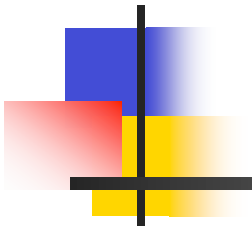


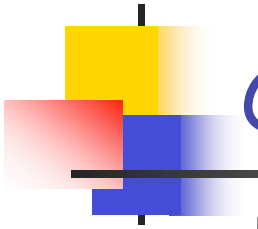
Implementing regularization implicitly via approximate eigenvector computation

Michael W. Mahoney

Stanford University

(Joint work with Lorenzo Orecchia of UC Berkeley.)

(For more info, see: http://cs.stanford.edu/people/mmahoney)



Overview (1 of 4)

Regularization in statistics, ML, and data analysis

- involves making (explicitly or implicitly) assumptions about the data
- arose in integral equation theory to "solve" ill-posed problems
- computes a better or more "robust" solution, so better inference

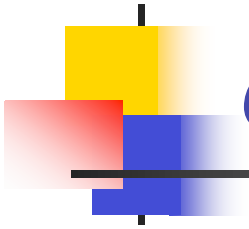
Usually *implemented* in 2 steps:

- add a norm/capacity constraint $g(x)$ to objective function $f(x)$
- then solve the modified optimization problem

$$x' = \text{argmin}_x\ f(x) + \lambda\, g(x)$$

- Often, this is a "harder" problem, e.g., L1-regularized L2-regression

$$x' = \text{argmin}_x\ ||Ax-b||_2 + \lambda\, ||x||_1$$



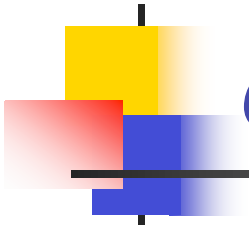
Overview (2 of 4)

Practitioners often use heuristics:

- e.g., "early stopping" or "binning"
- these heuristics often have the "side effect" of regularizing the data
- similar results seen in graph approximation algorithms (where at most linear time algorithms can be used!)

Question:

- Can we formalize the idea that performing approximate computation can *implicitly* lead to more regular solutions?



Overview (3 of 4)

Question:

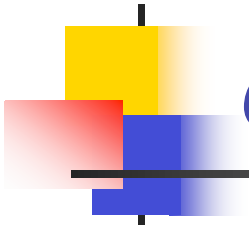
• Can we formalize the idea that performing approximate computation can *implicitly* lead to more regular solutions?

Special case today:

• Computing the first nontrivial eigenvector of a graph Laplacian?

Answer:

• Consider three random-walk-based procedures (heat kernel, PageRank, truncated lazy random walk), and show that each procedure is *implicitly* solving a regularized optimization *exactly*!



Overview (4 of 4)

What objective does the exact eigenvector optimize?

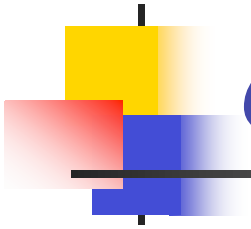
- Rayleigh quotient $R(A,x) = x^T Ax / x^T x$, for a *vector* x.
- But can also express this as an SDP, for a SPSD *matrix* X.
- We will put regularization on this SDP!

Basic idea:

- Power method starts with v_0, and iteratively computes

$$v_{t+1} = Av_t / ||Av_t||_2 \quad .$$

- Then, $v_t = \Sigma_i \gamma_i^t v_i \rightarrow v_1$.
- If we truncate after (say) 3 or 10 iterations, still have some mixing from other eigen-directions ... so don't overfit the data!



