Computational Graphics: Lecture 10

The CVDlab Team

Mon, Mar 24, 2014

1. Motivations
2. Towards a standard for all topological structures
   - The LAR proposal
   - LAR models: chain complexes
   - LAR representations: CSR matrices
3. Examples
   - 2D simplicial complex (solid & non-manifold pointset)
4. Some LAR operations
   - Boundary and coboundary
   - Topological queries
5. Application: quotient chains from 3D images
   - Model extraction
   - Divide et impera

Rethinking some foundations

of geometric and topological computing

Computational problems in science and technology must deal with increasingly complex geometric information and applications.

Complexity of geometric information stems from dramatic increase in size, diversity, and complexity of geometric data:

- point clouds,
- boundary meshes,
- NURBs representations,
- finite element meshes,
- CT scans,
- and so on

Rethinking some foundations

of geometric and topological computing

Emerging applications (e.g. medical 3D) require the convergence of shape synthesis and analysis from:

- computer imaging
- computer graphics
- computer-aided geometric design
- discrete meshing of domains
- physical simulations

The goals of unification, scalability, and massively parallel distributed computing

call for rethinking of the foundations of geometric and topological computing

The evolution of 3D geometric representations

can be generally understood in terms of graph-based data structures representing one of several possible cells complexes partitioning either the boundary or the interior of the represented model

Remark

Variety of assumptions about the cell complexes and graph representations:

- *make standardization difficult*
- *complicate the issues of data exchange and transfer*
- *and lead to proliferation of incompatible algorithms*

Towards rethinking some foundations

E.g.: boundary representation algorithms are dominated by graph searching algorithms (boundary traversals) that tend to force serial processing

Remark

In short, the specialized data structures based on cell complexes,

that have driven the evolutionary development of the field,

are no longer adequate for dealing with the emerging challenges and opportunities

A possible solution

Define a standard for model topology using a very general and simple repr scheme:

Models: (co)chain complexes $\rightarrow$ Reprs: sparse binary matrices

LAR scheme

Using standard concepts from landmark paper (Requicha, 1980)

Let $\Lambda(X)$ is a finite cellular complex, such that $X = \Lambda_0 \cup \cdots \cup \Lambda_d$

Definition (LAR scheme)

is a map from chain complexes to (sparse) characteristic matrices:

$$\text{LAR} : \mathbf{M} \rightarrow \mathbf{R}; \quad \mathbb{C}\text{h}(\Lambda(X)) \mapsto (\text{CSR}(M_p))_{p=1}^{d}$$

Definition (Math models)

M is the set of chain complexes $\mathbb{C}$h supported by $\Lambda(X)$.

Definition (Computer representations)

R is the set of d-tuples of compressed sparse row (CSR) characteristic matrices.

Complex

Basic definitions

A complex C is a sequence

$$\cdots \longrightarrow C_{d+1} \xrightarrow{\partial_{d+1}} C_d \xrightarrow{\partial_d} C_{d-1} \longrightarrow \cdots$$

of linear spaces (Abelian groups) C_d and linear maps (homomorphisms) ∂_d, called boundary operators, such that

$$\boxed{\partial_{d+1} \circ \partial_d = 0}$$

for all d

Chain space

Linear space (over $\mathbb{Z}_2$) of cell **subsets**

Definition (cell complex)

Equivalence class of meshes with same topology
(same sets of cells of each dimension and same incidence relations)

Cells can be transformed into Chains, by attaching a value to each cell.
Values taken from a field of scalars, in order to add and scale chains

Definition (chain space)

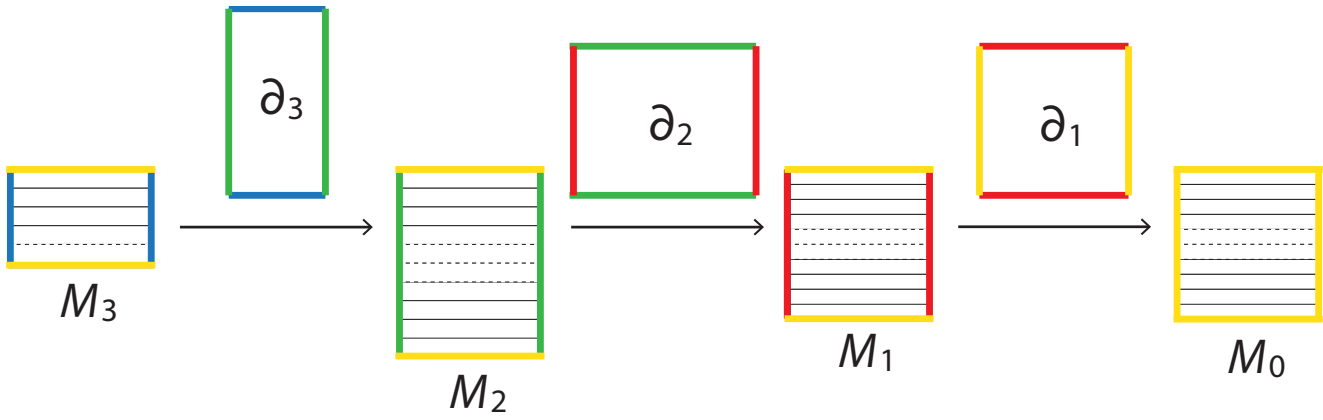
Vector space spanned by unit chains
(chains attaching 1 to a single cell and 0 to the others)

Chain complex

Basic definitions

A chain complex C is a complex of chain spaces and boundary maps:

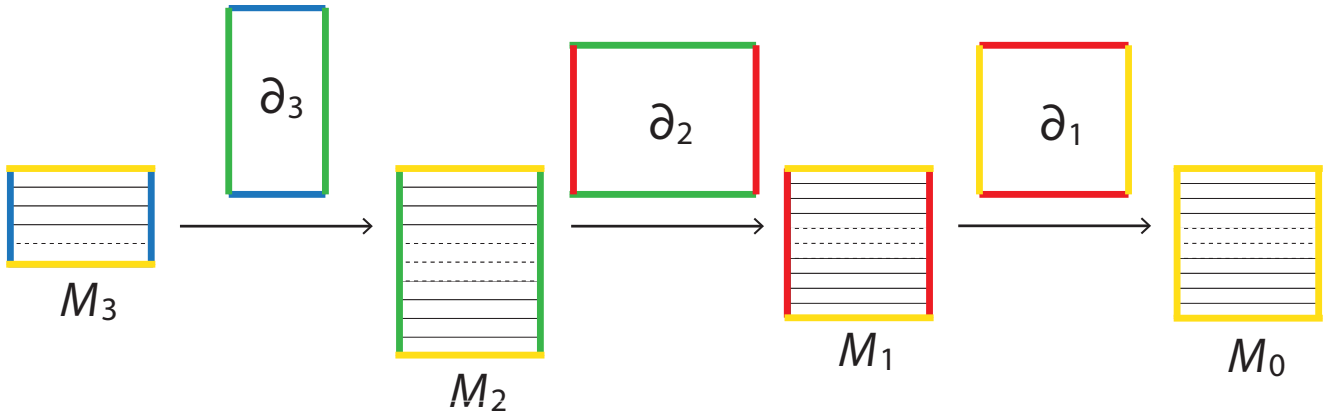
$$\cdots \longrightarrow C_{d+1} \xrightarrow{\partial_{d+1}} C_d \xrightarrow{\partial_d} C_{d-1} \longrightarrow \cdots$$

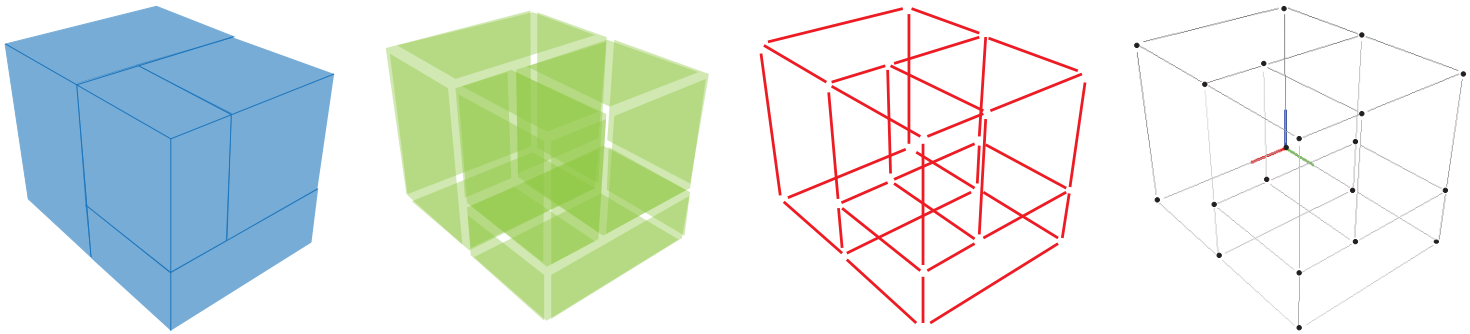


Chain complex (of chain spaces)

Sequence of linear spaces (over $\mathbb{Z}_2$) of d-cell **subsets**

Unit d-chains (single d-cell subsets), are the standard bases (M_d rows) of d-chain spaces

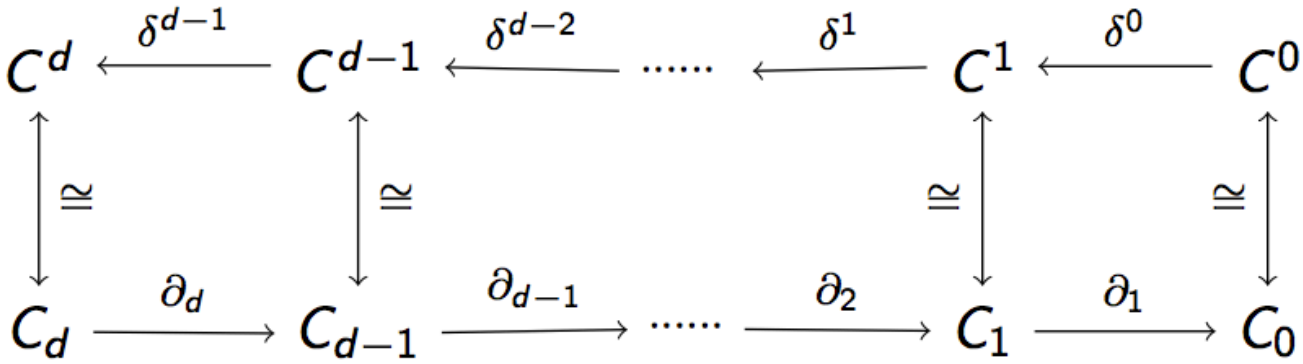




Chain and **cochain** complex

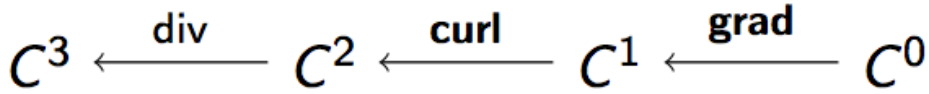
$C^0, C^1, C^2, C^3 \equiv \mathcal{V}, \mathcal{E}, \mathcal{F}, \mathcal{P}$ $\quad \Delta_p \equiv$ *Laplacian* $\quad \delta^0, \delta^1, \delta^2 \equiv$ **grad**, **curl**, div

cochains (all maps, discrete fields) and coboundary maps (δ^d operators)



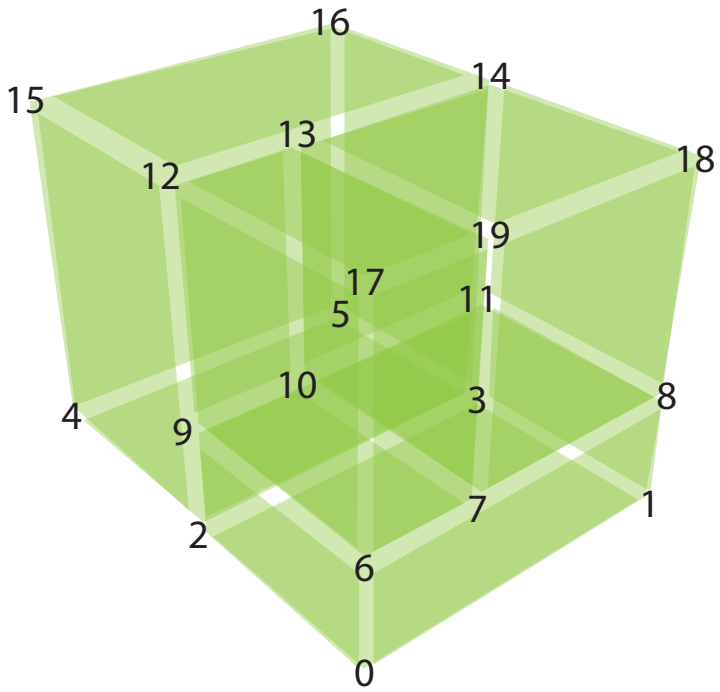
chains (linear spaces of model subsets) and boundary maps (∂_d operators)

$$\boxed{\delta^p = \partial_{p+1}^\top} \qquad \boxed{\Delta_p = (\delta^p)^\top \delta^p + (\delta^{p-1})^\top \delta^{p-1}}$$



Characteristic matrix of d-chain spaces

Matrix representation of the **standard basis** (the d-cells as **subsets** of vertices)

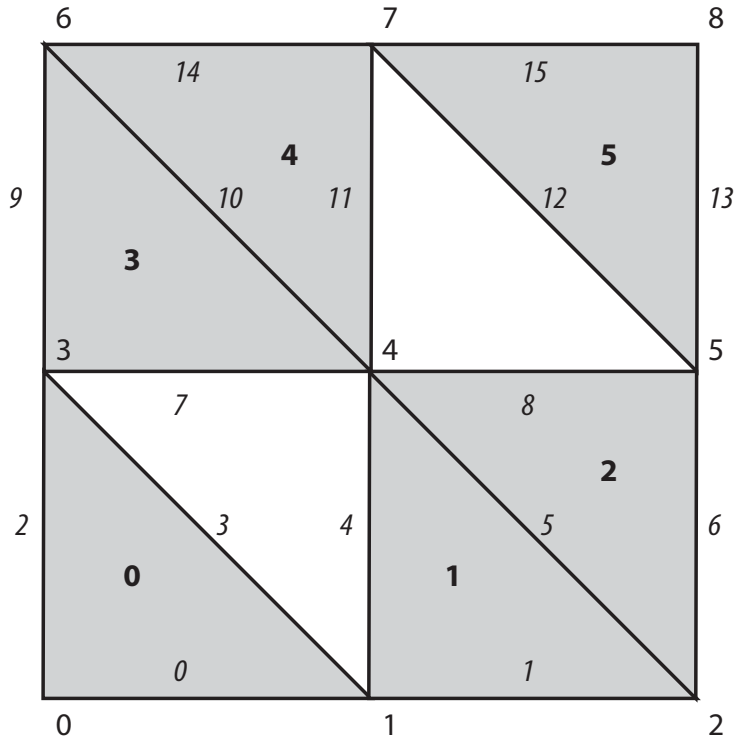


$$M_3 = \begin{pmatrix}
0&0&1&1&1&1&0&0&0&1&1&1&1&1&1&1&0&0&0\\
1&1&1&1&0&0&1&1&1&1&1&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&1&0&1&1&0&0&0&1&0&1\\
0&0&0&0&0&0&0&1&1&0&1&1&0&1&1&0&0&0&1&1
\end{pmatrix}$$

$$M_2 = \begin{pmatrix}
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&0&0&0&0&1&1&1&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&1&0&0&0&1&0&0&1&0&0&1&0&0&0&0&0&0\\
0&0&0&1&0&1&0&0&0&1&0&0&1&0&1&0&0&0&0&0&0\\
0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&1&0&1&0&0\\
0&0&0&0&0&0&1&0&0&1&0&0&1&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&1&1&0&0\\
0&0&0&0&0&0&0&1&0&0&1&0&0&1&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&1&0&0&1&0&0&1&0&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&1&1&0&1&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&1&0&1&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&1&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&1
\end{pmatrix}$$

Example: Characteristic matrix

0-cells associated to columns; d-cells associated to rows



$$M_d : C_0 \rightarrow C_d$$

$$M_2 = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix}$$

Remark (Reduced LAR)

when d-cells are convex, M_d is the only input

Remark (CSR binary matrix)

