Critical Groups of Simplicial Complexes

Art Duval[1] Caroline Klivans[2] Jeremy Martin[3]

[1]University of Texas at El Paso

[2]University of Chicago

[3]University of Kansas

Formal Power Series and Algebraic Combinatorics
Reykjavik, Iceland
June 17, 2011

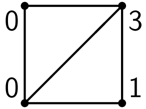
Sandpiles and chip-firing

Motivation Think of a sandpile, with grains of sand on vertices of a graph. When the pile at one place is too large, it topples, sending grains to all its neighbors.

Sandpiles and chip-firing

Motivation Think of a sandpile, with grains of sand on vertices of a graph. When the pile at one place is too large, it topples, sending grains to all its neighbors.

Abstraction Graph G with vertices $v_1, \dots, v_n$. Degree of v_i is d_i. Place $c_i \in \mathbb{Z}$ chips (grains of sand) on v_i.

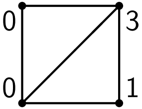


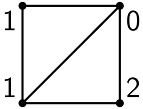
Sandpiles and chip-firing

Motivation Think of a sandpile, with grains of sand on vertices of a graph. When the pile at one place is too large, it topples, sending grains to all its neighbors.

Abstraction Graph G with vertices $v_1, \ldots, v_n$. Degree of v_i is d_i. Place $c_i \in \mathbb{Z}$ chips (grains of sand) on v_i.

Toppling If $c_i \geq d_i$, then v_i may fire by sending one chip to each of its neighbors.



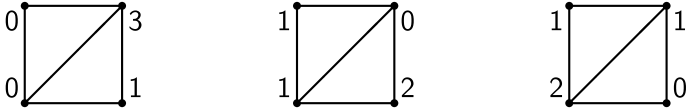


Sandpiles and chip-firing

Motivation Think of a sandpile, with grains of sand on vertices of a graph. When the pile at one place is too large, it topples, sending grains to all its neighbors.

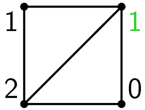
Abstraction Graph G with vertices $v_1, \ldots, v_n$. Degree of v_i is d_i. Place $c_i \in \mathbb{Z}$ chips (grains of sand) on v_i.

Toppling If $c_i \geq d_i$, then v_i may fire by sending one chip to each of its neighbors.



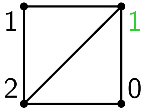
Source vertex

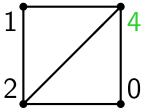
- To keep things going, pick one vertex v_r to be a source vertex. We can always add chips to v_r.



Source vertex

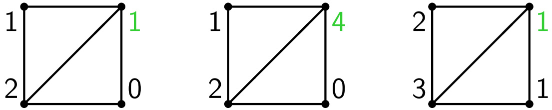
- To keep things going, pick one vertex v_r to be a source vertex. We can always add chips to v_r.
- Put another way: c_r can be any value.





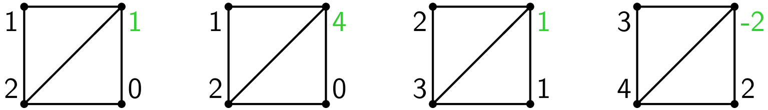
Source vertex

- To keep things going, pick one vertex v_r to be a source vertex. We can always add chips to v_r.
- Put another way: c_r can be any value.
- We might think $c_r \leq 0$, and $c_i \geq 0$ when $i \neq r$, or that v_r can fire even when $c_r \leq d_r$.



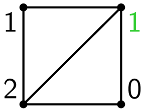
Source vertex

- To keep things going, pick one vertex v_r to be a source vertex. We can always add chips to v_r.
- Put another way: c_r can be any value.
- We might think $c_r \leq 0$, and $c_i \geq 0$ when $i \neq r$, or that v_r can fire even when $c_r \leq d_r$.



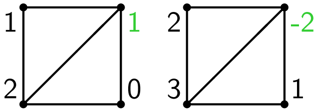
Critical configurations

- A configuration is stable when no vertex (except the source vertex) can fire.



