as a simple, uniform representation of finite and *regular* CW-complexes over subdivided d-manifolds. The cell-tuple unified preexisting work and provided intuitive clarity in all dimensions, however at the cost of a combinatorial explosion of the representation, actually not suitable for big models. More general set-theoretic and topological operators were provided by [Rossignac and O'Connor 1990] to represent inhomogeneous objects, building upon the *Selective Geometric Complex* (SGC), a superset of *CW-complexes* allowing for cells non homeomorphic to open balls, proposed to handle dimension-independent models of point sets with internal structures and incomplete boundaries. A dimension-independent generalisation of simplicial schemes, and various operators and algorithms were discussed by [Paoluzzi et al. 1993]. [Shapiro 2002] reviewed the main ideas and foundations of solid modeling in engineering. In particular, he remarked that solid modeling was conceived as a *universal technology* for developing engineering languages and systems with guaranteed geometric validity. Today, information and communication technologies are changing at a furious pace, and new challenges are posed by old and new application fields, producing highly demanding requirements. Despite the tremendous amount of research done and the progress made, the most used industrial softwares still follow the basic approach established twenty years ago, centered on boundary representations and non-manifold data structures. Only recently the field started moving away from boundary representations and akin off-line meshing, towards more general cellular decompositions [Wang et al. 2012] and direct integration between geometry and physics [DiCarlo et al. 2007; DiCarlo et al. 2009]. Similarly, the iso-geometric analysis by [Cottrell et al. 2009] requires 3D meshes, embedded using three-variate NURBs, so "allowing models to be designed, tested and adjusted in one integrated stage".

2 Background

A compact topological subspace is a *convex cell* if it is the set of solutions of affine equalities and inequalities. A *face* of a cell is the convex cell obtained by replacing some of the inequalities by equalities. A *facet* of a cell is a face defined by just one equality. The *dimension* n of a n-cell is that of its *affine hull*, the smallest affine subspace that contains it. A c*onvex-cell complex* or *polytopal complex* P is a finite union of convex cells such that: (i) if A is a cell of P, so are the faces of A; (ii) the intersection of two cells of P is a common face of each of them. A simplicial (respectively, cuboidal) complex is a polytopal complex where all cells are simplices (respectively, cuboids). The dimension of P is the maximal cell dimension of P. The r-skeleton P_r is the subcomplex formed by the cells of dimension $\leq r$. The 0-skeleton coincides with the set $V(P)$ of *vertices* of P.

Let X be a topological space, and Δ^m be the standard m-simplex. A *singular m-simplex* of X is a continuous map $\sigma : \Delta^m \to X$. The set of singular m-simplexes of X is denoted by $\mathcal{S}_m(X)$. The set of singular m-cochains of X is denoted by $C^m(X)$ and that of singular m-chains by $C_m(X)$. DEF A and DEF B endow $C^m(X)$ and $C_m(X)$ with the structure of a $\mathbb{Z}_2$-vector space, where singletons provide a basis of $C_m(X)$, in bijection with $\mathcal{S}_m(X)$. Thus, the following definitions are equivalent [Hausmann 2012]:

DEF A (a) A *singular m-cochain* of X is a subset of $\mathcal{S}_m(X)$;
(b) a *singular m-chain* of X is a finite subset of $\mathcal{S}_m(X)$.

DEF B (a) A *singular m-cochain* is a function $\phi : \mathcal{S}_m(X) \to \mathbb{Z}_2$;
(b) a *singular m-chain* is a function $\alpha : \mathcal{S}_m(X) \to \mathbb{Z}_2$ with finite support.

DEF C $C_m(X)$ is the $\mathbb{Z}_2$-vector space with basis $\mathcal{S}_m(X)$, such that $C_m(X) = \bigoplus_{\sigma \in \mathcal{S}_m(X)} \mathbb{Z}_2 \sigma$.

Let $\Lambda(X) = \cup_n \Lambda_{n \in \mathbb{N}}$ be a partition of X, with Λ_p the set of p-cells. A *CW-structure* on the space X is a filtration $\emptyset = X^{-1} \subset X_0 \subset X_1 \subset \cdots \subset X = \cup_n X_n$, such that, for each n, the space X_n is homeomorphic to a space obtained from X_{n-1} by attachment of n-cells of X in $\Lambda_n = \Lambda_n(X)$. A space endowed with a CW-structure is a *CW-complex*, also said a *cellular complex*. A cellular complex is *finite* when it contains a finite number of cells.