Outline

Overview

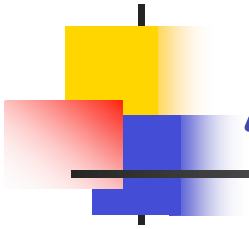
- Summary of the basic idea

Empirical motivations

- Finding clusters/communities in large social and information networks
- Empirical regularization and different graph approximation algorithms

Main technical results

- Implicit regularization defined precisely in one simple setting



A lot of loosely related* work

Machine learning and statistics

• Belkin-Niyogi-Sindhwan-06; Saul-Roweis-03; Rosasco-DeVito-Verri-05; Zhang-Yu-05; Shi-Yu-05; Bishop-95

Numerical linear algebra

• O'Leary-Stewart-Vandergraft-79; Parlett-Simon-Stringer-82

Theoretical computer science

• Spielman-Teng-04; Andersen-Chung-Lang-06; Chung-07

Internet data analysis

• Andersen-Lang-06; Leskovec-Lang-Mahoney-08; Lu-Tsaparas-Ntoulas-Polanyi-10

*"loosely related" = "very different" when the devil is in the details!

Networks and networked data

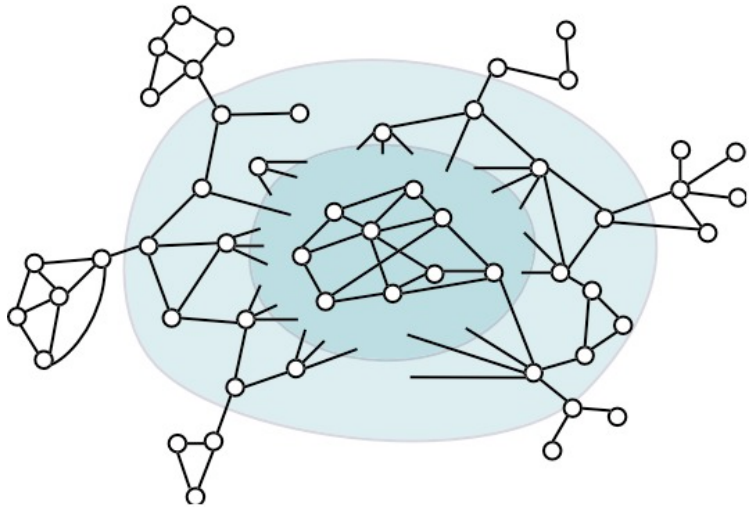
Lots of "networked" data!!

- technological networks
  - AS, power-grid, road networks
- biological networks
  - food-web, protein networks
- social networks
  - collaboration networks, friendships
- information networks
  - co-citation, blog cross-postings, advertiser-bidded phrase graphs...
- language networks
  - semantic networks...
- ...

Interaction graph model of networks:

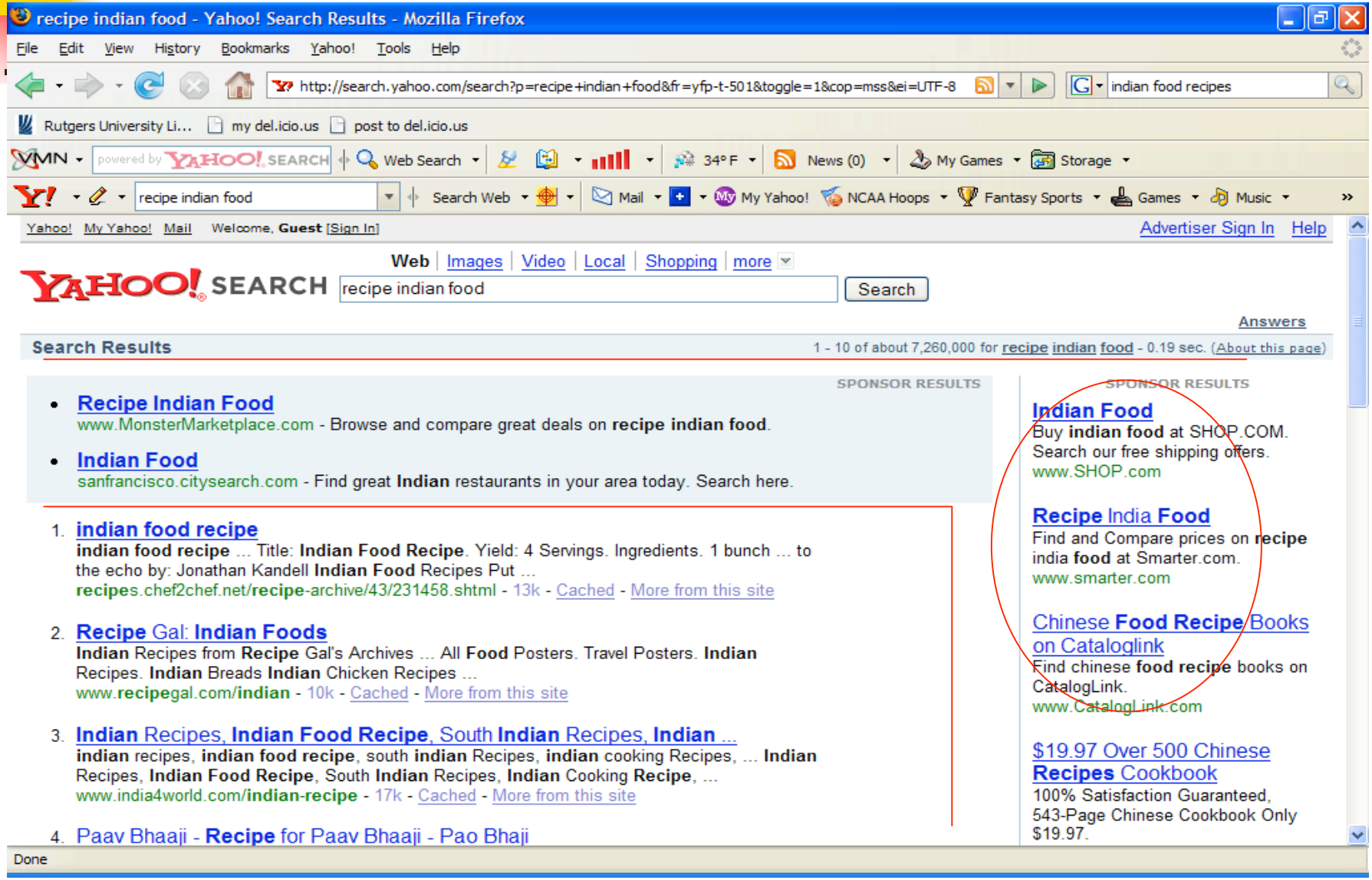
- Nodes represent "entities"
- Edges represent "interaction"

between pairs of entities

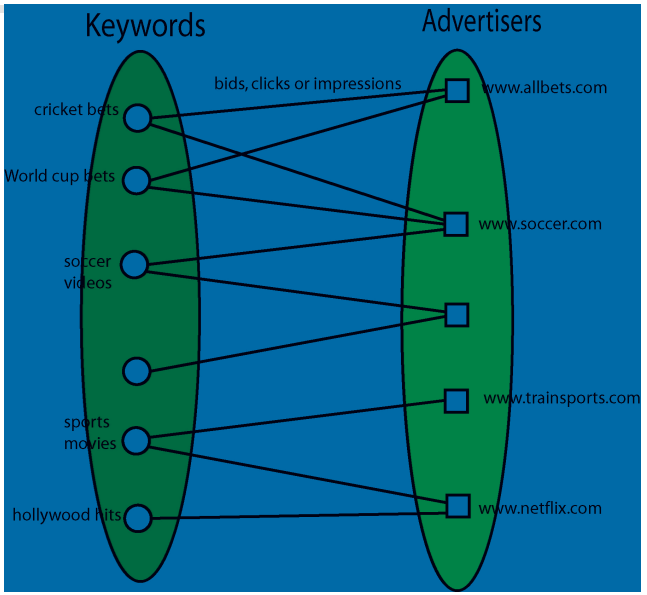


Sponsored ("paid") Search

Text-based ads driven by user query



Sponsored Search Problems



Keyword-advertiser graph:

- provide new ads
- maximize CTR, RPS, advertiser ROI

"Community-related" problems:

- Marketplace depth broadening:

  find new advertisers for a particular query/submarket

- Query recommender system:

  suggest to advertisers new queries that have high probability of clicks

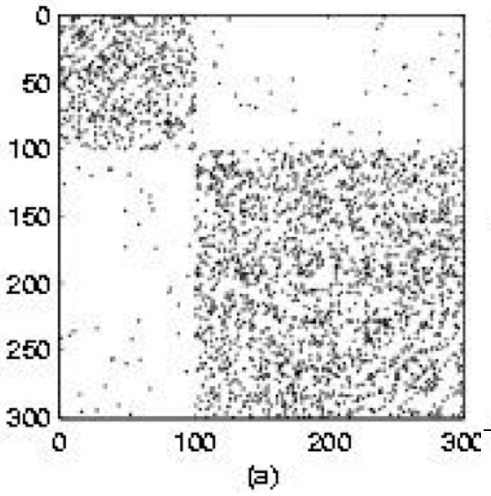
- Contextual query broadening:

  broaden the user's query using other context information

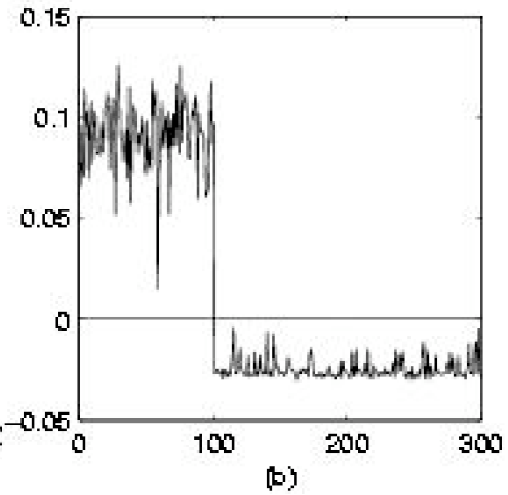
Spectral Partitioning and NCuts

$$\begin{aligned} \text{minimize} \quad & x^T L_G x \\ \text{s.t.} \quad & \langle x, x \rangle_D = 1 \\ & \langle x, 1 \rangle_D = 0 \end{aligned}$$

Adjacency matrix



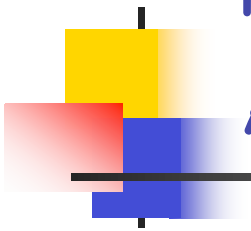
Eigenvector q_2



- Solvable via eigenvalue problem
- Bounds via Cheeger's inequality
- Used in parallel scientific computing, Computer Vision (called Normalized Cuts), and Machine Learning

- *But, what if there are not "good well-balanced" cuts (as in "low-dim" data)?*

Probing Large Networks with Approximation Algorithms



Idea: Use approximation algorithms for NP-hard graph partitioning problems as experimental probes of network structure.

Spectral - (quadratic approx) - confuses "long paths" with "deep cuts"

Multi-commodity flow - (log(n) approx) - difficulty with expanders

SDP - (sqrt(log(n)) approx) - best in theory

Metis - (multi-resolution for mesh-like graphs) - common in practice

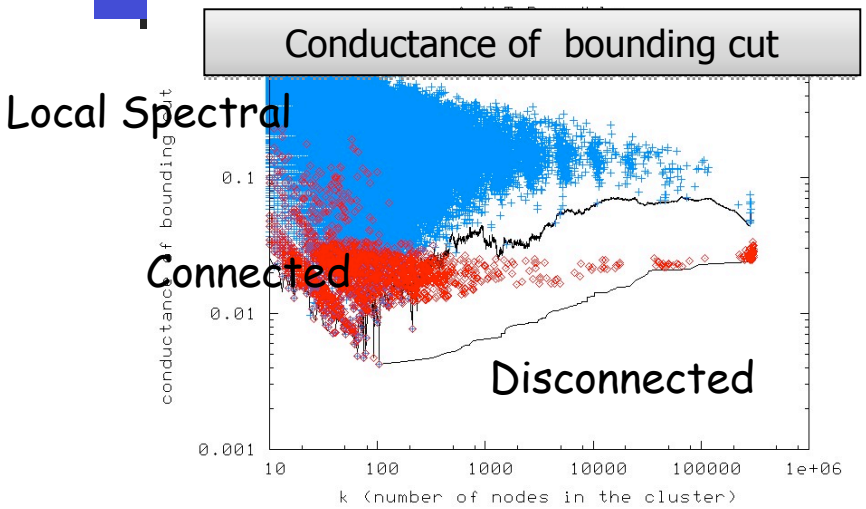
X+MQI - post-processing step on, e.g., Spectral of Metis

