Convex Optimization of Graph Laplacian Eigenvalues

Stephen Boyd

Stanford University

(Joint work with Persi Diaconis, Arpita Ghosh, Seung-Jean Kim, Sanjay Lall, Pablo Parrilo, Amin Saberi, Jun Sun, Lin Xiao. Thanks to Almir Mutapcic.)

ICM 2006 Madrid, August 29, 2006

Outline

- some basic stuff we'll need
  - graph Laplacian eigenvalues
  - convex optimization and semidefinite programming
- the basic idea
- some example problems
  - distributed linear averaging
  - fastest mixing Markov chain on a graph
  - fastest mixing Markov process on a graph
  - its dual: maximum variance unfolding
- conclusions

Outline

- some basic stuff we'll need
  - graph Laplacian eigenvalues
  - convex optimization and semidefinite programming
- the basic idea
- some example problems
  - distributed linear averaging
  - fastest mixing Markov chain on a graph
  - fastest mixing Markov process on a graph
  - its dual: maximum variance unfolding
- conclusions

(Weighted) graph Laplacian

- graph $G = (V, E)$ with $n = |V|$ nodes, $m = |E|$ edges
- edge weights $w_1, \ldots, w_m \in \mathbf{R}$
- $l \sim (i, j)$ means edge l connects nodes i, j
- incidence matrix: $A_{il} = \begin{cases} 1 & \text{edge } l \text{ enters node } i \\ -1 & \text{edge } l \text{ leaves node } i \\ 0 & \text{otherwise} \end{cases}$
- (weighted) Laplacian: $L = A\,\mathbf{diag}(w)A^T$
- $L_{ij} = \begin{cases} -w_l & l \sim (i, j) \\ \sum_{l \sim (i,k)} w_l & i = j \\ 0 & \text{otherwise} \end{cases}$

Laplacian eigenvalues

- L is symmetric; $L\mathbf{1} = 0$

- we'll be interested in case when $L \succeq 0$ ($i.e.$, L is PSD)
  (always the case when weights nonnegative)

- Laplacian eigenvalues (eigenvalues of L):

$$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$$

- *spectral graph theory* connects properties of graph, and λ_i (with $w = \mathbf{1}$)

  $e.g.$: G connected iff $\lambda_2 > 0$ (with $w = \mathbf{1}$)

Convex spectral functions

- suppose ϕ is a symmetric convex function in $n-1$ variables
- then $\psi(w) = \phi(\lambda_2, \ldots, \lambda_n)$ is a convex function of weight vector w
- examples:
  - $\phi(u) = \mathbf{1}^T u$ (*i.e.*, the sum):

$$\psi(w) = \sum_{i=2}^{n} \lambda_i = \sum_{i=1}^{n} \lambda_i = \mathbf{Tr}\, L = 2\mathbf{1}^T w \qquad \text{(twice the total weight)}$$

  - $\phi(u) = \max_i u_i$:

$$\psi(w) = \max\{\lambda_2, \ldots, \lambda_n\} = \lambda_n \qquad \text{(spectral radius)}$$

More examples

- $\phi(u) = \min_i u_i$ (concave) gives $\psi(w) = \lambda_2$, *algebraic connectivity* (concave function of w)

- $\phi(u) = \sum_i 1/u_i$ (with $u_i > 0$):

$$\psi(w) = \sum_{i=2}^{n} \frac{1}{\lambda_i}$$

proportional to *total effective resistance* of graph, $\mathbf{Tr}\, L^\dagger$

Outline

- some basic stuff we'll need
  - graph Laplacian eigenvalues
  - convex optimization and semidefinite programming
- the basic idea
- some example problems
  - distributed linear averaging
  - fastest mixing Markov chain on a graph
  - fastest mixing Markov process on a graph
  - its dual: maximum variance unfolding
- conclusions

Convex optimization

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \mathcal{X} \end{array}$$

- $x \in \mathbf{R}^n$ is optimization variable
- f is convex function (can maximize concave f by minimizing $-f$)
- $\mathcal{X} \subseteq \mathbf{R}^n$ is closed convex set
- roughly speaking, convex optimization problems are tractable, 'easy' to solve numerically (tractability depends on how f and $\mathcal{X}$ are described)

Symmetry in convex optimization

- permutation (matrix) π is a symmetry of problem if $f(\pi z) = f(z)$ for all z, $\pi z \in \mathcal{X}$ for all $z \in \mathcal{X}$

- if π is a symmetry and the convex optimization problem has a solution, it has a solution invariant under π

  (if $x^\star$ is a solution, so is average over $\{x^\star, \pi x^\star, \pi^2 x^\star, \ldots\}$)

Duality in convex optimization

primal:	minimize subject to	$f(x)$ $x \in \mathcal{X}$	*dual:*	maximize subject to	$g(y)$ $y \in \mathcal{Y}$

- y is dual variable; dual objective g is concave; $\mathcal{Y}$ is closed, convex (various methods can be used to generate g, $\mathcal{Y}$)

- $p^\star$ ($d^\star$) is optimal value of primal (dual) problem

- *weak duality*: for any $x \in \mathcal{X}$, $y \in \mathcal{Y}$, $f(x) \geq g(y)$; hence, $p^\star \geq g(y)$

- *strong duality*: for convex problems, provided a 'constraint qualification' holds, there exist $x^\star \in \mathcal{X}$, $y^\star \in \mathcal{Y}$ with $f(x^\star) = g(y^\star)$

  hence, $x^\star$ is primal optimal, $y^\star$ is dual optimal, and $p^\star = d^\star$

Semidefinite program (SDP)

a particular type of convex optimization problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & \sum_{i=1}^n x_i A_i \preceq B, \quad Fx = g \end{array}$$

- variable is $x \in \mathbf{R}^n$; data are c, F, g, symmetric matrices A_i, B
- $\preceq$ means with respect to positive semidefinite cone
- generalization of linear program (LP)

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & \sum_{i=1}^n x_i a_i \leq b, \quad Fx = g \end{array}$$

(here a_i, b are vectors; $\leq$ means componentwise)

SDP dual

primal SDP:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & \sum_{i=1}^n x_i A_i \preceq B, \quad Fx = g \end{array}$$

dual SDP:

$$\begin{array}{ll} \text{maximize} & -\mathbf{Tr}\, ZB - \nu^T g \\ \text{subject to} & Z \succeq 0, \quad \left(F^T \nu + c\right)_i + \mathbf{Tr}\, Z A_i = 0, \quad i = 1, \ldots, n \end{array}$$

with (matrix) variable Z, (vector) ν

SDP algorithms and applications

since 1990s,

- recently developed interior-point algorithms solve SDPs very effectively

  (polynomial time, work well in practice)

- many results for LP extended to SDP

- SDP widely used in many fields

  (control, combinatorial optimization, machine learning, finance, signal processing, communications, networking, circuit design, mechanical engineering, . . .)

