Spectra of the graph.

$M = A/d$ adjacency matrix, A

Eigenvector: a vector v where $Mv = \lambda v$

Real, symmetric.

Claim:
Two eigenvectors with different eigenvalues are orthogonal.

Proof: Eigenvectors: v, v' with eigenvalues λ, λ'.
$v^T M v' = v^T(\lambda' v') = \lambda' v^T v'$
$v^T M v' = \lambda v^T v' = \lambda v^T v$. □

Distinct eigenvalues → orthonormal basis.

In basis: matrix is diagonal..

$$M = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix}$$

Action of M.

v - assigns weights to vertices.

Mv replaces v_i with $\frac{1}{d}\sum_{e=(i,j)} v_j$.

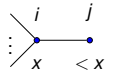
Eigenvector with highest value? $v = \mathbf{1}$. $\lambda_1 = 1$.

$\rightarrow v'_i = (M\mathbf{1})_i = \frac{1}{d}\sum_{e\in(i,j)} 1 = 1$.

Claim: For a connected graph $\lambda_2 < 1$.

Proof: Second Eigenvector: $v \perp \mathbf{1}$. Max value x.

Connected → path from x valued node to lower value.
$\rightarrow \exists\ e = (i,j)$, $v_i = x$, $v_j < x$.



$(Mv)_i \le \frac{1}{d}(x + x \cdots + v_j) < x$.

Therefore $\lambda_2 < 1$. □

Claim: Connected if $\lambda_2 < 1$.
Proof: Assign $+1$ to vertices in one component, $-\delta$ to rest.
$x_i = (Mx_i) \implies$ eigenvector with $\lambda = 1$.
Choose δ to make $\sum_i x_i = 0$, i.e., $x \perp \mathbf{1}$. □

Examples: Eigenvectors.

For graph laplacians: $L = dI - A$.

$dI - A$. Constant eigenvector, $\bar{\mathbf{1}}$ has eigenvalue 0.

$\frac{x^T Ax}{d|x|^2}$ is proxy for "eigenvalue".
$x^T(dI - A)x = d|x|^2 - x^T Ax$.
If different for different x, then λ_1/λ_n is large:
$Av_i = \lambda_i v_i \rightarrow \frac{v_i^T Av_i}{|v_i|^2} = \lambda_i$.

Expander Graph:
All $x \perp \bar{\mathbf{1}}$: $\sum_i x_i = 0$.
Degree d, think of random ± 1 labels.
$(Ax)_i =$ average value of labels.
Roughly $\pm\sqrt{d}$.
Expander: "no good cuts" "all cuts same".
$\implies$ every ± 1 vector is about the same!

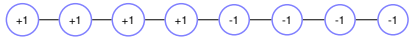
Fast convergence of iterative method.
Intuitively: fast communication across graph,
$\implies$ fast iterative algorithm.

Path (or Cycle)

Consider $\frac{x^T Ax}{d|x|^2}$.

Random ± 1, same argument as for expander.
$+1/-1$ edges change mass a lot.
$\implies \frac{x^T Ax}{d|x|^2} << 1$.

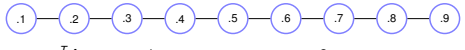
Good cut: $+1$ on left half, -1 on right half.
Lose mass only in middle.



$\implies \frac{x^T Ax}{d|x|^2} \approx 1 - \frac{1}{n}$.

$\implies \lambda_1$ and λ_n differ by a factor of n.

Worse example:



$\implies \frac{x^T Ax}{|x|^2} \approx 1 - \frac{1}{n^2}$. For Laplacian $1/n^2$.

Convergence is $O(n^2)$. Or $O(n)$.

An aside: Cheeger

Cut with few edges and good "balance".

Cut with few edges per unit of vertices "Cut off".

$$h(G) = \min_{S, |S| \le |V|/2} \frac{E(S, \bar{S})}{d|S|}$$

Expander graph: $h(G) = \Theta(1)$ and $1 - \lambda_2 = \Theta(1)$.

