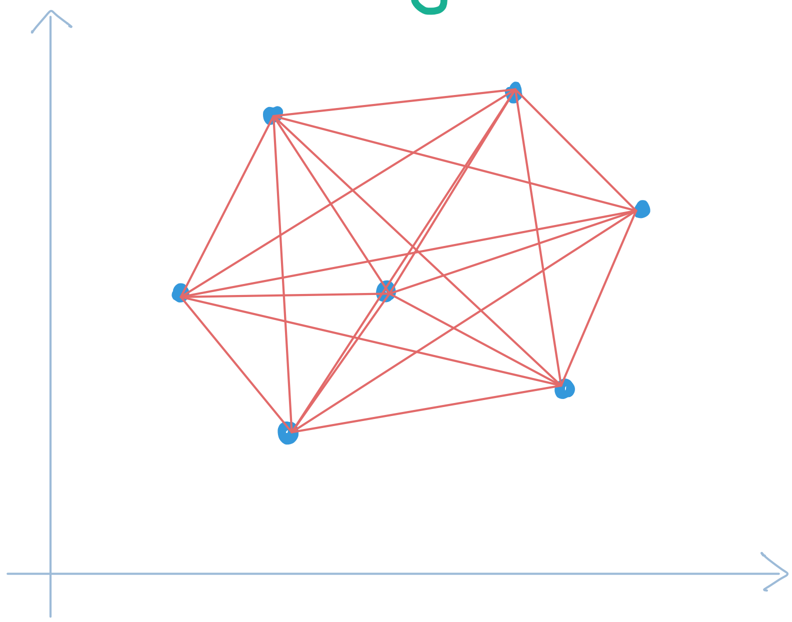
Spectral Sparsification of Metrics and Kernels

Kent Quanrud
Purdue University

Geometric graph problems



nodes = points

edges = distances

(or some function of)

$\Rightarrow$ many natural graph problems

goal: $\tilde{O}(\text{input size}) = \tilde{O}(n)$ time

(n = # points)

Obvious bottleneck:

graph dense, size $O(n^2)$

(no time to construct graph)

e.g. degrees

graph	metric
what is weighted degree of v?	what is sum of distances from v to all other vertices?

"trivial" problem in graphs

* sublinear time approximations in metrics

Indyk '99; Chechik, Cohen, Kaplan '15; Esfandiari & Mitzenmacher '18

Replacing metric w/ <u>*kernel*</u> function k

(e.g. $k(x,y) = 1/(1+\|x-y\|^2)$)

$\Rightarrow$ "density estimation"

Low-dimensions: Greengard & Rokhlin Jr., ~

High-dimensions: Holmes, Gray, & Isbell Jr. '07

Siminelakis & Charikar '17, Backurs et al., ~

Other natural problems (e.g.)

Max cut [I97, EM18 for metrics]

total sum of distances [I97, CCK15, EM18 for metrics]

min-cut?

sparsest cut?

graph embeddings?

Broad question:

Can we approximate geometric graphs in $\tilde{O}(n)$ time? (in some general sense)

Inspiration:

Spanners in low-dimensional $\mathbb{R}^d$

Sparse subgraphs where shortest path metric approximates original metric

(Salowe '91, Vaidya '91, Callahan & Kosaraju '92)

What about other structures?

The Laplacian

Quadratic form / PSD matrix L_G

associated w/ graph $G=(V,E)$

$$\langle x, L_G x\rangle = \sum_{\substack{e\in E \\ e=\{u,v\}}} w(e)\,(x_u - x_v)^2$$

($w(e)$ = weight of e)

Applications

- Encodes all cuts
- eigenvectors → embeddings
- spectral clustering
- graph partitioning

see (eg.) notes by Spielman, Trevisan

Sparsifying Laplacians

Input: graph G w/ Laplacian L_G

Goal: smaller (fewer edges) graph H w/ Laplacian L_H s.t.

