Using Local Spectral Methods in Theory and in Practice

Michael W. Mahoney

ICSI and Dept of Statistics, UC Berkeley

(For more info, see:
http://www.stat.berkeley.edu/~mmahoney
or Google on "Michael Mahoney")

December 2015

Global spectral methods

Given the Laplacian L of a graph $G = (V, E)$, solve the following:

$$\begin{aligned} \text{minimize} \quad & x^T L x \\ \text{subject to} \quad & x^T D x = 1 \\ & x^T D \mathbb{1} = 0 \end{aligned}$$

Good News: Things one can prove about this.

- Solution can be found by computing an eigenvector of L.
- Solution can be found by running a random walk to ∞.
- Solution is "quadratically good" (i.e., Cheeger's Inequality).
- Solution can be used for clustering, classification, ranking, etc.

Bad News: This is a very *global* thing and so often not useful.

- Can we localize it in some meaningful way?

Outline

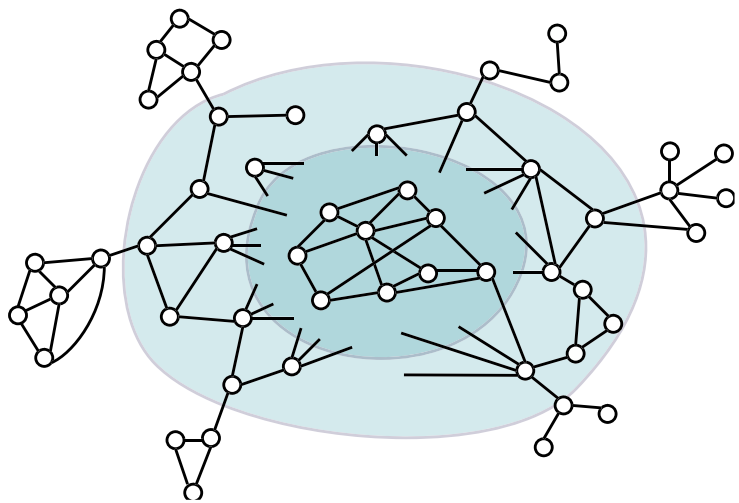
Networks and networked data

Lots of "networked" data!!

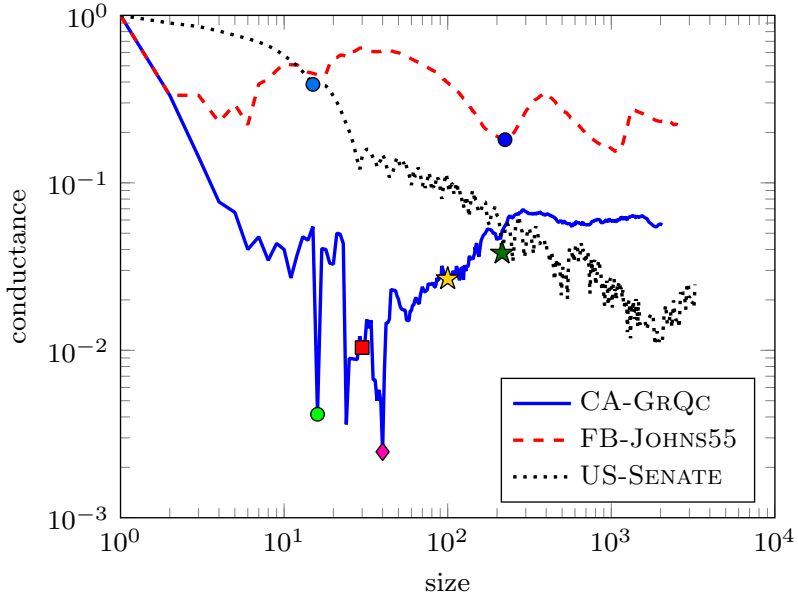
- technological networks (AS, power-grid, road networks)
- biological networks (food-web, protein networks)
- social networks (collaboration networks, friendships)
- information networks (co-citation, blog cross-postings, advertiser-bidded phrase graphs ...)
- language networks (semantic networks ...)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

Interaction graph model of networks:

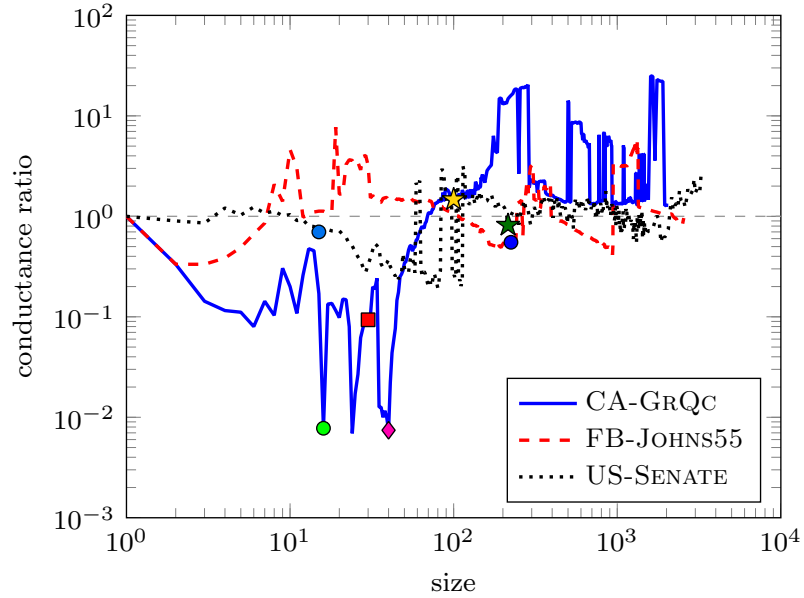
- Nodes represent "entities"
- Edges represent "interaction" between pairs of entities



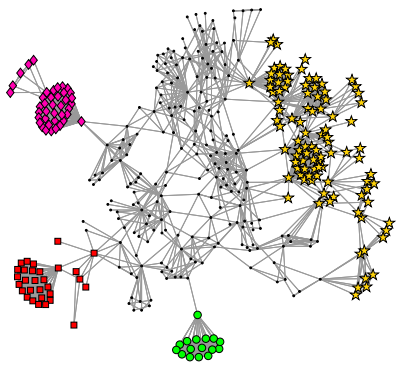
Three different types of real networks



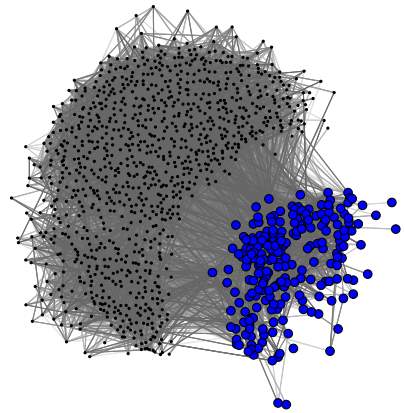
(a) NCP: conductance value of best conductance set, as function of size