The definitions above can be repeated for *cellular (co)chains* supported by a cellular complex. In particular, $C_m(X)$ is the $\mathbb{Z}_2$-vector space with basis $\Lambda_m(X)$, such that $C_m(X) = \bigoplus_{\lambda \in \Lambda_m(X)} \mathbb{Z}_2 \lambda$. The cellular and the singular (co)homology of a CW-complex are isomorphic. A cellular *chain complex* consists of $\mathbb{Z}_2$-vector spaces $C_i(X)$ $(0 \leq i \leq n)$ and linear *boundary* operators $\partial_i : C_i(X) \to C_{i-1}(X)$ satisfying $\partial_{i-1} \circ \partial_i = 0$.

For a topological space X, let $\langle , \rangle : C^n(X) \otimes C_n(X) \to \mathbb{Z}_2$ be the *Kronecker pairing*, given by

$$\langle \phi, \alpha \rangle := \phi(\alpha) = \sharp(\phi \cap \alpha) (\text{mod } 2) = \sum_{\lambda \in \alpha} \phi(\lambda).$$

The Kronecker pairing has an intuitive geometric interpretation (chain-cochain intersection), making the cohomology as the dual of the homology in an elementary way [Hausmann 2012]. In terms of this pairing, the *coboundary* map $\delta : C^n(X) \to C^{n+1}(X)$ is defined by $\langle \delta\phi, \alpha \rangle = \langle \phi, \partial\alpha \rangle$ for all $\alpha \in C_{n+1}(X)$. Therefore, δ may be seen as the Kronecker *adjoint* of ∂, and their matrix representations are transposed of each other.

3 LAR scheme

Here we introduce the *Linear Algebraic Representation* (LAR) scheme. The aim is to provide a representation that could support (at least in principle) all topological and geometric queries and constructions that may be asked of the corresponding model. In this paper, we deal only with topological queries and constructions, as opposed to intersection, distances, and membership tests.

Let a Hausdorff space X and a finite cellular complex $\Lambda(X)$ be given, such that $X = \Lambda_0 \cup \cdots \cup \Lambda_d$, and let $M_p \in \mathbb{Z}_2^{m \times n}$ be binary matrices, with $0 \leq p \leq d$, $m = \sharp\Lambda_p$ rows, and $n = \sharp\Lambda_0$ columns.

Definition 1 (Math models) *The LAR domain is the set M of* chain complexes *supported by a finite cellular complex $\Lambda(X)$.*

Definition 2 (Computer representations) *The LAR codomain is the set R of d-tuples of* CSR sparse binary matrices[3]

$$\text{CSR}(M_n^m(\mathbb{Z}_2)), \qquad m = \sharp\Lambda_p, \quad n = \sharp\Lambda_0,$$

where d is the dimension of the space $X = \bigcup_p \Lambda_p$, and $1 \leq p \leq d$.

Each $M_p = M_p(\Lambda(X))$ is an *incidence matrix*, i.e., its entry $M_p(\mu, \nu) = 1$ if and only if the cell $\mu \in \Lambda_p$ contains the (vertex) cell $\nu \in \Lambda_0$.

[3] Compressed Sparse Row (CSR) format, for which efficient implementations on high-performance hardware exist. See [Buluç and Gilbert 2012] and [Lokhmotov 2012].

The interesting point is that, for a given cellular d-complex $\Lambda(X)$, all of the M_p matrices ($1 \leq p \leq d$) have the same number of columns. This simple fact provides a convenient tool for computing boundary and coboundary operators and topological relations between cells.

A polytopal d-complex is *regular* when each cell is contained in a d-cell. For a regular polytopal d-complex the only M_d matrix already suffices to compute everything needed: giving just CSR(M_d) is enough to fully characterize the chain complex induced by $\Lambda(X)$. For a given decomposition, no ambiguity may arise. Also we note that LAR is very compact: the boundary representation of a 3-manifold is $|FV| = 2|E|$. For a 3-cube this means $|FV| = 6 \times 4 = 2 \times 12 = 2|E|$. It is worth noting that CSR(M_d) exactly coincides with the bulk of common representations in Computer Graphics — for instance, the OBJ and PLY file formats.