Outline

- some basic stuff we'll need
  - graph Laplacian eigenvalues
  - convex optimization and semidefinite programming
- the basic idea
- some example problems
  - distributed linear averaging
  - fastest mixing Markov chain on a graph
  - fastest mixing Markov process on a graph
  - its dual: maximum variance unfolding
- conclusions

The basic idea

some interesting weight optimization problems have the common form

$$\begin{array}{ll} \text{minimize} & \phi(w) = \phi(\lambda_2, \ldots, \lambda_n) \\ \text{subject to} & w \in \mathcal{W} \end{array}$$

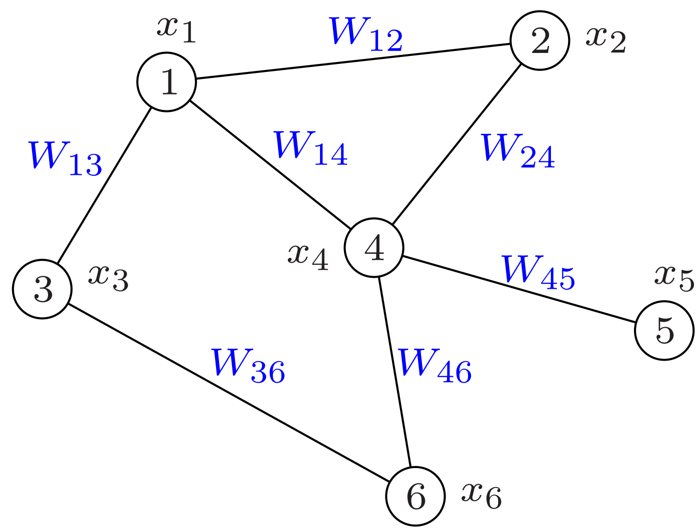
where ϕ is symmetric convex, and $\mathcal{W}$ is closed convex

- these are convex optimization problems
- we can solve them numerically
  (up to our ability to store data, compute eigenvalues . . .)
- for some simple graphs, we can get analytic solutions
- associated dual problems can be quite interesting

Outline

- some basic stuff we'll need
  - graph Laplacian eigenvalues
  - convex optimization and semidefinite programming
- the basic idea
- some example problems
  - distributed linear averaging
  - fastest mixing Markov chain on a graph
  - fastest mixing Markov process on a graph
  - its dual: maximum variance unfolding
- conclusions

Distributed averaging



- each node of connected graph has initial value $x_i(0) \in \mathbf{R}$; goal is to compute average $\mathbf{1}^T x(0)/n$ using distributed iterative method
- applications in load balancing, distributed optimization, sensor networks

Distributed linear averaging

- simple linear iteration: replace each node value with weighted average of its own and its neighbors' values; repeat

$$\begin{aligned} x_i(t+1) &= W_{ii}x_i(t) + \sum_{j \in \mathcal{N}_i} W_{ij}x_j(t) \\ &= x_i(t) - \sum_{j \in \mathcal{N}_i} W_{ij}\left(x_i(t) - x_j(t)\right) \end{aligned}$$

  where $W_{ii} + \sum_{j \in \mathcal{N}_i} W_{ij} = 1$

- we'll assume $W_{ij} = W_{ji}$, *i.e.*, weights symmetric
- weights W_{ij} determine whether convergence to average occurs, and if so, how fast
- classical result: convergence if weights W_{ij} $(i \neq j)$ are small, positive

Convergence rate

- vector form: $x(t+1) = Wx(t)$ (we take $W_{ij} = 0$ for $i \neq j$, $(i,j) \notin E$)
- W satisfies $W = W^T$, $W\mathbf{1} = \mathbf{1}$
- convergence $\iff \lim_{t\to\infty} W^t = (1/n)\mathbf{1}\mathbf{1}^T \iff$

$$\rho(W - (1/n)\mathbf{1}\mathbf{1}^T) = \|W - (1/n)\mathbf{1}\mathbf{1}^T\| < 1$$

  ρ is spectral radius; $\|\cdot\|$ is spectral norm

- asymptotic convergence rate given by $\|W - (1/n)\mathbf{1}\mathbf{1}^T\|$
- convergence time is $\tau = -1/\log\|W - (1/n)\mathbf{1}\mathbf{1}^T\|$

Connection to Laplacian eigenvalues

- identifying $W_{ij} = w_l$ for $l \sim (i,j)$, we have $W = I - L$
- convergence rate given by

$$\begin{aligned} \|W - (1/n)\mathbf{1}\mathbf{1}^T\| &= \|I - L - (1/n)\mathbf{1}\mathbf{1}^T\| \\ &= \max\{|1-\lambda_2|, \ldots, |1-\lambda_n|\} \\ &= \max\{1-\lambda_2, \lambda_n - 1\} \end{aligned}$$

. . . a convex spectral function, with $\phi(u) = \max_i |1 - u_i|$

Fastest distributed linear averaging

$$\begin{array}{ll} \text{minimize} & \|W - (1/n)\mathbf{1}\mathbf{1}^T\| \\ \text{subject to} & W \in \mathcal{S}, \quad W = W^T, \quad W\mathbf{1} = \mathbf{1} \end{array}$$

optimization variable is W; problem data is graph (sparsity pattern $\mathcal{S}$)

in terms of Laplacian eigenvalues

$$\text{minimize} \quad \max\{1 - \lambda_2, \lambda_n - 1\}$$

with variable $w \in \mathbf{R}^m$

- these are convex optimization problems
- so, we can efficiently find the weights that give the fastest possible averaging on a graph

Semidefinite programming formulation

introduce scalar variable s to bound spectral norm

$$\begin{array}{ll} \text{minimize} & s \\ \text{subject to} & -sI \preceq I - L - (1/n)\mathbf{1}\mathbf{1}^T \preceq sI \end{array}$$

(for $Z = Z^T$, $\|Z\| \leq s \Longleftrightarrow -sI \preceq Z \preceq sI$)

an SDP (hence, can be solved efficiently)