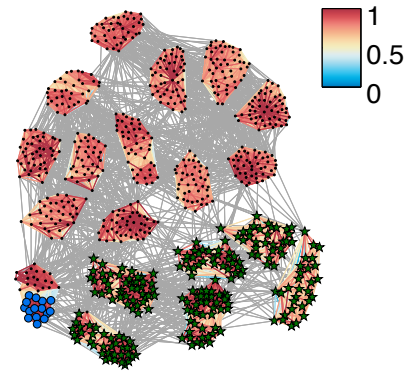
(b) CRP: ratio of internal to external conductance, as function of size



(c) CA-GrQc

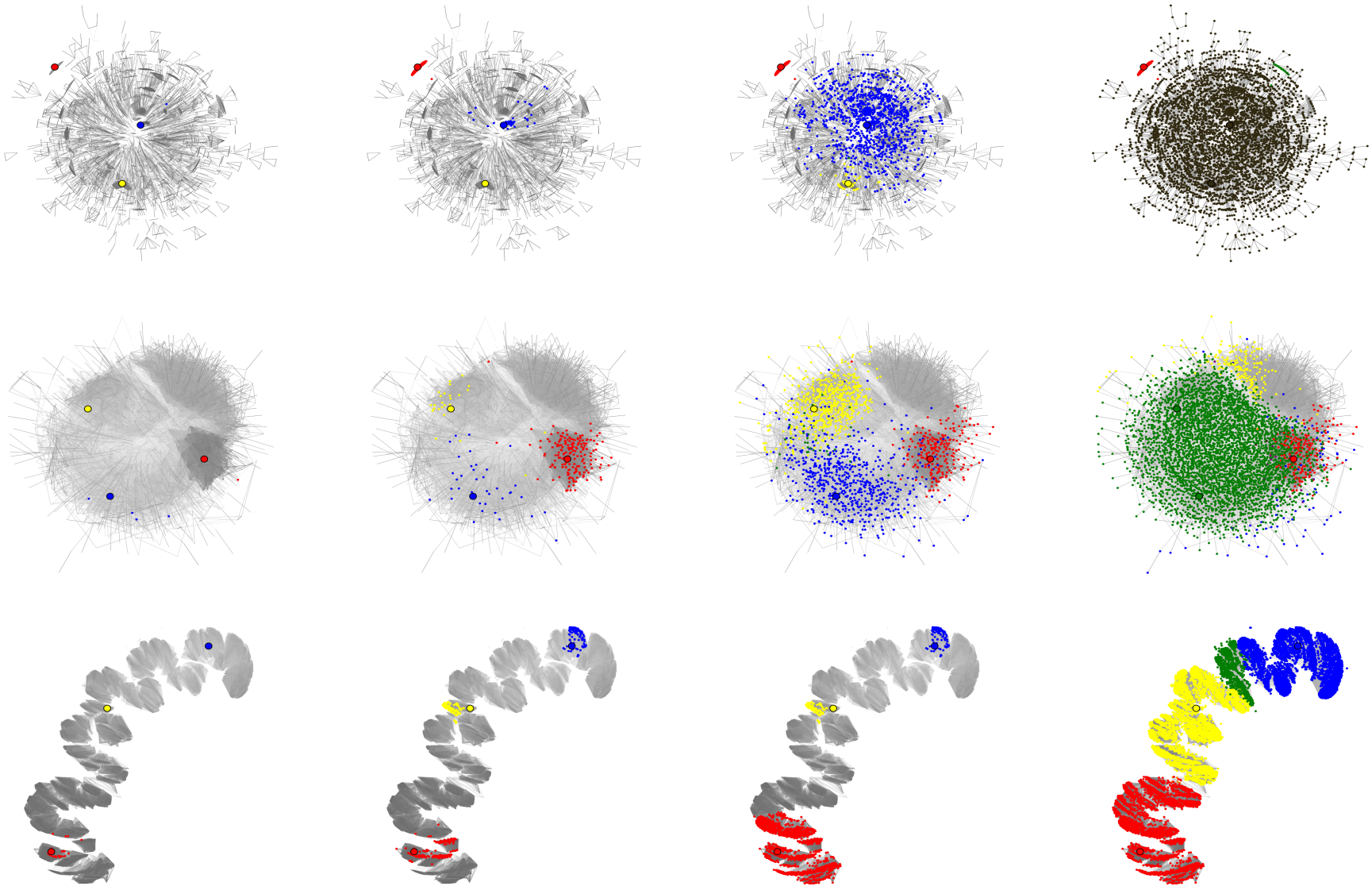


(d) FB-Johns55



(e) US-Senate

Information propagates local-to-glocal in different networks in different ways



Outline

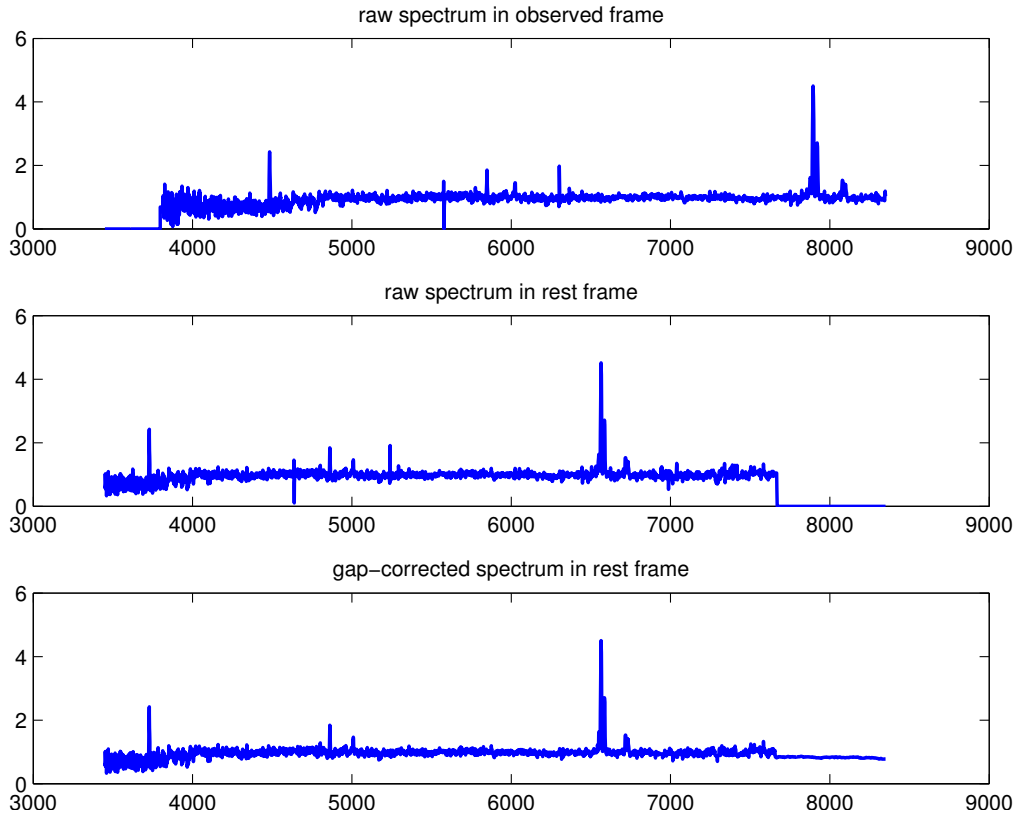
Use case[1]: Galactic spectra from SDSS

$x_i \in \mathbb{R}^{3841}$, $N \approx 500k$

photon fluxes in ≈ 10 Å wavelength bins

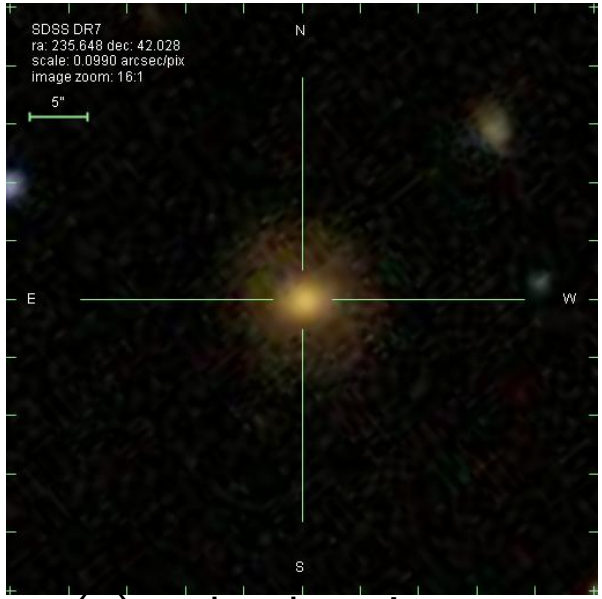
preprocessing corrects for redshift, gappy regions

normalized by median flux at certain wavelengths

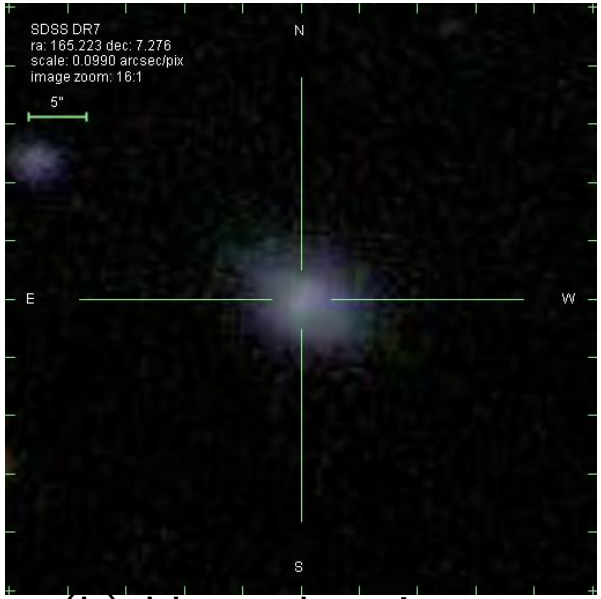


[1] Also results in neuroscience as well as genetics and mass spec imaging.

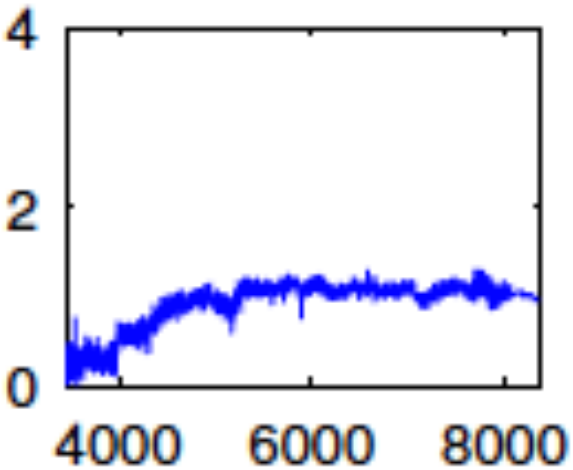
Red vs. blue galaxies



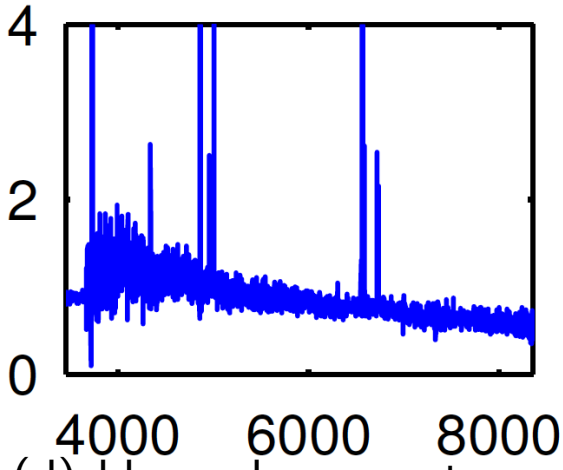
(a) red galaxy image



(b) blue galaxy image



(c) red galaxy spectrum



(d) blue galaxy spectrum

Dimension reduction in astronomy

An incomplete history:

- (Connolly et al., 1995): principal components analysis
- (Vanderplas & Connolly, 2009): locally linear embedding
- (Richards et al., 2009): diffusion maps regression
- (Yip, Mahoney, et al., 2013): CUR low-rank decompositions

Here:

- Apply global/local spectral methods to astronomical spectra
- Address both nonlinear structure and nonuniform density
- Explore locally-biased versions of these embeddings for a downstream classification task

Constructing global diffusion embeddings

(Belkin & Niyogi, 2003; Coifman & Lafon, 2006)

Given data $\{x_j\}_{j=1}^N$, form graph with edge weights

$$W_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{\varepsilon_i \varepsilon_j}\right), \quad D_{ii} = \sum_j W_{ij}$$