4 LAR operations

Incidence/adjacency operators The operators corresponding to relations VV $\subset V \times V$, VE $\subset V \times E$, and VF $\subset V \times F$ are given below. Similar sparse matrix representations are obtained for the operators that associate the incident vertices, edges and faces (better: the incident 0-, 1-, and 2-chains) with 1-chains, and for operators that associate 0-, 1-, and 2-chains with 2-chains:

$$\begin{array}{lll} \mathcal{VV} : C_0 \to C_0, & \mathcal{EV} : C_0 \to C_1, & \mathcal{FV} : C_0 \to C_2; \\ \mathcal{VE} : C_1 \to C_0, & \mathcal{EE} : C_1 \to C_1, & \mathcal{FE} : C_1 \to C_2; \\ \mathcal{VF} : C_2 \to C_0, & \mathcal{EF} : C_2 \to C_1, & \mathcal{FF} : C_2 \to C_2. \end{array}$$

All relations between boundary entities can be computed by at most a sparse matrix transposition and a sparse matrix multiplication. Also, any topological query, i.e. the computation of the subset of elements incident on a given element, only requires the extraction of a row of the matrix of the appropriate operator. For a cellular decomposition $\Lambda(X)$ of a 2D space, and hence for a solid 2-shape or for the boundary of a solid 3-shape, we have 9 incidence or adjacency relations between pairs of boundary elements. More binary relations appear in higher dimensions.

$$\begin{array}{lcl} \mathcal{VV} = \mathcal{VE} \circ \mathcal{EV} = \mathcal{EV}^\top \circ \mathcal{EV} & \Rightarrow & [\mathcal{VV}] = M_1^t M_1 \\ \mathcal{VE} = \mathcal{EV}^\top & \Rightarrow & [\mathcal{VE}] = M_1^t \\ \mathcal{VF} = \mathcal{FV}^\top & \Rightarrow & [\mathcal{VF}] = M_2^t \\ \mathcal{EV} & & [\mathcal{EV}] = M_1 \\ \mathcal{EE} = \mathcal{EV} \circ \mathcal{VE} = \mathcal{EV} \circ \mathcal{EV}^\top & \Rightarrow & [\mathcal{EE}] = M_1 M_1^t \\ \mathcal{EF} = \mathcal{EV} \circ \mathcal{VF} = \mathcal{EV} \circ \mathcal{FV}^\top & \Rightarrow & [\mathcal{EF}] = M_1 M_2^t \\ \mathcal{FV} & & [\mathcal{FV}] = M_2 \\ \mathcal{FE} = \mathcal{FV} \circ \mathcal{VE} = \mathcal{FV} \circ \mathcal{EV}^\top & \Rightarrow & [\mathcal{FE}] = M_2 M_1^t \\ \mathcal{FF} = \mathcal{FV} \circ \mathcal{VF} = \mathcal{FV} \circ \mathcal{FV}^\top & \Rightarrow & [\mathcal{FF}] = M_2 M_2^t \end{array}$$

Topological queries The p-chain (i.e. the p-cell subset) being in a specific incidence/adjacency relation with a given unit q-chain (i.e. a given q-cell) is computed by a sparse matrix-vector multiplication. Let $[\epsilon_0^i]$, $[\epsilon_1^j]$ and $[\epsilon_2^h]$ denote the elements of the standard bases of linear spaces C_0, C_1, and C_2, respectively. The whole set of elementary queries on a 2D cellular complex is listed below.

