Linear Algebraic Primitives Beyond Laplacian Solvers

FOCS 2018 Workshop
Laplacian Paradigm 2.0

Stanford | ENGINEERING
Management Science & Engineering

Aaron Sidford

Contact Info:

- email: *sidford@stanford.edu*
- website: *www.aaronsidford.com*

An algorithmic revolution over the last decade …

"The Laplacian Paradigm"

Can obtain faster graph algorithms

Spectral Graph Theory

Can solve $\mathcal{L}x = b$ in nearly linear time [ST04, …]

Combinatorial Object

Undirected Graph

Linear Algebraic Object

Laplacian Matrix

$\{i, j\} \in E \Leftrightarrow \mathcal{L}_{ij} = -w_{ji}$

$G = (V, E, w)$

- n vertices V
- m edges E
- $w_e \geq 0$ weight of edge E

Natural bijection

$(\mathcal{L}(G) = \boldsymbol{D}(G) - \boldsymbol{A}(G))$

$\mathcal{L} \in \mathbb{R}^{n \times n}$

- $\mathcal{L} = \mathcal{L}^{\top}$
- $\mathcal{L}_{ij} \leq 0$ for $i \neq j$
- $\mathcal{L}_{ii} = \sum_{j \neq i} -\mathcal{L}_{ji}$

Very successful paradigm and many faster algorithms for undirected graph problems.

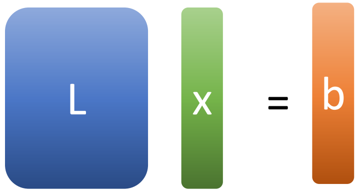
Laplacian System Solving

Beyond applications of solvers, broader applications of solving machinery.

<u>Linear Algebraic Problem</u>

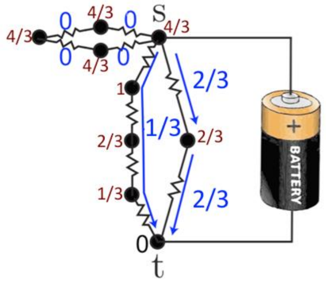
Solve $\mathcal{L}x = b$

- $\mathcal{L} = \mathcal{L}^\top, \mathcal{L}\vec{1} = \vec{0}$
- $\mathcal{L}_{ij} \leq 0,\ i \neq j$



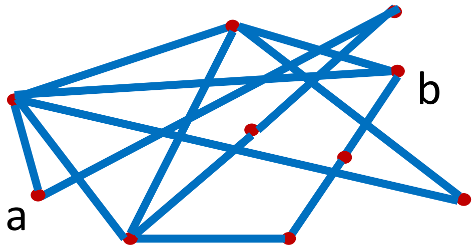
<u>Combinatorial Problem</u>

Compute electric current in a resistor network.



<u>Random Walk Problem</u>

For all vertices v the probability random walk on undirected graph gets to a before b.



<u>Continuous Optimization</u>

Iterative methods that converge to answer (maybe slowly)

(*e.g. gradient descent*)

<u>**Idea**</u>

- Couple them together
- Improve each

<u>Combinatorial Optimization</u>

Graph decompositions to decreasing iteration costs and speeding up convergence.

(e.g. trees, spanners)

Applications*: maximum flow, multicommodity flow, matrix scaling, sampling random spanning trees, graph sparsification, graph partitioning, graph routing, lossy flow, and more.*

Beyond Laplacian Systems?

Frontier
Directed Graphs?
Asymmetric Matrices?

- **Symmetric diagonally dominant (SDD) systems**
  - $\boldsymbol{A} = \boldsymbol{A}^\top$ where $\boldsymbol{A}_{ii} \geq \sum_{j \neq i} |\boldsymbol{A}_{ij}|$
  - Nearly linear time by direct reduction to Laplacians

- **Block diagonally dominant (BDD) Systems**
  - $\boldsymbol{A} = \boldsymbol{A}^\top$ where $\lambda_{min}(\boldsymbol{A}_{II}) \geq \sum_{J \neq I} \left\| \boldsymbol{A}_{IJ} \right\|_2$
  - Nearly linear time solver when block sizes are constant [KLPSS16]

- **Factored Factor Width 2 Matrices**
  - Given $\boldsymbol{C} \in \mathbb{R}^{m \times n}$ where every row of $\boldsymbol{C}$ has at most two non-zero entries
  - Can solve $(\boldsymbol{C}^\top \boldsymbol{C})x = b$ in nearly linear time [DS08]
  - Applications to lossy flow problems

Directed Graphs?

Nearly linear time algorithms have been more elusive….

