Representing Edge Flows via Sparse Cell Complexes

Josef Hoppe and Michael T. Schaub

Learning on Graphs – November 29, 2023

RWTH Aachen University

Funded by the European Union
European Research Council
Established by the European Commission
Computational Network Science

Motivation

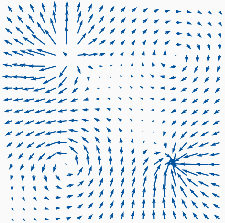
Motivation

Motivation

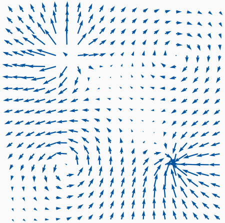
Motivation

Motivation

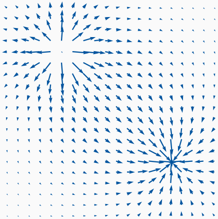
Intuition – Interpreting complicated Data



Intuition – Interpreting complicated Data

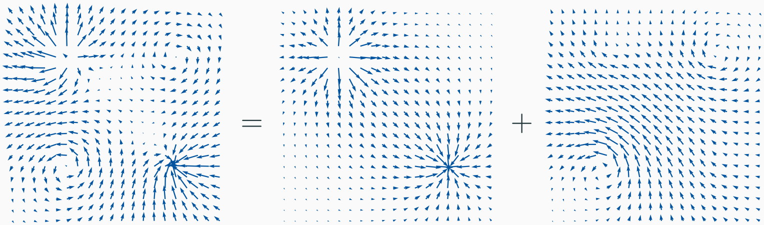


$=$

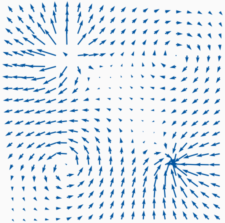


$+$

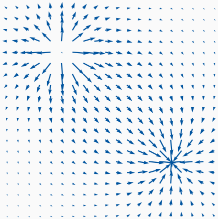
Intuition – Interpreting complicated Data



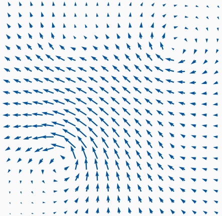
Intuition – Interpreting complicated Data