In practice, only add k nearest-neighbor edges

Set ε_i = distance to point i's $k/2$ nearest-neighbor

The "lazy" transition matrix is $M = \frac{1}{2} D^{-1/2}(D + W)D^{-1/2}$

Embedding given by leading non-trivial eigenvectors of M

Global embedding: effect of k

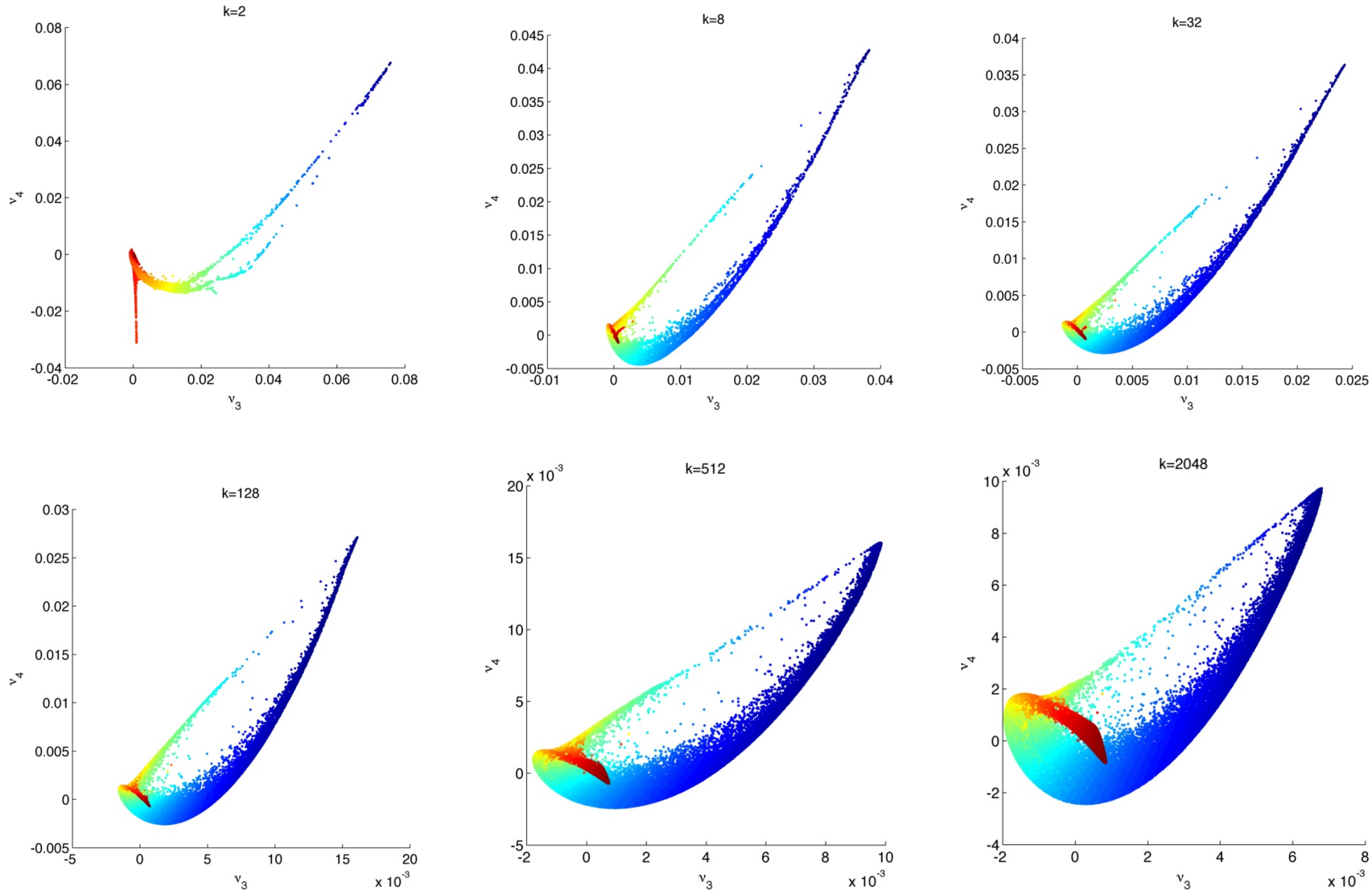
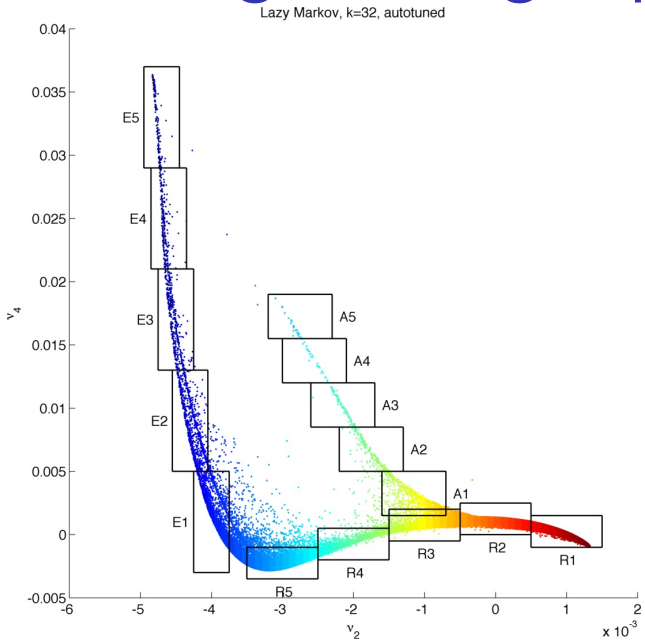
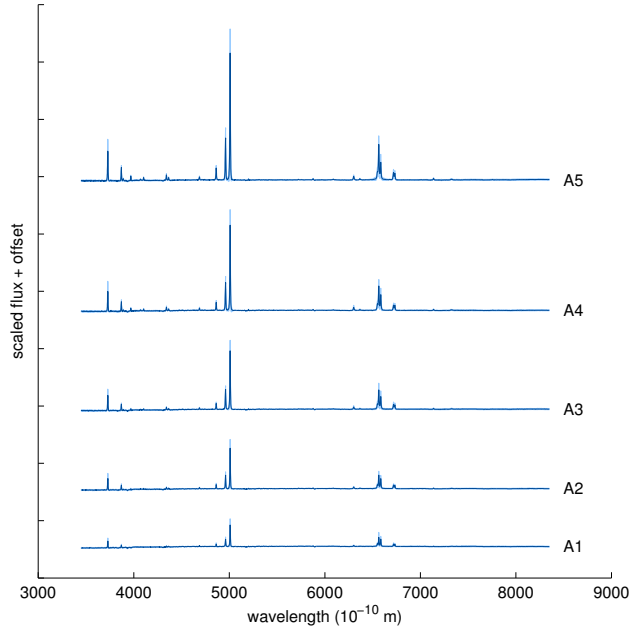
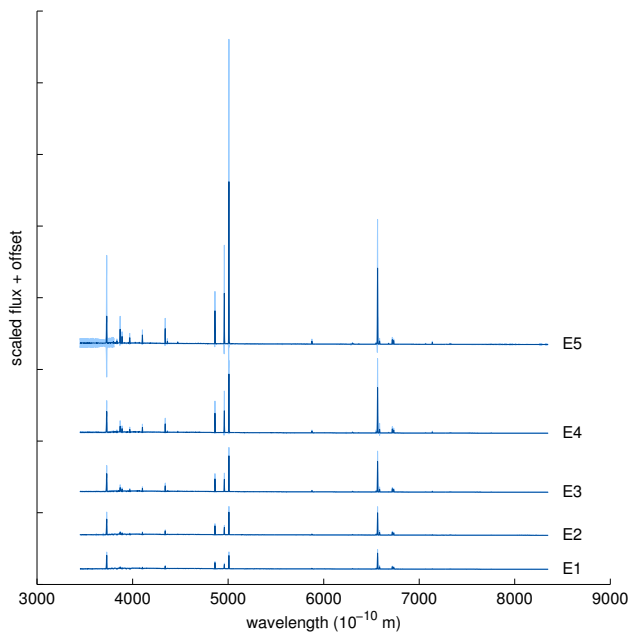