$$\underset{(1-\varepsilon)}{.99}\langle x, L_G x\rangle \leq \langle x, L_H x\rangle \leq \underset{(1+\varepsilon)}{1.01}\langle x, L_G x\rangle$$

for all x

	Edges in H	Time
Spielman & Teng '04	$O(n\,\mathrm{polylog}(n))$	$\tilde{O}(m)$
Spielman & Srivastava '08	$O(n\log(n))$	$\tilde{O}(m)$
Batson, Spielman & Srivastava '09	$O(n)$	$O(\mathrm{poly}(m))$
Lee & Sun '17	$O(n)$	$\tilde{O}(m)$

+ many more related results

Geometric graphs & Laplacians

The Laplacian of any geometric graph can be approximated w/ $\tilde{O}(n)$ edges

But $\tilde{O}(m)$ time graph algorithm
$\Rightarrow$ $\tilde{O}(n^2)$ time geometric algorithm

Question Can we approximate geometric Laplacians in $\tilde{O}(n)$ time?

Primary contribution: Yes!

for metrics & smooth kernels
(more detailed bounds later)

High level approach:

1. Quick (and sometimes simple) randomized estimates of *effective resistance* by sampling
2. importance sample edges w/r/t effective resistances (per Spielman-Srivastava)

Related independent work

"Algorithms and Hardness for Linear Algebra and Geometric Graphs"

by Alman, Chu, Schild, & Song

Geometric graphs & Laplacians

The Laplacian of any geometric graph can be approximated w/ $\tilde{O}(n)$ edges

But $\tilde{O}(m)$ time graph algorithm
$\Rightarrow$ $\tilde{O}(n^2)$ time geometric algorithm

Question Can we approximate geometric Laplacians in $\tilde{O}(n)$ time?

Primary contribution: Yes!

for metrics & smooth kernels

(more detailed bounds later)

High level approach:

1. Quick (and sometimes simple) randomized estimates of *effective resistance* by sampling
2. importance sample edges w/r/t effective resistances (per Spielman-Srivastava)

Effective Resistance: a brief primer

for edge $e=\{u,v\}$, let L_e = Laplacian of (only) edge e

e.g. $\langle x, L_e x\rangle = (x_u - x_v)^2$

for graph $G=(V,E)$,

$$L_G = \sum_{e\in E} w(e) L_e$$

Effective resistance measures importance of L_e

$$(\text{eff. resist. } e) = \max_x \frac{\langle x, L_e x\rangle}{\langle x, L_G x\rangle}$$

Given effective resistances, one can importance sample edges $e \in E$ in proportion to

$$(\text{weight } e)\cdot(\text{eff. resist. } e)^{*}$$

to obtain spectral sparsifier.

* upper bounds, approximations suffice.

Metrics

A theorem, and a simple algorithm

Theorem. $G=(V,E)$

w/ edge weights $d: E \to \mathbb{R}_{>0}$ forming a metric

Let $D = \sum_{e \in E}' d(e)$ = total sum of distances.

Then for all $e \in E$,

$$(\text{effective resistance of } e) \leq \frac{O(1)n}{D}$$

Now, suppose we use upper bound when sampling:

(a) uniformity => sampling proportional to edge lengths

(b) the fact that

$$\sum_{e \in E}' d(e) \cdot \frac{n}{D} = n \sum_{e \in E}' \frac{d(e)}{D} = n$$

=> the upper bound is tight enough to produce sparse graphs.

Given theorem, remains to estimate edge lengths for sampling. Crude upper bounds suffice.

- Mitzenmacher & Esfandiari subroutine gives estimates w/ $O(n \log^2 n)$ queries
- we provide simpler alternative w/ $O(n \log n)$ queries:

1. uniformly sample & query $O(n \log n)$ edges
2. sum up, for each vertex, the lengths of incident edges that were sampled
3. for each edge, use sum of endpoints (divided by $\log n$) as estimate

easy to sample w/ these estimates

A simple algorithm (in hindsight)

1. uniformly sample edges to get weak estimates on edge lengths
2. Importance sample $O(n \log(n)/\varepsilon^2)$ edges w/r/t estimates.