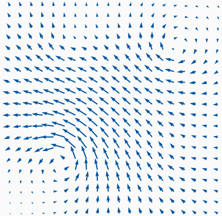


=



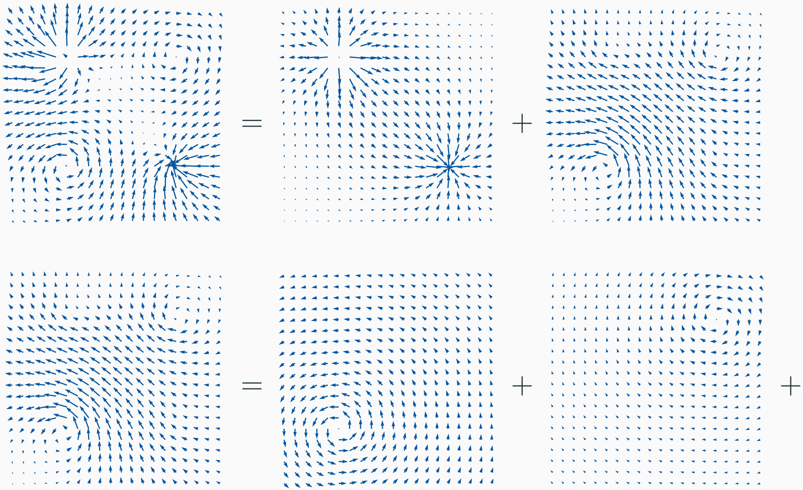
+



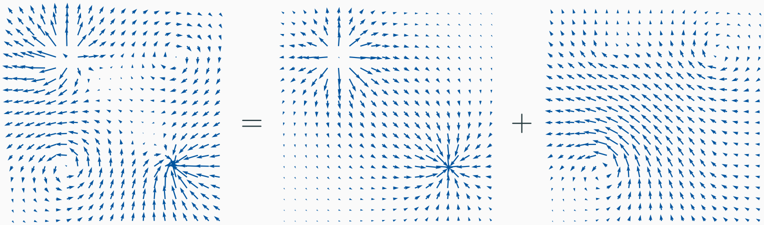


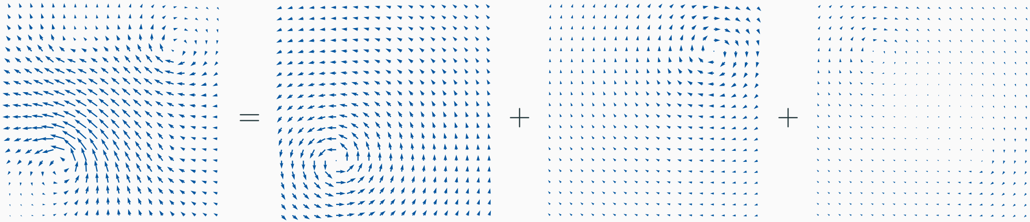
=

Intuition – Interpreting complicated Data

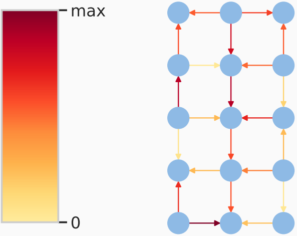


Intuition – Interpreting complicated Data



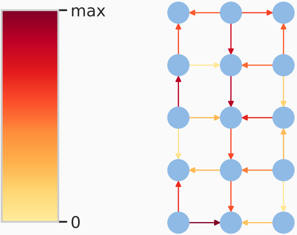


Topological Signal Processing: The Hodge Decomposition



edge flows

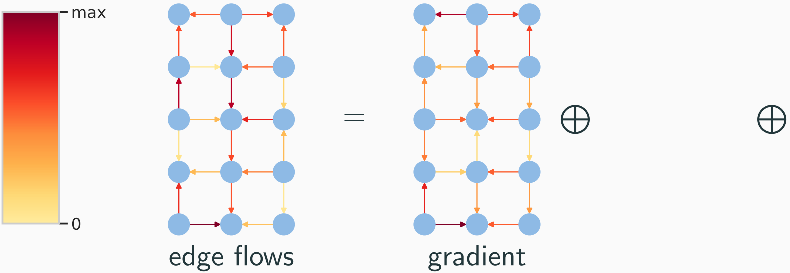
Topological Signal Processing: The Hodge Decomposition



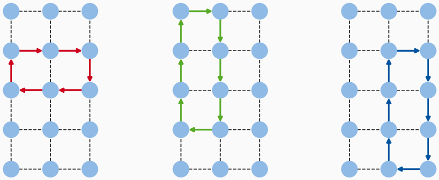
edge flows

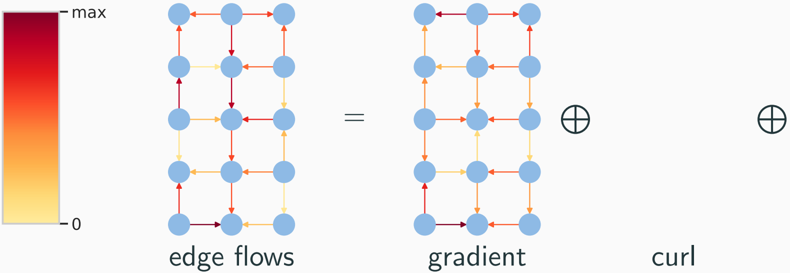
$$= \qquad \oplus \qquad \oplus$$

Topological Signal Processing: The Hodge Decomposition

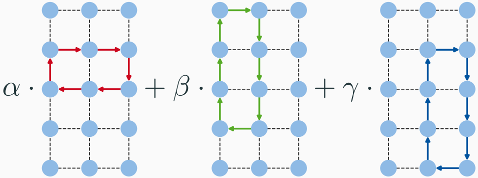


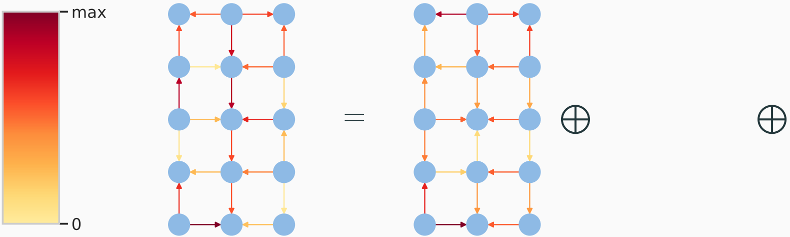
Topological Signal Processing: The Hodge Decomposition





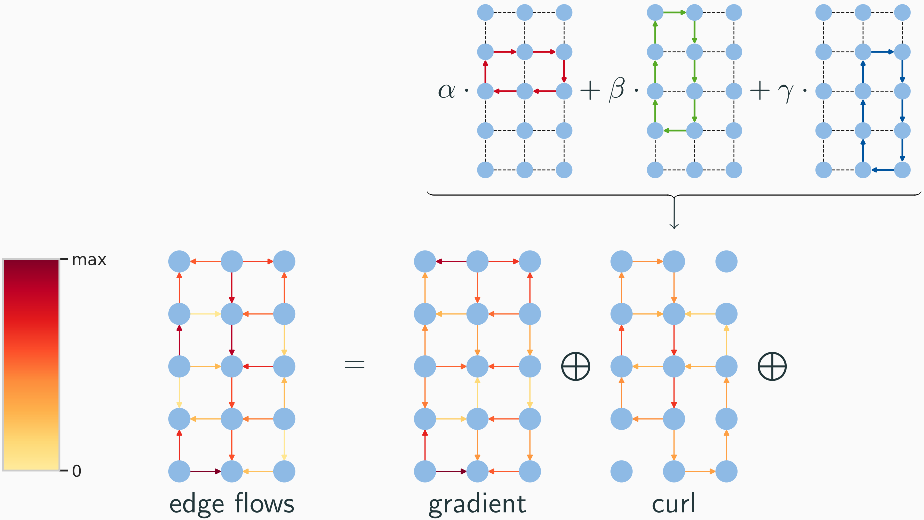
Topological Signal Processing: The Hodge Decomposition