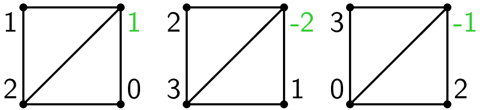
Critical configurations

- A configuration is stable when no vertex (except the source vertex) can fire.
- A configuration is recurrent when a series of topplings leads back to that configuration, without letting any vertex (except the source vertex) go negative.



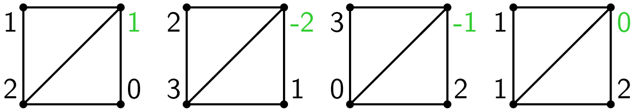
Critical configurations

- A configuration is stable when no vertex (except the source vertex) can fire.
- A configuration is recurrent when a series of topplings leads back to that configuration, without letting any vertex (except the source vertex) go negative.
- A configuration is critical when it is stable and recurrent.



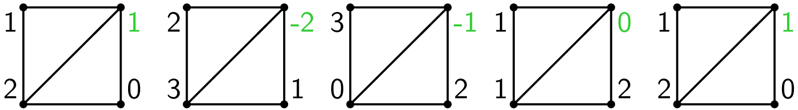
Critical configurations

- A configuration is stable when no vertex (except the source vertex) can fire.
- A configuration is recurrent when a series of topplings leads back to that configuration, without letting any vertex (except the source vertex) go negative.
- A configuration is critical when it is stable and recurrent.



Critical configurations

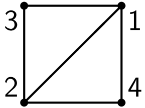
- A configuration is stable when no vertex (except the source vertex) can fire.
- A configuration is recurrent when a series of topplings leads back to that configuration, without letting any vertex (except the source vertex) go negative.
- A configuration is critical when it is stable and recurrent.



Fact: Every configuration topples to a unique critical configuration.

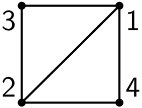
Laplacian

Let's make a matrix of how chips move when each vertex fires:



Laplacian

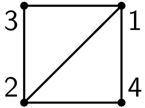
Let's make a matrix of how chips move when each vertex fires:



$$L = \begin{pmatrix} 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 2 & 0 \\ -1 & -1 & 0 & 2 \end{pmatrix} = D - A$$

Laplacian

Let's make a matrix of how chips move when each vertex fires:



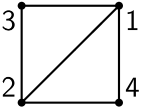
$$L = \begin{pmatrix} 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 2 & 0 \\ -1 & -1 & 0 & 2 \end{pmatrix} = D - A = \partial_1 \partial_1^T,$$

where

$$\partial_1 = \begin{array}{c|ccccc} & 12 & 13 & 14 & 23 & 24 \\ \hline 1 & -1 & -1 & -1 & 0 & 0 \\ 2 & 1 & 0 & 0 & -1 & -1 \\ 3 & 0 & 1 & 0 & 1 & 0 \\ 4 & 0 & 0 & 1 & 0 & 1 \end{array}$$

Laplacian

Let's make a matrix of how chips move when each vertex fires:



$$L = \begin{pmatrix} 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 2 & 0 \\ -1 & -1 & 0 & 2 \end{pmatrix} = D - A = \partial_1 \partial_1^T,$$

where

$$\partial_1 = \begin{array}{c|ccccc} & 12 & 13 & 14 & 23 & 24 \\ \hline 1 & -1 & -1 & -1 & 0 & 0 \\ 2 & 1 & 0 & 0 & -1 & -1 \\ 3 & 0 & 1 & 0 & 1 & 0 \\ 4 & 0 & 0 & 1 & 0 & 1 \end{array}$$

So firing v is subtracting Lv (row/column v from L) from $(c_1, \ldots, c_n)$.

Kernel ∂_0

- Did you notice?: Sum of chips stays constant.

Kernel ∂_0

- Did you notice?: Sum of chips stays constant.
- Also recall value of the source vertex can be anything, including negative (other vertices should stay positive).

Kernel ∂_0

- Did you notice?: Sum of chips stays constant.
- Also recall value of the source vertex can be anything, including negative (other vertices should stay positive).
- So we may as well insist that

$$\sum_i c_i = 0.$$

In other words, $\partial_0 c = 0$, i.e., $c \in \ker \partial_0$.