$$\begin{array}{c} \mathcal{VV}(\epsilon_0^i) = [\mathcal{VV}][\epsilon_0^i],\ \mathcal{EV}(\epsilon_0^i) = [\mathcal{EV}][\epsilon_0^i],\ \mathcal{FV}(\epsilon_0^i) = [\mathcal{FV}][\epsilon_0^i], \\ \mathcal{VE}(\epsilon_1^j) = [\mathcal{VE}][\epsilon_1^j],\ \mathcal{EE}(\epsilon_1^j) = [\mathcal{EE}][\epsilon_1^j],\ \mathcal{FE}(\epsilon_1^j) = [\mathcal{FE}][\epsilon_1^j], \\ \mathcal{VF}(\epsilon_2^h) = [\mathcal{VF}][\epsilon_2^h],\ \mathcal{EF}(\epsilon_2^h) = [\mathcal{EF}][\epsilon_2^h],\ \mathcal{FF}(\epsilon_2^h) = [\mathcal{FF}][\epsilon_2^h]. \end{array}$$

Of course, any query concerning a general (non unit) chain equally requires a single (sparse) matrix-vector multiplication. The complexity of any such a query is optimal, as linear in the output size.

Star operator The star of a cell λ in a cellular complex $\Lambda(X)$, denoted St λ, is the set of all cells in Λ that have at least a face in λ. In algebraic terms, it is a linear endomorphism of $\mathcal{C} = \oplus_p C_p$: St : $\mathcal{C} \to \mathcal{C}$. In practical terms, it is easy to compute. To get the star of an elementary chain $\{\lambda\} \in C_p$, just consider its unit covector representation $[\lambda]^t$, multiply it by all the matrices $M_{p,q} := M_p\, M_q^t$, and make the union of the resulting sets of cells. An important application of the star operator is the following. Let suppose to allocate a *big geometric model* across a distributed computational environment. In order to execute any distributed simulation, the (image of the) star of each submodel must be computed, and allocated on both sides of the pair of processes sharing a portion of the submodel boundary.

Boundary and coboundary operators The *boundary* operator $\partial : C_m \to C_{m-1}$ is the $\mathbb{Z}_2$-linear map defined by

$$\partial(\lambda) = \{\tau \in \Lambda_{m-1}(\lambda)\}, \qquad \lambda \in \Lambda_m(X).$$

The *coboundary* $\delta : C^{m-1} \to C^m$ is defined by the identity

$$\langle \delta\phi, \alpha \rangle = \langle \phi, \partial\alpha \rangle, \qquad \forall \alpha \in C_m.$$

In particular, if $\lambda \in \Lambda_m$ and $\tau \in \Lambda_{m-1}$, then

$$\tau \in \partial(\lambda) \Leftrightarrow \tau \subset \lambda \Leftrightarrow \lambda \in \delta(\tau).$$

The coboundary is the *Kronecker adjoint* of the boundary:

$$\delta_{p-1} = \partial_p^\top \quad \Longleftrightarrow \quad [\,\delta_{p-1}\,] = [\,\partial_p\,]^t, \qquad 1 \leq p \leq d.$$

Let us consider the incidence operator $\mathcal{I}_{p-1,p} : C_p \to C_{p-1}$ and its matrix $[\mathcal{I}_{p-1,p}] := M_{p-1,p} = M_{p-1}\, M_p^t$. The entry $M_{p-1,p}(i,j)$ stores the value of the application of the $(p-1)$-cochain generator μ_{p-1}^i on the p-chain generator λ_p^j:

$$M_{p-1,p}(i,j) = \langle \mu_{p-1}^i, \lambda_p^j \rangle = \mu_{p-1}^i(\lambda_p^j)$$

By standard multiplication in $\mathbb{Z}$ (*not* $\mathbb{Z}_2$), we get

$$\begin{aligned} M_{p-1,p}(i,j) &= \sum_{h=0}^{k_0-1} \left(M_{p-1}(i,h)\right)\left(M_p(j,h)\right) \\ &= \sharp(\mu_{p-1}^i \cap \lambda_p^j), \end{aligned}$$

thus computing the number of vertices of the intersection (i.e., of the common face) between $\mu_{p-1}^i \subset \Lambda_0$ and $\lambda_p^j \subset \Lambda_0$. This face coincides with μ_{p-1}^i if and only if $\sharp(\mu_{p-1}^i \cap \lambda_p^j) = \sharp\mu_{p-1}^i$. In such a case, we have $\mu_{p-1}^i(\lambda_p^j) = \mu_{p-1}^i \in \partial\lambda_p^j$.