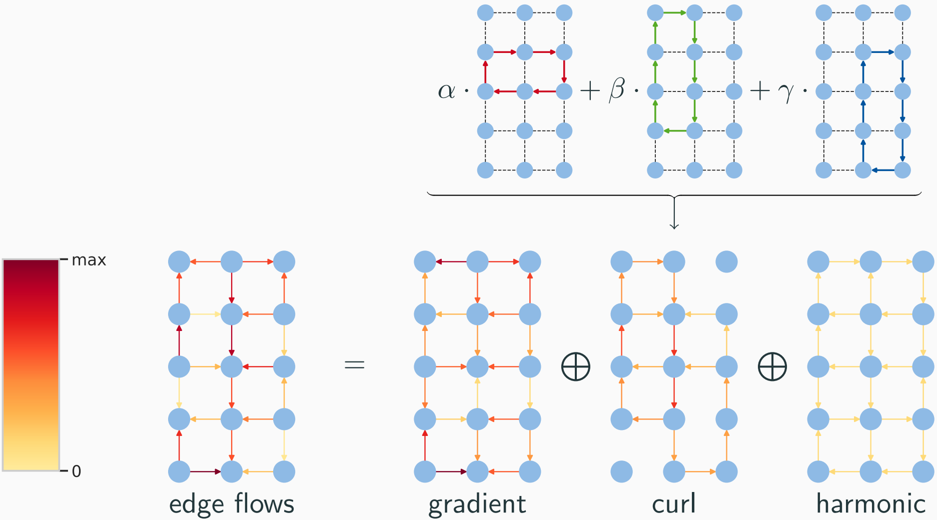


edge flows gradient curl

Topological Signal Processing: The Hodge Decomposition



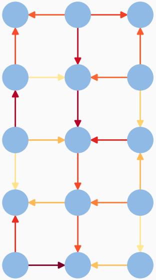
Topological Signal Processing: The Hodge Decomposition



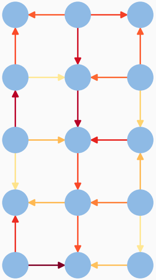
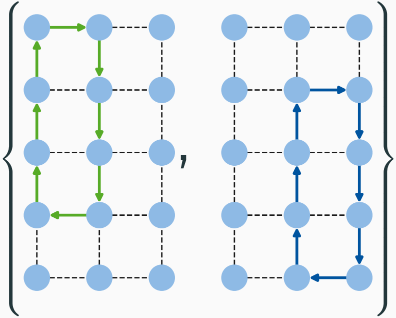
Problem Definition

Given

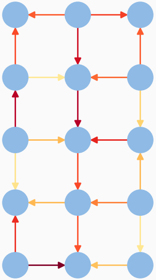
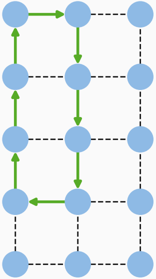
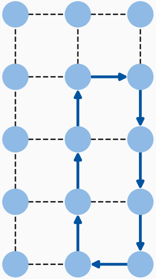
Problem Definition

Given edge flows ,

Problem Definition

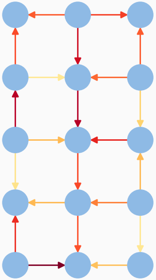
Given edge flows , find 2-cells 

Problem Definition

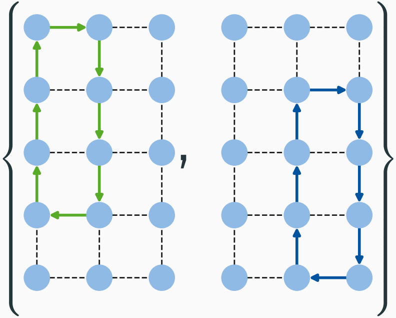
Given edge flows , find 2-cells { ,  } (cycles)

Problem Definition

Given edge flows



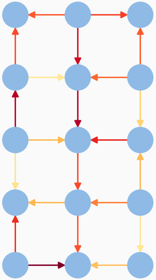
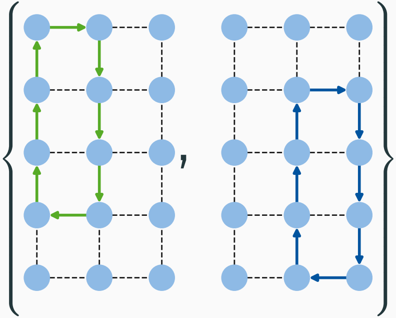
, find 2-cells



(cycles)

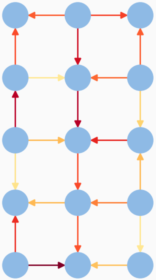
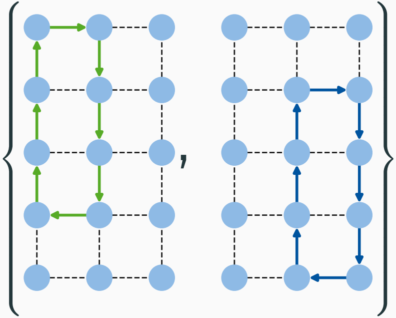
that minimize the loss