Characteristic matrices are binary matrices

Remark (Compressed LAR)

when the number of 1's per row are is constant

Compressed Sparse Row (CSR) matrix storage

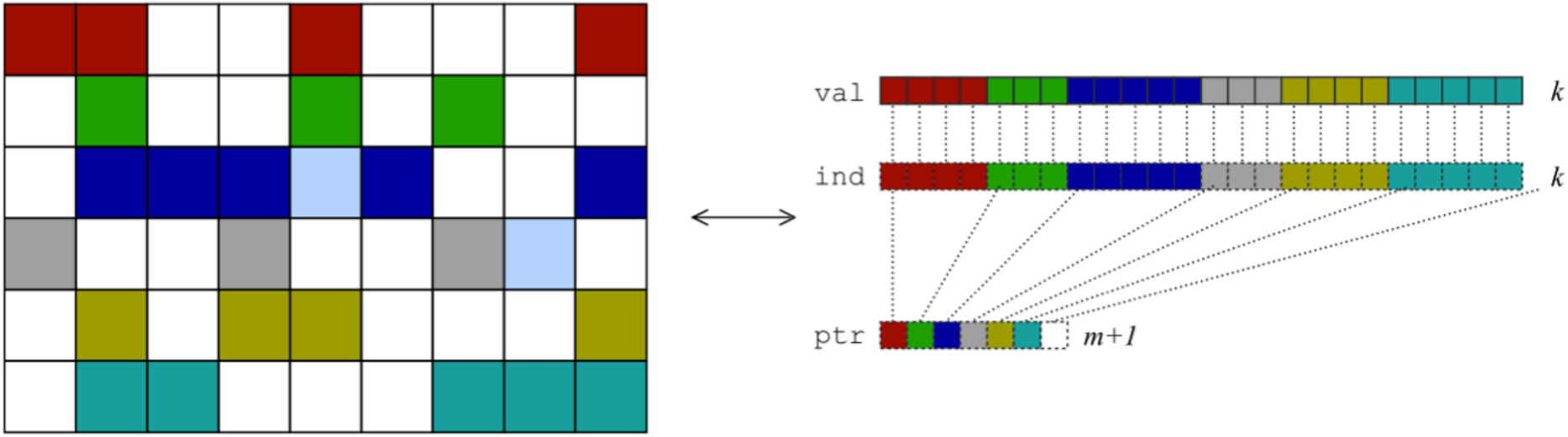
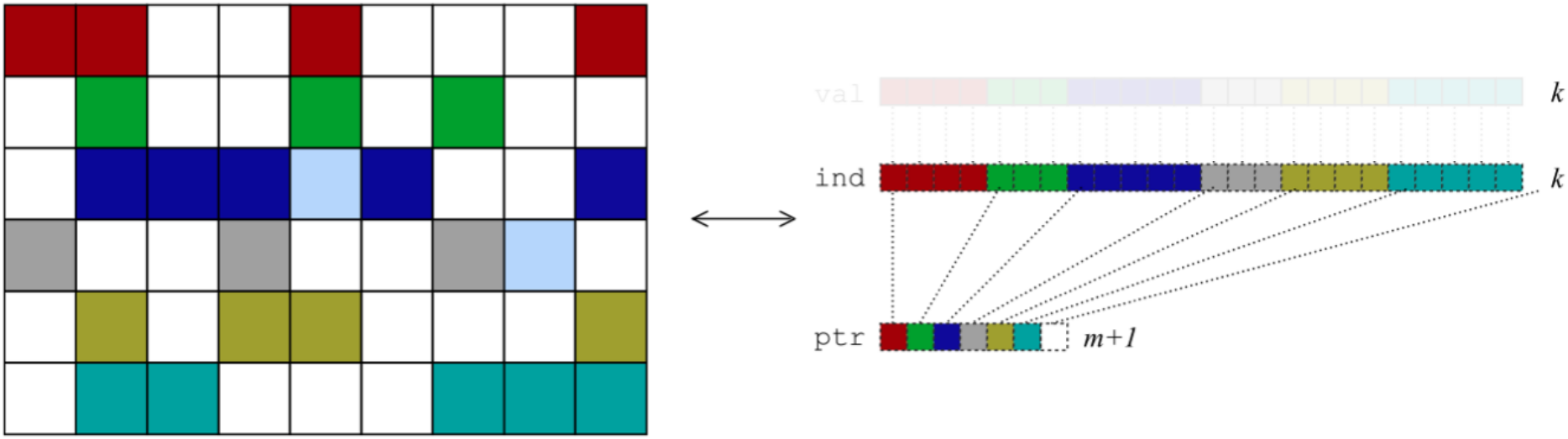


image from

Samuel Williams, Leonid Oliker, Richard Vuduc, John Shalf, Katherine Yelick, and James Demmel, Optimization of sparse matrix-vector multiplication on emerging multicore platforms, Proceedings of the 2007 ACM/IEEE conference on Supercomputing (New York, NY, USA), SC '07, ACM, 2007, pp. 38:1–38:12.

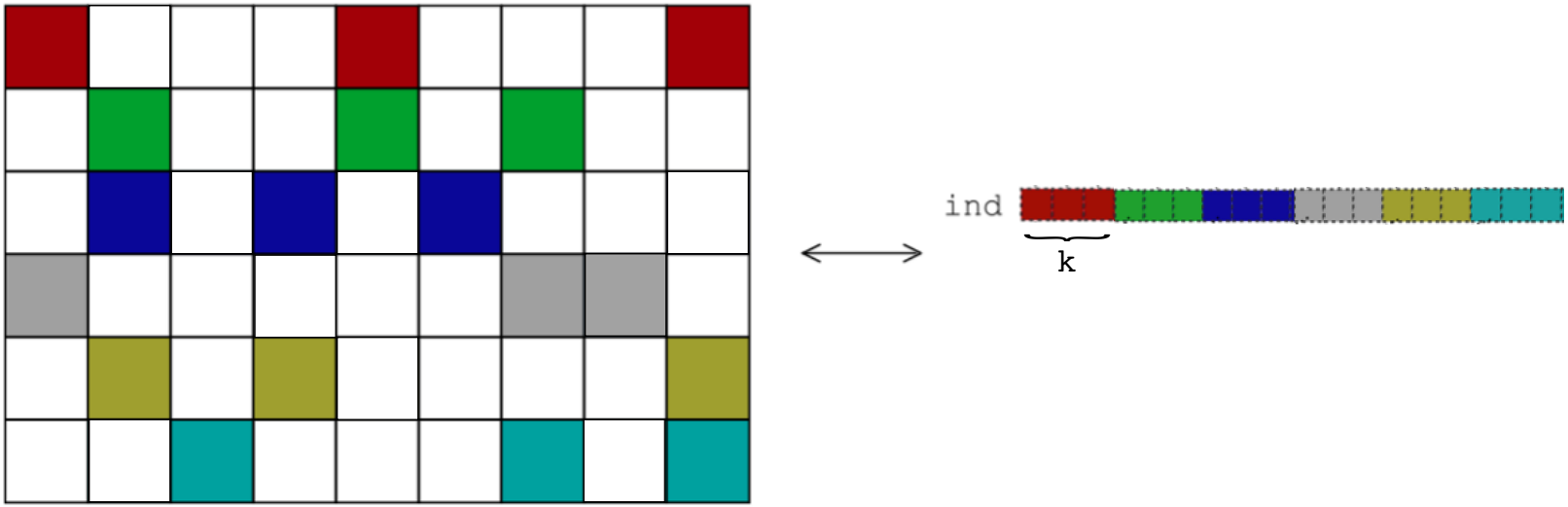
CSR storage of a BINARY matrix



of course, non-zero values (all ones) do not require storage

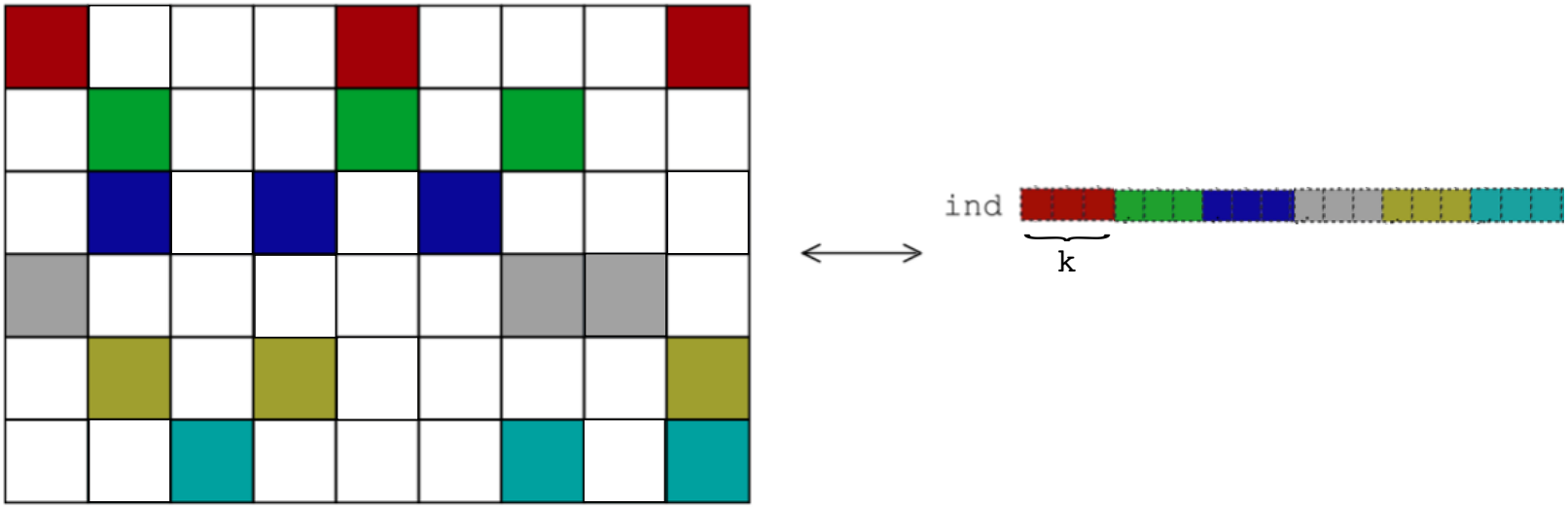
CSR storage of a binary matrix M_d such that

$$M_d\,\mathbf{1} = \mathbf{k}$$



CSR storage of a binary matrix M_d such that

$$M_d \, \mathbf{1} = \mathbf{k}$$

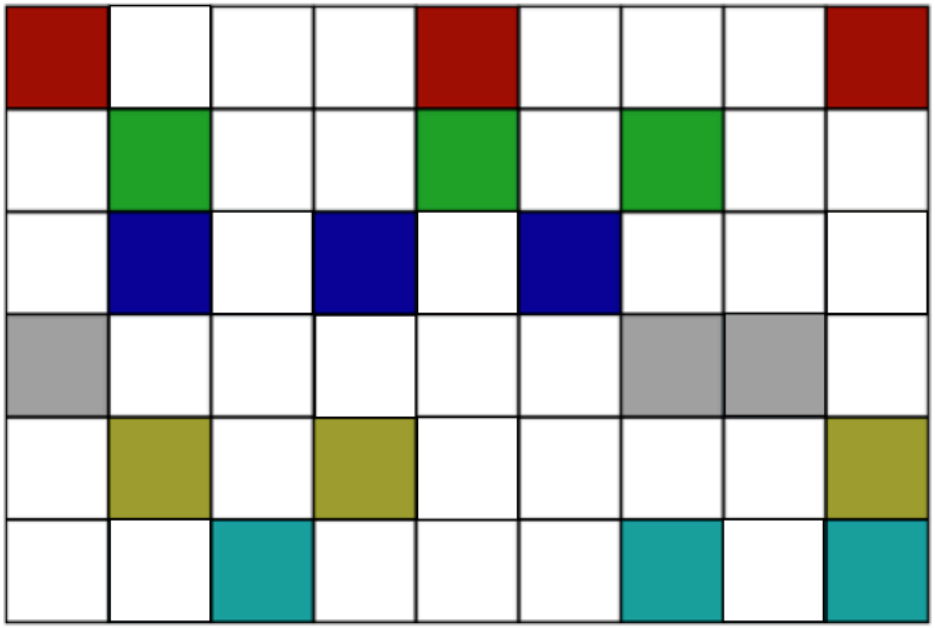


important cases:

- regular simplicial d-complexes: $k = d + 1$
- regular cuboidal d-complexes: $k = 2^d$

Storage of LAR(X) $\equiv$ CSR(M_d) matrix

for triangulated B-reps:
(very common case)

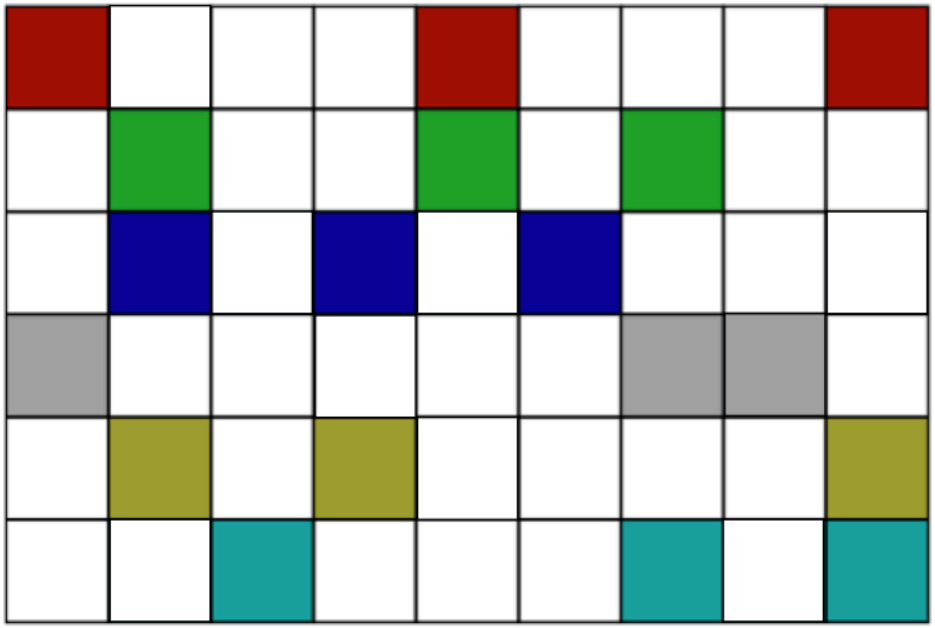


Remark (Storage occupancy)

$$|FV| = 2|E| \ !!!$$

Storage of LAR(X) $\equiv$ CSR(M_d) matrix

for triangulated B-reps:
(very common case)



Remark (Storage occupancy)

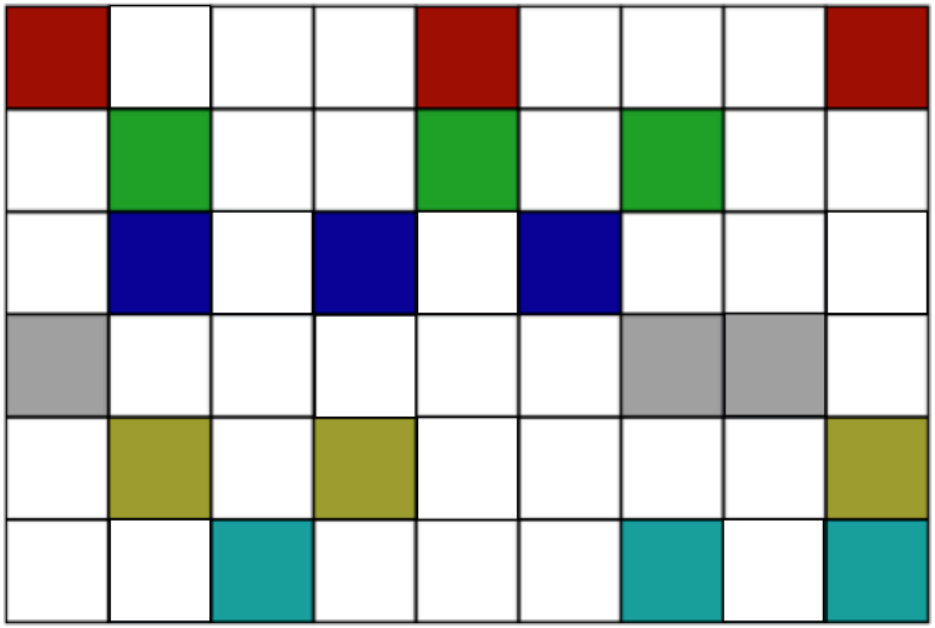
$$|FV| = 2|E| \ !!!$$

Remark (All topology representation)

$$|VE| + |VF| = 4|E|$$

Storage of LAR(X) $\equiv$ CSR(M_d) matrix

for triangulated B-reps:
(very common case)