Kernel ∂_0

- Did you notice?: Sum of chips stays constant.
- Also recall value of the source vertex can be anything, including negative (other vertices should stay positive).
- So we may as well insist that

$$\sum_i c_i = 0.$$

  In other words, $\partial_0 c = 0$, i.e., $c \in \ker \partial_0$.
- We can pick $c_i, i \neq r$, arbitrarily, and keep $c \in \ker \partial_0$ by picking c_r appropriately.

Kernel ∂_0

- Did you notice?: Sum of chips stays constant.
- Also recall value of the source vertex can be anything, including negative (other vertices should stay positive).
- So we may as well insist that

$$\sum_i c_i = 0.$$

  In other words, $\partial_0 c = 0$, i.e., $c \in \ker \partial_0$.
- We can pick $c_i, i \neq r$, arbitrarily, and keep $c \in \ker \partial_0$ by picking c_r appropriately.

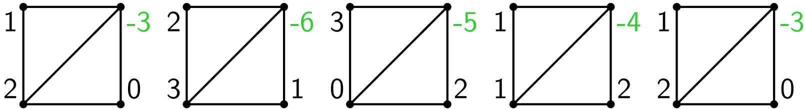
Kernel ∂_0

- Did you notice?: Sum of chips stays constant.
- Also recall value of the source vertex can be anything, including negative (other vertices should stay positive).
- So we may as well insist that

$$\sum_i c_i = 0.$$

In other words, $\partial_0 c = 0$, i.e., $c \in \ker \partial_0$.

- We can pick $c_i, i \neq r$, arbitrarily, and keep $c \in \ker \partial_0$ by picking c_r appropriately.



Critical group

- Consider two configurations (in $\ker \partial_0$) to be equivalent when you can get from one to the other by chip-firing.

Critical group

- Consider two configurations (in $\ker \partial_0$) to be equivalent when you can get from one to the other by chip-firing.
- Recall every configuration is equivalent to a critical configuration.

Critical group

- Consider two configurations (in $\ker \partial_0$) to be equivalent when you can get from one to the other by chip-firing.
- Recall every configuration is equivalent to a critical configuration.
- This equivalence means adding/subtracting integer multiples of Lv_i.

Critical group

- Consider two configurations (in $\ker \partial_0$) to be equivalent when you can get from one to the other by chip-firing.
- Recall every configuration is equivalent to a critical configuration.
- This equivalence means adding/subtracting integer multiples of Lv_i.
- In other words, instead of $\ker \partial_0$, we look at

$$K(G) := (\ker \partial_0)/(\operatorname{im} L) = (\ker \partial_0)/(\operatorname{im} \partial_1 \partial_1^T)$$

the critical group. (It is a graph invariant.)

Reduced Laplacian and spanning trees

Theorem (Biggs '99)

$$K := (\ker \partial_0)/(\operatorname{im} L) \cong \mathbb{Z}^{n-1}/(\operatorname{im} L_r),$$

where L_r denotes reduced Laplacian; remove row and column corresponding to source vertex.

Reduced Laplacian and spanning trees

Theorem (Biggs '99)

$$K := (\ker \partial_0)/(\operatorname{im} L) \cong \mathbb{Z}^{n-1}/(\operatorname{im} L_r),$$

where L_r denotes reduced Laplacian; remove row and column corresponding to source vertex.

Corollary

$|K(G)|$ is the number of spanning trees of G.

Reduced Laplacian and spanning trees

Theorem (Biggs '99)

$$K := (\ker \partial_0)/(\operatorname{im} L) \cong \mathbb{Z}^{n-1}/(\operatorname{im} L_r),$$

where L_r denotes reduced Laplacian; remove row and column corresponding to source vertex.

Corollary

$|K(G)|$ *is the number of spanning trees of* G*.*

Proof.

If M is a full rank t-dimensional matrix, then

$$|(\mathbb{Z}^t)/(\operatorname{im} M)| = \pm \det M$$

Reduced Laplacian and spanning trees

Theorem (Biggs '99)

$$K := (\ker \partial_0)/(\operatorname{im} L) \cong \mathbb{Z}^{n-1}/(\operatorname{im} L_r),$$

where L_r denotes reduced Laplacian; remove row and column corresponding to source vertex.