Problem Definition

Given edge flows , find 2-cells  (cycles)

that minimize the loss (= least squares of harmonic flow)

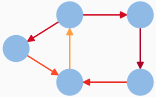
Problem Definition

Given edge flows , find 2-cells  (cycles)

that minimize the loss (= least squares of harmonic flow)

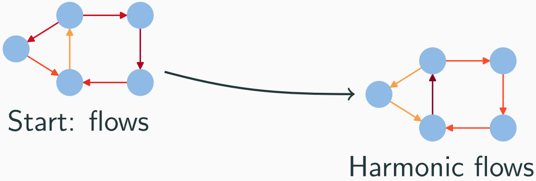
Note: The Decision Variant is **NP-Hard**.

Overview of the Algorithm



Start: flows

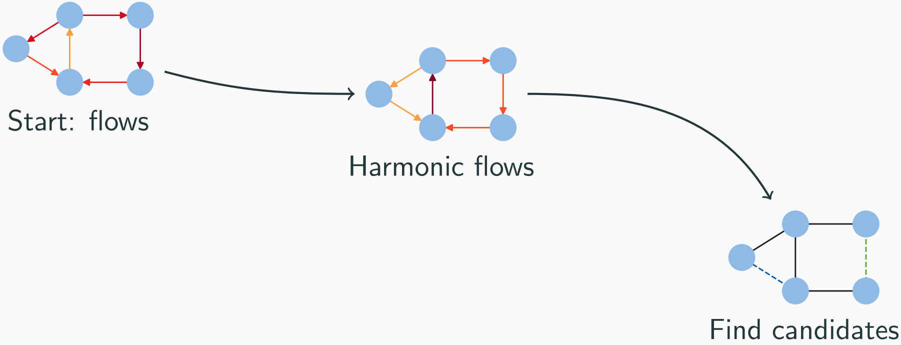
Overview of the Algorithm



Start: flows

Harmonic flows

Overview of the Algorithm

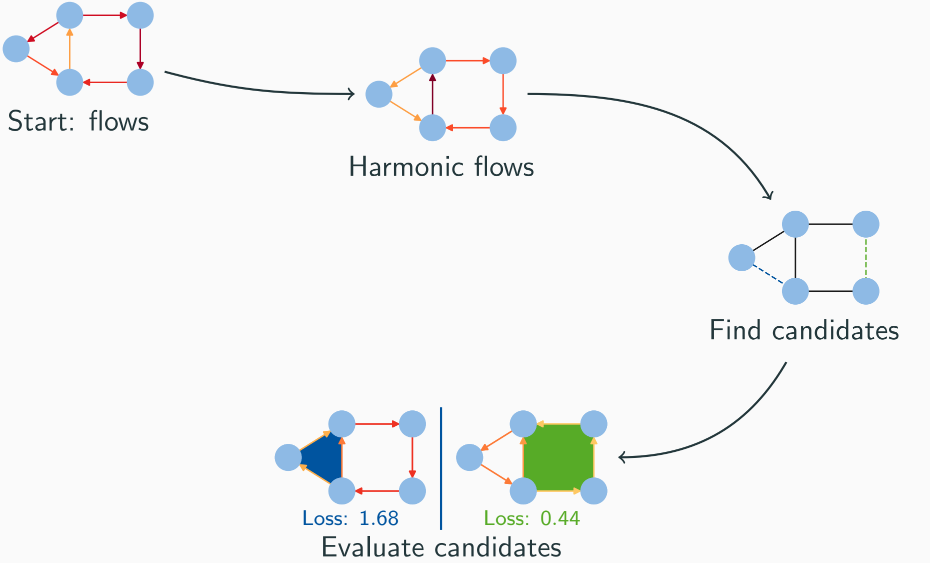


Start: flows

Harmonic flows

Find candidates

Overview of the Algorithm