Algorithm Therefore, as a computational procedure to calculate the *unoriented boundary* operator, we have the following algorithm:

1. Compute $M_{p-1,p} := M_{p-1} M_p^t$ by standard matrix product of sparse matrices.
2. For each $0 \leq i \leq k_{p-1} - 1$, compute the number of stored elements in row i of CSR(M_{p-1}):

$$k = \sum_h M_{p-1}(i,h) =: \sharp\mu_{p-1}^i,$$

and, for each $0 \leq j \leq k_p - 1$, set:

$$[\,\partial_p\,](i,j) = \begin{array}{ll} 1 & \text{if } M_{p-1,p}(i,j) = k; \\ 0 & \text{otherwise.} \end{array}$$

Product of cell complexes Let $A = \Lambda(X)$ and $B = \Lambda(Y)$ be cell complexes of dimension m and n, respectively. Then, $C = A \times B = \Lambda(X \times Y)$ is a cell complex of dimension $m + n$, with cells the products $\lambda^h \times \lambda^k$, with $\lambda^h \in \Lambda_m(X)$ and $\lambda^k \in \Lambda_n(Y)$. Assuming $X \subset \mathbb{E}^c$ and $Y \subset \mathbb{E}^d$, it is $X \times Y \subset \mathbb{E}^{c+d}$. If $V(X)$ and $V(Y)$ are the column matrices of positions of the X and Y vertices, then we have

$$V(X \times Y) = V(X) \otimes V(Y)^t.$$

The implementation in the LAR scheme is very simple, reducing to a doble loop on cells and vertices. Given two triangles $\alpha = (v_1, v_2, v_3)$ and $\beta = (v_4, v_5, v_6)$, the product cell $\alpha \times \beta$ has representation $(v_{14}, v_{15}, v_{16}, v_{24}, v_{25}, v_{26}, v_{34}, v_{35}, v_{36})$, with $v_i, v_j \in \mathbb{E}^2$ and $v_{ij} = (v_i, v_j) \in \mathbb{E}^4$. Examples are shown in Figures 1c and 1d.

Facet extraction from polytopes A subset of facets of d-cells of a polytopal complex $\Lambda_d(X)$ can be extracted by $\varphi : C_d \to C_{d-1}$, analyzing the symmetric adjacency matrix $M_{d,d} = M_d M_d^t$, whose row i contains the numbers of vertices shared by the cell λ_d^i with other d-cells λ_d^j, $(0 \leq j \leq d)$. Therefore, where $M_{d,d}(i, j) \geq d$, the two incident cells necessarily share a $(d-1)$-face. This shared cell $\mu \in C_{d-1}$ (a unit chain) is computed as $\lambda_d^i \cap \lambda_d^j = \langle \lambda_d^i, \lambda_d^j \rangle$. The other facets, i.e., the ones not shared by two d-cells, are computed using the *qhull* algorithm [Barber et al. 1996]. Combinatorial optimizations are possible when the cells are cuboidal or simplicial. Of course, the proportion of *qhull* evaluations to the total is $O(\partial(X)/\Lambda(X))$.

5 LAR examples

We discuss a few elementary examples, in order to demonstrate the simple computations required by LAR. In all of our examples, FV and EV give a CSR representation of the binary matrices M_2 and M_1, respectively. Since the LAR representation is purely topological, the actual shape of the cells does not matter.

Non-manifold simplicial complex Let's consider the $\Lambda(X)$ decomposition in Figure 2a, as a simplicial complex that is locally non manifold: $\Lambda = \Lambda_0 \cup \Lambda_1 \cup \Lambda_2$, with $k_0, k_1, k_2 = 9, 16, 6$.

```
FV = [[0,1,3],[1,2,4],[2,4,5],[3,4,6],[4,6,7],[5,7,8]]
```

The array FV = CSR(M_2) encodes the linear space $C_2(\Lambda)$ of 2-chains, generated by the matrix rows, through a standard basis (isomorphic to the 2-cells). The 2-cells are given as linear combinations of the standard basis of 0-chains (isomorphic to the 0-cells). This information is provided for the linear space $C_1(\Lambda)$ of 1-chains by the array EV = CSR(M_1). Note that FV is the only needed input: for a simplicial complex, the array EV may be computed as φ(FV), where φ is the facet extraction discussed in Section 4.