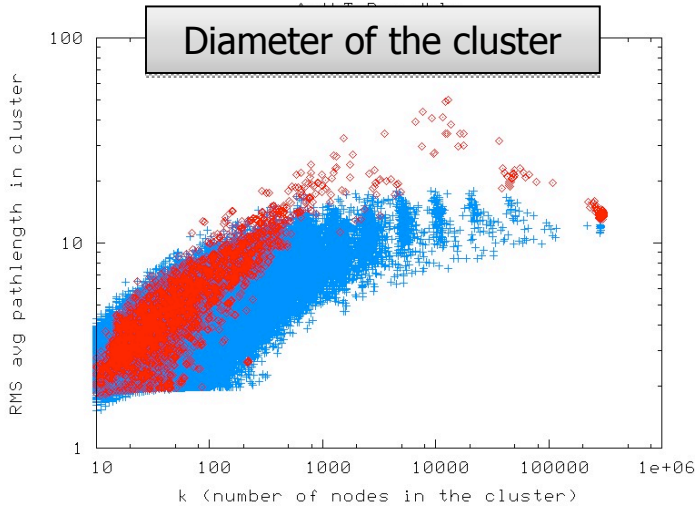
Metis+MQI - best conductance (empirically)

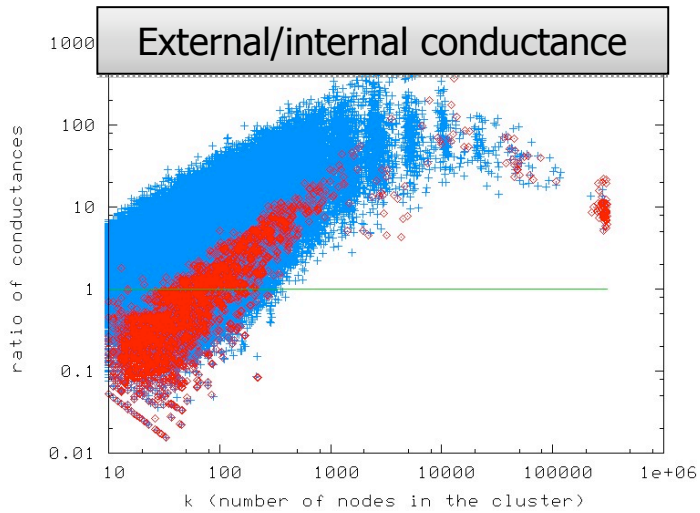
Local Spectral - connected and tighter sets (empirically, regularized communities!)

We are not interested in partitions per se, but in probing network structure.

Regularized and non-regularized communities (1 of 2)





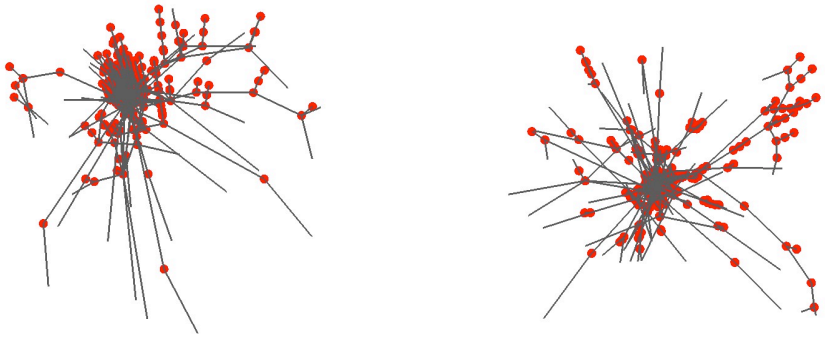


- Metis+MQI (red) gives sets with better conductance.
- Local Spectral (blue) gives tighter and more well-rounded sets.