Remark (Storage occupancy)

$$|FV| = 2|E| \ !!!$$

Remark (All topology representation)

$$|VE| + |VF| = 4|E|$$

Remark (Any topological queries)

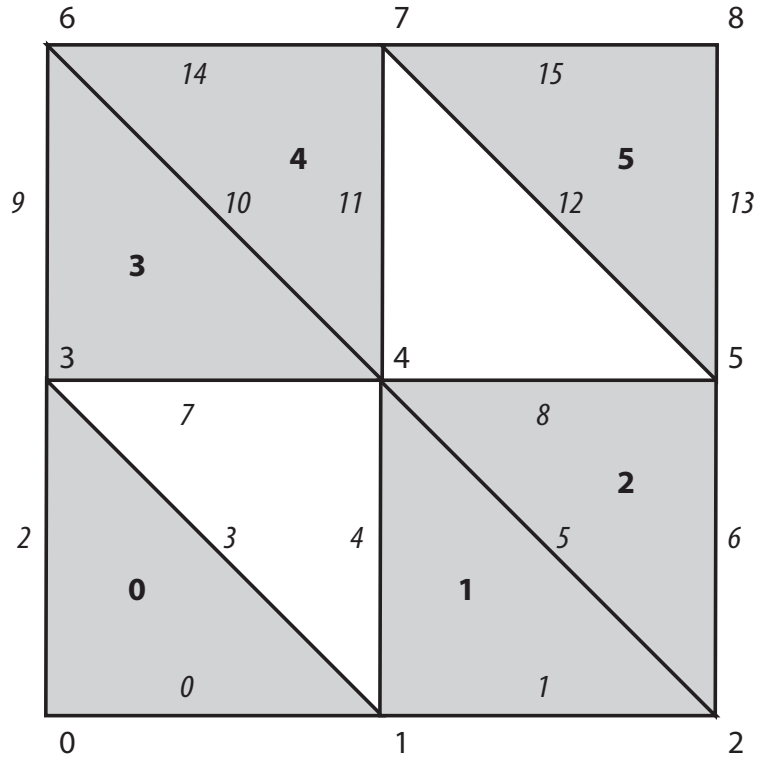
single SpMV multiplication

Example: input of simplicial complex

Reduced and compressed LAR

$\mathrm{FV} := \mathrm{CSR}(M_2)$

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

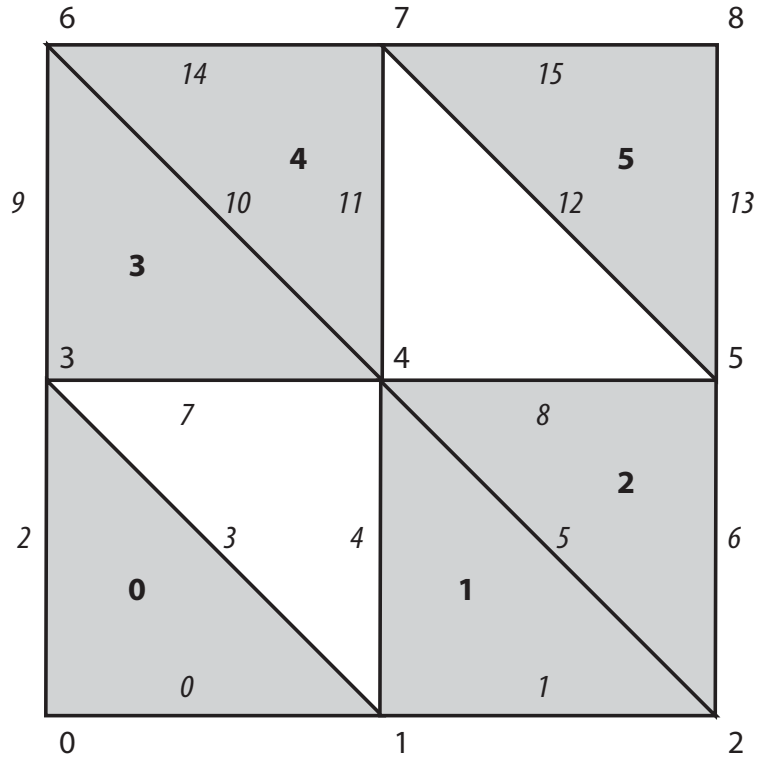


Example: input of simplicial complex

Reduced and compressed LAR

$\mathrm{FV} := \mathrm{CSR}(M_2)$

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



Remark

The bulk of common graphics formats !! *(e.g.: .OBJ or .PLY)*

2D simplicial complex (non manifold pointset)

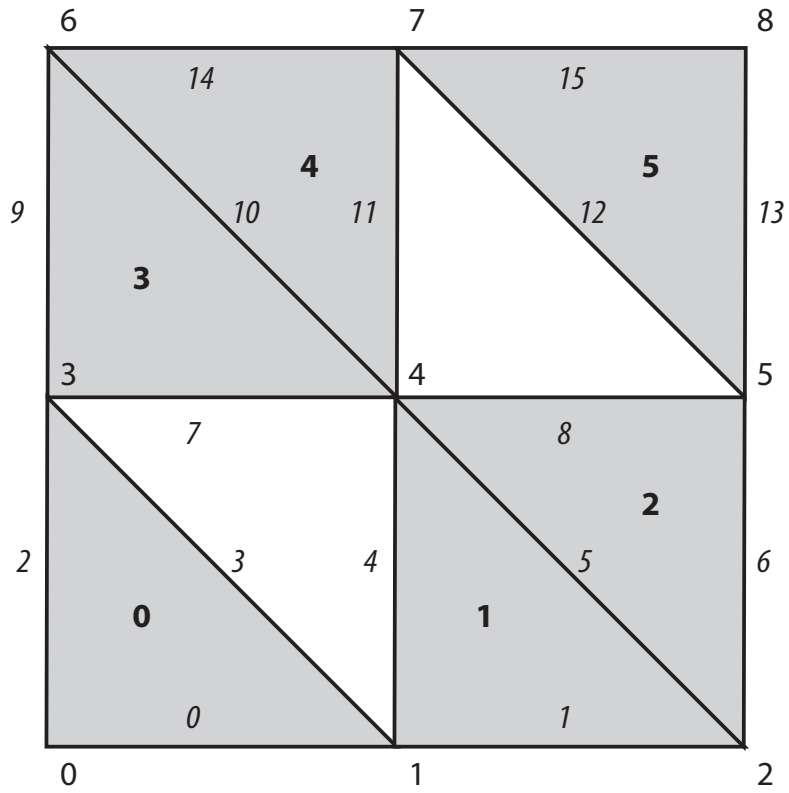
$K = K_0 \cup K_1 \cup K_2 \qquad \#K_0 =: k_0 = 9; \quad \#K_1 =: k_1 = 16; \quad \#K_2 =: k_2 = 6$

2-cells by 0-cells

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

1-cells by 0-cells

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



2D Example

Characteristic matrices

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

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

$$[\mathcal{EV}] = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

$$[\mathcal{FV}] = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

Sparse row-compressed matrices (CSR)[1]

operators $\mathcal{EV} : C_0 \to C_1$ and $\mathcal{FV} : C_0 \to C_2$ between spaces of p-chains $(0 \leq p \leq 2)$

coordinate representation w.r.t. the standard p-chain basis (the single p-cells)

$$[\mathcal{EV}] = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

$$[\mathcal{FV}] = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

sparse matrix representation

[[0,0,1],[0,1,1],[1,1,1],
[1,2,1],[2,0,1],[2,3,1],[3,1,1],
[3,3,1],[4,1,1],[4,4,1],[5,2,1],
[5,4,1],[6,2,1],[6,5,1],[7,3,1],
[7,4,1],[8,4,1],[8,5,1],[9,3,1],
[9,6,1],[10,4,1],[10,6,1],[11,4,1],
[11,7,1],[12,5,1],[12,7,1],[13,5,1],
[13,8,1],[14,6,1],[14,7,1],[15,7,1],
[15,8,1]]

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

[1] The actual CSR representation is different.

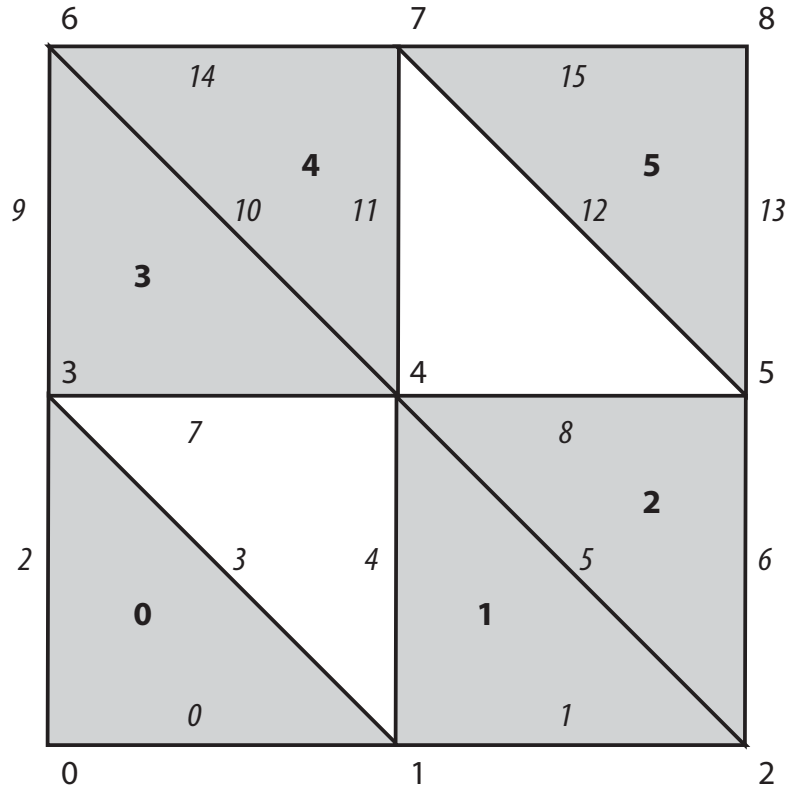
Incidence on vertices

$\mathcal{VV} : C_0 \to C_0$; $\mathcal{EV} : C_0 \to C_1$; $\mathcal{FV} : C_0 \to C_2$

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

```
EV =
[[0, 2],
 [0, 1, 3, 4],
 [1, 5, 6],
 [2, 3, 7, 9],
 [4, 5, 7, 8, 10, 11],
 [6, 8, 12, 13],
 [9, 10, 14],
 [11, 12, 14, 15],
 [13, 15]]
```

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



Computation examples:

$$[c_0^k] = [0, \ldots, 1, \ldots, 0]^\top \quad \text{(0-chain basis element)}$$

$\mathcal{VV}(c_0^k) \equiv [\mathcal{VV}][c_0^k]$; $\mathcal{EV}(c_0^k) \equiv [\mathcal{EV}][c_0^k]$; $\mathcal{FV}(c_0^k) \equiv [\mathcal{FV}][c_0^k]$)

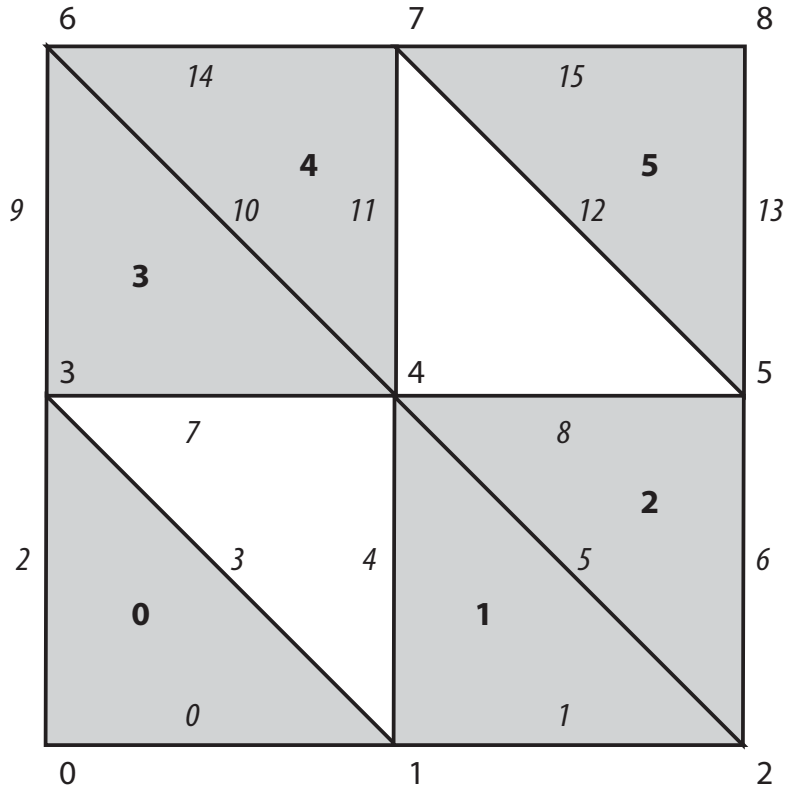
Incidence on edges

$\mathcal{VE} : C_1 \to C_0$; $\mathcal{EE} : C_1 \to C_1$; $\mathcal{FE} : C_1 \to C_2$

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

```
EE =
[[0, 1, 2, 3, 4],
 [0, 1, 3, 4, 5, 6],
 [0, 2, 3, 7, 9],
 [0, 1, 2, 3, 4, 7, 9],
 [0, 1, 3, 4, 5, 7, 8, 10, 11],
 [1, 4, 5, 6, 7, 8, 10, 11],
 [1, 5, 6, 8, 12, 13],
 [2, 3, 4, 5, 7, 8, 9, 10, 11],
 [4, 5, 6, 7, 8, 10, 11, 12, 13],
 [2, 3, 7, 9, 10, 14],
 [4, 5, 7, 8, 9, 10, 11, 14],
 [4, 5, 7, 8, 10, 11, 12, 14, 15],
 [6, 8, 11, 12, 13, 14, 15],
 [6, 8, 12, 13, 15],
 [9, 10, 11, 12, 14, 15],
 [11, 12, 13, 14, 15]]
```

```
FE =
[[0, 1],
 [0, 1, 2],
 [0, 3],
 [0, 1, 3],
 [0, 1, 2, 3, 4],
 [1, 2, 3, 4],
 [1, 2, 5],
 [0, 1, 2, 3, 4],
 [1, 2, 3, 4, 5],
 [0, 3, 4],
 [1, 2, 3, 4],
 [1, 2, 3, 4, 5],
 [2, 4, 5],
 [2, 5],
 [3, 4, 5],
 [4, 5]]
```



Computation examples:

$$[c_1^k] = [0, \ldots, 1, \ldots, 0]^\top \quad \text{(1-chain basis element)}$$

$\mathcal{VE}(c_1^k) = [\mathcal{VE}][c_1^k]$; $\mathcal{EE}(c_1^k) = [\mathcal{EE}][c_1^k]$; $\mathcal{FE}(c_1^k) = [\mathcal{FE}][c_1^k]$

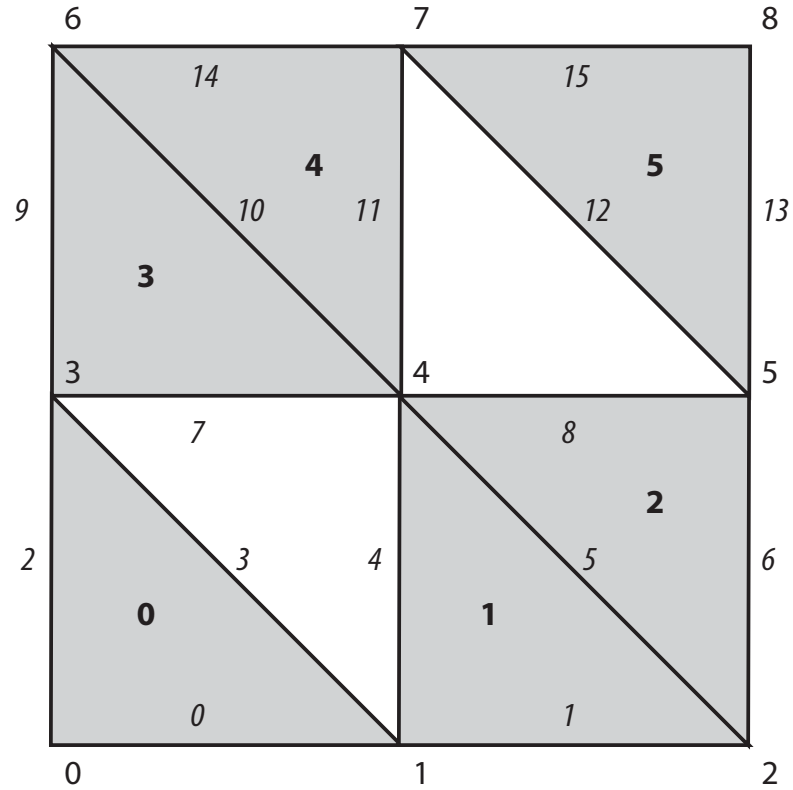
Incidence on faces

$\mathcal{VF} : C_2 \to C_0$; $\mathcal{EF} : C_2 \to C_1$; $\mathcal{FF} : C_2 \to C_2$