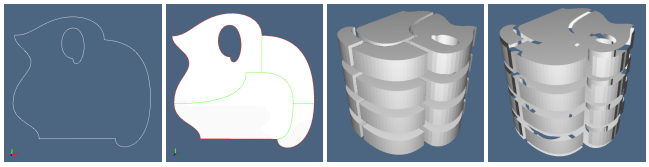
Complexes with non-convex and curved cells Examples of non-linear and/or non-convex cells are given in Figures 1b, 1c, 1d. Since vertices are used in this example as Bézier control points, their order and location may affect the cell topology. Therefore, this geometric information is here topologically relevant.

```
# input of vertex locations (numbered from zero)
V = [[5.,29.],[17.,29.],[8.,25.],[11.,25.],[14.,25.],[0.,
23.],[5.,23.],[17.,23.],[27.,23.],[0.,20.],[5.,20.],[8.,
19.],[11.,19.],[11.,17.],[14.,17.],[0.,16.],[5.,16.],[14.
,16.],[17.,16.],[23.,16.],[0.,10.],[14.,10.],[23.,10.],[
27.,10.],[0.,6.],[5.,6.],[5.,3.],[20.,3.],[23.,3.],[20.,
0.],[27.,0.]]
```

Edges and faces describe the 2D and 1D cells partitioning the geometry, named here according to their meaning, i.e. FV (faces-by-vertices) and EV (edges-by-vertices), respectively. Edge lists are ordered and will be rendered as Bézier curves.

```
# input of faces as lists of control-point indices
FV = [[0,1,2,3,4,5,6,7,9,10,11,12,13,14,15,16,17,18,20,
21],[7,8,18,19,22,23],[17,18,19,20,21,22,24,25,26,27,28],
[22,23,27,28,29,30]]
# input of edges as lists of control points indices
EV = [[5,6,0,1,7],[7,18],[18,17],[17,21,20],[20,15,16,10,
9,5], [12,11,2,3],[3,4,14,13,12],[7,8,23],[22,23],[22,19,
18],[22,28,27],[26,27],[26,25,24,20],[23,30,29,27]]
```

A canonical (numerically ordered) description of 2-cells is used for FV, without any topological ordering. On the contrary, control points of Bézier curves have to be properly ordered.



(a) (b) (c) (d)

Figure 1: *A cellular complex $B = \Lambda(X) \subset \mathbb{E}^2$, with curved 2-cells non homeomorphic to the 2-ball: (a) boundary 1-chain; (b) boundary 1-chain (red) and interior 1-chain (green); (c) solid complex $B \times C$, with $C \subset \mathbb{E}$; (d) 2D complex $(B \times \partial C) \cup (\partial B \times C)$.*

Boundary operator The LAR representation of the cellular complex $\Lambda(X)$ in Figure 2d, with $\Lambda_1 = \{e_0, e_1, \ldots, e_9\}$, and with non-convex faces $\Lambda_2 = \{f_0, f_1\}$, is:

```
FV = [[0,1,3,5,6,7],[0,2,3,4,5,6]]
EV = [[0,1],[0,2],[0,6],[1,3],[2,3],[3,5],[4,5],[4,6],
      [5,7],[6,7]]
```

The binary matrices M_1 and M_2, whose columns are indexed by 0-cells, and rows by 1- or 2-cells, respectively, are given by

$$M_1 = \begin{pmatrix} 1&1&0&0&0&0&0&0 \\ 1&0&1&0&0&0&0&0 \\ & & \cdots & & & & & \\ 0&0&0&0&0&1&0&1 \\ 0&0&0&0&0&0&1&1 \end{pmatrix}, \quad M_2 = \begin{pmatrix} 1&1&0&1&0&1&1&1 \\ 1&0&1&1&1&1&1&0 \end{pmatrix}.$$

To obtain the matrix representation of $\partial_2 : C_2 \to C_1$, the edge-face incidence is computed first:

$$[\mathcal{EF}] = [\mathcal{I}_{1,2}] = M_1 M_2^t = \begin{pmatrix} 2&1&2&2&1&2&1&1&2&2 \\ 1&2&2&1&2&2&2&2&1&1 \end{pmatrix}^t,$$

where the (i, j) entry denotes the number of vertices shared by the (elementary) chains $\mu_i \in C_1$ and $\lambda_j \in C_2$. Let us recall from Section 4 that

$$[\partial_p](i,j) = \begin{array}{ll} 1 & \text{if } M_{p-1,p}(i,j) = \sharp\mu_i = 2 \\ 0 & \text{otherwise.} \end{array}$$