Corollary

$|K(G)|$ *is the number of spanning trees of* G*.*

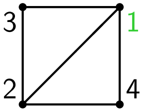
Proof.

If M is a full rank t-dimensional matrix, then

$$|(\mathbb{Z}^t)/(\operatorname{im} M)| = \pm \det M$$

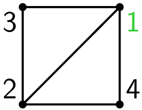
and $|\det L_r|$ counts spanning trees. □

Example



$$L = \begin{pmatrix} 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 2 & 0 \\ -1 & -1 & 0 & 2 \end{pmatrix}$$

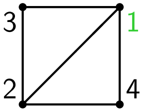
Example



$$L = \begin{pmatrix} 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 2 & 0 \\ -1 & -1 & 0 & 2 \end{pmatrix}$$

$$L_r = \begin{pmatrix} 3 & -1 & -1 \\ -1 & 2 & 0 \\ -1 & 0 & 2 \end{pmatrix}$$

Example



$$L = \begin{pmatrix} 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 2 & 0 \\ -1 & -1 & 0 & 2 \end{pmatrix}$$

$$L_r = \begin{pmatrix} 3 & -1 & -1 \\ -1 & 2 & 0 \\ -1 & 0 & 2 \end{pmatrix}$$

$\det L_r = 8$, and there are 8 spanning trees of this graph

Generalize to simplicial complexes

Let Δ be a d-dimensional simplicial complex.

$$C_d(\Delta;\mathbb{Z}) \underset{\partial_d}{\overset{\partial_d^T}{\leftrightarrows}} C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{\partial_{d-1}} C_{d-2}(\Delta;\mathbb{Z}) \to \cdots$$

$$C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{L_{d-1}} C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{\partial_{d-1}} C_{d-2}(\Delta;\mathbb{Z}) \to \cdots$$

Generalize to simplicial complexes

Let Δ be a d-dimensional simplicial complex.

$$C_d(\Delta;\mathbb{Z}) \underset{\partial_d}{\overset{\partial_d^T}{\leftrightarrows}} C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{\partial_{d-1}} C_{d-2}(\Delta;\mathbb{Z}) \to \cdots$$

$$C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{L_{d-1}} C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{\partial_{d-1}} C_{d-2}(\Delta;\mathbb{Z}) \to \cdots$$

Define

$$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1})$$

where $L_{d-1} = \partial_d \, \partial_d^T$ is the $(d-1)$-dimensional up-down Laplacian.

Generalize to simplicial complexes

Let Δ be a d-dimensional simplicial complex.

$$C_d(\Delta;\mathbb{Z}) \underset{\partial_d}{\overset{\partial_d^T}{\leftrightarrows}} C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{\partial_{d-1}} C_{d-2}(\Delta;\mathbb{Z}) \rightarrow \cdots$$

$$C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{L_{d-1}} C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{\partial_{d-1}} C_{d-2}(\Delta;\mathbb{Z}) \rightarrow \cdots$$

Define

$$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1})$$

where $L_{d-1} = \partial_d\, \partial_d^T$ is the $(d-1)$-dimensional up-down Laplacian.
Can we compute it with a reduced Laplacian?

Generalize to simplicial complexes

Let Δ be a d-dimensional simplicial complex.

$$C_d(\Delta;\mathbb{Z}) \underset{\partial_d}{\overset{\partial_d^T}{\leftrightarrows}} C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{\partial_{d-1}} C_{d-2}(\Delta;\mathbb{Z}) \to \cdots$$

$$C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{L_{d-1}} C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{\partial_{d-1}} C_{d-2}(\Delta;\mathbb{Z}) \to \cdots$$

Define

$$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1})$$

where $L_{d-1} = \partial_d\, \partial_d^T$ is the $(d-1)$-dimensional up-down Laplacian. Can we compute it with a reduced Laplacian? How do we reduce the Laplacian?