Key point: Theorem about eff. resist. permits simple sampling

Smooth Kernels in $\mathbb{R}^d$

(simplifying)

$K(x,y)$ decaying polynomially in $\|x-y\|$

e.g. $K(x,y) = \dfrac{1}{1+\|x-y\|}$ "1-smooth"

$K(x,y) = \dfrac{1}{1+\|x-y\|^2}$ "2-smooth"

Classically, in low dimensions,

fast multipole method

Recent interest in high dimensions from ML.

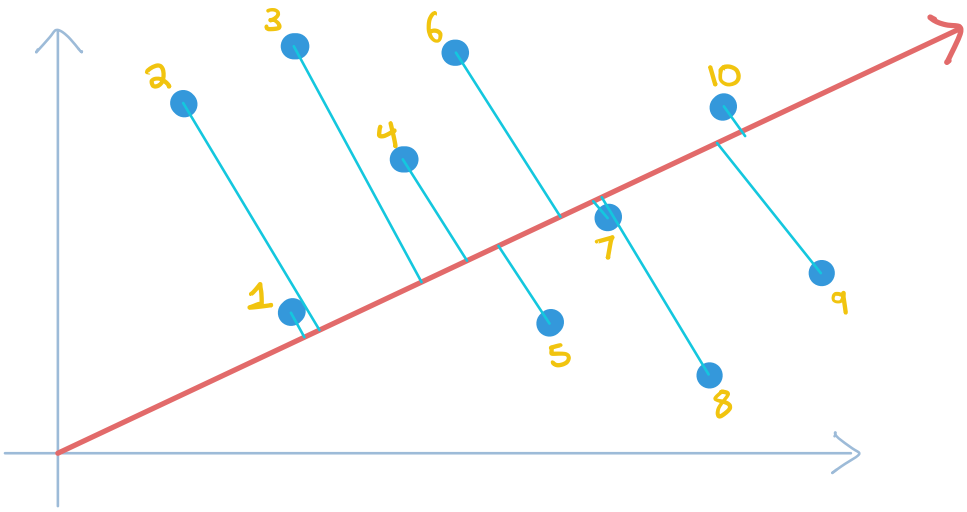
Simple algorithm for 1-smooth kernels

e.g. $k(x,y) = \frac{1}{1+\|x-y\|}$

intermediate goal: quick and small upper bounds on (eff. resist. e) $\cdot$ $k(e)$ for all $e \in E$

Theorem: let $a \sim \mathcal{N}^d$ be random Gaussian vector

rank $v \in V$ in increasing order of $\langle a, v \rangle$



for all $e = \{x,y\} \in E$,

$$(\text{eff. resist. } e) \cdot k(e) \leq O(1) \frac{1}{|\text{rank}(x) - \text{rank}(y)|}$$

w/ constant probability

Algorithmic consequences

① $O(\log n)$ random rankings

$\Rightarrow$ upper bounds w/ high probability

② the fact that

$$\sum_{x,y}{}' \frac{1}{|\mathrm{rank}(x)-\mathrm{rank}(y)|} \leq O(n \log n)$$

$\Rightarrow$ upper bounds tight enough to produce sparse graph

Thus the algorithm is:

(a) project $O(\log n)$ times $\Rightarrow$ $O(\log n)$ rankings

(b) importance sample $O(n \log^2 n)$ edges in proportion to

$$\sum_{i=1}^{O(\log n)} \frac{1}{|\mathrm{rank}_i(x)-\mathrm{rank}_i(y)|}$$

for edge $e = \{x, y\}$.

t-smooth kernels $(t \geq 1)$

e.g. $k(x,y) = \dfrac{1}{1+\|x-y\|^t}$

Similar high-level idea:

- project into $O(t)$ dimensions instead of 1
- use quadtrees to avoid all-to-all computations

- Overall running time (very roughly)

$$\tilde{O}(n(d+Q)/\varepsilon^2)$$

where Q denotes query to $k(x,y)$

$\tilde{O}(\sim)$ also hides expon. dep. on t

Byproduct: density queries

- Density queries correspond to weighted degrees
- Preserving entire Laplacian is overkill
- But by skipping unneeded steps

$\Rightarrow$ data structure competitive w/
(in fact, slightly improving)
state-of-the-art (Backurs et al.)