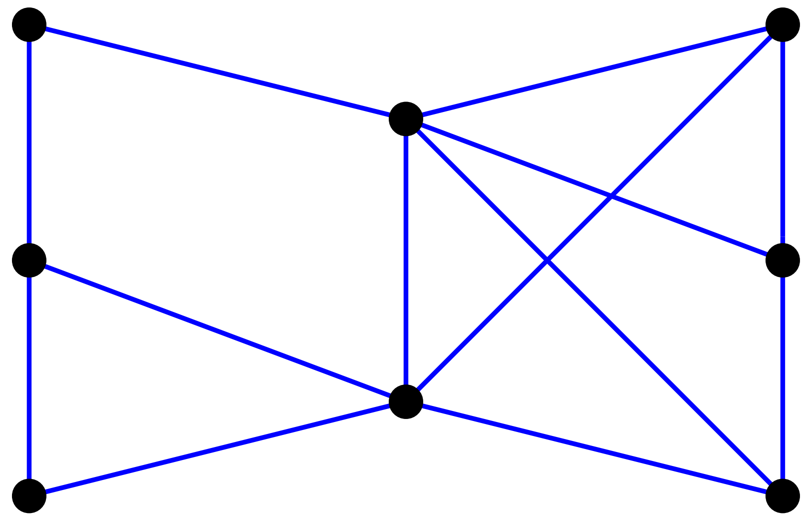
Constant weight averaging

- a simple, traditional method: constant weight on all edges, $w = \alpha \mathbf{1}$
- yields update

$$x_i(t+1) = x_i(t) + \sum_{j \in \mathcal{N}_i} \alpha(x_j(t) - x_i(t))$$

- a simple choice: max-degree weight, $\alpha = 1/\max_i d_i$
  d_i is degree (number of neighbors) of node i
- best constant weight: $\alpha^\star = \dfrac{2}{\lambda_2 + \lambda_n}$
  (λ_2, λ_n are eigenvalues of unweighted Laplacian, *i.e.*, with $w = \mathbf{1}$)
- for edge transitive graph, $w_l = \alpha^\star$ is optimal

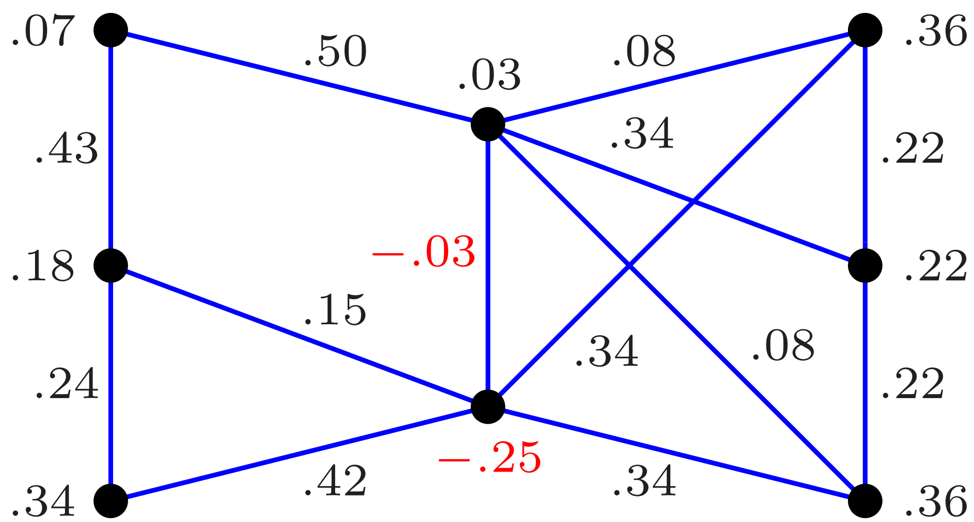
A small example



	max-degree	best constant	optimal
$\rho = \|W - (1/n)\mathbf{1}\mathbf{1}^T\|$	0.779	0.712	0.643
$\tau = 1/\log(1/\rho)$	4.01	2.94	2.27

Optimal weights

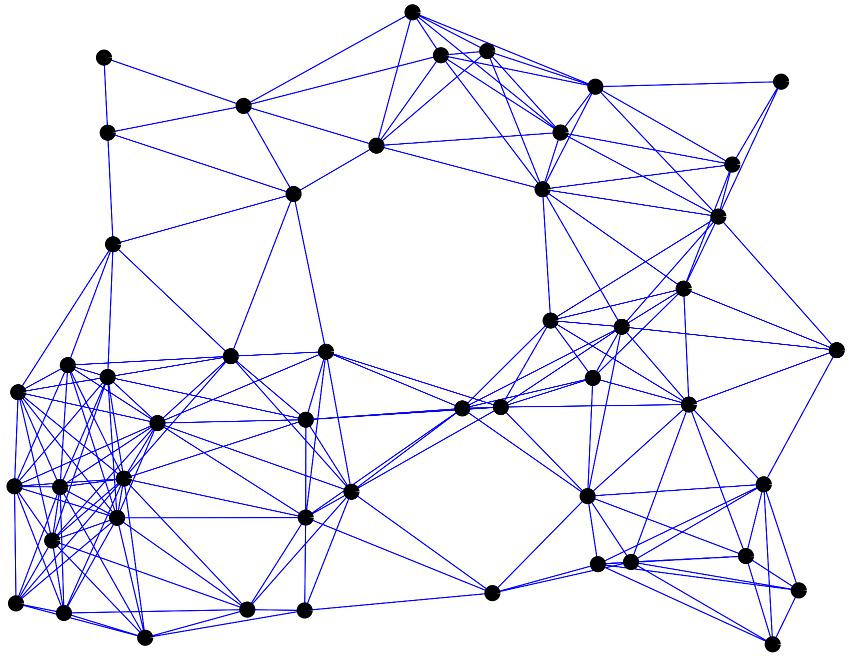
(note: some are negative!)