Generalize to simplicial complexes

Let Δ be a d-dimensional simplicial complex.

$$C_d(\Delta;\mathbb{Z}) \underset{\partial_d}{\overset{\partial_d^T}{\leftrightarrows}} C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{\partial_{d-1}} C_{d-2}(\Delta;\mathbb{Z}) \to \cdots$$

$$C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{L_{d-1}} C_{d-1}(\Delta;\mathbb{Z}) \xrightarrow{\partial_{d-1}} C_{d-2}(\Delta;\mathbb{Z}) \to \cdots$$

Define

$$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1})$$

where $L_{d-1} = \partial_d\, \partial_d^T$ is the $(d-1)$-dimensional up-down Laplacian. Can we compute it with a reduced Laplacian? How do we reduce the Laplacian? And what about the trees?

Simplicial spanning trees of arbitrary simplicial complexes

Let Δ be a d-dimensional simplicial complex, and assume it is **APC**, acyclic in positive codimension, i.e., $\tilde{H}_j(\Delta;\mathbb{Z})$ is finite for all $j < d$.

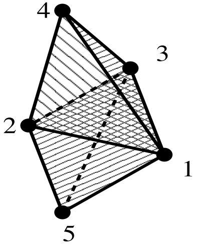
$\Upsilon \subseteq \Delta$ is a **simplicial spanning tree** of Δ when:

0. $\Upsilon_{(d-1)} = \Delta_{(d-1)}$ ("spanning");
1. $\tilde{H}_{d-1}(\Upsilon;\mathbb{Z})$ is a finite group ("connected");
2. $\tilde{H}_d(\Upsilon;\mathbb{Z}) = 0$ ("acyclic");
3. $f_d(\Upsilon) = f_d(\Delta) - \tilde{\beta}_d(\Delta) + \tilde{\beta}_{d-1}(\Delta)$ ("count").

- If 0. holds, then any two of 1., 2., 3. together imply the third condition.
- When $d = 1$, coincides with usual definition.

Example

Bipyramid with equator, $\langle 123, 124, 125, 134, 135, 234, 235 \rangle$



Let's figure out all its simplicial spanning trees.

Reduced Laplacians to count spanning trees

Let $\mathcal{T}(\Delta)$ denote the spanning trees of Δ.

- Δ a d-dimensional APC complex
- $\Gamma \in \mathcal{T}(\Delta_{(d-1)})$
- ∂_Γ = restriction of ∂_d to faces not in Γ
- reduced (up-down) $(d-1)$-dimensional Laplacian $L_\Gamma = \partial_\Gamma {\partial^T}_\Gamma$

Simplicial Matrix-Tree Theorem [DKM '09]

$$h_d = \sum_{\Upsilon \in \mathcal{T}(\Delta)} |\tilde{H}_{d-1}(\Upsilon)|^2 = \frac{|\tilde{H}_{d-2}(\Delta;\mathbb{Z})|^2}{|\tilde{H}_{d-2}(\Gamma;\mathbb{Z})|^2} \det L_\Gamma.$$

Note: The $|\tilde{H}_{d-2}|$ terms are often trivial.

Bipyramid again

$\Gamma = 12, 13, 14, 15$ spanning tree of 1-skeleton

Bipyramid again

$\Gamma = 12, 13, 14, 15$ spanning tree of 1-skeleton

$$L_\Gamma = \begin{array}{c|ccccc} & 23 & 24 & 25 & 34 & 35 \\ \hline 23 & 3 & -1 & -1 & 1 & 1 \\ 24 & -1 & 2 & 0 & -1 & 0 \\ 25 & -1 & 0 & 2 & 0 & -1 \\ 34 & 1 & -1 & 0 & 2 & 0 \\ 35 & 1 & 0 & -1 & 0 & 2 \end{array}$$

Bipyramid again

$\Gamma = 12, 13, 14, 15$ spanning tree of 1-skeleton

$$L_\Gamma = \begin{array}{c|ccccc} & 23 & 24 & 25 & 34 & 35 \\ \hline 23 & 3 & -1 & -1 & 1 & 1 \\ 24 & -1 & 2 & 0 & -1 & 0 \\ 25 & -1 & 0 & 2 & 0 & -1 \\ 34 & 1 & -1 & 0 & 2 & 0 \\ 35 & 1 & 0 & -1 & 0 & 2 \end{array}$$

$\det L_\Gamma = 15.$

How to reduce Laplacian?