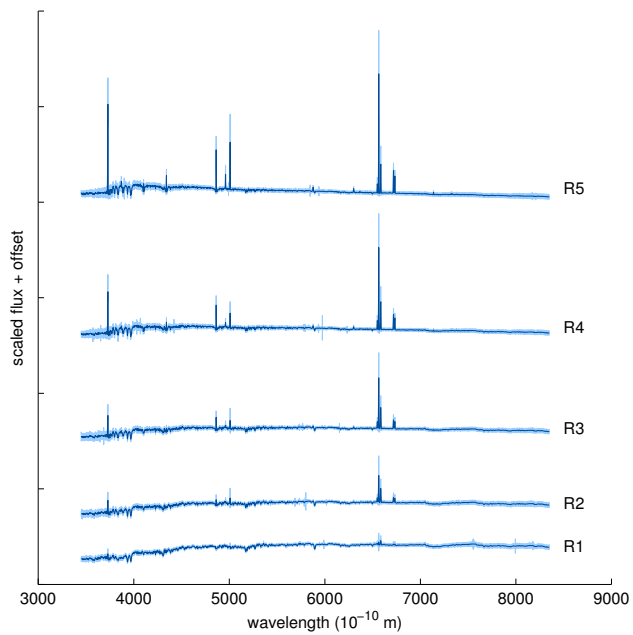
Figure: Eigenvectors 3 and 4 of Lazy Markov operator, $k = 2 : 2048$

Global embedding: average spectra









Optimization approach to global spectral methods

Markov matrix related to combinatorial graph Laplacian L:

$$L \stackrel{\text{def}}{=} D - W = 2D^{1/2}(I - M)D^{1/2}$$

Can write v_2, the first nontrivial eigenvector of L, as the solution to

$$\begin{aligned} \text{minimize} \quad & x^T L x \\ \text{subject to} \quad & x^T D x = 1 \\ & x^T D \mathbb{1} = 0 \end{aligned}$$

Similarly for v_t with additional constraints $x^T D v_j = 0,\ j < t$.

Theorem. Solution can be found by computing an eigenvector. It it is "quadratically good" (i.e., Cheeger's Inequality).

MOV optimization approach to local spectral methods

(Mahoney, Orecchia, and Vishnoi, 2009; Hansen and Mahoney, 2013; Lawlor, Budavari, Mahoney, 2015)

Suppose we have:

1. a seed vector $s = \chi_S$, where S is a subset of data points
2. a correlation parameter κ

MOV objective. The first semi-supervised eigenvector w_2 solves:

$$\begin{aligned} \text{minimize} \quad & x^T L x \\ \text{subject to} \quad & x^T D x = 1 \\ & x^T D \mathbb{1} = 0 \\ & x^T D s \geq \sqrt{\kappa} \end{aligned}$$

Similarly for w_t with addition constraints $x^T D w_j = 0,\ j < t$.

Theorem. Solution can be found by solving a linear equation. It is "quadratically good" (with a local version of Cheeger's Inequality).

Local embedding: scale parameter and effect of seed

For an appropriate choice of c and $\gamma = \gamma(\kappa) < \lambda_2$, one can show

$$\begin{aligned} w_2 &= c(L - \gamma D)^+ Ds \\ &= c(L_G - \gamma L_{k_n})^+ Ds \end{aligned}$$

(In practice, binary search to find "correct" γ.)

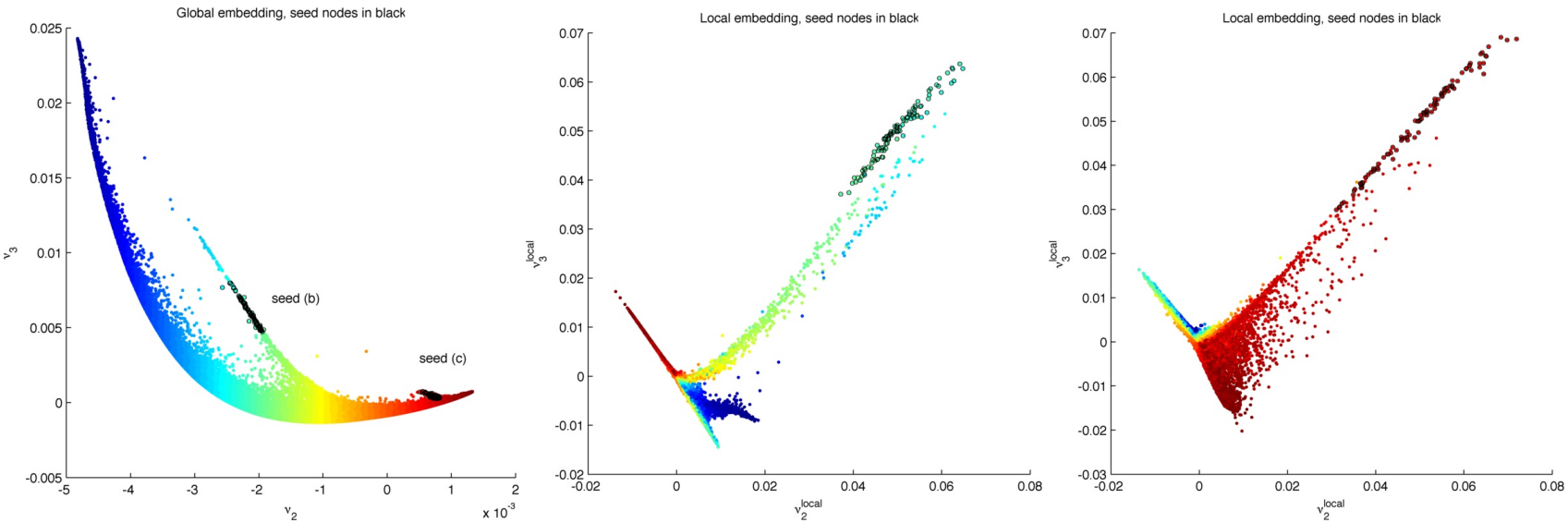


Figure: (left) Global embedding with seeds in black. (middle, right) Local embeddings using specified seeds.

