Linear Algebraic Primitives Beyond Laplacian Solvers

FOCS 2018 Workshop
Laplacian Paradigm 2.0

Aaron Sidford

Contact Info:
• email: sidford@stanford.edu
• website: www.aaronsidford.com
“The Laplacian Paradigm”

An algorithmic revolution over the last decade ...

Can obtain faster graph algorithms

Spectral Graph Theory

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

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

Undirected Graph

$G = (V, E, w)$

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

Laplacian Matrix

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

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

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

Combinatorial Object

Linear Algebraic Object

Natural bijection

$(\mathcal{L}(G) = D(G) - A(G))$
Laplacian System Solving

Linear Algebraic Problem
Solve $\mathcal{L}x = b$
- $\mathcal{L} = \mathcal{L}^T, \mathcal{L}^1 = 0$
- $\mathcal{L}_{ij} \leq 0, i \neq j$

Combinatorial Problem
Compute electric current in a resistor network.

Continuous Optimization
Iterative methods that converge to answer (maybe slowly) (e.g. gradient descent)

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

Combinatorial Optimization
Graph decompositions to decreasing iteration costs and speeding up convergence.
(e.g. trees, spanners)

Idea
- Couple them together
- Improve each

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

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

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

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

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 [LS14]
- 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 [EMPS16] + [P16]

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

Nearly linear time algorithms have been more elusive....
New Linear Algebraic Primitives?  

Directed Spectral Graph Theory  
• 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?

Is there a directed primitive missing from our toolkit?
“The Laplacian Paradigm”

**Combinatorial Object**

- Undirected Graph

\[ G = (V, E, w) \]
- \( n \) vertices \( V \)
- \( m \) edges \( E \)
- \( w_e \geq 0 \) weight of edge \( E \)

**Linear Algebraic Object**

- Laplacian Matrix

\[ \mathcal{L} \in \mathbb{R}^{n \times n} \]
- \( \mathcal{L} = \mathcal{L}^T \)
- \( \mathcal{L}_{ij} \leq 0 \) for \( i \neq j \)
- \( \mathcal{L}_{ii} = \sum_{j \neq i} -\mathcal{L}_{ji} \)

\[ \{i, j\} \in E \iff \mathcal{L}_{ij} = -w_{ji} \]

Natural bijection

\[ (\mathcal{L}(G) = D(G) - A(G)) \]
**The Laplacian Paradigm**

Directed?

“The Laplacian Paradigm”

Combinatorial Object

- Undirected Graph
- \( G = (V, E, w) \)
  - \( n \) vertices \( V \)
  - \( m \) edges \( E \)
  - \( w_e \geq 0 \) weight of edge \( e \)

Linear Algebraic Object

- Laplacian Matrix
- \( \mathcal{L} \in \mathbb{R}^{V \times V} \)
  - \( \mathcal{L} = \mathcal{L}^\top \)
  - \( \mathcal{L}_{ij} \leq 0 \) for \( i \neq j \)
  - \( \mathcal{L}_{ii} = \sum_{j \neq i} -\mathcal{L}_{ji} \)

Natural bijection

\( (\mathcal{L}(G) = D_{out}(G) - A^\top(G)) \)

\( \{i, j\} \in E \iff \mathcal{L}_{ij} = -w_{ji} \)

Is this actually meaningful?
Directed Graph Problems

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

Natural Problems

- **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.

Can solve all of these problems essentially in time needed to solve directed Laplacian.
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)

(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?

Eulerian Laplacians

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

Graph Connection

- $\mathcal{L} = D(G) - A(G)^T$
- $G$ is an Eulerian graph, i.e. in-degree = out degree
- $D(G) = $ degree matrix
- $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 [CKPPSV16]
- Sparsifiers and almost linear time algorithms [CKPPRSV16]
- Sparse approximate LU factorizations and nearly linear time algorithms [CKPPRS18, Tues!]

Can solve directed Laplacian in time needed to solve Eulerian Laplacians [CKPPSV16]

We’ll get back to this
What else can we do with an Eulerian solver?

Directed Laplacians
- Properties of random walk on directed graph

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

A New Proof

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

What else?