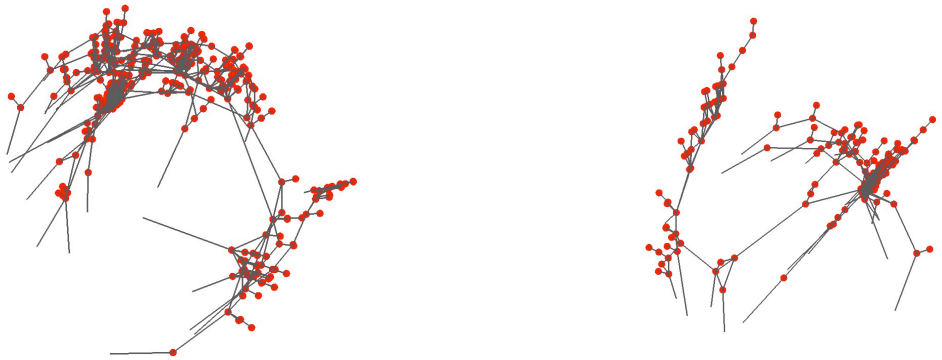
Lower is good

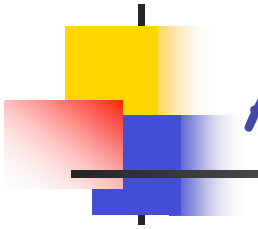
Regularized and non-regularized communities (2 of 2)

Two ca. 500 node communities from Local Spectral Algorithm:



Two ca. 500 node communities from Metis+MQI:





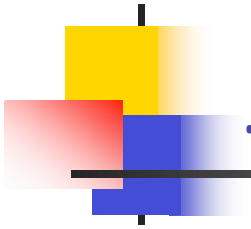
Approximate eigenvector computation …

Many uses of Linear Algebra in ML and Data Analysis involve *approximate* computations

- Power Method, Truncated Power Method, HeatKernel, Truncated Random Walk, PageRank, Truncated PageRank, Diffusion Kernels, TrustRank, etc.
- Often they come with a "generative story," e.g., random web surfer, teleportation preferences, drunk walkers, etc.

What are these procedures *actually* computing?

- E.g., what optimization problem is 3 steps of Power Method solving?
- Important to know if we really want to "scale up"



… and *implicit* regularization

Regularization: A general method for computing "smoother" or "nicer" or "more regular" solutions - useful for inference, etc.

Recall: Regularization is usually *implemented* by adding "regularization penalty" and optimizing the new objective.

$$\hat{x} = \text{argmin}_x f(x) + \lambda g(x)$$

Empirical Observation: Heuristics, e.g., binning, early-stopping, etc. often implicitly perform regularization.

Question: Can approximate computation* *implicitly* lead to more regular solutions? If so, can we exploit this algorithmically?

*Here, consider approximate eigenvector computation. But, can it be done with graph algorithms?

Views of approximate spectral methods

Three common procedures (L=Laplacian, and M=r.w. matrix):

- Heat Kernel: $H_t = \exp(-tL) = \sum_{k=0}^{\infty} \frac{(-t)^k}{k!} L^k$
- PageRank: $\pi(\gamma, s) = \gamma s + (1-\gamma) M \pi(\gamma, s)$

  $R_\gamma = \gamma \left(I - (1-\gamma) M\right)^{-1}$
- q-step Lazy Random Walk: $W_\alpha^q = (\alpha I + (1-\alpha) M)^q$

Ques: Do these "*approximation* procedures" *exactly* optimizing some regularized objective?