Electric Flow View

Can find minimum norm projection onto subspace of circulations in a graph and use in interior point methods.

- Unit capacity directed maximum flow [M13,M17]
- Dense directed minimum cost [L**S**14]
- Shortest path with negative costs [CMSV16]

Undirected Enough

If directed graph is undirected in some way, can get fast algorithms.

- Approximate max flow on "balanced graphs" in nearly linear time [EMP**S**16] + [P16]

Maximum Flow Running Time on Unit Capacitated Graphs

- $O(\min\{m^{3/2}, mn^{2/3}\})$ [ET75,K73]
- $O(m^{10/7})$ [M13]
- $\tilde{O}(m\sqrt{n})$ [L**S**14]

Inherent Barriers for Directed Graphs

- *Don't always have sparse cut sparsifiers*
- *Don't always have sparse spanners*
- *Low radius decompositions don't always exist*

New Linear Algebraic Primitives?

Is there a directed primitive missing from our toolkit?

<u>Directed Spectral Graph Theory</u>

- Directed cheeger inequality [C05]
- Directed local partitioning [ACL07]

Are there nearly linear time linear algebraic primitives for directed graphs / asymmetric matrices?

What is the right notion of a directed Laplacian system?

“The Laplacian Paradigm”

Combinatorial Object

Undirected Graph

$G = (V, E, w)$

- n vertices V
- m edges E
- $w_e \geq 0$ weight of edge E

$\longleftrightarrow$

$\{i, j\} \in E \Leftrightarrow \boldsymbol{\mathcal{L}}_{\boldsymbol{ij}} = -w_{ji}$

Natural bijection

$(\boldsymbol{\mathcal{L}}(G) = \boldsymbol{D}(G) - \boldsymbol{A}(G))$

Linear Algebraic Object

Laplacian Matrix

$\boldsymbol{\mathcal{L}} \in \mathbb{R}^{n \times n}$

- $\boldsymbol{\mathcal{L}} = \boldsymbol{\mathcal{L}}^\top$
- $\boldsymbol{\mathcal{L}}_{\boldsymbol{ij}} \leq 0$ for $i \neq j$
- $\boldsymbol{\mathcal{L}}_{ii} = \sum_{j \neq i} -\boldsymbol{\mathcal{L}}_{\boldsymbol{ji}}$

Directed?

"The Laplacian Paradigm"

Combinatorial Object

~~Un~~Directed Graph

$\{i,j\} \in E \Leftrightarrow \boldsymbol{\mathcal{L}_{ij}} = -w_{\mathrm{ji}}$

$G = (V, E, w)$

- n vertices V
- m edges E
- $w_e \geq 0$ weight of edge e

Natural bijection

$(\boldsymbol{\mathcal{L}}(G) = \boldsymbol{D}_{out}(G) - \boldsymbol{A}^{\top}(G))$