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

```
EF =
[[0, 1, 2, 3, 4, 7, 9],
 [0, 1, 3, 4, 5, 6, 7, 8, 10, 11],
 [1, 4, 5, 6, 7, 8, 10, 11, 12, 13],
 [2, 3, 4, 5, 7, 8, 9, 10, 11, 14],
 [4, 5, 7, 8, 9, 10, 11, 12, 14, 15],
 [6, 8, 11, 12, 13, 14, 15]]
```

```
FF =
[[0, 1, 3],
 [0, 1, 2, 3, 4],
 [1, 2, 3, 4, 5],
 [0, 1, 2, 3, 4],
 [1, 2, 3, 4, 5],
 [2, 4, 5]]
```



Computation examples:

$$[c_2^k] = [0, \ldots, 1, \ldots, 0]^\top \qquad \text{(2-chain basis element)}$$

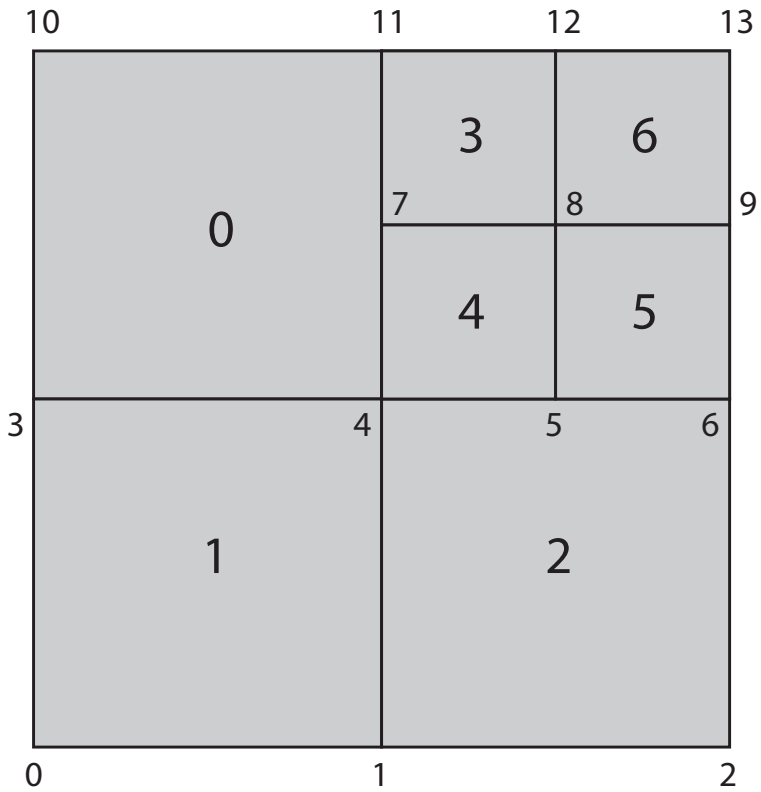
$\mathcal{VF}(c_2^k) = [\mathcal{VF}][c_2^k]$; $\mathcal{EF}(c_2^k) = [\mathcal{EF}][c_2^k]$; $\mathcal{FF}(c_2^k) = [\mathcal{FF}][c_2^k]$

Example: input of Tmesh

Reduced (non compressed) LAR: of course, it is the same in 3D

$FV := \mathrm{CSR}(M_2)$

```
FV = [[3,4,7,10,11],
      [0,1,3,4],
      [1,2,4,5,6],
      [7,8,11,12],
      [4,5,7,8],
      [5,6,8,9],
      [8,9,12,13]]
```

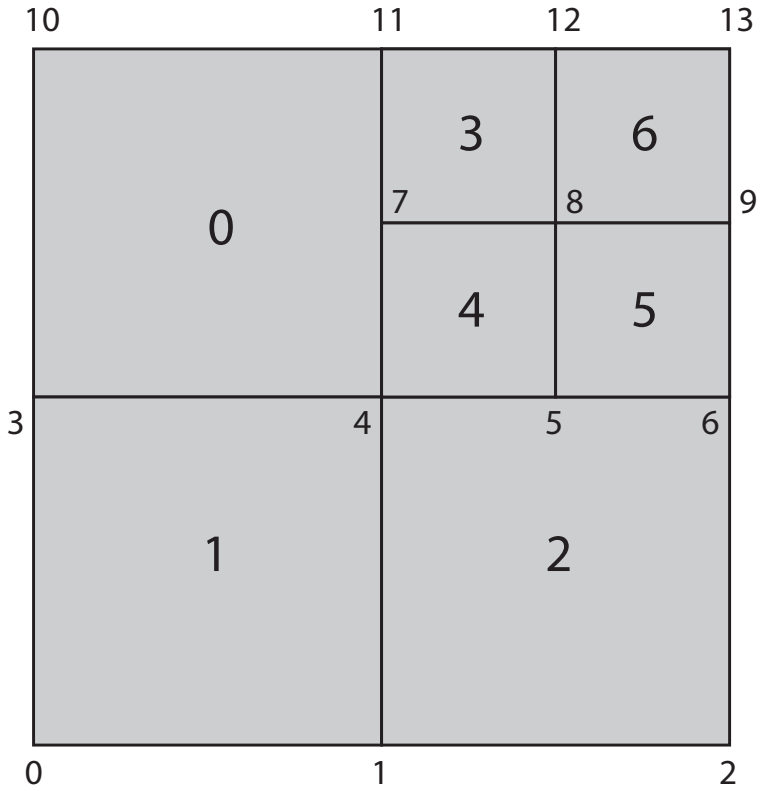


Example: input of Tmesh

Reduced (non compressed) LAR: of course, it is the same in 3D

$\mathrm{FV} := \mathrm{CSR}(M_2)$

```
FV = [[3,4,7,10,11],
      [0,1,3,4],
      [1,2,4,5,6],
      [7,8,11,12],
      [4,5,7,8],
      [5,6,8,9],
      [8,9,12,13]]
```



Remark

The bulk of common graphics formats !! (e.g.: `IndexedFaceSets`)

Algorithm: computation of $\partial_p = \delta_{p-1}^\top$

Knowledege of $[\partial_p]$ gives the boundary of every p-chain by a single SpMV product

1. compute the numbers of vertices of $(d-1)$-cells $\mu_{p-1}^i \in \Lambda_{p-1}$:

$$k_i := \sharp \mu_{p-1}^i \qquad (\mathbf{k} = M_{p-1}\mathbf{1})$$

2. Compute

$$M_{p-1}^p := M_{p-1}\, M_p^t$$

3. For each $1 \leq i \leq |\Lambda_{p-1}|$, and for each $1 \leq j \leq |\Lambda_p|$:

$$\text{if} \quad M_{p-1}^p(i,j) = k_i \quad \begin{cases} \text{then} & [\partial_p](i,j) := 1 \\ \text{else} & [\partial_p](i,j) := 0 \end{cases}$$

Boundary operators $\partial_n : C_n \rightarrow C_{n-1}$

Linear operators to compute the $(n-1)$-chains that are boundary of a n-chain

$$[\partial_n](i,j) = 1$$

$$\text{if} \quad [\mathcal{C}_{n-1}\mathcal{C}_n](i,j) = \max_j [\mathcal{C}_{n-1}\mathcal{C}_n](i,j)$$

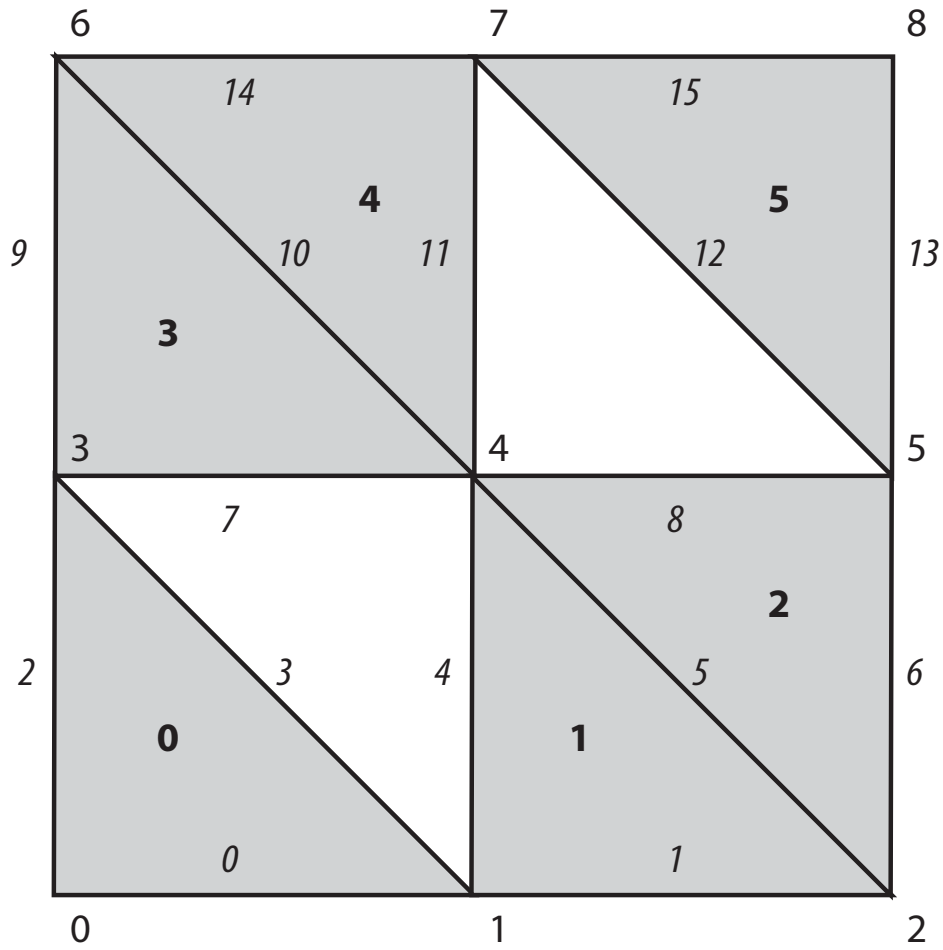
example of $[\partial_2]$

$[\mathcal{EF}] =$

```
[[2 1 0 0 0 0]
 [1 2 1 0 0 0]
 [2 0 0 1 0 0]
 [2 1 0 1 0 0]
 [1 2 1 1 1 0]
 [0 2 2 1 1 0]
 [0 1 2 0 0 1]
 [1 1 1 2 1 0]
 [0 1 2 1 1 1]
 [1 0 0 2 1 0]
 [0 1 1 2 2 0]
 [0 1 1 1 2 1]
 [0 0 1 0 1 2]
 [0 0 1 0 0 2]
 [0 0 0 1 2 1]
 [0 0 0 0 1 2]]
```

$[\partial_2] =$

```
[[1 0 0 0 0 0]
 [0 1 0 0 0 0]
 [1 0 0 0 0 0]
 [1 0 0 0 0 0]
 [0 1 0 0 0 0]
 [0 1 1 0 0 0]
 [0 0 1 0 0 0]
 [0 0 0 1 0 0]
 [0 0 1 0 0 0]
 [0 0 0 1 0 0]
 [0 0 0 1 1 0]
 [0 0 0 0 1 0]
 [0 0 0 0 0 1]
 [0 0 0 0 0 1]
 [0 0 0 0 1 0]
 [0 0 0 0 0 1]]
```



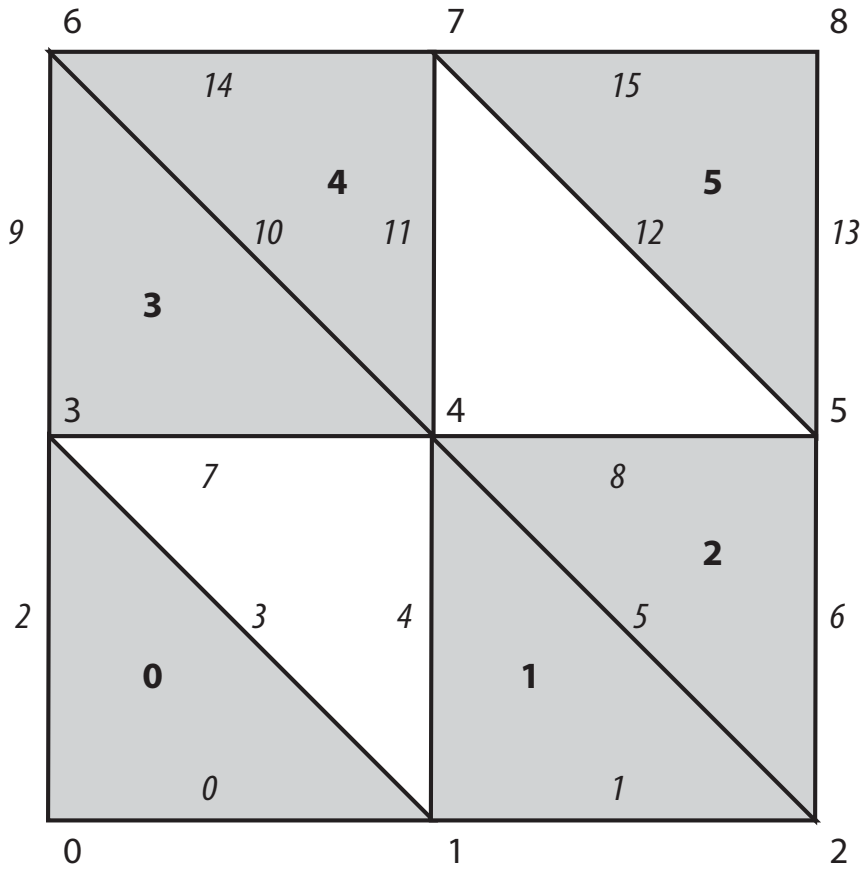
Computation of 1-boundaries

Some examples: take 2-chains $c_2, d_2, e_2 \in C_2$

$$
\begin{aligned}
[c_2] &= [1,1,1,1,1,1]^\top \in C_2 \\
[c_1] &= \mathbb{Z}_2([\partial_2][c_2]) \in C_1 \\
&= [1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1]^\top
\end{aligned}
$$

$$
\begin{aligned}
[d_2] &= [1,0,0,0,0,0]^\top \in C_2 \\
[d_1] &= \mathbb{Z}_2([\partial_2][d_2]) \in C_1 \\
&= [1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0]^\top
\end{aligned}
$$

$$
\begin{aligned}
[e_2] &= [0,0,0,1,1,1]^\top \in C_2 \\
[e_1] &= \mathbb{Z}_2([\partial_2][e_2]) \in C_1 \\
&= [0,0,0,0,0,0,0,1,0,1,0,1,1,1,1,1]^\top
\end{aligned}
$$