Classification on global and local embeddings

Try to reproduce 5 astronomer-defined classes

Train multiclass logistic regression on global and local embeddings

Seeds chosen to discriminate one class (e.g., AGN) vs. rest

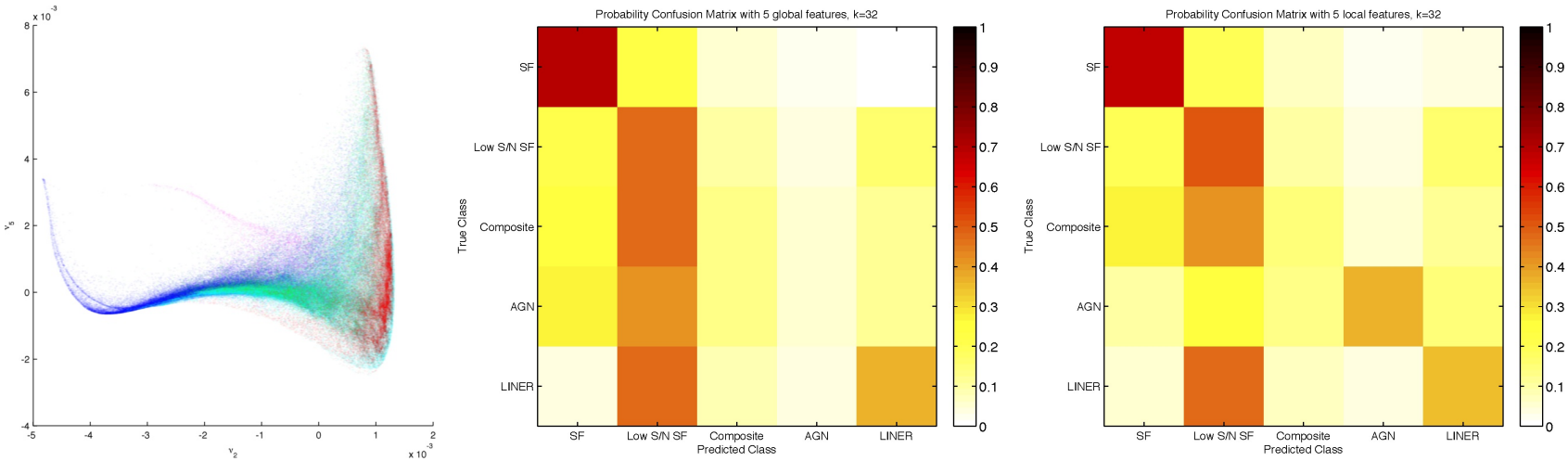


Figure: (left) Global embedding, colored by class. (middle, right) Confusion matrices for classification on global and local embeddings.

Outline

Local versus global spectral methods

Global spectral methods:

- Compute eigenvectors of matrices related to the graph.
- Provide "quadratically good" approximations to the best partitioning of the graph (Cheeger's Inequality).
- This provide inference control for classification, regression, etc.

Local spectral methods:

- Use random walks to find locally-biased partitions in large graphs.
- Can prove locally-biased versions of Cheeger's Inequality.
- Scalable worst-case running time; non-trivial statistical properties.

Success stories for local spectral methods:

- Getting nearly-linear time Laplacian-based linear solvers.
- For finding local clusters in very large graphs.
- For analyzing large social and information graphs.

Two different types of local spectral methods

Strongly-local methods:

- ST; ACL; C; AP: run short random walks.
- Theorem: If there is a small cluster near the seed node, they you will find it, otherwise you will stop; and running time depends on size of output, not the graph.
- You don't even touch most of the nodes in a large graph.
- Very good in practice, especially the ACL push algorithm.

Weakly-local methods:

- MOV; HM: optimization objective with locality constraints.
- Theorem: If there is a small cluster near the seed node, they you will find it, otherwise you will stop; and running time depends on time to solve linear systems.
- You do touch all of the nodes in a large graph.
- Many semi-supervised learning methods have similar form.

The ACL push procedure

1. $\vec{x}^{(1)} = 0, \vec{r}^{(1)} = (1-\beta)\vec{e}_i$, $k = 1$
2. *while* any $r_j > \tau d_j$ $\quad$ d_j *is the degree of node* j
3. $\quad \vec{x}^{(k+1)} = \vec{x}^{(k)} + (r_j - \tau d_j \rho)\vec{e}_j$
4. $$\vec{x}_i^{(k+1)} = \begin{cases} \tau d_j \rho & i = j \\ r_i^{(k)} + \beta(r_j - \tau d_j \rho)/d_j & i \sim j \\ r_i^{(k)} & \text{otherwise} \end{cases}$$
5. $\quad k \leftarrow k + 1$

Things to note:

- This approximates the solution to the personalized PageRank problem:
  - $(I - \beta A D^{-1})\vec{x} = (1-\beta)\vec{v}$;
  - $(I - \beta \mathcal{A})\vec{y} = (1-\beta)D^{-1/2}\vec{v}$, where $\begin{cases} \mathcal{A} = D^{-1/2} A D^{-1/2} \\ \vec{x} = D^{1/2}\vec{y} \end{cases}$
  - $[\alpha D + L]\vec{z} = \alpha \vec{v}$, where $\beta = 1/(1+\alpha)$ and $\vec{x} = D\vec{z}$.
- The global invariant $\vec{r} = (1-\beta)\vec{v} - (I - \beta A D^{-1})\vec{x}$ is maintained throughout, even though $\vec{r}$ and $\vec{x}$ are supported locally.
- *Question: What does this algorithm compute—approximately or exactly?*

ACL theory, TCS style

Informally, here is the ACL algorithm:

- Diffuse from a localized seed set of nodes.
- Maintain two localized vectors such that a global invariant is satisfied.
- Stop according to a stopping rule.

Informally, here is the ACL worst-case theory:

- If there is a good conductance cluster near the initial seed set, then the algorithm will find it, and otherwise it will stop.
- The output satisfied Cheeger-like quality-of-approximation guarantees.
- The running time of the algorithm depends on the size of the output but is independent of the size of the graph.