$\lambda_i(W)$: $-.643$, $-.643$, $-.106$, 0.000, $.368$, $.643$, $.643$, 1.000

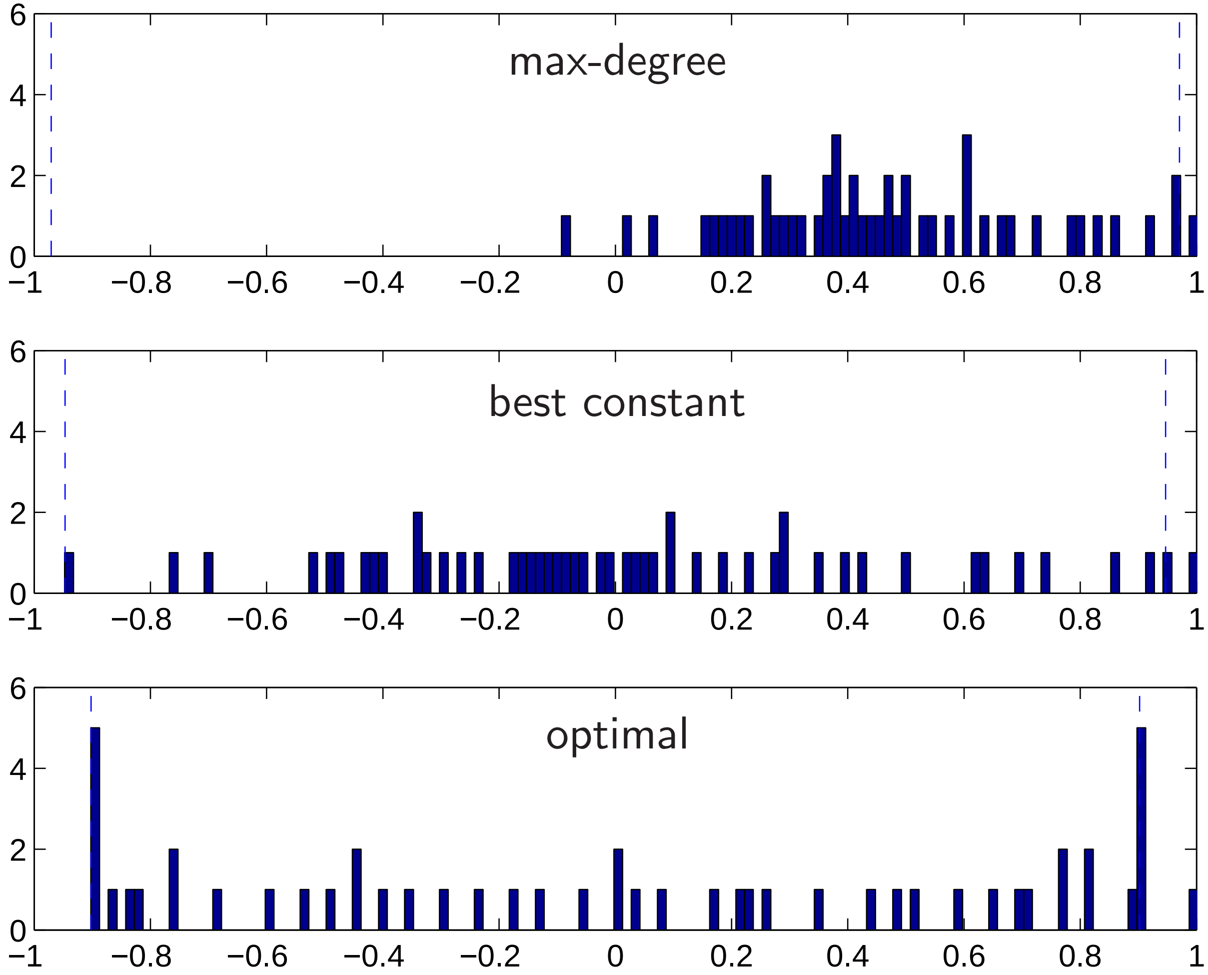
Larger example

50 nodes, 200 edges

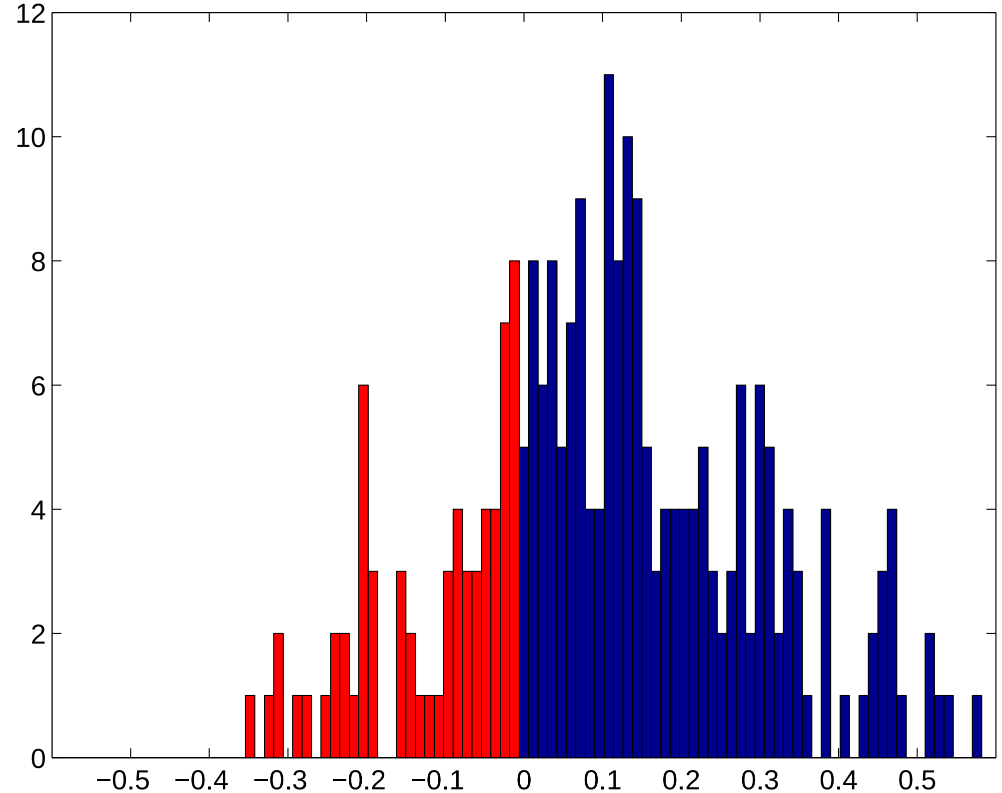


	max-degree	best constant	optimal
$\rho = \|W - (1/n)\mathbf{1}\mathbf{1}^T\|$	.971	.947	.902
$\tau = 1/\log(1/\rho)$	33.5	18.3	9.7

Eigenvalue distributions



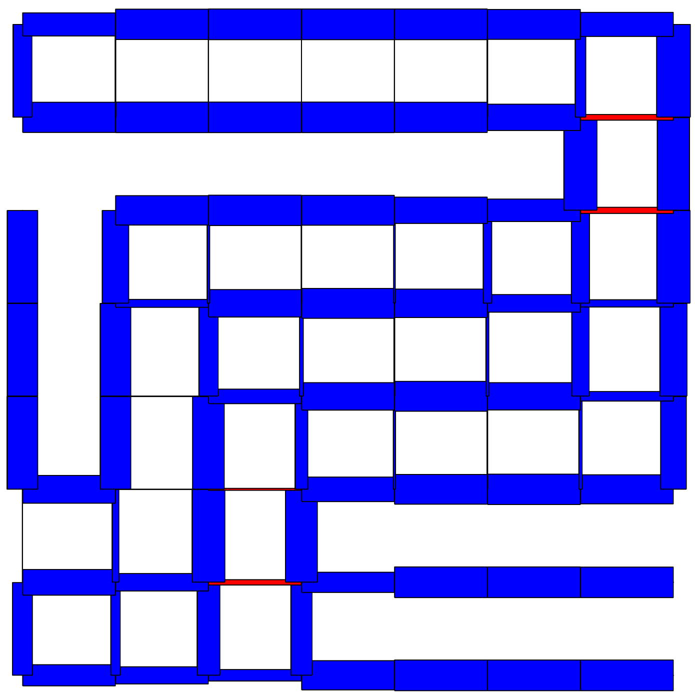
Optimal weights



69 out of 250 are negative

Another example

- a cut grid with $n = 64$ nodes, $m = 95$ edges
- edge width shows weight value (red for negative)
- $\tau = 85$; max-degree $\tau = 137$