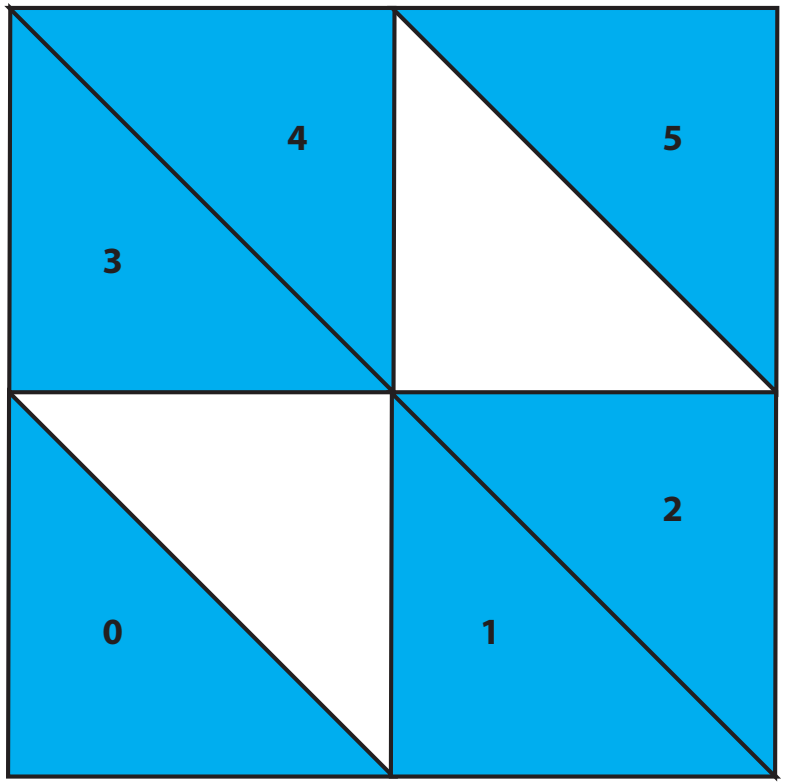
Computation of 1-boundaries

Some examples: take 2-chains $c_2, d_2, e_2 \in C_2$

$$
\begin{aligned}
[c_2] &= [1,1,1,1,1,1]^\top \in C_2 \\
[c_1] &= \mathbb{Z}_2([\partial_2][c_2]) \in C_1 \\
&= [1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1]^\top
\end{aligned}
$$

$$
\begin{aligned}
[d_2] &= [1,0,0,0,0,0]^\top \in C_2 \\
[d_1] &= \mathbb{Z}_2([\partial_2][d_2]) \in C_1 \\
&= [1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0]^\top
\end{aligned}
$$

$$
\begin{aligned}
[e_2] &= [0,0,0,1,1,1]^\top \in C_2 \\
[e_1] &= \mathbb{Z}_2([\partial_2][e_2]) \in C_1 \\
&= [0,0,0,0,0,0,0,1,0,1,0,1,1,1,1,1]^\top
\end{aligned}
$$



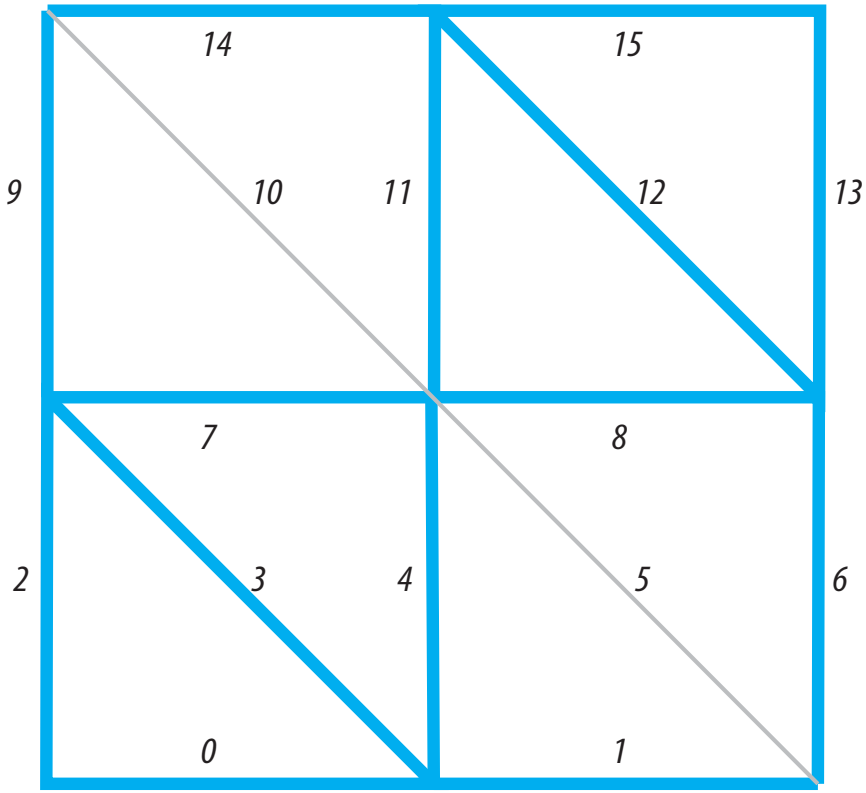
Computation of 1-boundaries

Some examples: take 2-chains $c_2, d_2, e_2 \in C_2$

$$
\begin{aligned}
[c_2] &= [1,1,1,1,1,1]^\top \in C_2 \\
[c_1] &= \mathbb{Z}_2([\partial_2][c_2]) \in C_1 \\
&= [1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1]^\top
\end{aligned}
$$

$$
\begin{aligned}
[d_2] &= [1,0,0,0,0,0]^\top \in C_2 \\
[d_1] &= \mathbb{Z}_2([\partial_2][d_2]) \in C_1 \\
&= [1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0]^\top
\end{aligned}
$$

$$
\begin{aligned}
[e_2] &= [0,0,0,1,1,1]^\top \in C_2 \\
[e_1] &= \mathbb{Z}_2([\partial_2][e_2]) \in C_1 \\
&= [0,0,0,0,0,0,0,1,0,1,0,1,1,1,1,1]^\top
\end{aligned}
$$



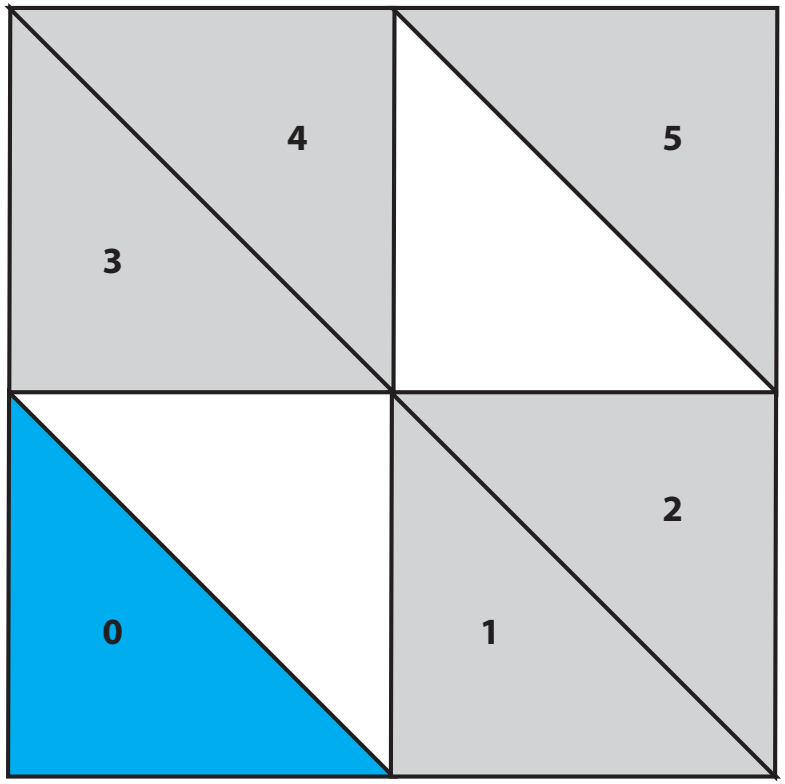
Computation of 1-boundaries

Some examples: take 2-chains $c_2, d_2, e_2 \in C_2$

$$
\begin{aligned}
[c_2] &= [1,1,1,1,1,1]^\top \in C_2 \\
[c_1] &= \mathbb{Z}_2([\partial_2][c_2]) \in C_1 \\
&= [1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1]^\top
\end{aligned}
$$

$$
\begin{aligned}
[d_2] &= [1,0,0,0,0,0]^\top \in C_2 \\
[d_1] &= \mathbb{Z}_2([\partial_2][d_2]) \in C_1 \\
&= [1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0]^\top
\end{aligned}
$$

$$
\begin{aligned}
[e_2] &= [0,0,0,1,1,1]^\top \in C_2 \\
[e_1] &= \mathbb{Z}_2([\partial_2][e_2]) \in C_1 \\
&= [0,0,0,0,0,0,0,1,0,1,0,1,1,1,1,1]^\top
\end{aligned}
$$



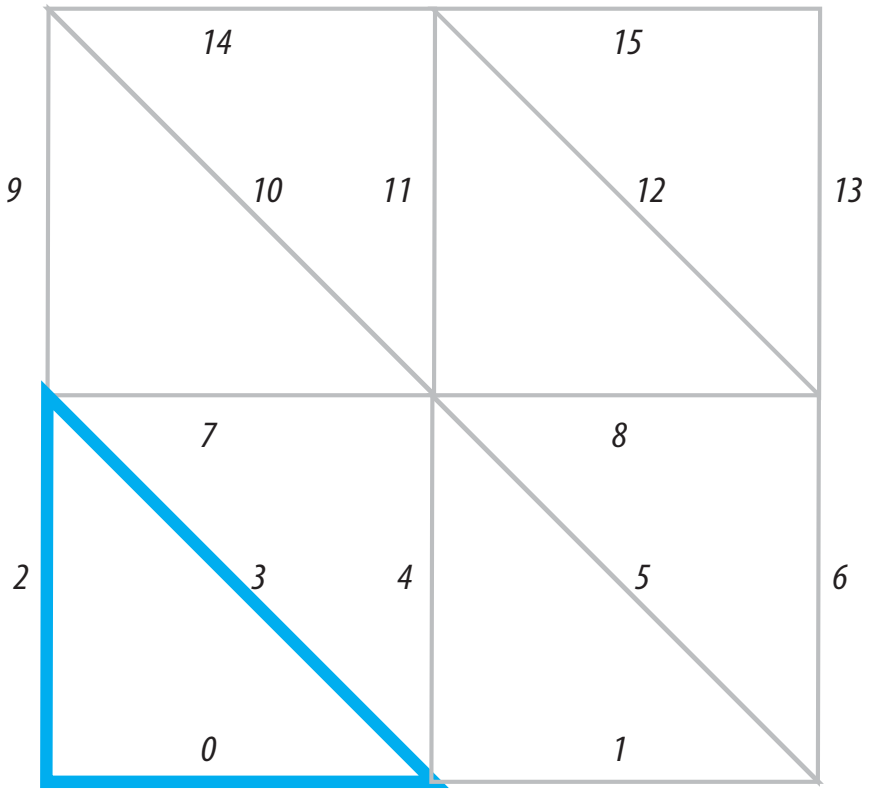
Computation of 1-boundaries

Some examples: take 2-chains $c_2, d_2, e_2 \in C_2$

$$
\begin{aligned}
[c_2] &= [1,1,1,1,1,1]^\top \in C_2 \\
[c_1] &= \mathbb{Z}_2([\partial_2][c_2]) \in C_1 \\
&= [1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1]^\top
\end{aligned}
$$

$$
\begin{aligned}
[d_2] &= [1,0,0,0,0,0]^\top \in C_2 \\
[d_1] &= \mathbb{Z}_2([\partial_2][d_2]) \in C_1 \\
&= [1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0]^\top
\end{aligned}
$$

$$
\begin{aligned}
[e_2] &= [0,0,0,1,1,1]^\top \in C_2 \\
[e_1] &= \mathbb{Z}_2([\partial_2][e_2]) \in C_1 \\
&= [0,0,0,0,0,0,0,1,0,1,0,1,1,1,1,1]^\top
\end{aligned}
$$



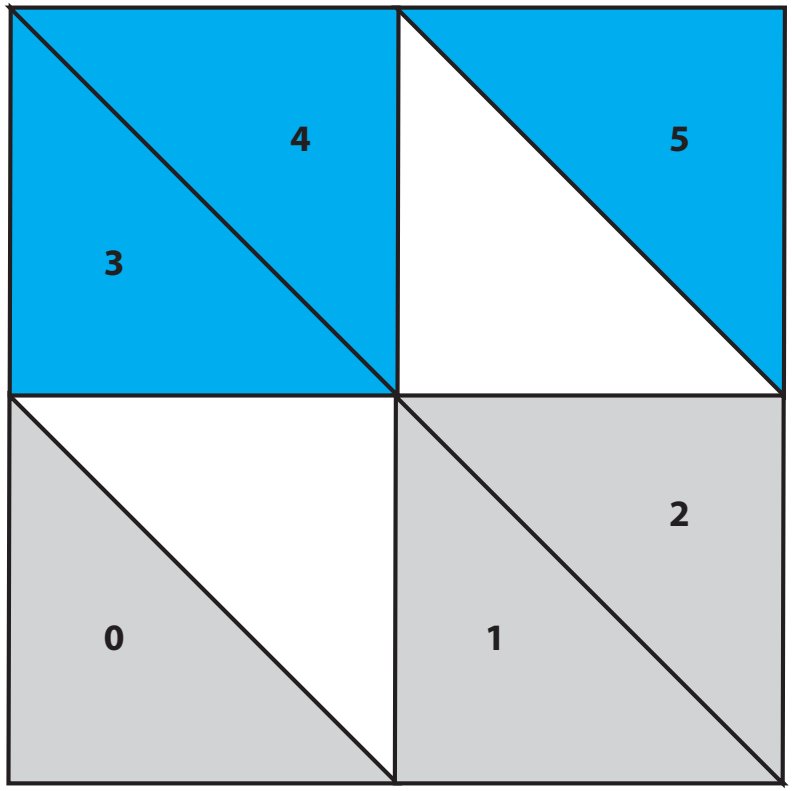
Computation of 1-boundaries

Some examples: take 2-chains $c_2, d_2, e_2 \in C_2$

$$
\begin{aligned}
[c_2] &= [1,1,1,1,1,1]^\top \in C_2 \\
[c_1] &= \mathbb{Z}_2([\partial_2][c_2]) \in C_1 \\
&= [1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1]^\top
\end{aligned}
$$

$$
\begin{aligned}
[d_2] &= [1,0,0,0,0,0]^\top \in C_2 \\
[d_1] &= \mathbb{Z}_2([\partial_2][d_2]) \in C_1 \\
&= [1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0]^\top
\end{aligned}
$$

$$
\begin{aligned}
[e_2] &= [0,0,0,1,1,1]^\top \in C_2 \\
[e_1] &= \mathbb{Z}_2([\partial_2][e_2]) \in C_1 \\
&= [0,0,0,0,0,0,0,1,0,1,0,1,1,1,1,1]^\top
\end{aligned}
$$



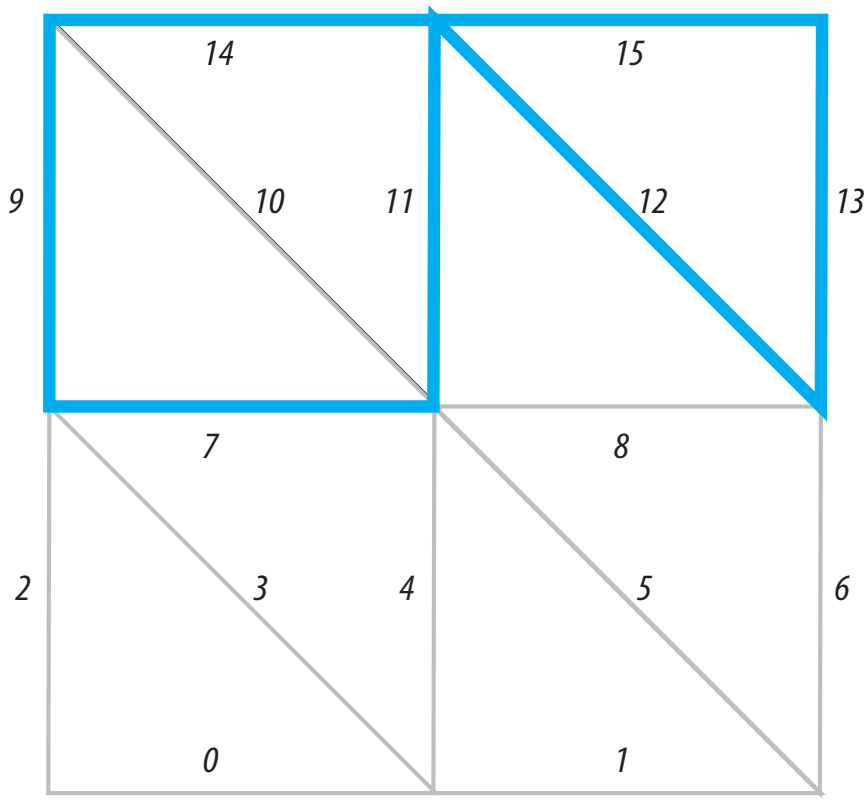
Computation of 1-boundaries

Some examples: take 2-chains $c_2, d_2, e_2 \in C_2$

$$
\begin{aligned}
[c_2] &= [1,1,1,1,1,1]^\top \in C_2 \\
[c_1] &= \mathbb{Z}_2([\partial_2][c_2]) \in C_1 \\
&= [1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1]^\top
\end{aligned}
$$

$$
\begin{aligned}
[d_2] &= [1,0,0,0,0,0]^\top \in C_2 \\
[d_1] &= \mathbb{Z}_2([\partial_2][d_2]) \in C_1 \\
&= [1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0]^\top
\end{aligned}
$$

$$
\begin{aligned}
[e_2] &= [0,0,0,1,1,1]^\top \in C_2 \\
[e_1] &= \mathbb{Z}_2([\partial_2][e_2]) \in C_1 \\
&= [0,0,0,0,0,0,0,1,0,1,0,1,1,1,1,1]^\top
\end{aligned}
$$