Start: flows

Harmonic flows

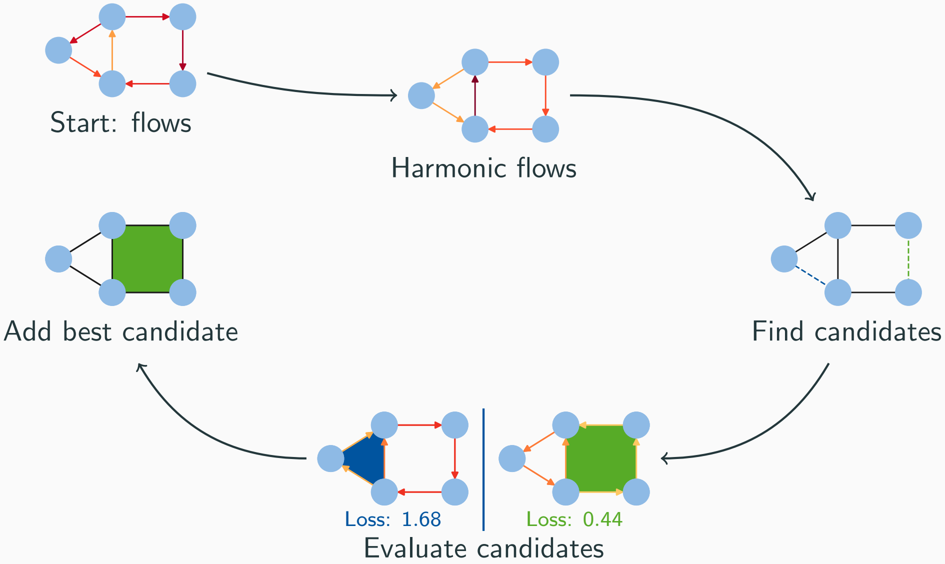
Find candidates

Loss: 1.68

Loss: 0.44

Evaluate candidates

Overview of the Algorithm



Start: flows

Harmonic flows

Find candidates

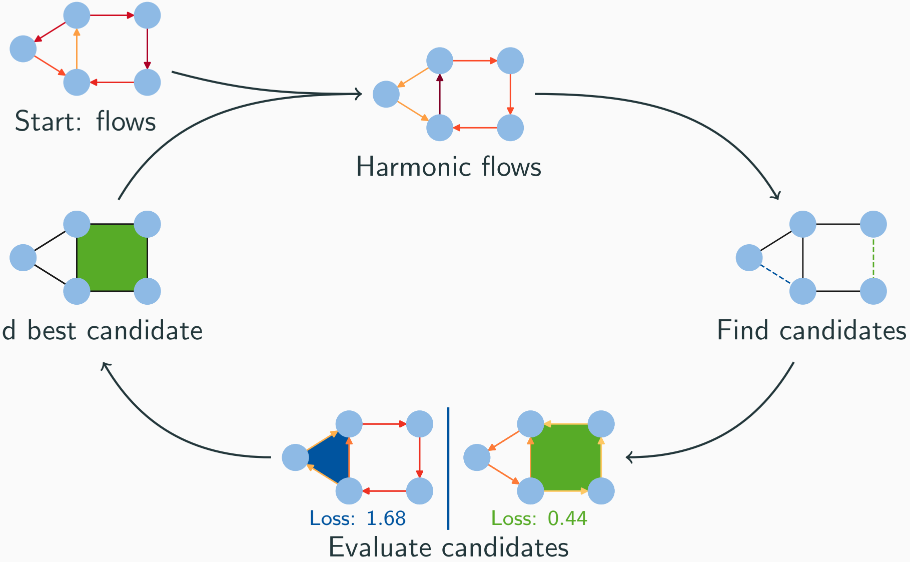
Evaluate candidates

Loss: 1.68

Loss: 0.44

Add best candidate

Overview of the Algorithm



Start: flows

Harmonic flows

Find candidates

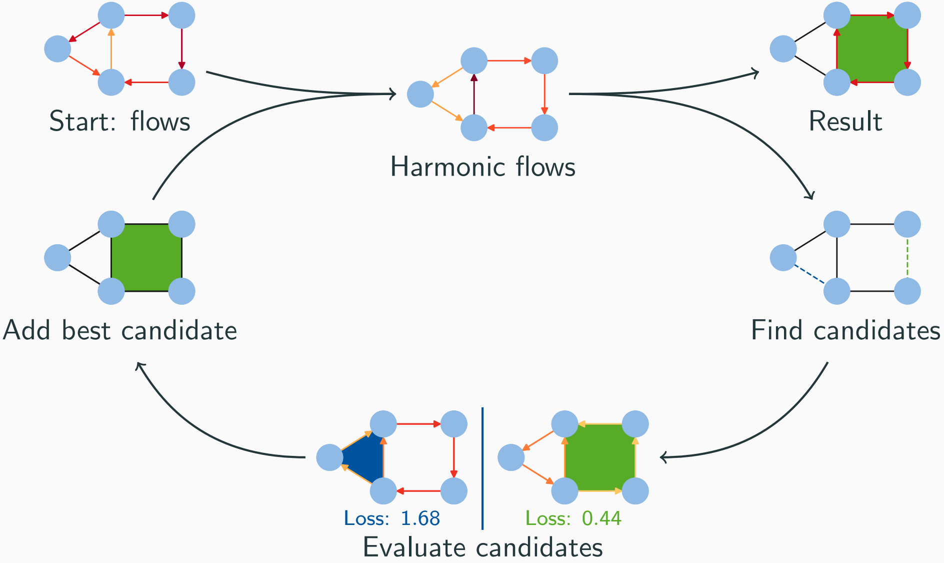
Loss: 1.68

Loss: 0.44

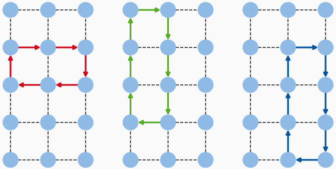
Evaluate candidates

Add best candidate

Overview of the Algorithm

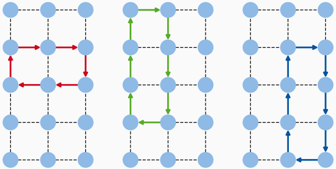


How can we get cell candidates?