Cycle: $h(G) = \Theta(1/n)$. and $1 - \lambda_2 = \Theta(1/n^2)$.

Cheeger's inequality.

$$\frac{\mu}{2} = \frac{1-\lambda_2}{2} \le h(G) \le \sqrt{2(1-\lambda_2)} = \sqrt{2\mu}$$

Cycle

Tight example for Other side of Cheeger?

$\frac{\mu}{2} = \frac{1-\lambda_2}{2} \le h(G) \le \sqrt{2(1-\lambda_2)} = \sqrt{2\mu}$

Cycle on n nodes.

Will show other side of Cheeger is tight.

Edge expansion:Cut in half.
$|S| = n/2$, $|E(S, \bar{S})| = 2$
$\rightarrow h(G) = \frac{2}{n}$.

Show eigenvalue gap $\mu \le \frac{1}{n^2}$.

Find $x \perp \mathbf{1}$ with Rayleigh quotient, $\frac{x^T Mx}{x^T x}$ close to 1.

Slow vector.

Find $x \perp \mathbf{1}$ with Rayleigh quotient, $\frac{x^T M x}{x^T x}$ close to 1.

$$x_i = \begin{cases} i - n/4 & \text{if } i \le n/2 \\ 3n/4 - i & \text{if } i > n/2 \end{cases}$$

Hit with M.

$$(Mx)_i = \begin{cases} -n/4 + 1/2 & \text{if } i = 1, n \\ n/4 - 1 & \text{if } i = n/2 \\ x_i & \text{otherwise} \end{cases}$$

$\to x^T M x = x^T x(1 - O(\frac{1}{n^2})) \quad \to \lambda_2 \ge 1 - O(\frac{1}{n^2})$

$\mu = \lambda_1 - \lambda_2 = O(\frac{1}{n^2})$

$h(G) = \frac{2}{n} = \Theta(\sqrt{\mu})$

$\frac{\mu}{2} = \frac{1-\lambda_2}{2} \le h(G) \le \sqrt{2(1-\lambda_2)} = \sqrt{2\mu}$

Tight example for upper bound for Cheeger.

Eigenvalues of cycle?

Eigenvalues: $\cos \frac{2\pi k}{n}$.

$x_i = \cos \frac{2\pi k i}{n}$

$(Mx)_i = \cos\left(\frac{2\pi k(i+1)}{n}\right) + \cos\left(\frac{2\pi k(i-1)}{n}\right) = 2\cos\left(\frac{2\pi k}{n}\right)\cos\left(\frac{2\pi k i}{n}\right)$

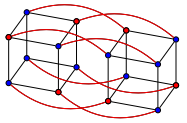
Eigenvalue: $\propto \cos \frac{2\pi k}{n}$.

Eigenvalues:
vibration modes of system.
Fourier basis.

Hypercube

$V = \{0,1\}^d \quad (x,y) \in E$ when x and y differ in one bit.

$|V| = 2^d \; |E| = d2^{d-1}$.



Good cuts? "Coordinate cut": d of them.
Edge expansion: $\frac{2^{d-1}}{d2^{d-1}} = \frac{1}{d}$

Ball cut: All nodes within $d/2$ of node, say $00\cdots0$.
Vertex cut size: $\binom{d}{d/2}$ bit strings with $d/2$ 1's.

$\approx \frac{2^d}{\sqrt{d}}$

Vertex expansion: $\approx \frac{1}{\sqrt{d}}$.

Edge expansion: $d/2$ edges to next level. $\approx \frac{1}{2\sqrt{d}}$

Worse by a factor of $\sqrt{d}$

Eigenvalues of hypercube.

Anyone see any symmetry?

Coordinate cuts. +1 on one side, -1 on other.

$(Mv)_i = (1 - 2/d)v_i$.

Eigenvalue $1 - 2/d$. d Eigenvectors. Why orthogonal?

Next eigenvectors?

Delete edges in two dimensions.

Four subcubes: bipartite. Color ± 1

Eigenvalue: $1 - 4/d$. $\binom{d}{2}$ eigenvectors.