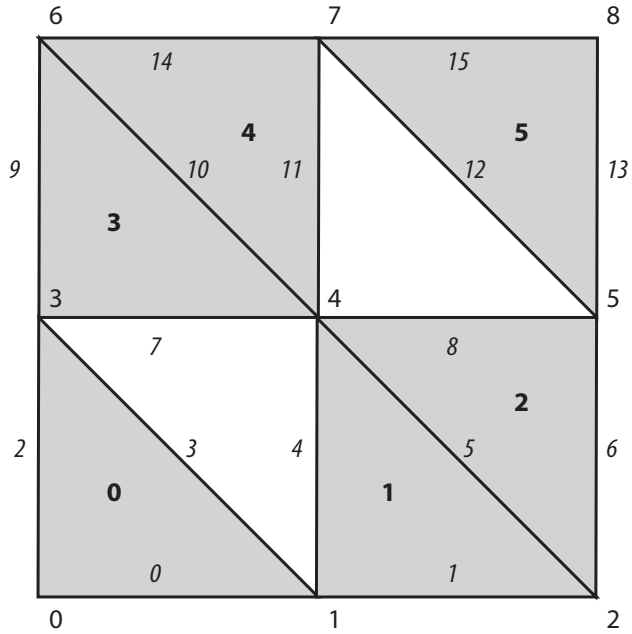


Boundary operators $\partial_n : C_n \rightarrow C_{n-1}$

Linear operators to compute the subset of $(n-1)$-cycles [2] that are boundary of a n-chain

$$[\partial_n](i,j) = 1$$

$$\text{if} \quad [\mathcal{C}_{n-1}\mathcal{C}_n](i,j) = \max_j [\mathcal{C}_{n-1}\mathcal{C}_n](i,j)$$



example of $[\partial_1]$

$$[\mathcal{V}\mathcal{E}] = \begin{bmatrix}
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&1&1&0&0&0&0&0&0&0&0&0\\
0&0&1&1&0&0&0&1&0&1&0&0&0&0&0&0\\
0&0&0&0&1&1&0&1&1&0&1&1&0&0&0&0\\
0&0&0&0&0&0&1&0&1&0&0&0&1&1&0&0\\
0&0&0&0&0&0&0&0&0&1&1&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1
\end{bmatrix}$$

$$[\partial_1] = \begin{bmatrix}
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&1&1&0&0&0&0&0&0&0&0&0\\
0&0&1&1&0&0&0&1&0&1&0&0&0&0&0&0\\
0&0&0&0&1&1&0&1&1&0&1&1&0&0&0&0\\
0&0&0&0&0&0&1&0&1&0&0&0&1&1&0&0\\
0&0&0&0&0&0&0&0&0&1&1&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1
\end{bmatrix}$$

[2] closed $(n-1)$-chains

$\partial\partial = 0$

Of course, the constraint equations of chain complexes are satisfied ...

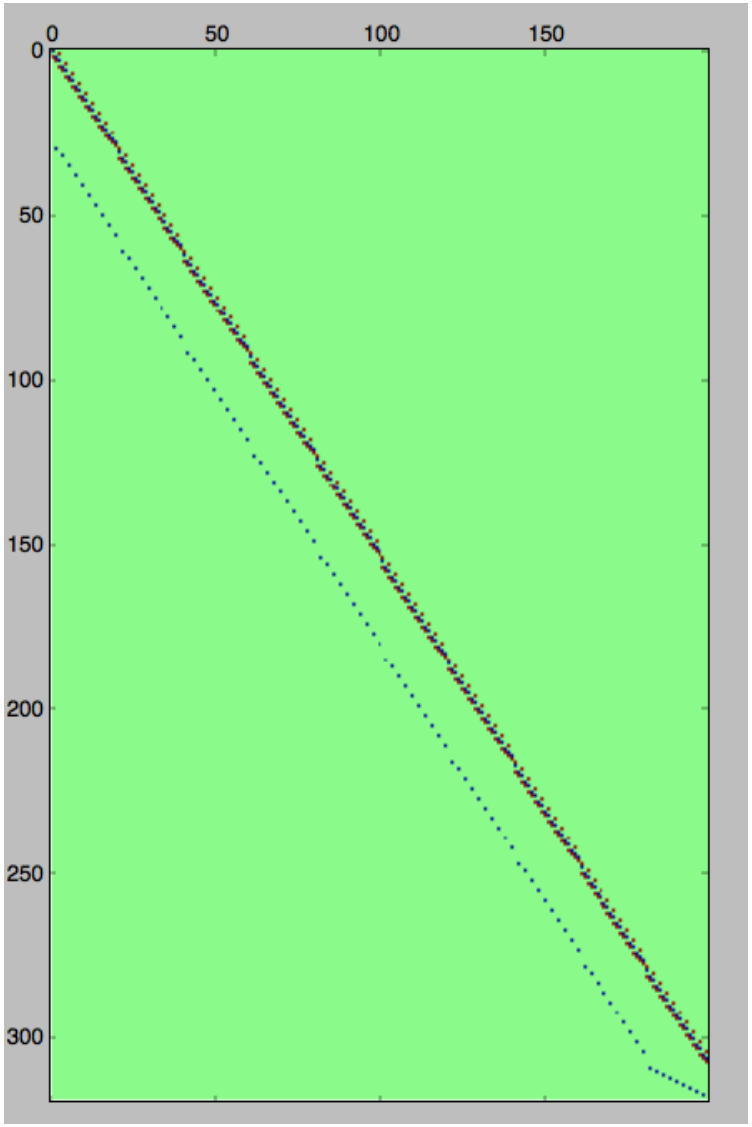
$$[\partial_1]\,[\partial_2]\,\begin{bmatrix} c & d & e \end{bmatrix} =$$

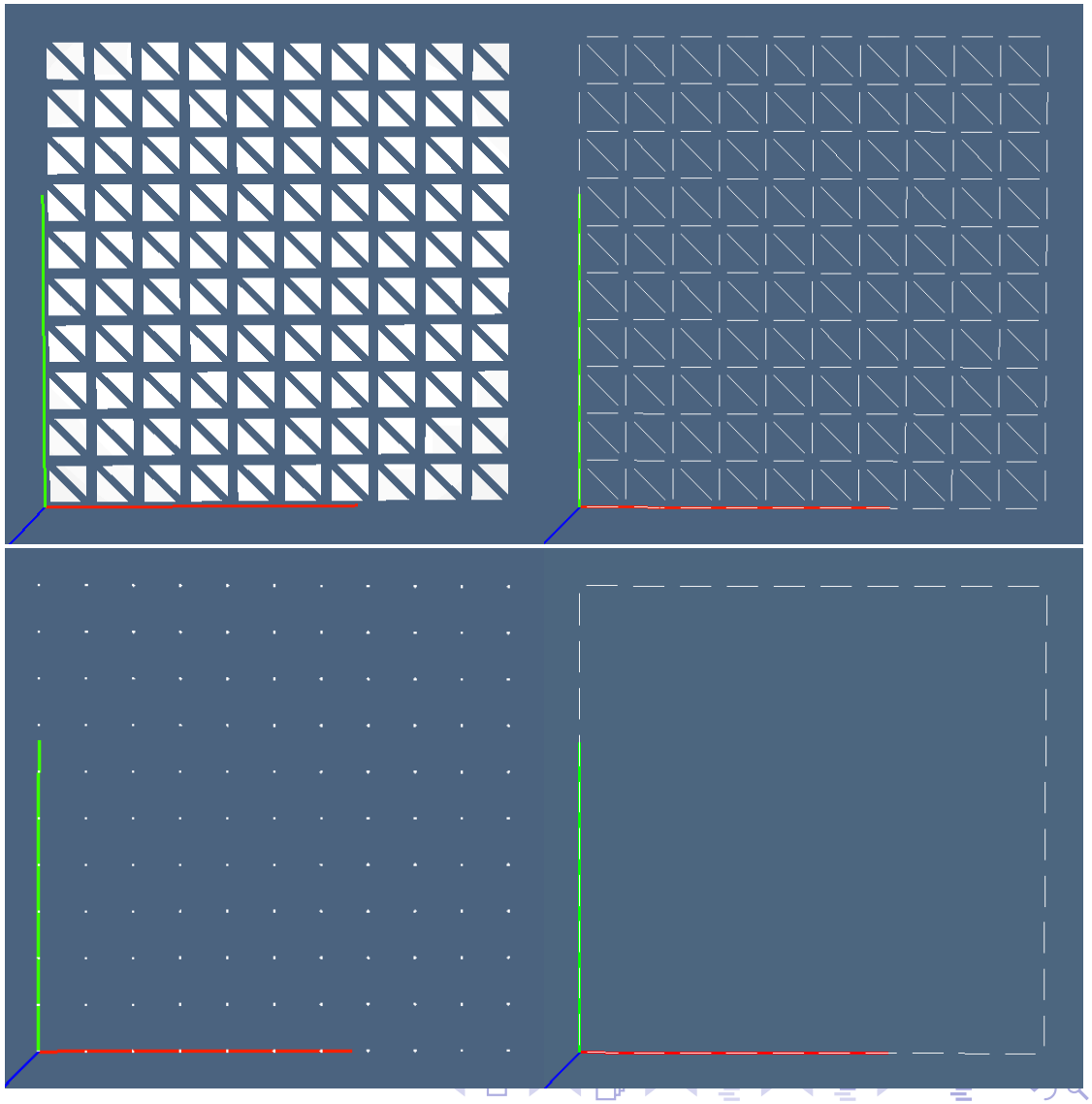
$$= \begin{bmatrix}
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&1&1&0&0&0&0&0&0&0&0&0\\
0&0&1&1&0&0&0&1&0&1&0&0&0&0&0&0\\
0&0&0&0&1&1&0&1&1&0&1&1&0&0&0&0\\
0&0&0&0&0&0&1&0&1&0&0&0&1&1&0&0\\
0&0&0&0&0&0&0&0&0&1&1&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1
\end{bmatrix} \times \begin{bmatrix}
1&0&0&0&0&0\\
0&1&0&0&0&0\\
1&0&0&0&0&0\\
1&0&0&0&0&0\\
0&1&0&0&0&0\\
0&1&1&0&0&0\\
0&0&1&0&0&0\\
0&0&0&1&0&0\\
0&0&1&0&0&0\\
0&0&0&1&0&0\\
0&0&0&1&1&0\\
0&0&0&0&1&0\\
0&0&0&0&0&1\\
0&0&0&0&0&1\\
0&0&0&0&1&0\\
0&0&0&0&0&1
\end{bmatrix} \times \begin{bmatrix}
1&1&0\\
1&1&0\\
1&0&0\\
1&0&1\\
1&0&1\\
1&0&1
\end{bmatrix} = \begin{bmatrix}
2&2&0\\
4&2&0\\
4&0&0\\
4&2&2\\
8&0&4\\
4&0&2\\
4&0&4\\
4&0&4\\
2&0&2
\end{bmatrix}$$

$$\mathbb{Z}_2\begin{pmatrix}
2&2&0\\
4&2&0\\
4&0&0\\
4&2&2\\
8&0&4\\
4&0&2\\
4&0&4\\
4&0&4\\
2&0&2
\end{pmatrix} = \begin{bmatrix}
0&0&0\\
0&0&0\\
0&0&0\\
0&0&0\\
0&0&0\\
0&0&0\\
0&0&0\\
0&0&0\\
0&0&0
\end{bmatrix}$$

Example: 2D simplicial grid

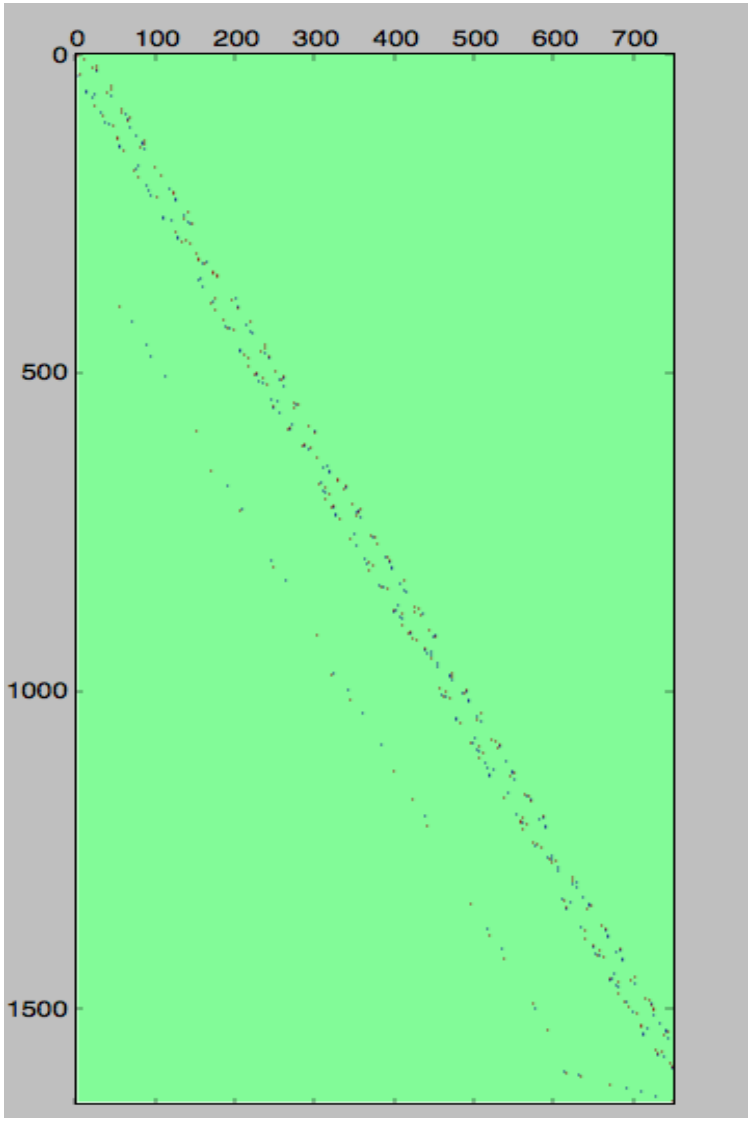
Boundary computation via a single SpMV product