Some questions & comments

- how much better are the optimal weights than the simple choices?
  - for barbell graphs $K_n - K_n$, optimal weights are unboundedly better than max-degree, optimal constant, and several other simple weight choices
- what size problems can be handled (on a PC)?
  - interior-point algorithms easily handle problems with 10^4 edges
  - subgradient-based methods handle problems with 10^6 edges
  - any symmetry can be exploited for efficiency gain
- what happens if we *require* the weights to be nonnegative?
  - (we'll soon see)

Least-mean-square average consensus

- include random noise in averaging process: $x(t+1) = Wx(t) + v(t)$
  $v(t)$ i.i.d., $\mathbf{E}\, v(t) = 0$, $\mathbf{E}\, v(t)v(t)^T = I$

- steady-state mean-square deviation:

$$\delta_{\mathrm{ss}} = \lim_{t\to\infty} \mathbf{E} \left(\frac{1}{n} \sum_{i<j} (x_i(t) - x_j(t))^2 \right) = \sum_{i=2}^{n} \frac{1}{\lambda_i(2-\lambda_i)}$$

  for $\rho = \max\{1-\lambda_2, \lambda_n - 1\} < 1$

- another convex spectral function

Outline

- some basic stuff we'll need
  - graph Laplacian eigenvalues
  - convex optimization and semidefinite programming
- the basic idea
- some example problems
  - distributed linear averaging
  - fastest mixing Markov chain on a graph
  - fastest mixing Markov process on a graph
  - its dual: maximum variance unfolding
- conclusions

Random walk on a graph

- Markov chain on nodes of G, with transition probabilities on edges

$$P_{ij} = \mathbf{Prob}\,(X(t+1) = j \mid X(t) = i)$$

- we'll focus on symmetric transition probability matrices P
  (everything extends to reversible case, with fixed equilibrium distr.)

- identifying P_{ij} with w_l for $l \sim (i,j)$, we have $P = I - L$

- same as linear averaging matrix W, but here $W_{ij} \geq 0$
  ($i.e.$, $w \geq 0$, $\mathbf{diag}(L) \leq \mathbf{1}$)

Mixing rate

- probability distribution $\pi_i(t) = \mathbf{Prob}(X(t) = i)$ satisfies $\pi(t+1)^T = \pi(t)^T P$
- since $P = P^T$ and $P\mathbf{1} = \mathbf{1}$, uniform distribution $\pi = (1/n)\mathbf{1}$ is stationary, $i.e.$, $((1/n)\mathbf{1})^T P = ((1/n)\mathbf{1})^T$
- $\pi(t) \to (1/n)\mathbf{1}$ for any $\pi(0)$ iff

$$\mu = \|P - (1/n)\mathbf{1}\mathbf{1}^T\| = \|I - L - (1/n)\mathbf{1}\mathbf{1}^T\| < 1$$

  μ is called second largest eigenvalue modulus (SLEM) of MC

- SLEM determines convergence (mixing) rate, $e.g.$,

$$\sup_{\pi(0)} \|\pi(t) - (1/n)\mathbf{1}\|_{\mathrm{tv}} \leq \left(\sqrt{n}/2\right) \mu^t$$

- associated mixing time is $\tau = 1/\log(1/\mu)$

Fastest mixing Markov chain problem

$$\begin{array}{ll} \text{minimize} & \mu = \|I - L - (1/n)\mathbf{1}\mathbf{1}^T\| = \max\{1-\lambda_2, \lambda_n - 1\} \\ \text{subject to} & w \geq 0, \quad \mathbf{diag}(L) \leq \mathbf{1} \end{array}$$

- optimization variable is w; problem data is graph G
- same as fast linear averaging problem, with additional nonnegativity constraint $W_{ij} \geq 0$ on weights
- convex optimization problem (indeed, SDP), hence efficiently solved

Two common suboptimal schemes

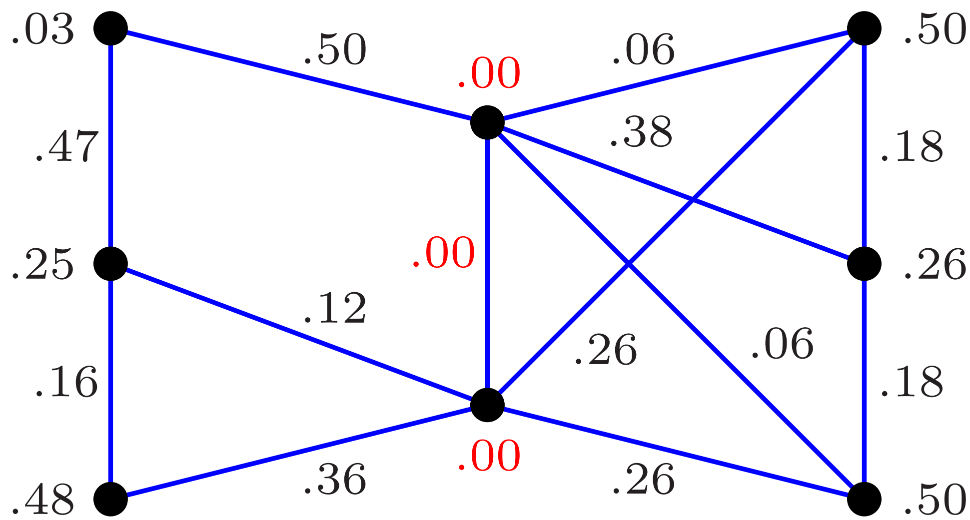
- max-degree chain: $w = (1/\max_i d_i)\mathbf{1}$

- Metropolis-Hastings chain: $w_l = \dfrac{1}{\max\{d_i, d_j\}}$, where $l \sim (i, j)$

  (comes from Metropolis method of generating reversible MC with uniform stationary distribution)

Small example

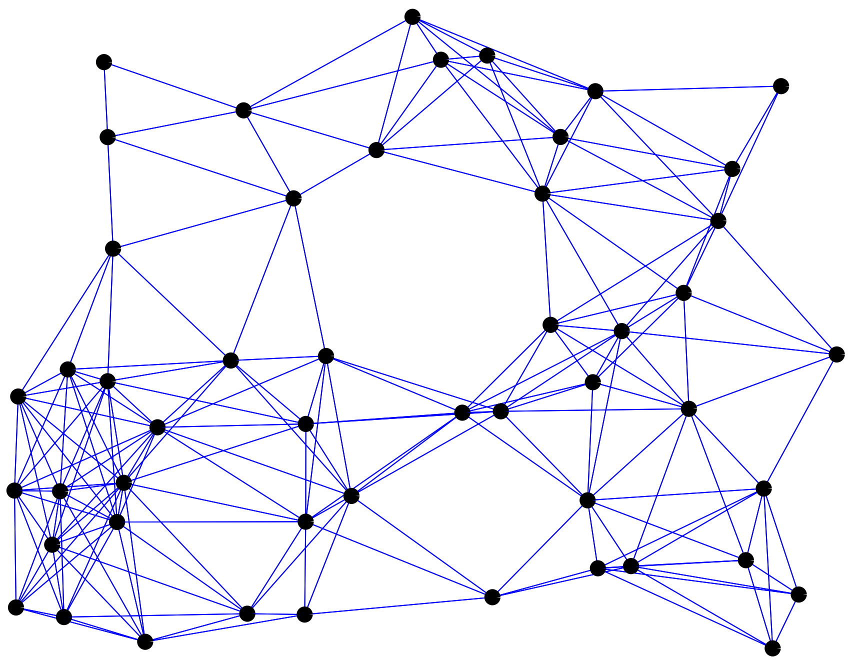
optimal transition probabilities (some are zero)



	max-degree	M.-H.	optimal	(fastest avg)
SLEM μ	.779	.774	.681	(.643)
mixing time τ	4.01	3.91	2.60	(2.27)