Note: This is an approximation algorithm.
Question: What does this algorithm compute—exactly?

Constructing graphs that algorithms implicitly optimize

Given $G = (V, E)$, add extra nodes, s and t, with weights connected to nodes in $S \subset V$ or to $\bar{S}$.

Then, the s, t-minimum cut problem is:

$$\min_{x_s=1, x_t=0} \|B\vec{x}\|_{C,1} = \sum_{(u,v)\in E} C_{(u,v)}|x_u - x_v|$$

The ℓ_2-minorant of this problem is:

$$\min_{x_s=1, x_t=0} \|B\vec{x}\|_{C,2} = \sqrt{\sum_{(u,v)\in E} C_{(u,v)}|x_u - x_v|^2}$$

or, equivalently, of this problem:

$$\min_{x_s=1, x_t=0} \frac{1}{2}\|B\vec{x}\|_{C,2}^2 = \frac{1}{2}\sum_{(u,v)\in E} C_{(u,v)}|x_u - x_v|^2 = \frac{1}{2}\vec{x}^T L\vec{x}$$

Implicit regularization in the ACL approximation algorithm

Let $B(S)$ be the incidence matrix for the "localized cut graph," for vertex subset S, and let $C(\alpha)$ be the edge-weight matrix.

Theorem. The PageRank vector $\vec{z}$ that solves $(\alpha D + L)\vec{z} = \alpha\vec{v}$, with $\vec{v} = \vec{d}_S/\mathrm{vol}(S)$ is a renormalized solution of the 2-norm cut computation:

$$\min_{x_s=1,x_t=0} \|B(S)\vec{x}\|_{C(\alpha),2}.$$

Theorem. Let $\vec{x}$ be the output from the ACL push procedure, set parameters right, and let $\vec{z}_G$ be the solution of:

$$\min_{z_s=1,z_t=0,\vec{z}\geq 0} \frac{1}{2}\|B(S)\vec{z}\|^2_{C(\alpha),2} + \kappa\|D\vec{z}\|_1,$$

where $\vec{z} = (1 \quad \vec{z}_G \quad 0)^T$. Then $\vec{x} = D\vec{z}_G/\mathrm{vol}(S)$.

Outline

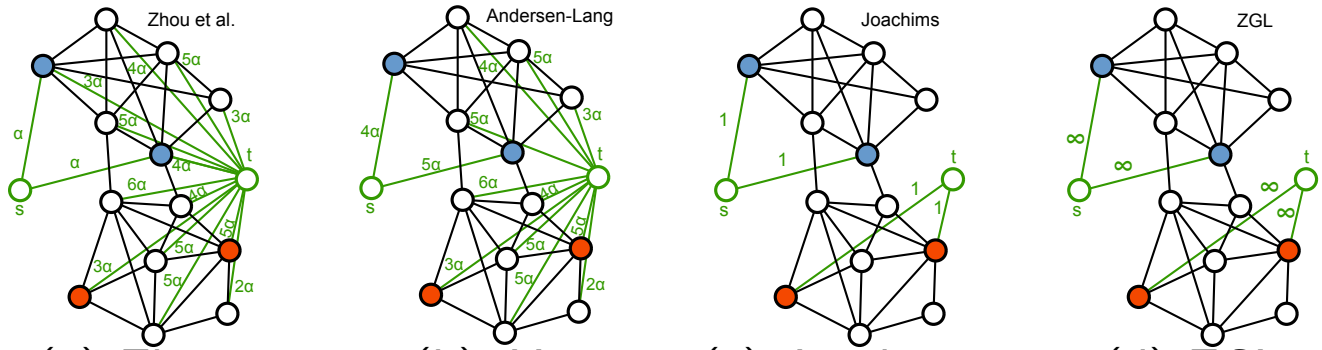
Problems that arise with explicit graph construction

Question: What is the effct of "noise" or "perturbations" or "arbitrary decisions" in the construction of the graph on the output of the subsequent graph algorithm.

Common problems with constructing graphs.

- Problems with edges/nonedges in explicit graphs (where arbitrary decisions are hidden from the user).
- Problems with edges/nonedges in constructed graphs (where arbitrary decisions are made by the user).
- Problems with labels associated with the nodes or edges (since the "ground truth" may be unreliable).

Semi-supervised learning and implicit graph construction



(a) Zhou (b) AL (c) Joachims (d) ZGL

Figure: The s, t-cut graphs associated with four different constructions for semi-supervised learning on a graph. The labeled nodes are indicated by the blue and red colors. This construction is to predict the blue-class.

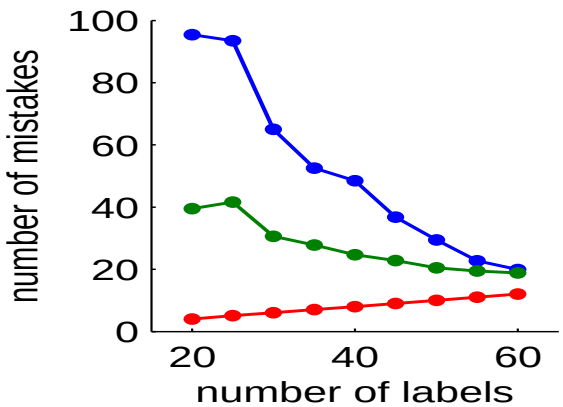
To do semi-supervised learning, these methods propose a diffusion equation:

$$Y = (L + \alpha D)^{-1} S.$$

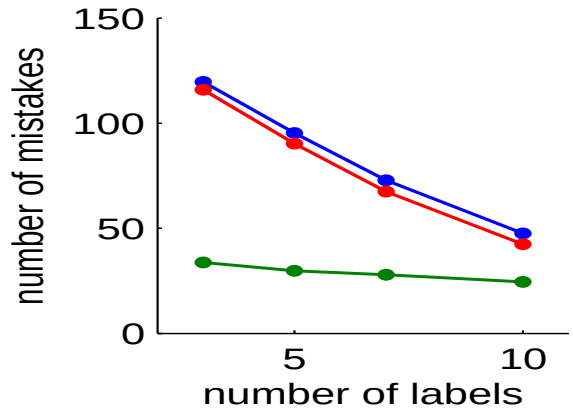
This equation "propagates" labels from a small set S of labeled nodes.

- This is equivalent to the minorant of an s, t-cut problem where the problem varies based on the class.
- This is *exactly* the MOV local spectral formulation—that ACL **approximates** the solution of and **exactly solves** a regularized version of.