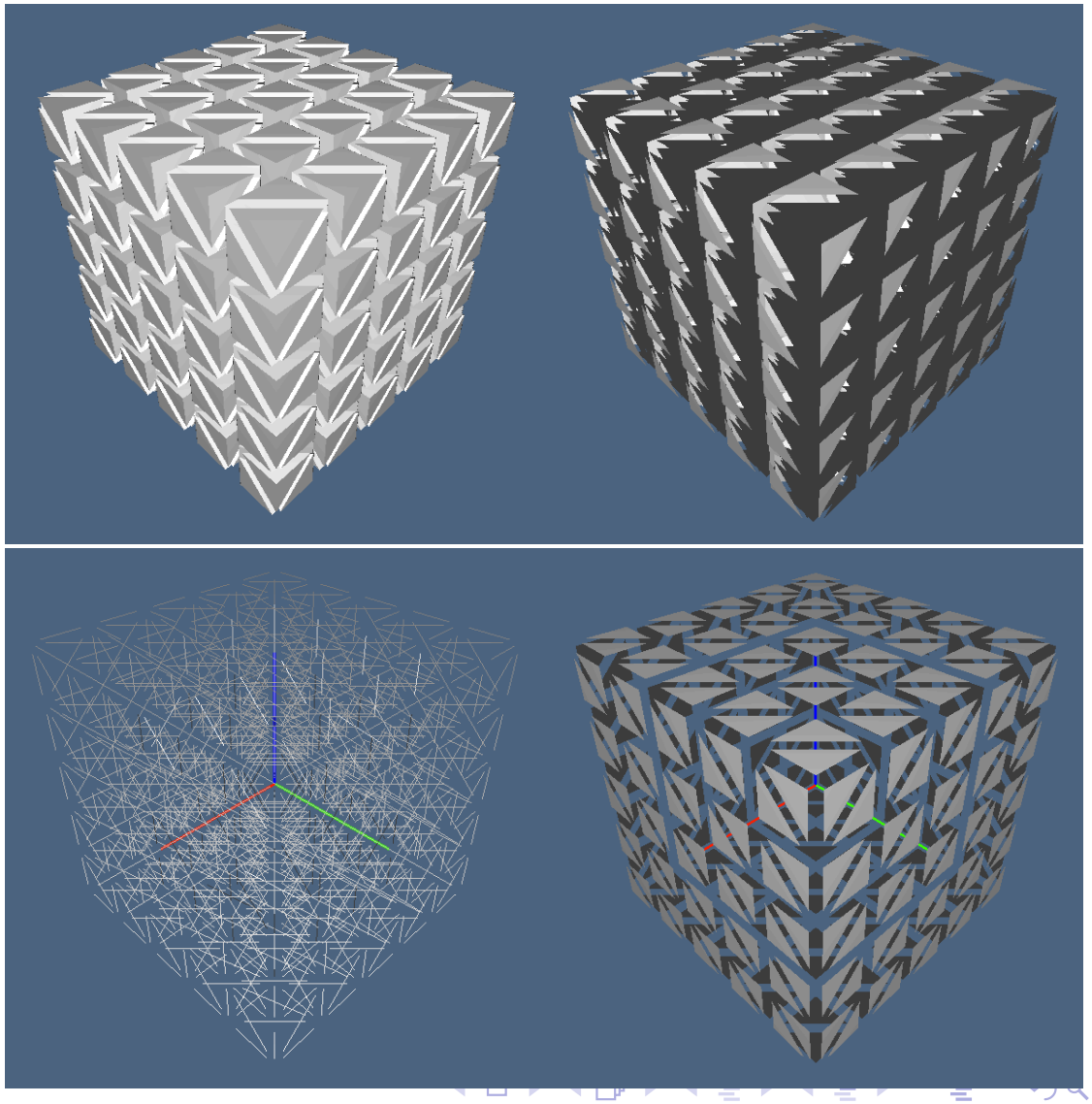




Example: 3D simplicial grid

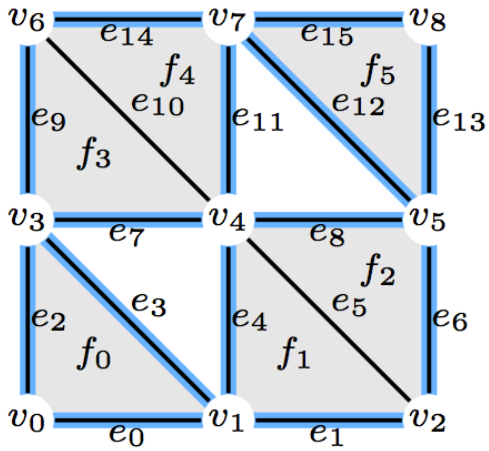
Boundary computation via a single SpMV product





Incidence and adjacency relations

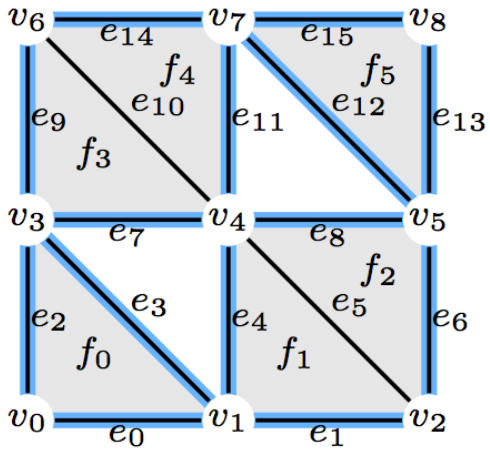
Computation of any topological query via a single SpMV product



for convex-cell complexes
(ex: simplicial or cuboidal)
only the CSR(M_d) is needed:

Incidence and adjacency relations

Computation of any topological query via a single SpMV product

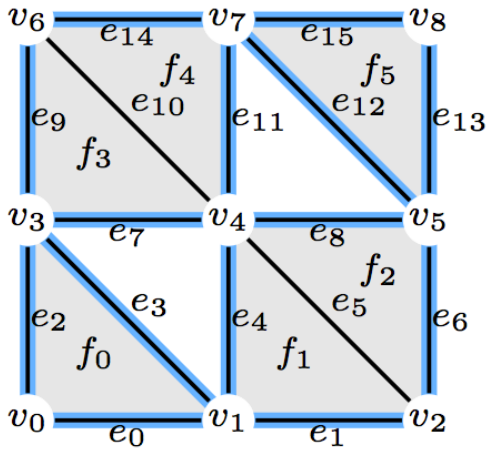


for convex-cell complexes
(ex: simplicial or cuboidal)
only the CSR(M_d) is needed:

	$M_1^t \Leftarrow$	$M_q^t \Leftarrow$	M_d^t
M_1		$\Downarrow$	
$\Uparrow$			
M_p	$\Longrightarrow$	$M_p M_q^t$	
$\Uparrow$			
M_d			

Incidence and adjacency relations

Computation of any topological query via a single SpMV product



for convex-cell complexes
(ex: simplicial or cuboidal)
only the CSR(M_d) is needed:

	$M_1^t \Leftarrow$	$M_q^t \Leftarrow$	M_d^t
M_1		$\Downarrow$	
$\Uparrow$			
M_p	$\Longrightarrow$	$M_pM_q^t$	
$\Uparrow$			
M_d			

$M_1^tM_1 =$ VV =
```
[[0,1,3],
 [0,1,2,3,4],
 [1,2,4,5],
 [0,1,3,4,6],
 [1,2,3,4,5,6,7],
 [2,4,5,7,8],
 [3,4,6,7],
 [4,5,6,7,8],
 [5,7,8]]
```

$M_1^t =$ VE =
```
[[0,2],
 [0,1,3,4],
 [1,5,6],
 [2,3,7,9],
 [4,5,7,8,10,11],
 [6,8,12,13],
 [9,10,14],
 [11,12,14,15],
 [13,15]]
```

$M_2^t =$ VF =
```
[[0],
 [0,1],
 [1,2],
 [0,3],
 [1,2,3,4],
 [2,5],
 [3,4],
 [4,5],
 [5]]
```

$M_1 =$ EV =
```
[[0,1],
 [1,2],
 [0,3],
 [1,3],
 [1,4],
 [2,4],
 [2,5],
 [3,4],
 [4,5],
 [3,6],
 [4,6],
 [4,7],
 [5,7],
 [5,8],
 [6,7],
 [7,8]]
```

$M_1M_1^t =$ EE =
```
[[0,1,2,3,4],
 [0,1,3,4,5,6],
 [0,2,3,7,9],
 [0,1,2,3,4,7,9],
 [0,1,3,4,5,7,8,10,11],
 [1,4,5,6,7,8,10,11],
 [1,5,6,8,12,13],
 [2,3,4,5,7,8,9,10,11],
 [4,5,6,7,8,10,11,12,13],
 [2,3,7,9,10,14],
 [4,5,7,8,9,10,11,14],
 [4,5,7,8,10,11,12,14,15],
 [6,8,11,12,13,14,15],
 [6,8,12,13,15],
 [9,10,11,12,14,15],
 [11,12,13,14,15]]
```

$M_1M_2^t =$ EF =
```
[[0,1],
 [0,1,2],
 [0,3],
 [0,1,3],
 [0,1,2,3,4],
 [1,2,3,4],
 [1,2,5],
 [0,1,2,3,4],
 [1,2,3,4,5],
 [0,3,4],
 [1,2,3,4],
 [1,2,3,4,5],
 [2,4,5],
 [2,5],
 [3,4,5],
 [4,5]]
```

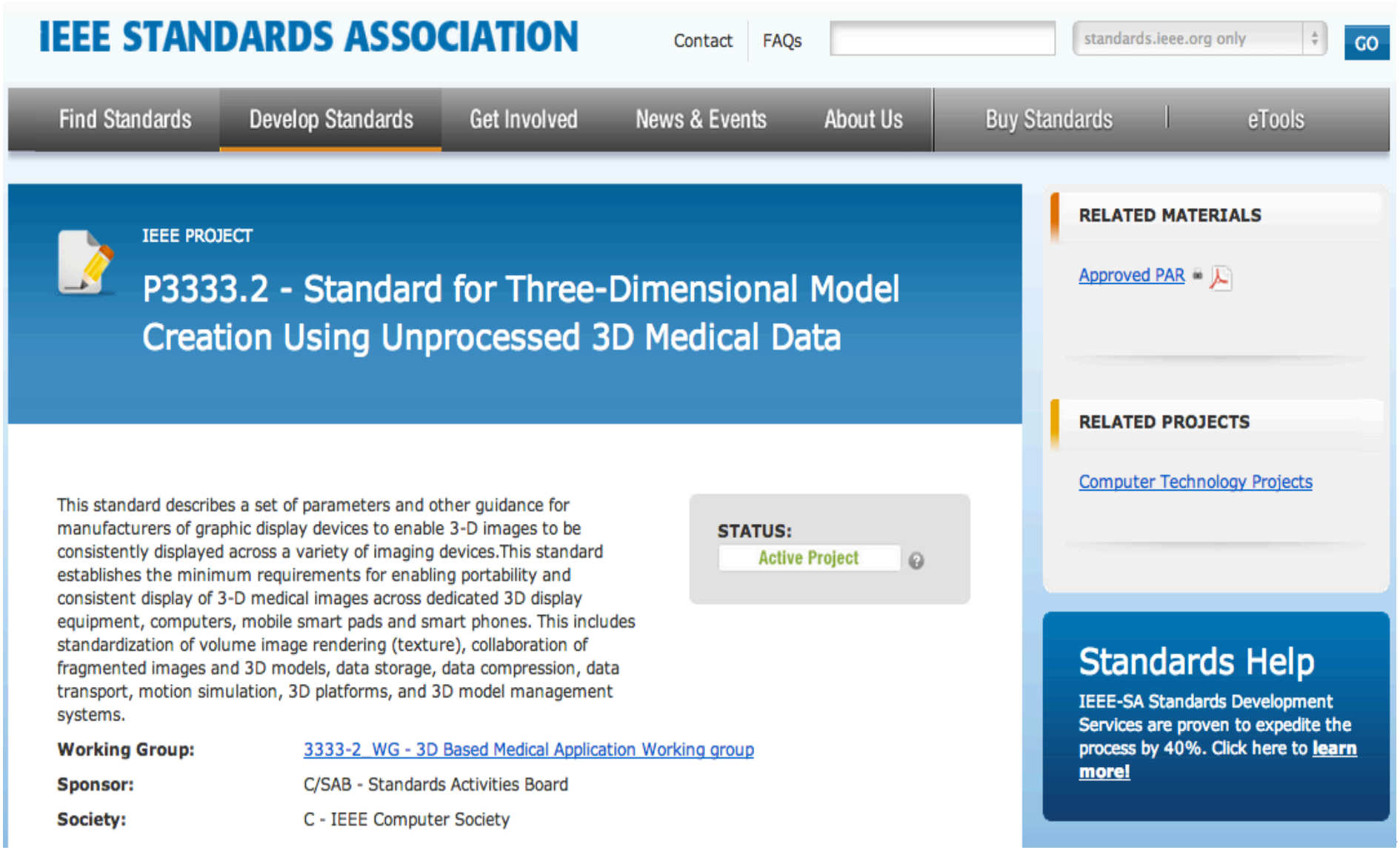
$M_2 =$ FV =
```
[[0,1,3],
 [1,2,4],
 [2,4,5],
 [3,4,6],
 [4,6,7],
 [5,7,8]]
```

$M_2M_1^t =$ FE =
```
[[0,1,2,3,4,7,9],
 [0,1,3,4,5,6,7,8,10,11],
 [1,4,5,6,7,8,10,11,12,13],
 [2,3,4,5,7,8,9,10,11,14],
 [4,5,7,8,9,10,11,12,14,15],
 [6,8,11,12,13,14,15]]
```

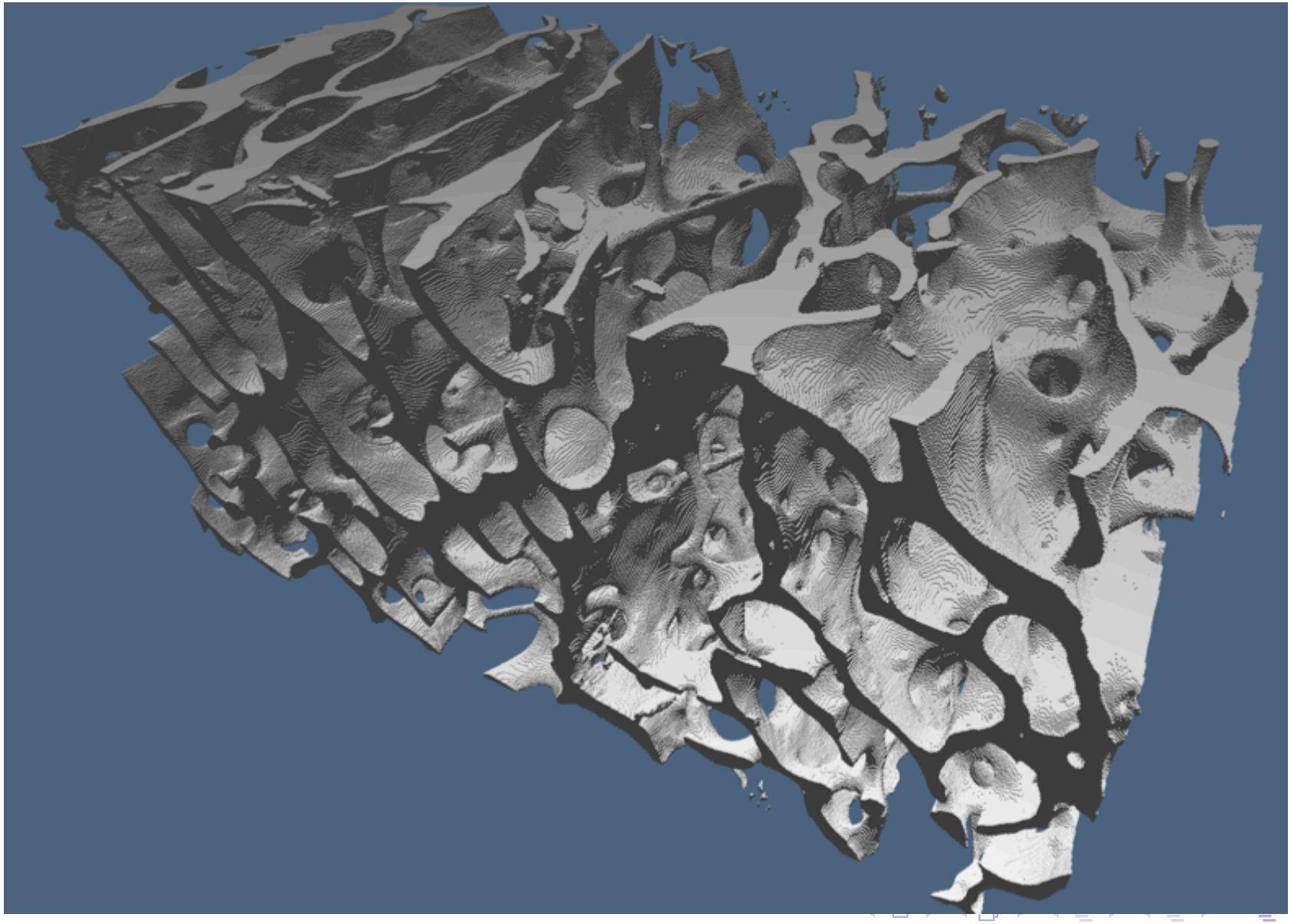
$M_2M_2^t =$ FF =
```
[[0,1,3],
 [0,1,2,3,4],
 [1,2,3,4,5],
 [0,1,2,3,4],
 [1,2,3,4,5],
 [2,4,5]]
```