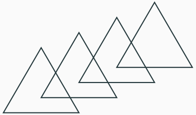


predefined

How can we get cell candidates?

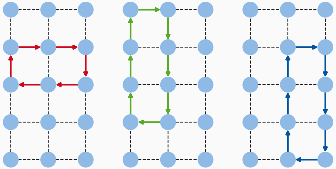


predefined

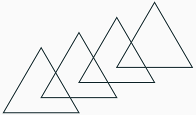


all triangles

How can we get cell candidates?

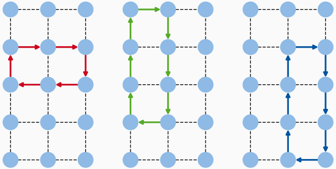


predefined

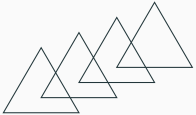


all triangles
(Simplicial Complex)

How can we get cell candidates?

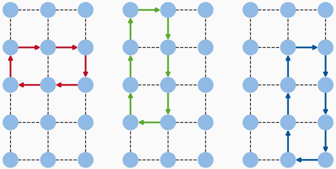


predefined

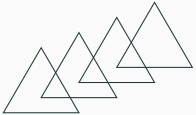


all triangles
(Simplicial Complex)

How can we get cell candidates?



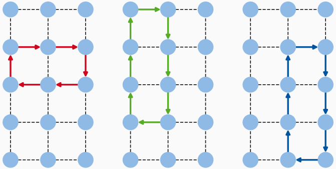
predefined



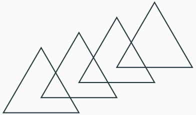
all triangles
(Simplicial Complex)

all cycles?

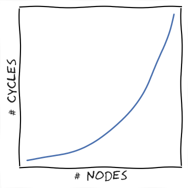
How can we get cell candidates?



predefined



all triangles
(Simplicial Complex)



all cycles?

Towards a heuristic for cell candidates

- Needs to consider “all” of the harmonic space

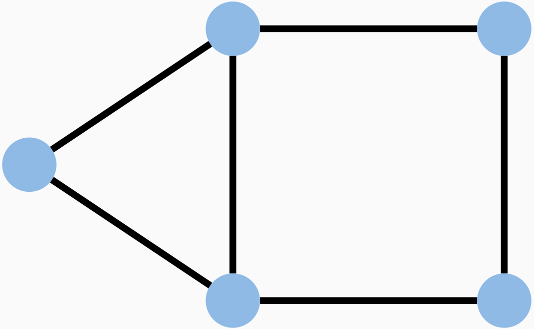
Towards a heuristic for cell candidates

- Needs to consider “all” of the harmonic space
- Exponentially many cycles $\Rightarrow$ Must not consider all cycles

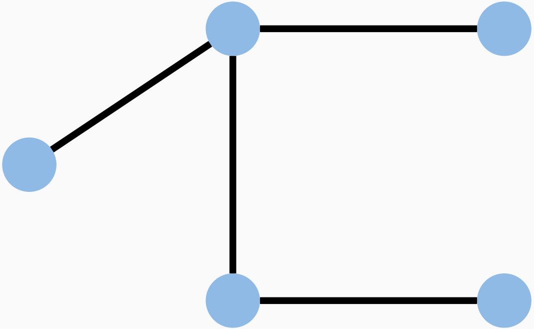
Towards a heuristic for cell candidates

- Needs to consider “all” of the harmonic space
- Exponentially many cycles ⇒ Must not consider all cycles
- Good start: Cycle basis
    - Spans entire harmonic space
    - Size linear in number of edges
    - Sparsity: Requires “good” cycles