Hence we get:

$$[\partial_2] = \begin{pmatrix} 1&0&1&1&0&1&0&0&1&1 \\ 0&1&1&0&1&1&1&1&0&0 \end{pmatrix}^t.$$

Finally, the total boundary 1-chain is computed as the mod 2 product of $[\partial_2]$ times the coordinate representation of the *total 2-chain* $[\Lambda_2] = [1, 1]^t$. The result is shown in blue in Figure 2d.

$$[\partial_2][\Lambda_2] = [1,1,0,1,1,0,1,1,1,1]^t = \{e_0, e_1, e_3, e_4, e_6, e_7, e_8, e_9\}.$$

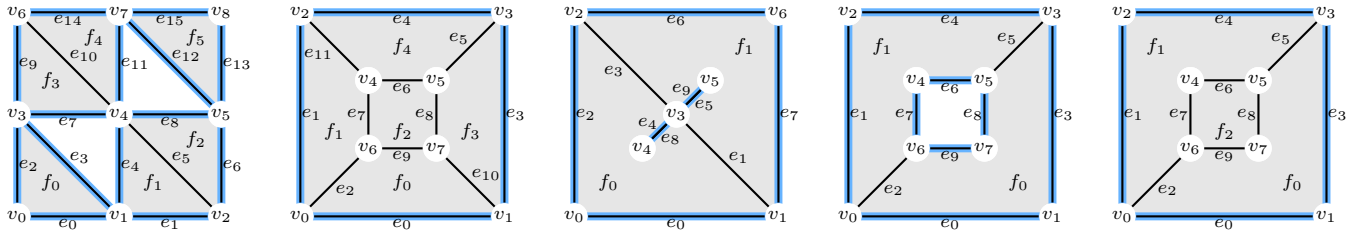


Figure 2: *Cellular 2-complexes and their boundaries (blue): (a) non-manifold simplicial complex; (b) cuboidal complex; (c) complex with internal boundaries; (d,e) complexes with non-convex 2-cells.*

Facet extraction Facet extraction using the mixed method suggested for general polytopal complexes is shown in Figure 3. In particular, the *qhull* algorithm, independently applied to each exterior 3-cell, returns a subset of simplicial boundary facets (see Figure 3c), that are pairwise attached, via set-union of their vertices (see Figure 3d), when sharing the same affine hull.

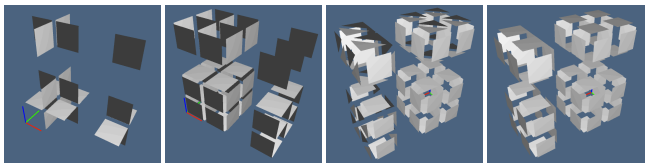


Figure 3: *Extraction of facets using the $M_d M_d^t$ matrix: (a) internal facets of general polytopal cells; (b) using the* a priori *knowledge that cells are cuboidal; (c) simplicial boundary facets from* qhull *algorithm; (d) attachment of boundary facets with same affine hull.*

6 Conclusion

In this paper we introduced the LAR representation scheme, a simple algebraic rep for cell complexes, using a CSR (Compressed Sparse Row) form for characteristic matrices of linear spaces of (co)chains. LAR allows us to compute globally and query locally a model topology through sparse matrix multiplication, and supports not only boundary reps, but also decompositions of the interior. This scheme may handle general piecewise-affine cells, and also curved cells. It is space-optimal for regular polytopal complexes: the storage size for boundary reps of 3-manifolds is $|\mathrm{FV}| = 2|\mathrm{E}|$. It is dimension-independent, not restricted to regular complexes, and allows for internal/external structures, including cracks and fibers of lower dimensions. From a technology viewpoint, LAR may obtain parallel performance via OpenCL, OpenGL and their web porting, since it does not require traversing or searching linked data structures. Furthermore, LAR enjoys a neat mathematical format—being based on *chains*, i.e. the domains of discrete integration, and *cochains*, i.e. the discrete prototype of differential forms. LAR is being implemented using the Khronos's APIs for industry standard heterogeneous computing. We are confident that this algebraic representation can help dealing with biomedical applications, which require fast performances with big geometric data.[4]

References

BARBER, C. B., DOBKIN, D. P., AND HUHDANPAA, H. 1996. The quickhull algorithm for convex hulls. *ACM Trans. Math. Softw. 22*, 4 (Dec.), 469–483.