Eigenvalues: $1 - 2k/d$. $\binom{d}{k}$ eigenvectors.

Back to Cheeger.

Coordinate Cuts:
Eigenvalue $1 - 2/d$. d Eigenvectors.

$\frac{\mu}{2} = \frac{1-\lambda_2}{2} \le h(G) \le \sqrt{2(1-\lambda_2)} = \sqrt{2\mu}$

For hypercube: $h(G) = \frac{1}{d}$ $\lambda_1 - \lambda_2 = 2/d$.
Left hand side is tight.

Note: hamming weight vector also in first eigenspace.

Lose "names" in hypercube, find coordinate cut?

Find coordinate cut?

Eigenvector v maps to line.
Cut along line.

Eigenvector algorithm gets a linear combination of coordinate cuts.

Something like ball cut.

Find coordinate cut?

Random Walk.

p - probability distribution.

Probability distrubtion after choose a random neighbor.
Mp.

Converge to uniform distribution.

Power method: $M^t x$ goes to highest eigenvector.

$M^t x = a_1 \lambda_1^t v_1 + a_2 \lambda_2 v_2 + \cdots$

rate of convergence $\propto \lambda_1 - \lambda_2$

$\Omega(n^2)$ steps to get close to uniform.

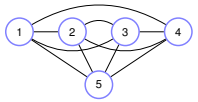
Start at node 0, probability distribution, $[1,0,0,\cdots,0]$.
Takes $\Omega(n^2)$ to get n steps away.

Recall drunken sailor.

Gaussian Elimination

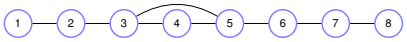
$Ax = b$.

$x_1 = b_1 - a_1 \cdot x$
$a'_{2,1} = 0$
$\forall j \geq 1, a'_{2,j} = a_{2,j} - a_{2,j}a_{1,j}$
$b'_2 = b_2 - a_{2,1}b_1$
$\vdots$



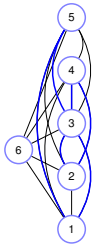
Elminate variable: graph view.
$\forall j \geq 1, a'_{2,j} = 0 - a_{2,j}a_{1,j}$

Expander: eliminate any half of vertices, (almost) get complete graph!
Really complex.
Path:



Kyung-Sachdeva

Elimination/sparsify.
Iteratively: $Ax_n = b$.
Find $x_{n+1} = A^{-1}(b - Ax_n)$
Would be done in one iteration!
Circular!



Eliminate a vertex.
The neighbors would form a complete graph.
Implicitly Sparsify:
Select edge with probability p, multiply by $1/p$.
every edge unchanged in expectation.
fewer edges though.
(Approximate Cholesky factorization.)

Didn't really solve $x_{n+1} = (A^{-1})(b - Ax_n)$.
Argued elimination process approximates eigenvalues of A^{-1}
Good condition number for $\tilde{A}^{-1}A$.
Note: the $\tilde{A}^{-1}$ is fake notation representing elimination. No matrix $\tilde{A}$.

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

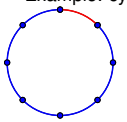
Nested Disection:
Planar graphs (elimination) in $O(n^{3/2})$ time:
complete graph only on separators.

Tradeoff in general:
maybe combine iterative $1/\sqrt{n}$-expanders
and eliminate between them (iteratively)?

Preconditioning:
Solve $B^{-1}Ax = B^{-1}b$, where B is easy to invert.
And $B^{-1}A$ has good condition number, i.e., λ_1/λ_n.
if $B = A$, $B^{-1}A = I$, where condition number is 1.
Vaidya: if B max weight spanning tree, condition number $O(m)$
regardless of weights on edges!

Low stretch trees.

Another cool approach:
Low stretch spanning tree.
Spanning tree T with small $\ell_T(u,v)$ for average edge $e = (u,v)$.
Example: expander. Any short tree. Diameter is $O(\log n)$.
Example: cycle.



Distance 1 goes to $n-1$!