Larger example

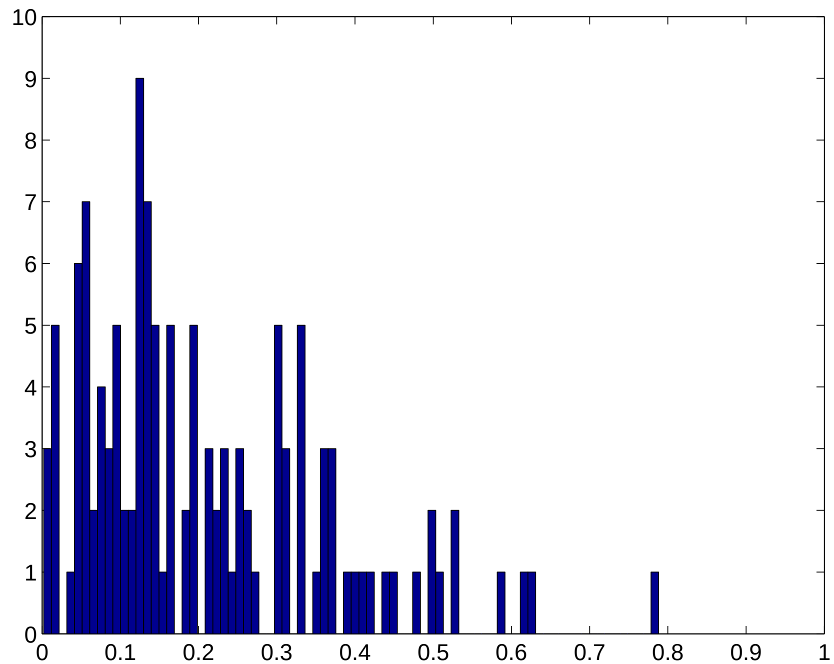
50 nodes, 200 edges



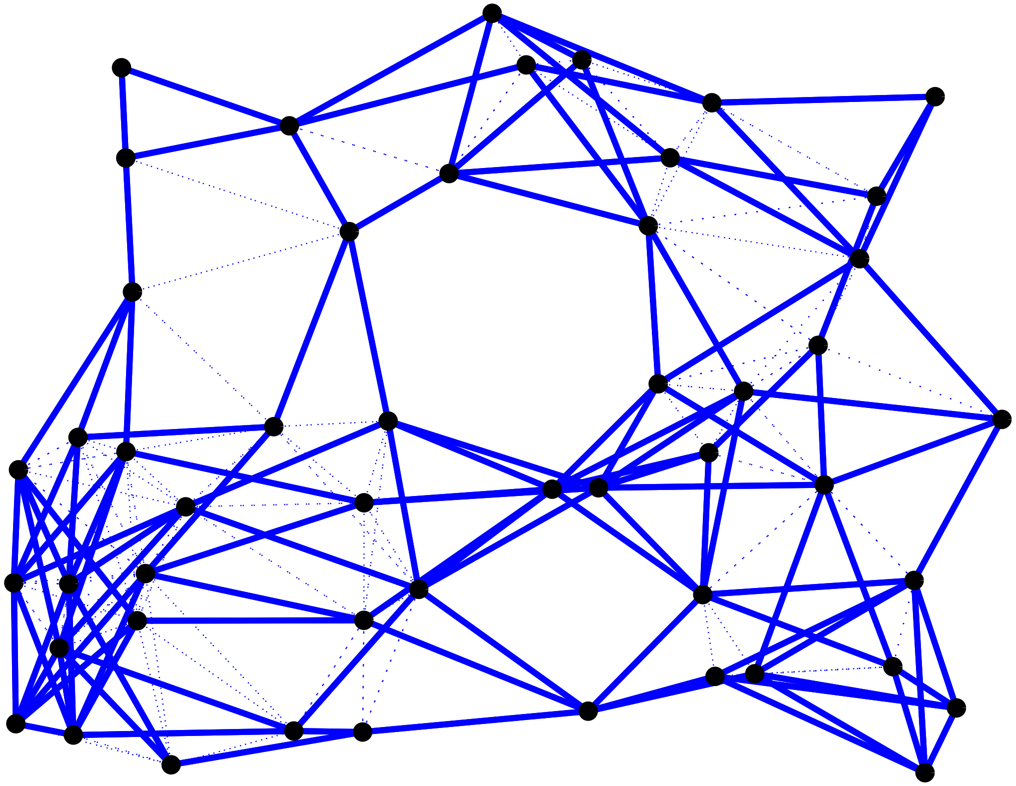
	max-degree	M.-H.	optimal	(fastest avg)
SLEM μ	.971	.949	.915	(.902)
mixing time τ	33.5	19.1	11.3	(9.7)

Optimal transition probabilities

- 82 edges (out of 200) edges have *zero* transition probability
- distribution of positive transition probabilities:

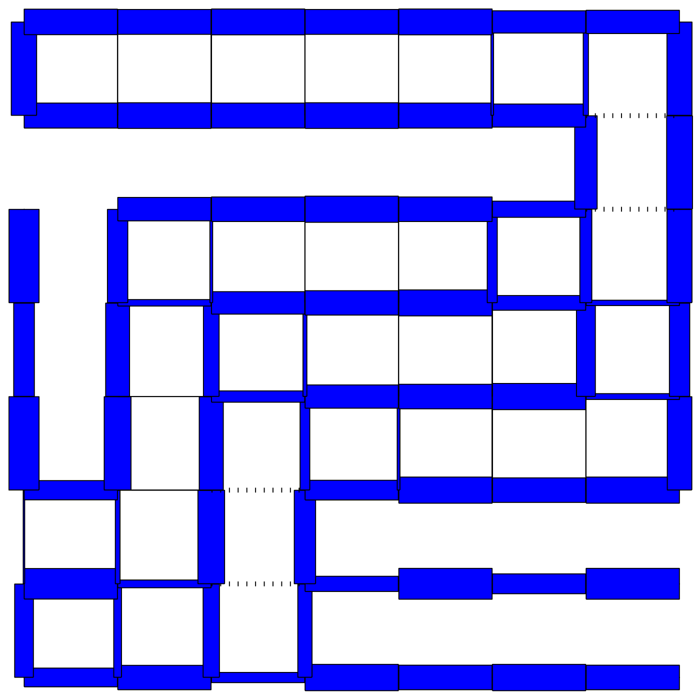


Subgraph with positive transition probabilities



Another example

- a cut grid with $n = 64$ nodes, $m = 95$ edges
- edge width shows weight value (dotted for zero)
- mixing time $\tau = 89$; Metropolis-Hastings mixing time $\tau = 120$