AmirMahdi Ahmadinejad
Arun Jambulapati
Amin Saberi
Aaron Sidford
Perron-Frobenius Theorem

• Let $A \in \mathbb{R}^{n \times n}_{\geq 0}$ be non-negative irreducible square matrix
  • (i.e. associated graph is strongly connected)

• Let $\rho(A) = \max_i |\lambda_i(A)| = \lim_{k \to \infty} \|A^k\|_2^{1/k}$ denote spectral radius of $A$

• Theorem
  • $\rho(A)$ is an eigenvalue of $A$
  • There exist unique left and right eigenvectors of eigenvalue $\rho(A)$
  • These eigenvectors, called Perron vector are all positive
  • $\exists v_\ell, v_r \in \mathbb{R}^n_{>0}$ such that $v_\ell^\top A = \rho(A)v_\ell^\top$ and $Av_r = \rho(A)v_r$

Can “compute” $\rho(A), v_\ell, v_r$ in Eulerian Laplacian solver time [AJSS18].
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 $M = sI - A$ where $A \in \mathbb{R}_{\geq 0}^{n \times n}$ with $\rho(A) \leq s$.

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

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

Can solve check if a matrix is a M-matrix and solve linear systems in it in nearly Eulerian Laplacian solver time [AJSS18].

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

Proof Sketch

Computing Perron Vectors

Solving $M$-Matrices

Solving Eulerian Laplacians

Just a sketch; will hide lots of details.

Technical, but very short.

For more on this, stay tuned to rest of workshop and conference.
Let $M = sI - A$ be an invertible M-matrix
Let $v_\ell, v_r \in \mathbb{R}_>^n$ be Perron vectors of $A$
- $v_\ell^T A = \rho(A)v_\ell^T$ and $Av_r = \rho(A)v_r$

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

Proof
- $[LMR]_{ij} \leq 0$ for all $i \neq j$
- $[LMR]^\top \geq \vec{0}$ and $\vec{1}^T [LMR] \geq \vec{0}^T$

RCDD Scaling
Any positive diagonal $L$ and $R$ such that $LMR$ is RCDD.
M-Matrix Solver $\Rightarrow$ RCDD Scaling

• Let $M = sI - A$ be an invertible M-matrix

• Claim: $r = M^{-1}\mathbf{1}$ and $\ell = [M^T]^{-1}\mathbf{1}$ yield RCDD scalings
  • $R = \text{diag}(r)$ and $L = \text{diag}(\ell)$

• Proof
  • $M^{-1} = \left(\frac{1}{s}\right) \sum_{i=0}^{\infty} \left(\frac{1}{s} A\right)^i$ and therefore $\ell$ and $r$ are positive
  • $[LMR]\mathbf{1} = \ell \geq \mathbf{0}$ and $\mathbf{1}^T [LMR] = r^T \geq \mathbf{0}^T$

RCDD Scaling
Any positive diagonal $L$ and $R$ such that $LMR$ is RCDD.

Chicken and Egg Problem
Given scaling can solve and given solver can scale.
Solution: Regularization + Preconditioning

Regularization
• Let $M_\alpha = \alpha I + M$
• If $M$ M-Matrix so is $M_\alpha$ for $\alpha \geq 0$
• Easy to solve for large $\alpha$
• Suffices to solve for small $\alpha$

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

Claim [AJSS18]
$M_\alpha \approx \frac{M_\alpha}{2}$
(in appropriate norm)*

This shows up over and over again [LMPS13,KLMMS14,..., CKPPSV16,...]
Algorithm

Notation
• Let $M_{\alpha} = \alpha I - M$
• Let $r_{\alpha} = M_{\alpha}^{-1} \vec{1}$
• Let $\ell_{\alpha} = [M_{\alpha}^T]^{-1} \vec{1}$

Algorithm
• Pick large $\alpha > 0$
• While $\alpha$ is too big
  • Use solver for $\alpha$ in preconditioned Richardson to compute $r_{\alpha/2}$ and $\ell_{\alpha/2}$
  • Use Eulerian solver to have solver for $L_{\alpha/2} M_{\alpha/2} R_{\alpha/2}$ and therefore $M_{\alpha/2}$

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 [MSRV17]

• More practical algorithms and broader implications
  • Stick around
Thank you

Questions?

Contact Info:
• email: sidford@stanford.edu
• website: www.aaronsidford.com