Linear Algebraic Object

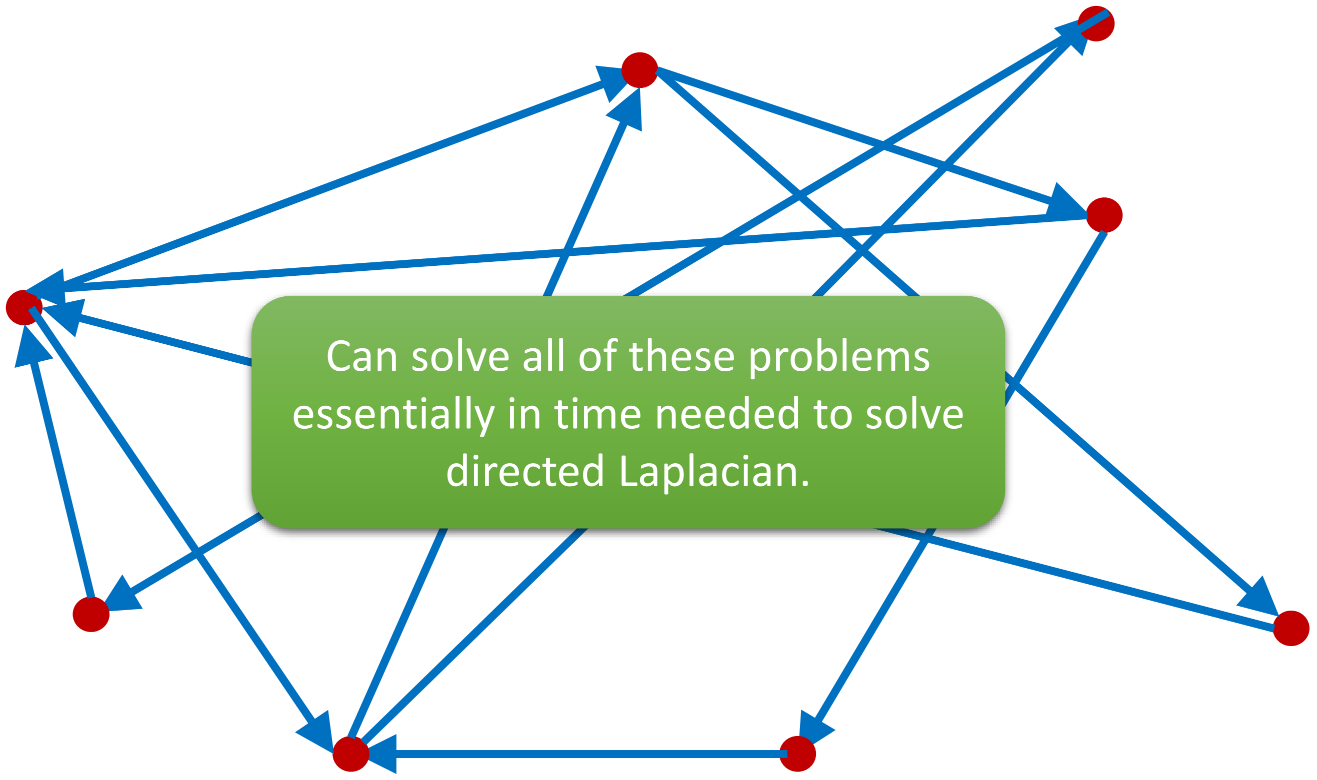
Laplacian Matrix

$\boldsymbol{\mathcal{L}} \in \mathbb{R}^{V \times V}$

- ~~$\boldsymbol{\mathcal{L}} = \boldsymbol{\mathcal{L}}^{\top}$~~
- $\boldsymbol{\mathcal{L}_{ij}} \leq 0$ for $i \neq j$
- $\boldsymbol{\mathcal{L}_{ii}} = \sum_{j \neq i} -\boldsymbol{\mathcal{L}_{ji}}$

Is this actually meaningful?

Directed Graph Problems



Random Walk Model

Pick a random outgoing edge proportional to weight, follow edge, repeat.

<u>Natural Problems</u>

- **Stationary distribution**: limit distribution of random walk.
- **Escape probabilities**: probability random from a gets to b before c.
- **Commute times**: expected amount of time random walk takes to go from a to b.
- **MDP Policy Evaluation**: each state yields some reward and want computed expected average reward per step.

Faster Algorithms for Computing the Stationary Distribution, Simulating Random Walks, and More (FOCS 2016)

(Michael B. Cohen, Jonathan A. Kelner, John Peebles, Richard Peng, Aaron Sidford, Adrian Vladu)

Almost-Linear-Time Algorithms for Markov Chains and New Spectral Primitives for Directed Graphs (STOC 2016)

(Michael B. Cohen, Jonathan A. Kelner, John Peebles, Richard Peng, Anup B. Rao, Aaron Sidford, Adrian Vladu)

Solving Directed Laplacian Systems in Nearly-Linear Time through Sparse LU Factorizations (FOCS 2018)

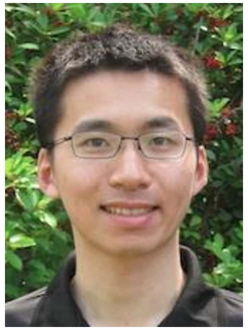
(Michael B. Cohen, Jonathan Kelner, Rasmus Kyng, John Peebles, Richard Peng, Anup Rao, and Aaron Sidford)


Michael B. Cohen


Jonathan Kelner


Rasmus Kyng


John Peebles


Richard Peng


Aaron Sidford


Adrian Vladu


Anup Rao

Solving Directed Laplacian?

Can solve directed Laplacian in time needed to solve Eulerian Laplacians [CKPP**S**V16]

Eulerian Laplacians

- $\boldsymbol{\mathcal{L}} \in \mathbb{R}^{V \times V}$
- $\boldsymbol{\mathcal{L}}_{ij} \leq 0$ for all $i \neq j$
- $\boldsymbol{\mathcal{L}}\vec{1} = \boldsymbol{\mathcal{L}}^{\top}\vec{1} = \vec{0}$

- Graph Connection
  - $\boldsymbol{\mathcal{L}} = \boldsymbol{D}(G) - \boldsymbol{A}(G)^{T}$
  - G is an Eulerian graph, i.e. in-degree = out degree
  - $\boldsymbol{D}(G) =$ degree matrix
  - $\boldsymbol{A}(G) =$ adjacency matrix

Runtime for Solving Eulerian Laplacians

- Naïve $O(n^{\omega})$ for $\omega < 2.373$ [W12]
- Faster algorithms than naïve for sparse systems [CKPP**S**V16]
- Sparsifiers and almost linear time algorithms [CKPPR**S**V16]
- Sparse approximate LU factorizationsand nearly linear time algorithms [CKKPPR**S**18, Tues!]