BAUMGART, B. G. 1972. Winged edge polyhedron representation. Tech. Rep. Stan-CS-320, Stanford, CA, USA.

BRISSON, E. 1989. Representing geometric structures in d dimensions: topology and order. In *Proc. of the 5-th Annual Symposium on Computational Geometry*, Acm, New York, NY, USA, SCG '89, 218–227.

BULUÇ, A., AND GILBERT, J. R. 2012. Parallel sparse matrix-matrix multiplication and indexing: Implementation and experiments. *SIAM Journal of Scientific Computing (SISC) 34*, 4, 170 – 191.

COTTRELL, J., HUGHES, T., AND BAZILEVS, Y. 2009. *Isogeometric analysis: toward integration of CAD and FEA*. Wiley.

DICARLO, A., MILICCHIO, F., PAOLUZZI, A., AND SHAPIRO, V. 2007. Solid and physical modeling with chain complexes. In *Proceedings of the 2007 ACM symposium on Solid and physical modeling*, ACM, New York, NY, USA, SPM '07, 73–84.

DICARLO, A., MILICCHIO, F., PAOLUZZI, A., AND SHAPIRO, V. 2009. Chain-based representations for solid and physical modeling. *Automation Science and Engineering, IEEE Transactions on 6*, 3 (July), 454 –467.

GUIBAS, L., AND STOLFI, J. 1985. Primitives for the manipulation of general subdivisions and the computation of voronoi. *ACM Trans. Graph. 4*, 2 (Apr.), 74–123.

HAUSMANN, J.-C. 2012. *Mod Two Homology and Cohomology*. Book Project, http://www.unige.ch/math/folks/hausmann/.

HELLERSTEIN, J. 2008. Parallel programming in the age of big data. *Gigaom Blog*, November. http:// gigaom.com/2008/ 11/09/mapreduce-leads-the-way-for-parallel-programming/.

LOKHMOTOV, A. 2012. Implementing sparse matrix-vector product in OpenCL. In *OpenCL tutorial, 6th Int. Conference on High Performance and Embedded Architectures and Compilers (HiPEAC'11)*.

PAOLUZZI, A., BERNARDINI, F., CATTANI, C., AND FERRUCCI, V. 1993. Dimension-independent modeling with simplicial complexes. *ACM Trans. Graph. 12*, 1 (Jan.), 56–102.

REQUICHA, A. G. 1980. Representations for rigid solids: Theory, methods, and systems. *ACM Comput. Surv. 12*, 4 (Dec.), 437–464.

ROSSIGNAC, J. R., AND O'CONNOR, M. A. 1990. SGC: a dimension-independent model for pointsets with internal structures and incomplete boundaries. In *Geometric modeling for product engineering*. North-Holland.

SHAPIRO, V. 2002. Solid modeling. In *Handbook of Computer Aided Geometric Design*, G. Farin, J. Hoschek, and S. Kim, Eds. Elsevier Science, ch. 20, 473–518.

WANG, K., LI, X., LI, B., XU, H., AND QIN, H. 2012. Restricted trivariate polycube splines for volumetric data modeling. *IEEE Trans. Vis. Comput. Graph. 18*, 5, 703–716.

WEILER, K. J. 1988. The radial edge structure: A topological representation for non-manifold geometric modelling. In *Geometric modelling for CAD applications*, M. Wozny, H. McLaughlin, and J. Encarnacao, Eds. North-Holland, Amsterdam, 3–12.

[4] In fact we are working towards a proposal and a prototypal implementation within the frame of the IEEE Project P3333.2 - Standard for Three-Dimensional Model Creation Using Unprocessed 3D Medical Data.