For cycle, remove a random edge get a tree.

Average Stretch of edge: $\frac{n-1}{n} \times 1 + \frac{1}{n} \times (n-1) \approx 2$

In general: $\tilde{O}(m) = O(m \log n \log\log n)$.

Spielman-Teng $\rightarrow \cdots \rightarrow$ Kyung-Sachdeva

Solve $B^{-1}Ax = b$, where B is easy to invert.

Spielman-Teng:
Form B by finding subclusters that are expanders,
ultra-sparsify using low stretch trees, eliminate, sparsify, ...

Kind of a big mess. Tons of ideas.

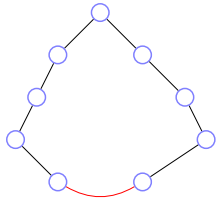
KOSZ

Laplacian systems $\equiv$ to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u,v)$.
(Assume resistances $r(e) = 1$.)

Pick non-tree edge with probability $\propto \ell_T(e) + 1$



$f(e) \neq \phi(u,v)$

Change flow by $\frac{f(u,v) - \phi(u,v)}{\ell_T(e)+1}$.
Reduces $\sum_{e=(u,v)} (f(u,v) - \phi(u,v))^2$
Increases a bit on $\ell_T(e)$ (no violation) edges)
reduces a lot on e since large violation

Analysis Idea

Total violation: $\sum_{e=(u,v)}(f(e)-\phi(u,v))^2$.

Edge w/stretch ℓ selected: reduce violation by $\approx \frac{1}{\ell(f(e)-\phi(u,v))^2}$.

Non-tree edge reduces $\propto \alpha(f(e)-\phi(u,v))^2$
tree edges increase $\propto \ell(\alpha^2(f(e)-\phi(u,v))^2$
$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell}(f(e)-\phi(u,v))^2$.

Selected $\propto \ell$.

$(f(e)-\phi(u,v))^2 \times \frac{1}{\ell} \times \ell$
$\implies$ expected reduction $\propto \sum_{e=(u,v)}(f(e)-\phi(u,v))^2$.

The constant of proportionality: $\sum_e \ell_T(e). = \tilde{O}(m)$
Probabilities: $\approx \frac{\ell_e}{\sum_e \ell_T(e)}$

$(1-\tilde{\Omega}(1/m))$ expected reduction.

$\tilde{O}(m)$ iterations halves error.

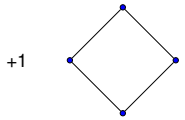
Again: used exact solve on tree to speed up iterative method.
Shout out to "Algebra".

Need data structure to update efficiently!
Interval trees, and path decomposition of trees.

Electrical Flow and Laplacian Systems.

A graph $G=(V,E)$.
Circuit: nodes V, resistors E, value 1 (for today.)

Given $\chi : V \to \Re$
Find flow that routes χ and minimizes
$\sum_e f(e)^2$.



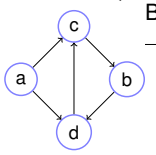
Claim: Minimizer is electrical flow.

Flow corresponds to flow induced by a set of potentials.

Some Matrices.

Given $G=(V,E)$, arbitrarily orient edges.

$$B_{v,e} = \begin{cases} -1 & e=(u,v) \\ 1 & e=(v,u) \\ 0 & \text{otherwise} \end{cases} \qquad L_{u,v} = \begin{cases} d & u=v \\ -1 & (u,v)\in E \\ 0 & \text{otherwise} \end{cases}$$



B

	a	b	c	d
(a,b)	1	-1	0	0
(a,c)	1	0	-1	0
(c,d)	0	0	1	-1
(d,b)	0	-1	0	1
(b,c)	0	1	-1	1

L

	a	b	c	d
a	2	-1	-1	0
b	-1	2	0	-1
c	-1	-1	3	-1
d	-1	-1	-1	3

Fun facts: $\mathbf{f} \in \Re^{|E|}$
$[B^T f]_u = \sum_{e=(u,v)} f_e - \sum_{e=(v,u)} f_e$
$B^T B = L$
$[Bx]_{e=(u,v)} = x_u - x_v$
$x^T L x = \sum_{e=(u,v)}(x_u - x_v)^2$

Duality..