Some analytical results

- for path, fastest mixing MC is obvious one ($P_{i,i+1} = 1/2$)
- for any edge-transitive graph (hypercube, ring, . . .), all edge weights are equal, with value $2/(\lambda_2 + \lambda_n)$ (unweighted Laplacian eigenvalues)

Commute time for random walk on graph

- P_{ij} proportional to w_l, for $l \sim (i,j)$; $P_{ii} = 0$
- P not symmetric, but MC is reversible
- can normalize w as $\mathbf{1}^T w = 1$
- commute time C_{ij}: time for random walk to return to i after visiting j
- expected commute time averaged over all pairs of nodes is

$$\overline{C} = \frac{1}{n^2} \sum_{i,j=1}^{n} \mathbf{E}\, C_{ij} = \frac{2}{(n-1)} \sum_{i=2}^{n} \frac{1}{\lambda_i}$$

(Chandra et al, 1989)

- called *total effective resistance* . . . another convex spectral function

Minimizing average commute time

find weights that minimize average commute time on graph:

$$\begin{array}{ll} \text{minimize} & \overline{C} = 2/(n-1)\sum_{i=2}^{n} 1/\lambda_i \\ \text{subject to} & w \geq 0, \quad \mathbf{1}^T w = 1 \end{array}$$

- another convex problem of our general form

Outline

- some basic stuff we'll need
  - graph Laplacian eigenvalues
  - convex optimization and semidefinite programming
- the basic idea
- some example problems
  - distributed linear averaging
  - fastest mixing Markov chain on a graph
  - fastest mixing Markov process on a graph
  - its dual: maximum variance unfolding
- conclusions

Markov process on a graph

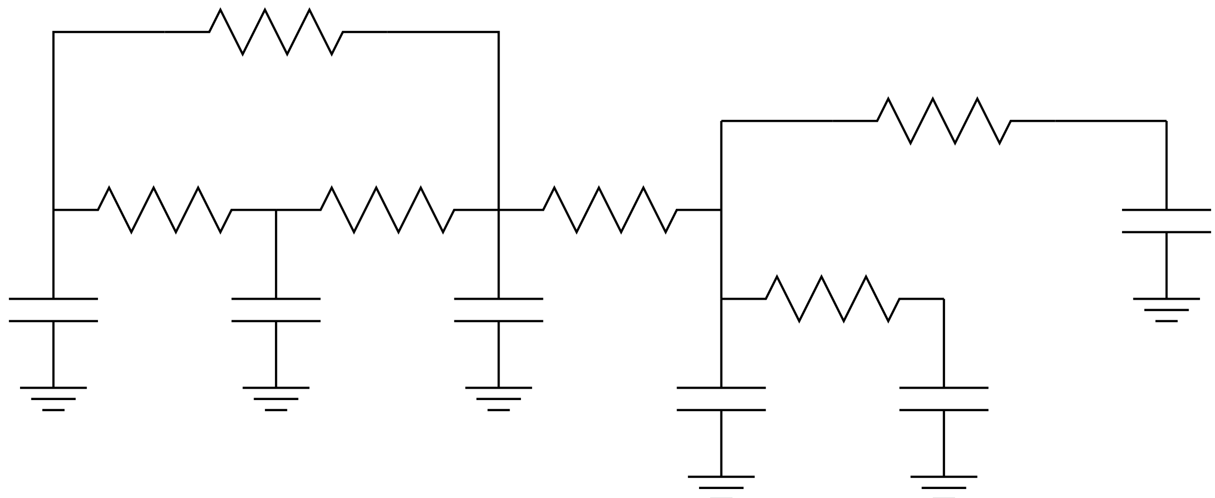
- (continuous-time) Markov process on nodes of G, with transition rate $w_l \geq 0$ between nodes i and j, for $l \sim (i, j)$
- probability distribution $\pi(t) \in \mathbf{R}^n$ satisfies heat equation $\dot{\pi}(t) = -L\pi(t)$
- $\pi(t) = e^{-tL}\pi(0)$
- $\pi(t)$ converges to uniform distribution $(1/n)\mathbf{1}$, for any $\pi(0)$, iff $\lambda_2 > 0$
- (asymptotic) convergence as $e^{-\lambda_2 t}$; λ_2 gives mixing rate of process
- λ_2 is concave, homogeneous function of w
  (come from symmetric concave function $\phi(u) = \min_i u_i$)

Fastest mixing Markov process on a graph

$$\begin{array}{ll} \text{maximize} & \lambda_2 \\ \text{subject to} & \sum_l d_l^2 w_l \leq 1, \qquad w \geq 0 \end{array}$$

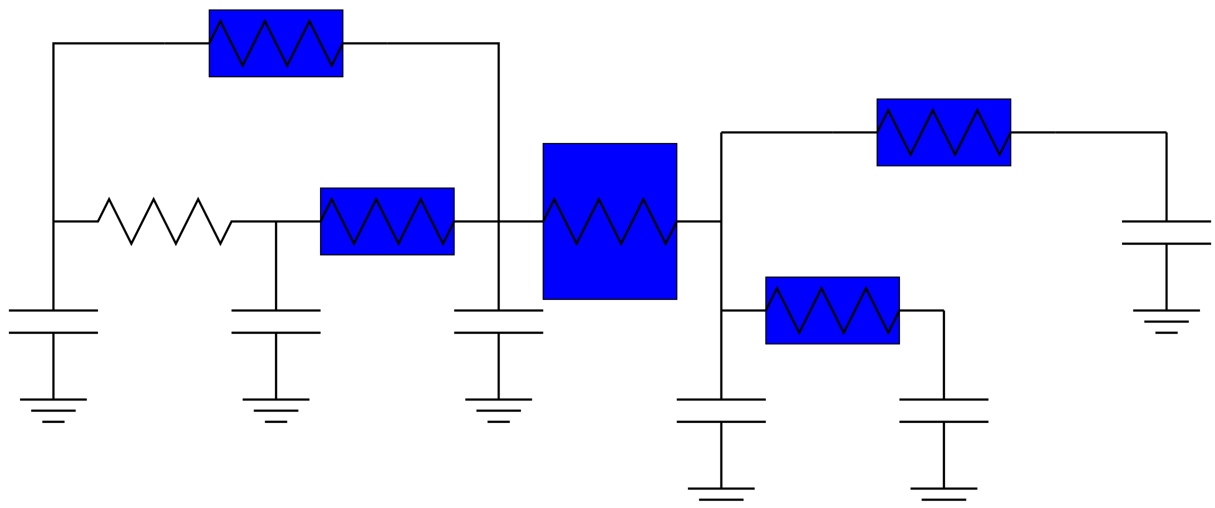
- variable is $w \in \mathbf{R}^m$; data is graph, normalization constants $d_l > 0$
- a convex optimization problem, hence easily solved
- allocate rate across edges so as maximize mixing rate
- constraint is always tight at solution, *i.e.*, $\sum_l d_l^2 w_l = 1$
- when $d_l^2 = 1/m$, optimal value is called *absolute algebraic connectivity*