Two versions of spectral partitioning

VP:

$$\begin{aligned} \text{min.} \quad & x^T L_G x \\ \text{s.t.} \quad & x^T L_{K_n} x = 1 \\ & < x, 1 >_D = 0 \end{aligned}$$

R-VP:

$$\begin{aligned} \text{min.} \quad & x^T L_G x + \lambda f(x) \\ \text{s.t.} \quad & constraints \end{aligned}$$

Two versions of spectral partitioning

VP: $\longleftrightarrow$ **SDP:**

$$\begin{aligned} \text{min.} \quad & x^T L_G x \\ \text{s.t.} \quad & x^T L_{K_n} x = 1 \\ & < x, 1 >_D = 0 \end{aligned} \qquad \begin{aligned} \text{min.} \quad & L_G \circ X \\ \text{s.t.} \quad & L_{K_n} \circ X = 1 \\ & X \succeq 0 \end{aligned}$$

R-VP: **R-SDP:**

$$\begin{aligned} \text{min.} \quad & x^T L_G x + \lambda f(x) \\ \text{s.t.} \quad & constraints \end{aligned} \qquad \begin{aligned} \text{min.} \quad & L_G \circ X + \lambda F(X) \\ \text{s.t.} \quad & constraints \end{aligned}$$

A simple theorem

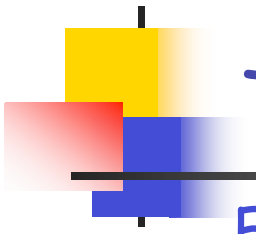
Mahoney and Orecchia (2010)

$$\text{(F,}\eta\text{)-SDP} \quad \min \quad L \bullet X + \frac{1}{\eta} \cdot F(X)$$
$$\text{s.t.} \quad I \bullet X = 1$$
$$X \succeq 0$$

Modification of the usual SDP form of spectral to have regularization (but, on the matrix X, not the vector x).

Theorem: Let G be a connected, weighted, undirected graph, with normalized Laplacian L. Then, the following conditions are sufficient for $X^\star$ to be an optimal solution to (F,η)-SDP.

- $X^\star = (\nabla F)^{-1}\left(\eta \cdot (\lambda^* I - L)\right)$, for some $\lambda^* \in R$,
- $I \bullet X^\star = 1$,
- $X^\star \succeq 0$.



Three simple corollaries

$F_H(X) = Tr(X \log X) - Tr(X)$ (i.e., generalized entropy)

gives scaled Heat Kernel matrix, with $t = \eta$

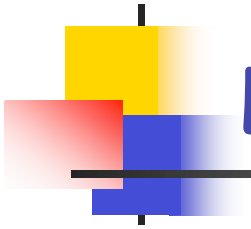
$F_D(X) = -\log\det(X)$ (i.e., Log-determinant)

gives scaled PageRank matrix, with $t \sim \eta$

$F_p(X) = (1/p)||X||_p^p$ (i.e., matrix p-norm, for p>1)

gives Truncated Lazy Random Walk, with $\lambda \sim \eta$

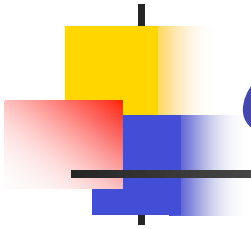
Answer: These "approximation procedures" compute regularized versions of the Fiedler vector *exactly*!



Large-scale applications

A lot of work on large-scale data already *implicitly* uses these ideas:

- Fuxman, Tsaparas, Achan, and Agrawal (2008): random walks on query-click for automatic keyword generation
- Najork, Gallapudi, and Panigraphy (2009): carefully "whittling down" neighborhood graph makes SALSA faster and better
- Lu, Tsaparas, Ntoulas, and Polanyi (2010): test which page-rank-like implicit regularization models are most consistent with data



Conclusion

Main technical result

• Approximating an exact eigenvector is exactly optimizing a regularized objective function

More generally

• Can regularization as a function of different graph approximation algorithms (seen empirically) be formalized?

• If yes, can we construct a toolbox (since, e.g., spectral and flow regularize differently) for interactive analytics on very large graphs?