What else can we do with an Eulerian solver?

Directed Laplacians
- Properties of random walk on directed graph

A New Proof

Eulerian Laplacian Solver

Row Column Diagonally Dominant (RCDD Systems)
- $\boldsymbol{A}_{ii} \geq \sum_{j \neq i} |\boldsymbol{A}_{ji}|$ and $\boldsymbol{A}_{ii} \geq \sum_{j \neq i} |\boldsymbol{A}_{ij}|$
- Can solve in time needed to solve Eulerian Laplacians
- Analogous to SDD → Laplacian reductions

What else?

Perron-Frobenius Theory in Nearly Linear Time:
Positive Eigenvectors, M-matrices, Graph Kernels, and Other Applications
(arXiv, to appear in SODA 2019)

AmirMahdi
Ahmadinejad

Arun
Jambulapati

Amin
Saberi

Aaron
Sidford

Perron-Frobenius Theorem

Can "compute" $\rho(\boldsymbol{A})$,v_ℓ,v_r in Eulerian Laplacian solver time [AJSS18].

- Let $\boldsymbol{A} \in \mathbb{R}_{\geq 0}^{n \times n}$ be non-negative irreducible square matrix
  - (i.e. associated graph is strongly connected)
- Let $\rho(\boldsymbol{A}) = \max_i |\lambda_i(\boldsymbol{A})| = \lim_{k \to \infty} \left\|\boldsymbol{A}^k\right\|_2^{1/k}$ denote spectral radius of $\boldsymbol{A}$
- **Theorem**
  - $\rho(\boldsymbol{A})$ is an eigenvalue of $\boldsymbol{A}$
  - There exist unique left and right eigenvectors of eigenvalue $\rho(\boldsymbol{A})$
  - These eigenvectors, called *Perron vector* are all positive
  - $\exists v_\ell, v_r \in \mathbb{R}_{>0}^n$ such that $v_\ell^\top \boldsymbol{A} = \rho(\boldsymbol{A}) v_\ell^\top$ and $\boldsymbol{A} v_r = \rho(\boldsymbol{A}) v_r$

M-Matrices

Can solve check if a matrix is a M-matrix and solve linear systems in it in nearly Eulerian Laplacian solver time [AJS**S**18].

M-Matrices

- Prevalent class of matrices containing directed Laplacians
- Many characterizations
  - "A Z-matrix where the real part of every eigenvalue is positive"
  - A matrix of the form $\boldsymbol{M} = s\boldsymbol{I} - \boldsymbol{A}$ where $\boldsymbol{A} \in \mathbb{R}_{\geq 0}^{n \times n}$ with $\rho(\boldsymbol{A}) \leq s$.

Can check if $\sum_i |\boldsymbol{A}|^i$ converges or diverges.

$$\boldsymbol{M}^{-1} = \left(\frac{1}{s}\right) \sum_{i=0}^{\infty} \left(\frac{1}{s}\boldsymbol{A}\right)^i$$

For geometrically distributed random walk compute expected product of edge weights.

Applications

- **Directed Laplacian** related results are a special case
  - Stationary distribution is a Perron vector of random walk matrix
  - Directed Laplacians are M-matrices
- **Singular Vectors**
  - Can compute top left-right singular vectors of positive matrix in nearly linear time
- **Graph Measures**
  - Faster algorithms for graph kernels and Katz centrality
- **Factor Width Two**
  - Can checking if a matrix is factor width two (without the factorization) and solving it in nearly linear time.
- **Leontief economies**

Rest of Talk

Just a sketch; will hide lots of details.

Technical, but very short.

Proof Sketch

Computing Perron Vectors

Solving M-Matrices

Solving Eulerian Laplacians

For more on this, stay tuned to rest of workshop and conference.