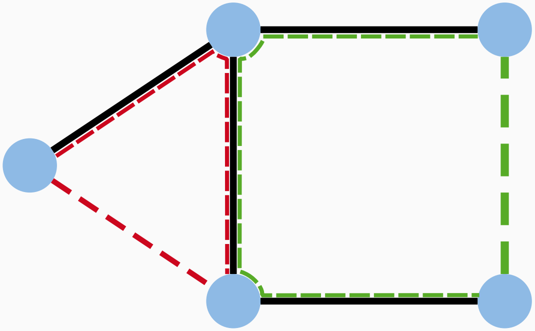
Spanning Trees induce Cycle Bases



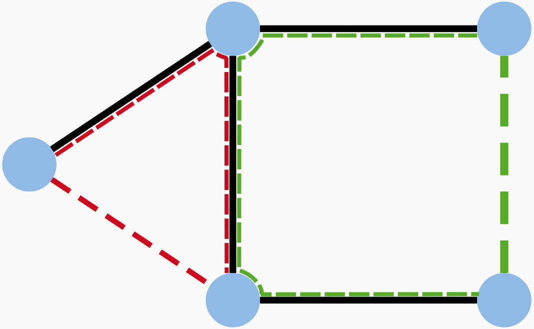
Spanning Trees induce Cycle Bases



Spanning Trees induce Cycle Bases

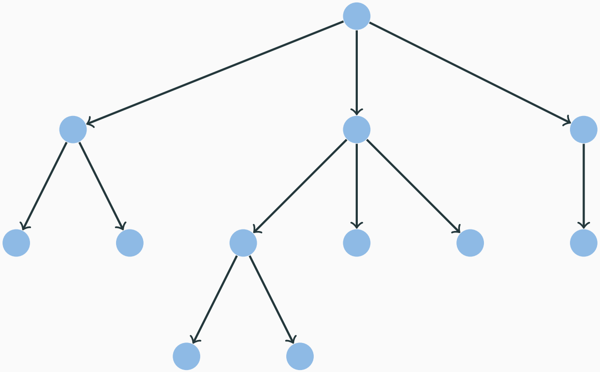


Spanning Trees induce Cycle Bases

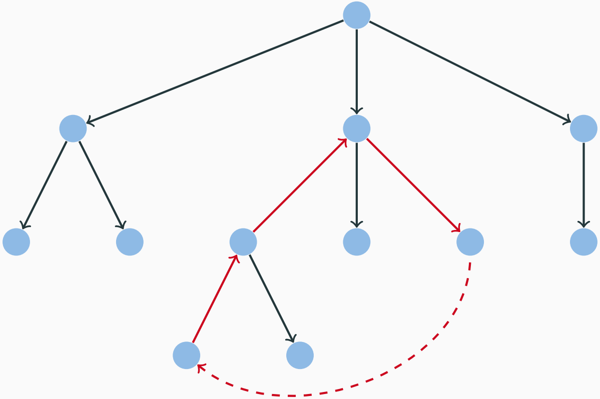


Complexity: exp(nodes) → nodes · edges

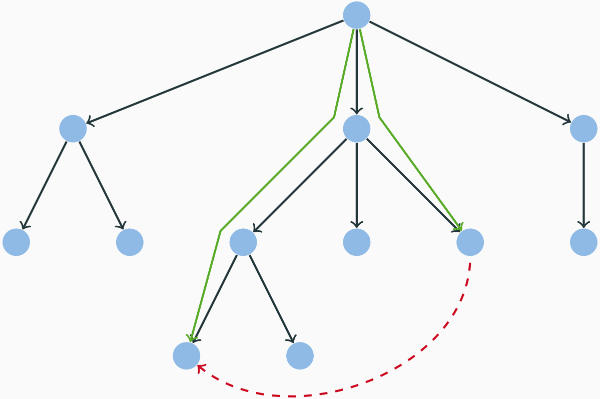
Utilizing Spanning Trees for efficiency



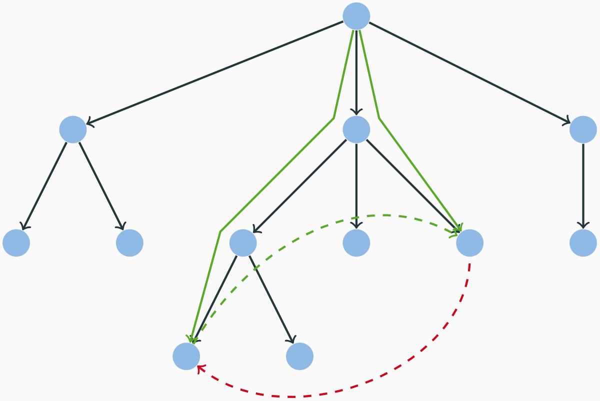
Utilizing Spanning Trees for efficiency



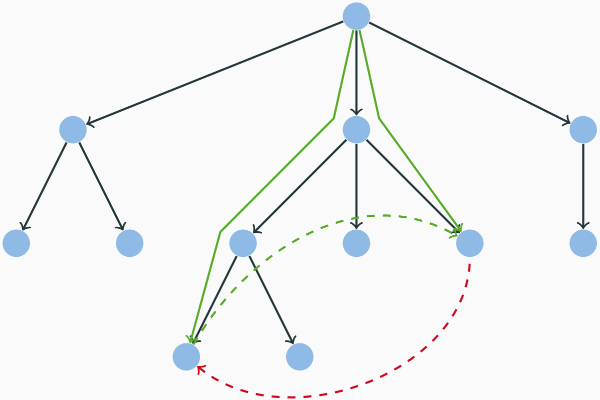
Utilizing Spanning Trees for efficiency



Utilizing Spanning Trees for efficiency

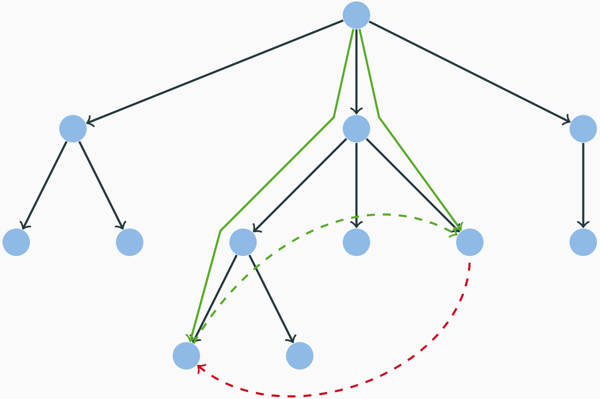


Utilizing Spanning Trees for efficiency



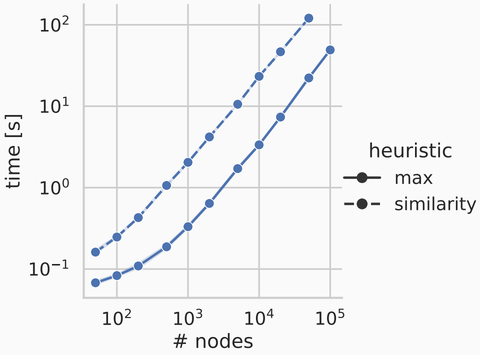
Complexity: nodes · edges → ∼edges + finding a spanning tree

Utilizing Spanning Trees for efficiency



Complexity: nodes · edges → ∼ edges + finding a spanning tree (edges log edges)

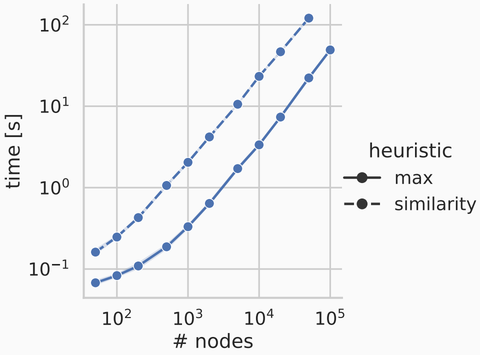
Results



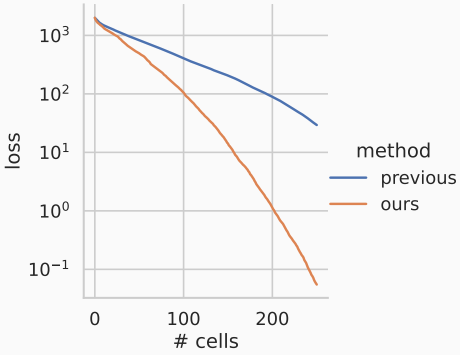
Runtime on triangulation graph