Graphs To count spanning trees, and compute critical group, remove a vertex. (Source vertex of sandpiles.)

How to reduce Laplacian?

Graphs To count spanning trees, and compute critical group, remove a vertex. (Source vertex of sandpiles.)

Simplicial complexes

How to reduce Laplacian?

Graphs To count spanning trees, and compute critical group, remove a vertex. (Source vertex of sandpiles.)

Simplicial complexes

- To count spanning trees, remove a $(d-1)$-dimensional spanning tree from up-down Laplacian.

How to reduce Laplacian?

Graphs: To count spanning trees, and compute critical group, remove a vertex. (Source vertex of sandpiles.)

Simplicial complexes

- To count spanning trees, remove a $(d-1)$-dimensional spanning tree from up-down Laplacian.
- To compute critical group, remove a $(d-1)$-dimensional spanning tree from up-down Laplacian.

Spanning trees

Theorem (DKM)

$$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1}) \cong \mathbb{Z}^t/(\operatorname{im} L_\Gamma)$$

where Γ is a torsion-free $(d-1)$-dimensional spanning tree and $t = \dim L_\Gamma$.

Spanning trees

Theorem (DKM)

$$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1}) \cong \mathbb{Z}^t/(\operatorname{im} L_\Gamma)$$

where Γ is a torsion-free $(d-1)$-dimensional spanning tree and $t = \dim L_\Gamma$.

Corollary

$|K(\Delta)|$ is the torsion-weighted number of d-dimensional spanning trees of Δ.

Proof.

$|K(\Delta)| = |\mathbb{Z}^t/(\operatorname{im} L_\Gamma)| = |\det L_\Gamma|$, which counts (torsion-weighted) spanning trees. □

What does it look like?

$$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1}) \subseteq \mathbb{Z}^m$$

What does it look like?

$$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1}) \subseteq \mathbb{Z}^m$$

- Put integers on $(d-1)$-faces of Δ. Orient faces arbitrarily.
  $d = 2$: flow; $d = 3$: circulation; etc.

What does it look like?

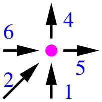
$$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1}) \subseteq \mathbb{Z}^m$$

- Put integers on $(d-1)$-faces of Δ. Orient faces arbitrarily. $d=2$: flow; $d=3$: circulation; etc.
- conservative flow

What does it look like?

$$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1}) \subseteq \mathbb{Z}^m$$

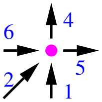
- Put integers on $(d-1)$-faces of Δ. Orient faces arbitrarily. $d = 2$: flow; $d = 3$: circulation; etc.
- conservative flow
  - $d = 2$: chips do not accumulate or deplete at any vertex;

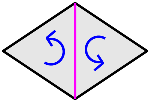


What does it look like?

$$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1}) \subseteq \mathbb{Z}^m$$

- Put integers on $(d-1)$-faces of Δ. Orient faces arbitrarily. $d = 2$: flow; $d = 3$: circulation; etc.
- conservative flow
  - $d = 2$: chips do not accumulate or deplete at any vertex;
  - $d = 3$: face circulation at each edge adds to zero.

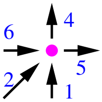


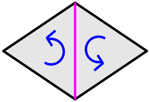


What does it look like?

$$K(\Delta) := (\ker \partial_{d-1})/(\mathrm{im}\, L_{d-1}) \subseteq \mathbb{Z}^m$$

- Put integers on $(d-1)$-faces of Δ. Orient faces arbitrarily. $d = 2$: flow; $d = 3$: circulation; etc.
- conservative flow
  - $d = 2$: chips do not accumulate or deplete at any vertex;
  - $d = 3$: face circulation at each edge adds to zero.
- By theorem, just specify values off the spanning tree.