RCDD Scaling $\Rightarrow$ M-Matrix Solver

- Let $\boldsymbol{M} = s\boldsymbol{I} - \boldsymbol{A}$ be an invertible M-matrix
- Let $v_\ell, v_r \in \mathbb{R}^n_{>0}$ be Perron vectors of $\boldsymbol{A}$
  - $v_\ell^\top \boldsymbol{A} = \rho(\boldsymbol{A}) v_\ell^\top$ and $\boldsymbol{A} v_r = \rho(\boldsymbol{A}) v_r$

- **Claim**: $\boldsymbol{LMR}$ is RCDD for $\boldsymbol{L} = diag(v_\ell)$ and $\boldsymbol{R} = diag(v_r)$

- **Proof**
  - $[\boldsymbol{LMR}]_{ij} \le 0$ for all $i \neq j$
  - $[\boldsymbol{LMR}]\vec{1} \ge \vec{0}$ and $\vec{1}^\top[\boldsymbol{LMR}] \ge \vec{0}^\top$

RCDD Scaling

Any positive diagonal $\boldsymbol{L}$ and $\boldsymbol{R}$ such that $\boldsymbol{LMR}$ is RCDD.

M-Matrix Solver ⇒ RCDD Scaling

RCDD Scaling
Any positive diagonal $\boldsymbol{L}$ and $\boldsymbol{R}$ such that $\boldsymbol{LMR}$ is RCDD.

- Let $\boldsymbol{M} = s\boldsymbol{I} - \boldsymbol{A}$ be an invertible M-matrix

- **Claim**: $r = \boldsymbol{M}^{-1}\vec{1}$ and $\ell = [\boldsymbol{M}^\top]^{-1}\vec{1}$ yield RCDD scalings
  - $\boldsymbol{R} = diag(r)$ and $\boldsymbol{L} = diag(\ell)$

- **Proof**
  - $\boldsymbol{M}^{-1} = \left(\frac{1}{s}\right)\sum_{i=0}^{\infty}\left(\frac{1}{s}\boldsymbol{A}\right)^i$ and therefore ℓ and r are positive
  - $[\boldsymbol{LMR}]\vec{1} = \ell \geq \vec{0}$ and $\vec{1}^\top[\boldsymbol{LMR}] = r^\top \geq \vec{0}^\top$

Chicken and Egg Problem
Given scaling can solve and given solver can scale.

This shows up over and over again [LMPS13,KLMM**S**14,…, CKPP**S**V16,…]

Solution: Regularization + Preconditioning

Regularization

- Let $\boldsymbol{M}_\alpha = \alpha\boldsymbol{I} + \boldsymbol{M}$
- If M M-Matrix so is M_α for $\alpha \geq 0$
- Easy to solve for large α
- Suffices to solve for small α

Preconditioning

- Suppose want to solve $\boldsymbol{A}x = b$
- Suppose can solve systems in $\boldsymbol{B}$
- Preconditioned Richardson
  - $x_{k+1} = x_k - \eta\boldsymbol{B}^{-1}[\boldsymbol{A}x_k - b]$
- Converges fast if $\boldsymbol{A} \approx \boldsymbol{B}$

Claim [AJSS18]

$M_\alpha \approx M_{\alpha/2}$

*(in appropriate norm)**

Algorithm

Note
This is hiding lots of precision issues. See paper for details.

Notation

- Let $M_\alpha = \alpha \boldsymbol{I} - \boldsymbol{M}$
- Let $r_\alpha = \boldsymbol{M}_\alpha^{-1} \vec{1}$
- Let $\ell_\alpha = [\boldsymbol{M}_\alpha^\top]^{-1} \vec{1}$

Algorithm

- Pick large $\alpha > 0$
- While α is too big
  - Use solver for α in preconditioned Richardson to compute $r_{\alpha/2}$ and $\ell_{\alpha/2}$
  - Use Eulerian solver to have solver for $\boldsymbol{L}_{\alpha/2} \boldsymbol{M}_{\alpha/2} \boldsymbol{R}_{\alpha/2}$ and therefore $\boldsymbol{M}_{\alpha/2}$

Note: if M is symmetric then symmetry is preserved (i.e. only need to use symmetric Laplacian solvers)

Rest of Talk

Proof Sketch

Computing Perron Vectors

Solving M-Matrices

Solving Eulerian Laplacians

For more on this, stay tuned to rest of workshop and conference.

Hopefully this is just the beginning

- “Directed Laplacian Paradigm”
  - We have new nearly linear time primitives for directed graphs and asymmetric matrices, can we use this to design faster algorithms for combinatorial problems? [e.g. MDPs]

- Broader classes of systems or hardness
  - For example, other Laplacian-like block structure [KZ17, KPSZ18]
  - For example, Laplacian inversion [MNSUW18]

- Complexity implications
  - For example, RL v.s. L [M**S**RV17]

- More practical algorithms and broader implications
  - Stick around

Thank you

Questions?

Contact Info:

- email: *sidford@stanford.edu*
- website: *www.aaronsidford.com*