Given G, χ, $\chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that maximizes $\min_f L(\phi, f)$.

Given ϕ, minimize $L(\phi, f)$? Calculus.

For $e=(u,v)$
$2f(e) + 2(\phi_v - \phi_u) = 0$ (Minimum when partial derivatives = 0.)

$\to f(e) = (\phi_u - \phi_v)$ Potential differences!!!

Matrix Form: $f = B\phi$ Again, flows should be potential differences.

Dual problem: Find ϕ that maximizes ...
$\max L(\phi, B\phi) = \phi^T B^T B\phi + 2\phi^T(\chi - B^T B\phi) \implies \max_\phi 2\phi^T\chi - \phi^T L\phi$
Note: want $\phi^T L\phi = \sum_e(\phi_u - \phi_v)^2$ to be small.
Minimize Squared Potential differences!

Why did we take dual?

Dual problem:
Find ϕ that maximizes ...
$\max_\phi 2\phi\chi - \phi L\phi$

Take the derivative:
$L\phi - \chi$
$L\phi = \chi$ at optimal point!

Optimal potential is solution to a Laplacian linear system.

Also useful for convergence.

Algorithm maintains feasible ϕ, f,

Primal value: $|f|^2$.
Dual value: $2\phi\chi - \phi^T L\phi$

Duality gap is "distance" from optimal!

Algorithm: Work on flow and potentials.
To drive gap to 0.

Alg.

Given: χ, G

Take a spanning tree T of G. (Which tree?)

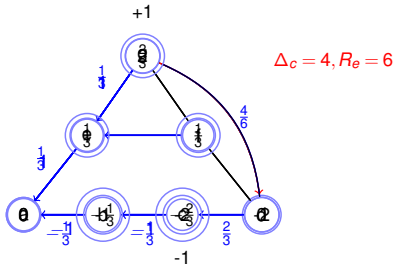
Route flow, f, to satisfy χ through T
Compute, ϕ, using tree ; $\phi_s = 0$, add f_e through T

Repeat:
Choose non-tree edge $e=(u,v)$ (Which non-tree edge?)
$f(e) = (\phi_u - \phi_v)/(\ell_T(u,v)+1)$
($\ell_T(u,v)$ path length in T)
Route excess on path through tree.

Which Tree?
Claim: Linear time algorithm for T w/ stretch $O(m \log n \log\log n)$!
Stretch: $\sum_{e=(u,v)} \ell_T(u,v)$

Which non-tree edge?
Choose an edge w/prob. proportional to $\ell_T(e)$.

Finds $(1+\varepsilon)$ approximation in $O(m \log n \log\log n \log(\frac{n}{\varepsilon}))$! ! ! ! ! !



$\Delta_c = 4, R_e = 6$

Energy reduction.

Given T,$e = (u,v)$, let $R_e = \ell_T(u,v)+1$.

Algorithm:

Repeatedly "Fix" edge $e = (u,v)$.

Route $-\delta = -\frac{\sum_{e' \in C_e} f(e')}{R_e}$ flow around cycle induced in T: C_e

(assume e' are oriented around cycle.)

Difference in energy from f and f'.

$\sum_{e' \in C_e} (f(e') - \delta)^2 - (f(e'))^2 = \sum_{e' \in C_e} -2f(e')\delta + \delta^2$

$= -(2\delta \sum_{e' \in C_e} f(e')) + R_E \delta^2$

Note: $\sum_{e' \in C_e} f(e') = R_e \delta$

$\rightarrow -\Delta_{C_e}^2 / R_e$ where $\Delta_{C_e} = \sum_{e' \in C_e} f_(e')$.

Fix a part of the potential difference, Δ_{c_e} around cyle!!

$\rightarrow$ reduction of $\Delta_{C_e}^2 / R_e$ in energy!

Fix $1/R_e$ of a cycle violation!