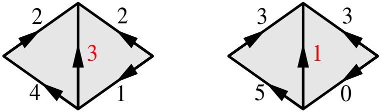




Firing faces

$$K(\Delta) := (\ker \partial_{d-1})/(\mathrm{im}\, L_{d-1}) \subseteq \mathbb{Z}^m$$

Toppling/firing moves the flow to "neighboring" $(d-1)$-faces, across d-faces.



Open problem: Critical configurations?

- What are the critical configurations?

Open problem: Critical configurations?

- What are the critical configurations?
  - i.e., canonical set of representatives

Open problem: Critical configurations?

- What are the critical configurations?
  - i.e., canonical set of representatives
- We could pick any set of representatives; by definition, there is some sequence of firings taking any configuration to the representative.

Open problem: Critical configurations?

- What are the critical configurations?
  - i.e., canonical set of representatives
- We could pick any set of representatives; by definition, there is some sequence of firings taking any configuration to the representative.
- But this misses the sense of "critical".

Open problem: Critical configurations?

- What are the critical configurations?
  - i.e., canonical set of representatives
- We could pick any set of representatives; by definition, there is some sequence of firings taking any configuration to the representative.
- But this misses the sense of "critical".
- Main obstacle is idea of what is "positive".

Example: Spheres

Theorem

If Δ is a sphere, with n facets, then $K(\Delta) \cong \mathbb{Z}_n$.

Example: Spheres

Theorem

If Δ is a sphere, with n facets, then $K(\Delta) \cong \mathbb{Z}_n$.

$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1})$

Proof.

- $K(\Delta)$ is generated by boundaries of facets ∂F.

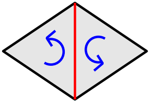
$\square$

Example: Spheres

Theorem

If Δ is a sphere, with n facets, then $K(\Delta) \cong \mathbb{Z}_n$.

$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1})$



Proof.

- $K(\Delta)$ is generated by boundaries of facets ∂F.
- In a sphere, the Laplacian of a ridge shows if facets F, G are adjacent, then $\partial F \equiv \pm\partial G \pmod{\operatorname{im} L}$.

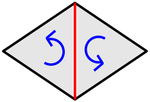
□

Example: Spheres

Theorem

If Δ is a sphere, with n facets, then $K(\Delta) \cong \mathbb{Z}_n$.

$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1})$



Proof.

- $K(\Delta)$ is generated by boundaries of facets ∂F.
- In a sphere, the Laplacian of a ridge shows if facets F, G are adjacent, then $\partial F \equiv \pm \partial G \pmod{\operatorname{im} L}$.
- So $K(\Delta)$ has a single generator, so it is cyclic.

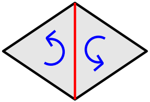
$\square$

Example: Spheres

Theorem

If Δ is a sphere, with n facets, then $K(\Delta) \cong \mathbb{Z}_n$.

$K(\Delta) := (\ker \partial_{d-1})/(\operatorname{im} L_{d-1})$



Proof.

- $K(\Delta)$ is generated by boundaries of facets ∂F.
- In a sphere, the Laplacian of a ridge shows if facets F, G are adjacent, then $\partial F \equiv \pm \partial G \pmod{\operatorname{im} L}$.
- So $K(\Delta)$ has a single generator, so it is cyclic.
- $|K(\Delta)|$ is the number of spanning trees, and there is one tree for every facet (remove that facet for the tree).

□

Riemann-Roch Theorem for Graphs

Baker-Norine ('07)

Algebraic geometry	**Graph theory**
Riemann surface	Graph
Divisor	Chip configuration
Linear equivalence (i.e., equivalence mod principal divisors)	Sequence of chip-firing moves
Picard group	Critical group

and now simplicial complexes?

Algebraic geometry	**Simplicial complexes**
Variety	Simplicial complex
Algebraic cycle	Simplicial (co)chain
Rational equivalence (i.e., equiv. mod principal divisors)	Flow redistribution (i.e., equiv. mod Laplacians)
Chow group	Simplicial critical group
Chow ring	?????

In other words, how can we define a multiplication on elements (equivalence classes) of the simplicial critical group?