Interpretation: Grounded unit capacitor RC circuit



- charge vector $q(t)$ satisfies $\dot{q}(t) = -Lq(t)$, with edge weights given by conductances, $w_l = g_l$
- charge equilibrates (*i.e.*, converges to uniform) at rate determined by λ_2
- with conductor resistivity ρ, length d_l, and cross-sectional area a_l, we have $g_l = a_l/(\rho d_l)$

- total conductor volume is $\sum_l d_l a_l = \rho \sum_l d_l^2 w_l$
- problem is to choose conductor cross-sectional areas, subject to a total volume constraint, so as to make the circuit equilibrate charge as fast as possible



optimal λ_2 is $.105$; uniform allocation of conductance gives $\lambda_2 = .073$

SDP formulation and dual

alternate formulation:

$$\begin{array}{ll} \text{minimize} & \sum d_l^2 w_l \\ \text{subject to} & \lambda_2 \geq 1, \quad w \geq 0 \end{array}$$

SDP formulation:

$$\begin{array}{ll} \text{minimize} & \sum d_l^2 w_l \\ \text{subject to} & L \succeq I - (1/n)\mathbf{1}\mathbf{1}^T, \quad w \geq 0 \end{array}$$

dual problem:

$$\begin{array}{ll} \text{maximize} & \mathbf{Tr}\, X \\ \text{subject to} & X_{ii} + X_{jj} - X_{ij} - X_{ji} \leq d_l^2, \quad l \sim (i,j) \\ & \mathbf{1}^T X \mathbf{1} = 0, \quad X \succeq 0 \end{array}$$

with variable $X \in \mathbf{R}^{n \times n}$

A maximum variance unfolding problem

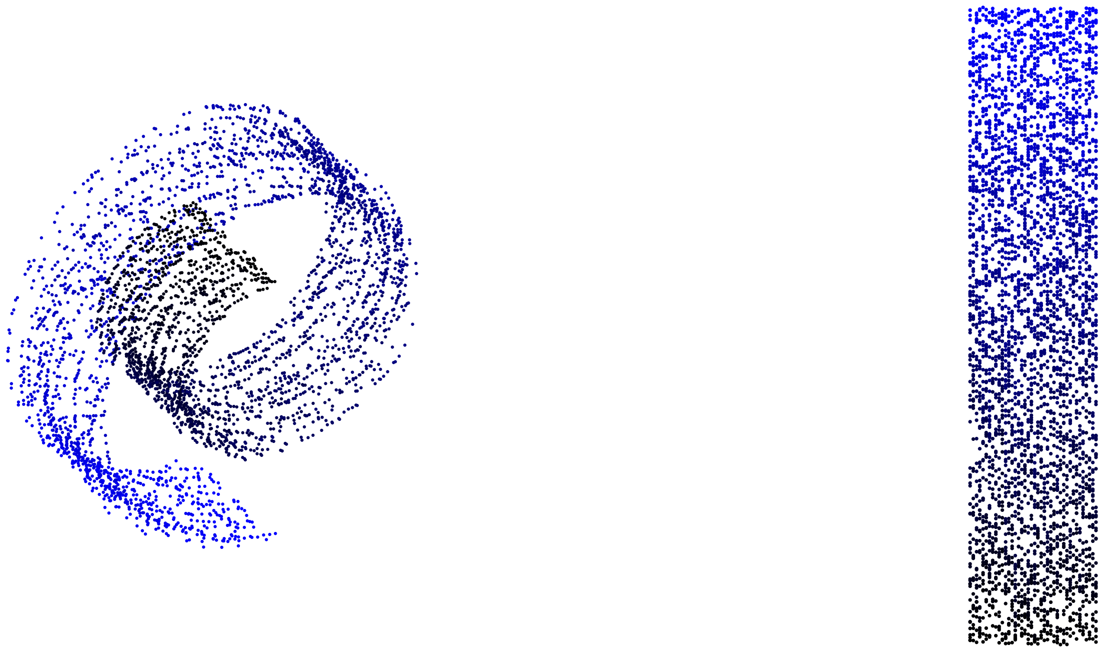
- use variables $x_1, \ldots, x_n \in \mathbf{R}^n$, with $X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} [x_1 \ \cdots \ x_n]$

- dual problem becomes **maximum variance unfolding** (MVU) problem

$$\begin{array}{ll} \text{maximize} & \sum_i \|x_i\|^2 \\ \text{subject to} & \|x_i - x_j\| \leq d_l, \quad l \sim (i,j) \\ & \sum_i x_i = 0 \end{array}$$

- position n points in $\mathbf{R}^n$ to maximize variance, while respecting local distance constraints

- similar to **semidefinite embedding** for unsupervised learning of manifolds (Weinberger & Saul 2004)



- **surprise:** duality between fastest mixing Markov process and maximum variance unfolding

Conclusions

some interesting weight optimization problems have the common form

$$\begin{array}{ll} \text{minimize} & \phi(w) = \phi(\lambda_2, \ldots, \lambda_n) \\ \text{subject to} & w \in \mathcal{W} \end{array}$$

where ϕ is symmetric convex, and $\mathcal{W}$ is closed convex

- these are convex optimization problems
- we can solve them numerically
  (up to our ability to store data, compute eigenvalues . . .)
- for some simple graphs, we can get analytic solutions
- associated dual problems can be quite interesting

Some references

- Convex Optimization
  Cambridge University Press, 2004
- Fast linear iterations for distributed averaging
  Systems and Control Letters 53:65-78, 2004
- Fastest mixing Markov chain on a graph
  SIAM Review 46(4):667-689, 2004
- Fastest mixing Markov process on a graph and a connection to a maximum variance unfolding problem
  to appear, *SIAM Review 48(4), 2006*
- Minimizing effective resistance of a graph
  to appear, *SIAM Review, 2007*
- Distributed average consensus with least-mean-square deviation
  MTNS p2768-2776, 2006

these and other papers available at `www.stanford.edu/~boyd`