Duality Gap?

Algorithm maintains feasible ϕ, f, $(B^T f = \chi)$

Primal value: $|f|^2$.
Dual value: $2\phi\chi - \phi L \phi$

ϕ is tree induced voltages.

Total Duality Gap?

Gap: $|f|^2 - (2\phi^T \chi - \phi^T L \phi)$.

$= |f|^2 - 2\phi^T B^T f + \phi^T B^T B \phi$ where $B^T f = \chi$ and $L = B^T B$.
$= (f - B\phi)^T (f - B\phi)$.

Gap $= \sum_e (f(e) - \Delta_\phi(e))^2$ Difference between ϕ flow and f.

$\Delta_\phi(u,v) = \sum_{e \in P_{u,v}} -f(e)$. assume $f(e)$ is oriented around cycle.

For $e \in T$, $\Delta_\phi(e) = 0$. For $e \notin T$ $f(e) + \sum_{e \in P_e} f(e) = \Delta_{c_e}(f)$

Duality Gap: $\sum_{e \notin T} \sum_e \Delta_{c_e}(f)^2$

Total distance from optimal is cycle violations!

Polishing off.

Claim: E[change in energy$|Gap] = \frac{Gap}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T.)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$ □

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation.

$O(\tau \log(n/\varepsilon))$ iterations gives $(1+\varepsilon)$ approximation.

$\tau = O(m \log n \log\log n)$ ...

$\tilde{O}(m)$ iterations

Iteration in $O(\log^2 n)$ time using balanced binary trees.

$\rightarrow \tilde{O}(m)$ time! ! ! ! ! !
Enough with the exclamations already !

Wrapup...

How to get low stretch tree?
Answer: Elkin-Spielman-Teng, ...,Abraham, Newman.
Open: Get $O(m \log n)$ stretch? Like for HSTs. (Later)
Ideas: Similar to HST construction, but much more subtle.

How to do update along cycle?
Answer: Data Structures.
Idea: Use a binary tree on paths.
Decompose tree into paths.

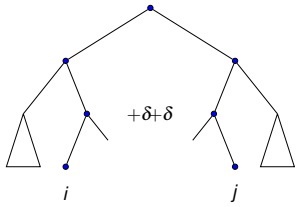
Interval Tree

Path: $G = (V,E)$, $V = \{1,\ldots,n\}$, $E = \{(i,i+1) : i \in \{1,n-1\}\}$

Init: $f(e) = 0, \forall e \in G$.

Update(i,j,δ)
$\forall k \in \{i,\ldots,j-1\}$, $f(i,i+1) = f(i,i+1) + \delta$

Lookup(i,j)
Return $\sum_{k=i}^{j-1} f(k,k+1)$.



Update: find common ancestor.
For left path:
For right children, add δ to edge's value.
Reflect for right children.

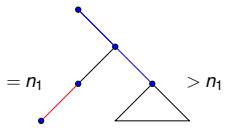
Lookup: Exercise.
Something like compute paths.
Add product of numbers on some edges.
and number of descendants in subtree.

$O(\log n)$ update/lookup.

Path Decomposition of Tree

Given tree T.

Decomposition Procedure:
Longest path from root to leaf.
From root, go towards heavier side.
Remove. Recurse



Decomposition Property:
Where a root to leaf path in tree only sees $O(\log n)$ paths.

Proof Idea:
Every path change, doubles number of vertices.

$O(\log^2 n)$ update time for updating flow values on tree edges!

What's going on?

Geometric View.
Cycles are constraints.
Flow around cycle $= 0$.
Each cycle update is approximate projection into subspace defined by constraint.
The solution is intersection of all cycle constraints and flow conservation.

Algorithmic Power. Algebra: Solving exactly on tree.
Calculus: make a local move.

Better Algorithm:
Recursive algorithm give $O(m\sqrt{\log n})$ iterations to halve error.

Correspondence to Practice: Random sparsification of Cholesky factorization.

Laplacian Systems are quite general:
Climate, physics, SDD-matrices.