[1] Transportation Networks for Research. `https://github.com/bstabler/TransportationNetworks`. Accessed 2023-08-18.

Results



Runtime on triangulation graph



Traffic flow in Anaheim[1]

[1] Transportation Networks for Research. `https://github.com/bstabler/TransportationNetworks`. Accessed 2023-08-18.

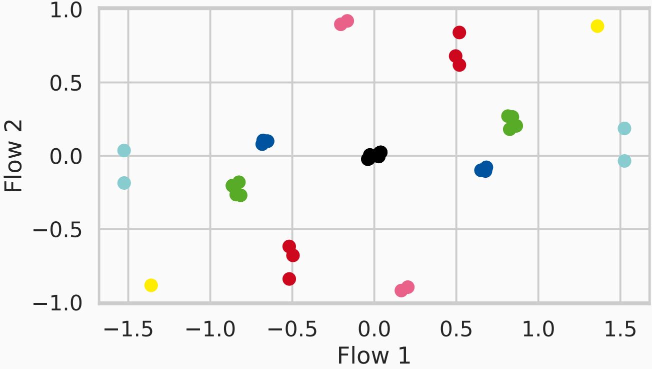
Conclusion

- Generalization of methodology for Topological Signal Processing
  - Cells: triangles → polygons
  - Candidates: pre-selected → heuristic selection
- Efficient approximation algorithm for NP-hard problem
  - Iterative greedy algorithm
  - Utilizing spanning trees for efficiency
  - Worked well in empirical experiments
- Future work:
  - Other downstream tasks / Adaptation to other tasks
  - Qualitative evaluation of inferred cells

Full Paper &
PyPI Package

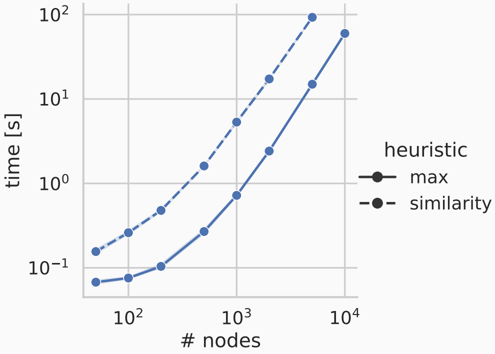
Heuristic: Finding Spanning Trees

1. Maximum spanning tree (by absolute flow)
2. Spanning tree of *similar* edges

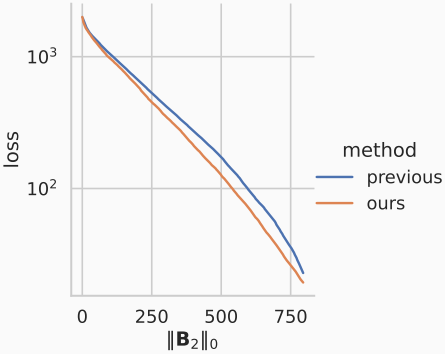


Edges clustered by flow value

Additional Plots



Runtime on smallworld graph



Traffic flow in Anaheim