Models from medical images

LAR being developed in the framework of new 3D medical standard

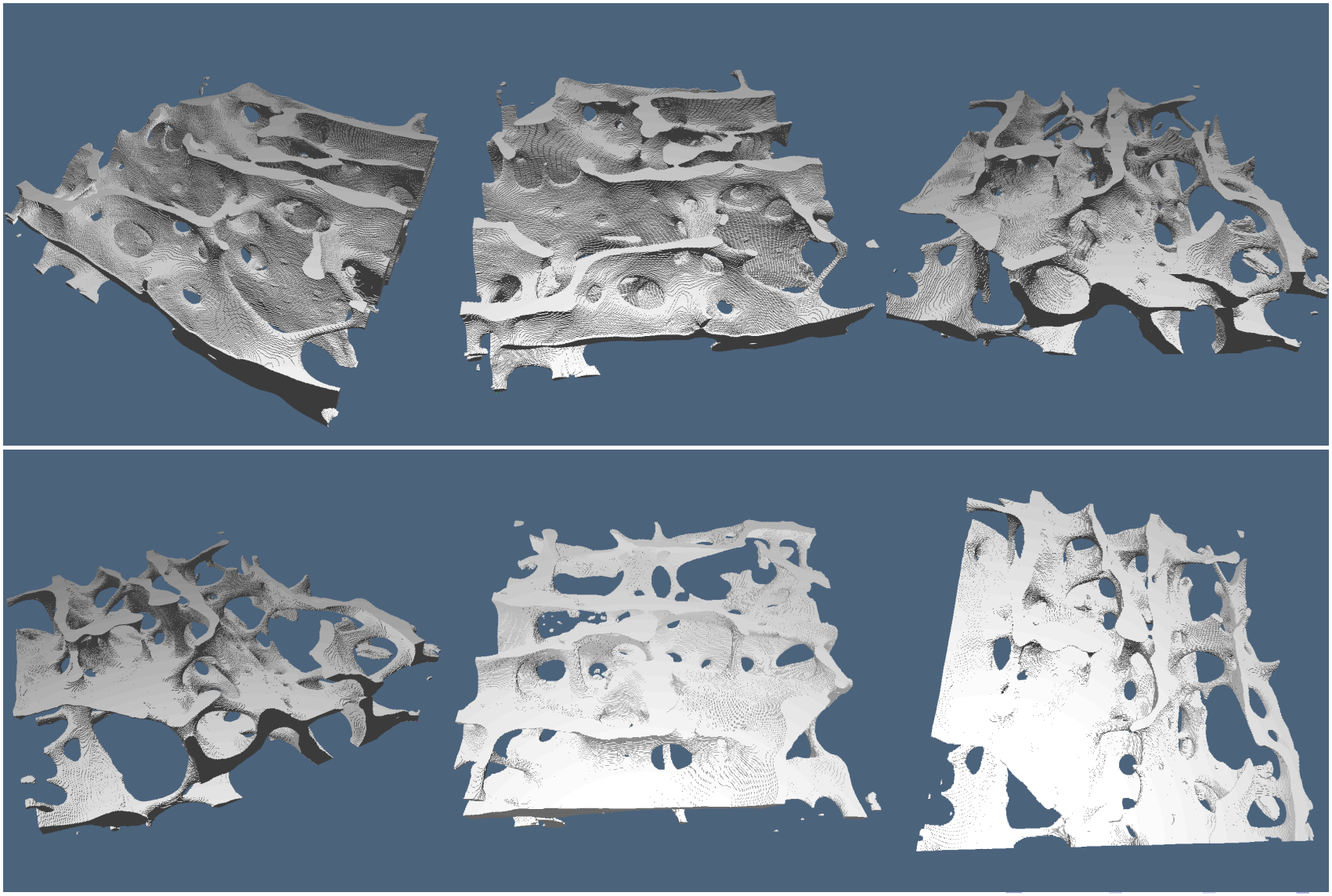


Solid model from 3D image (spongy bone)



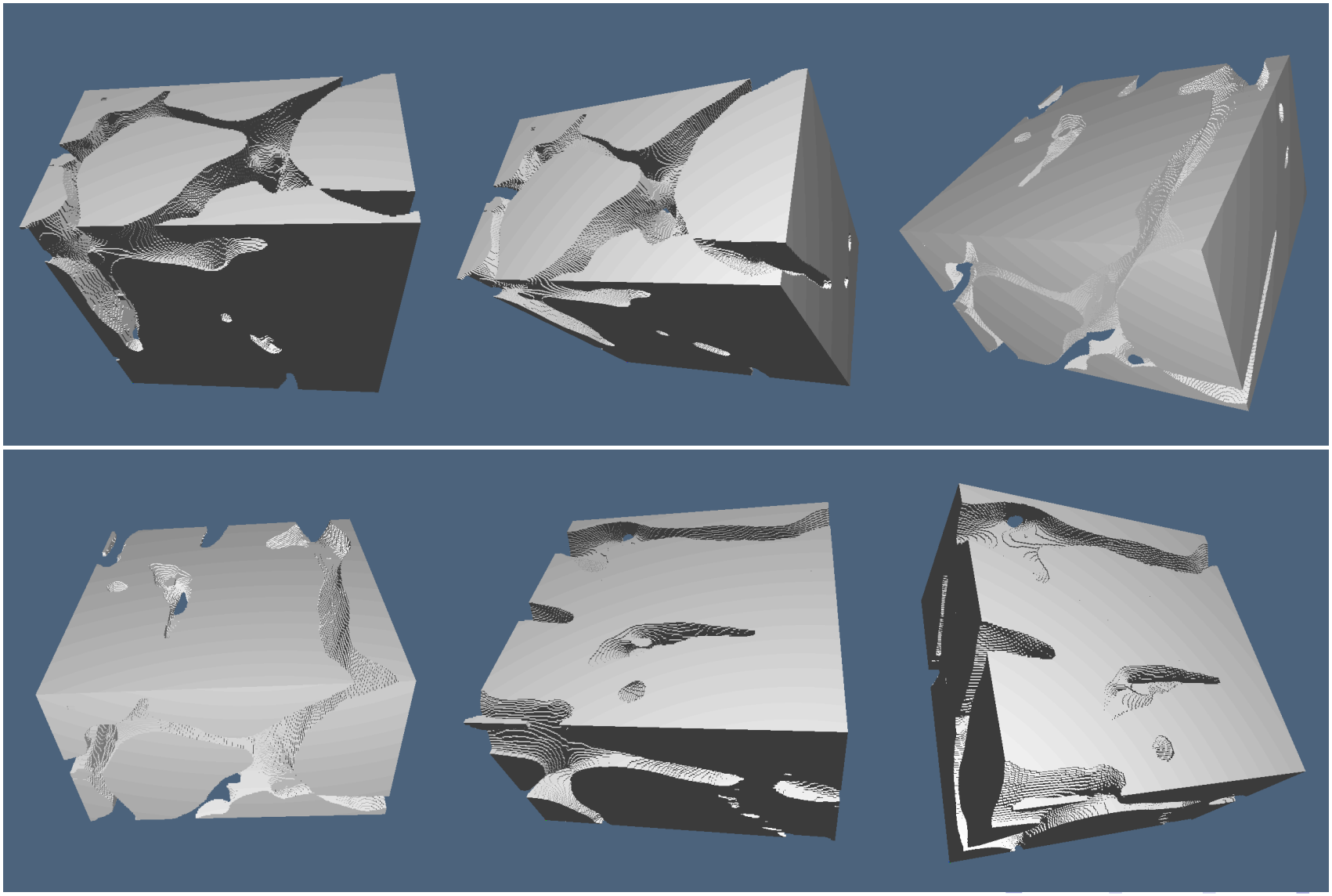
Example

Solid model from 3D image (spongy bone)



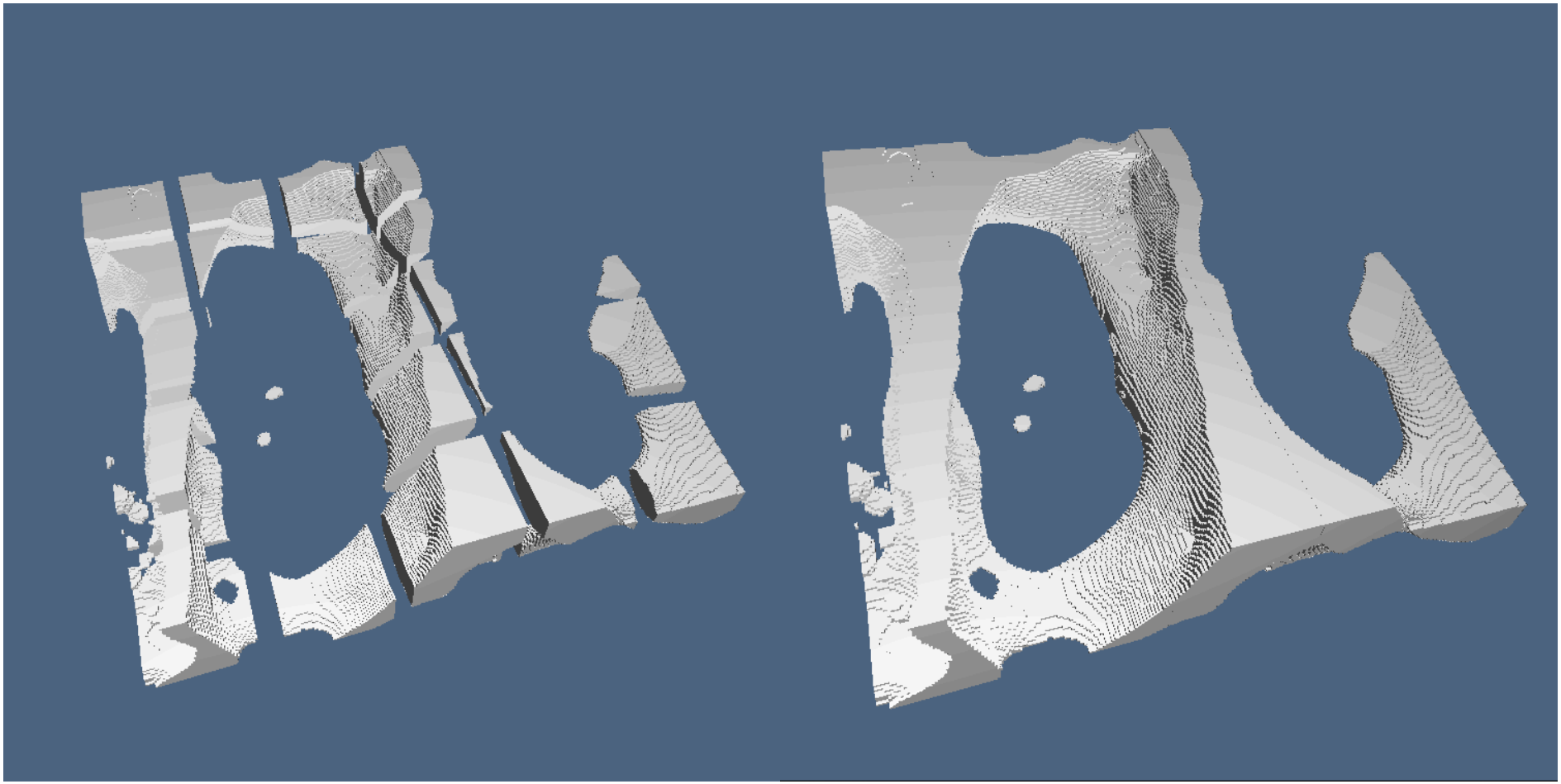
Spongy bone complement

if $c \in C_3$ is a chain of given density (bone), just use the complement chain $C_3 - c$



Paradigm: *Divide et Impera*

Bottleneck of GPGPU: moving data from global to local memory



Solution: store the (sparse) $[\partial_3]$ of n^3 voxels in device Constant Memory, and move the (binary) coordinate vectors of chains in Private Memory

Example: boundary model of spongy bone

Bottleneck of GPGPU: moving data from global to local memory

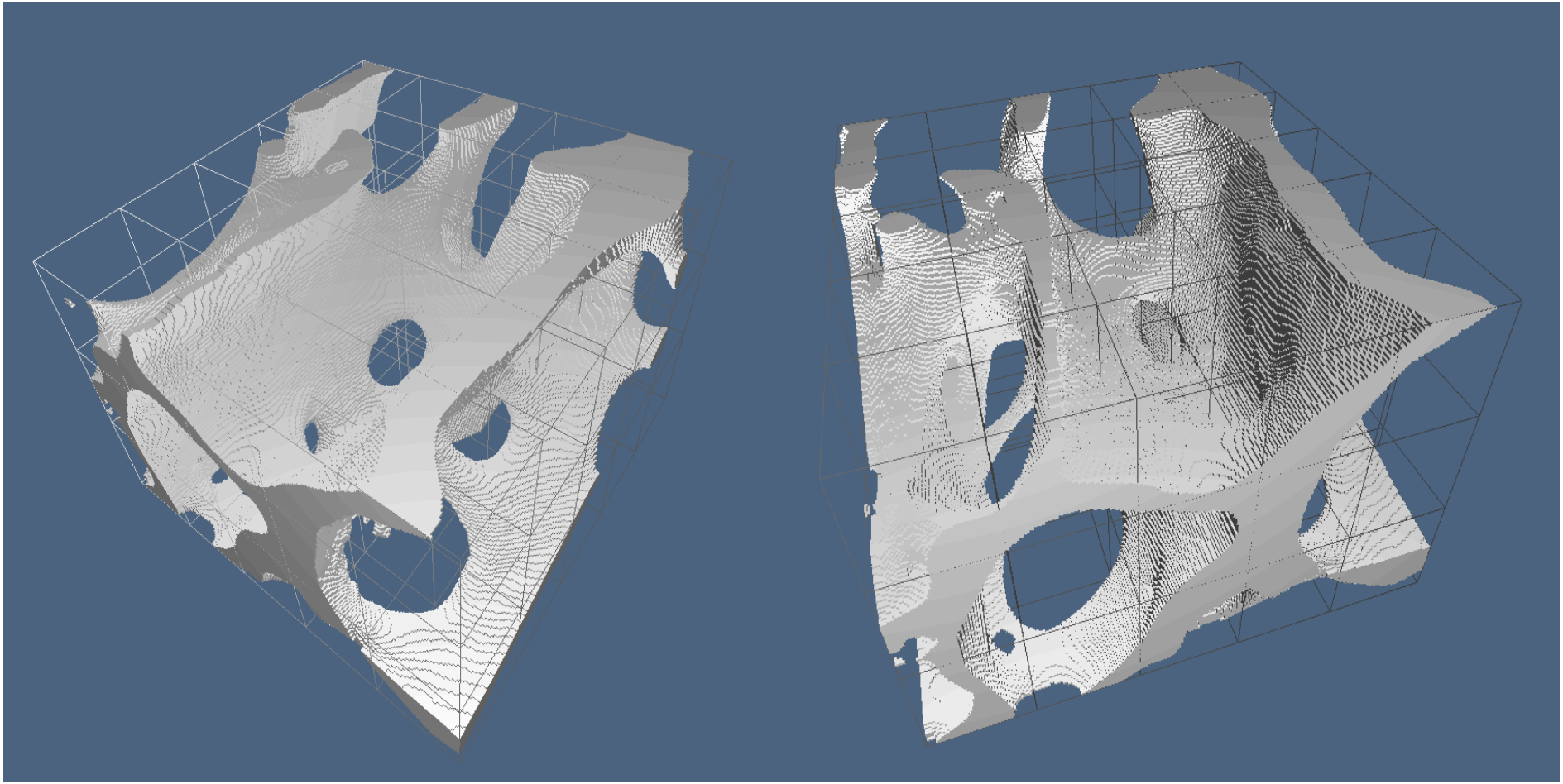
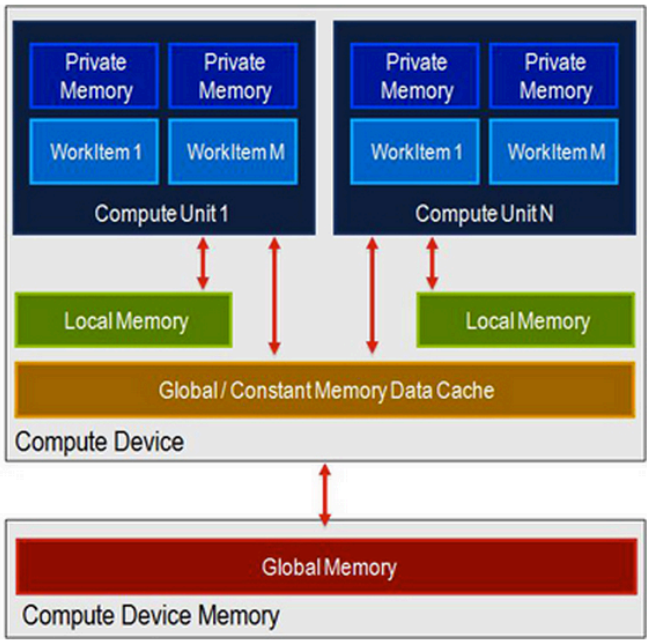


Image serialization & model extraction

Serialization is the conversion of a subimage to a stream of bytes

to be easily moved across the memory hierarchy or streamed across a communication link



Remark

- *CSR ∂_3 and ∂_2 matrices stored in Constant Memory data cache (read only)*
- *single binary (sparse) chains sent to in Local Memory of work groups*
- *result vectors from work groups in Local Memory converted to quads in Global Memory*
- *each work-group responsible for a chain chunk (subimage) of* $16k$

Input: serialized 3D image $\rightarrow$ Output: stream of 3D model blocks