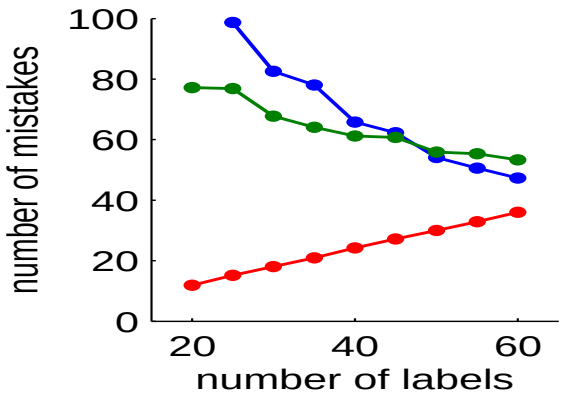
Comparing diffusions: sparse and dense graphs, low and high error rates



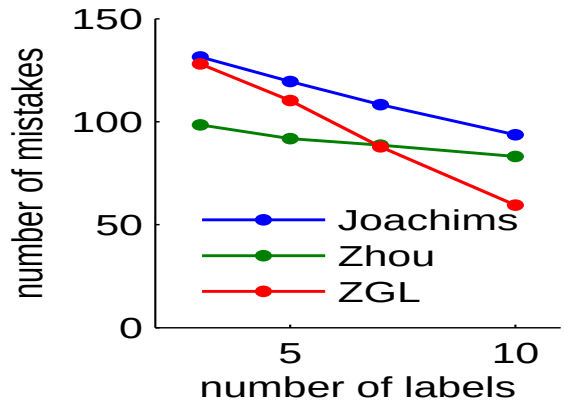
(a) dense; low label error



(b) sparse; low label error

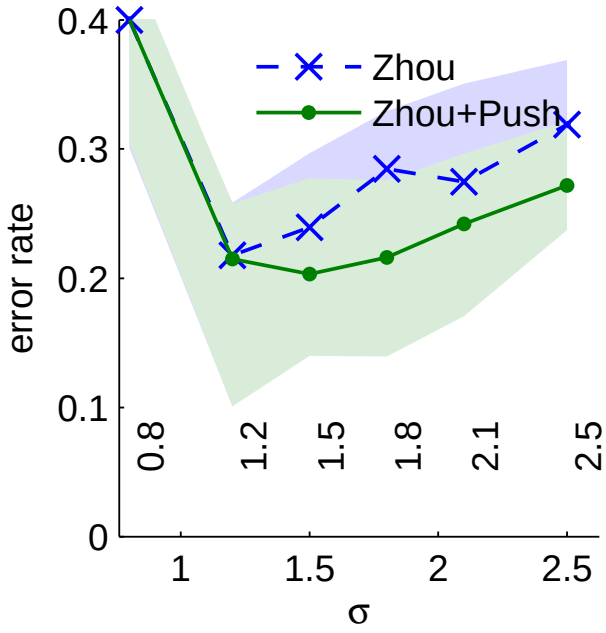


(c) dense; high label error

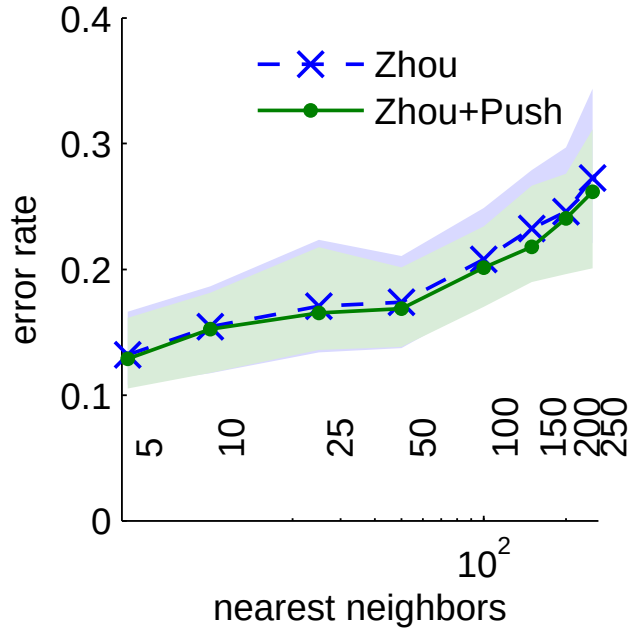


(d) sparse; high label error

Case study with toy digits data set



(e) Varying σ



(f) Varying nearest neighbors

Performance of the diffusions while varying the density by changing (a) $\boldsymbol{\sigma}$ or (b) varying r in the nearest neighbor construction. In both cases, making the graph "denser" results in worse performance.

Densifying sparse graphs with matrix polynomials

Do it the usual way:

- Vary the kernel density width parameter σ.
- Convert the weighted graph into a highly sparse unweighted graph through a nearest neighbor construction.

Do it by coundint paths of different lengths:

- Run *the A_k construction*: given a graph with adjacency matrix A, the graph A_k counts the number of paths of length up to k between pairs of nodes:

$$A_k = \sum_{\ell=1}^{k} A^\ell.$$

- That is, oversparsify, compute A_k for $k = 2, 3, 4$, and *then* do nearest neighbor construction.

(BTW, this is essentially what local spectral methods do.)

Error rates on densified sparse graphs

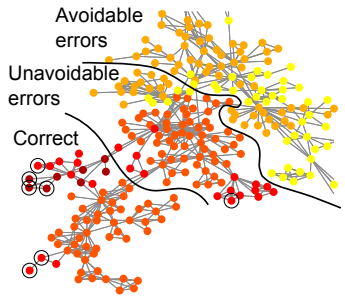
Neighs.	Avg. Deg	Neighs.	k	Avg. Deg
13	19.0	3	2	18.1
28	40.5	5	2	39.2
37	53.3	3	3	52.3
73	104.4	10	2	103.8
97	138.2	3	4	127.1

Table: Paired sets of parameters that give us the same non-zeros in a nearest neighbor graph and a densified nearest neighborgraph A_k.

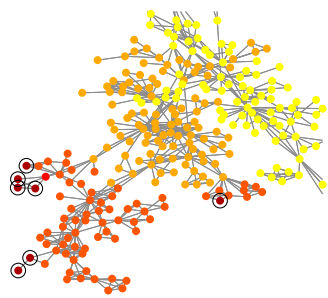
	Zhou		Zhou w. Push	
Avg. Deg	$k = 1$	$k \geq 1$	$k = 1$	$k \geq 1$
19	0.163	0.114	0.156	0.117
41	0.156	0.132	0.158	0.113
53	0.183	0.142	0.179	0.136
104	0.193	0.145	0.178	0.144
138	0.216	0.102	0.204	0.101

Table: Median error rates show the benefit to densifying a sparse graph with the A_k construction. Using average degree of 138 outperforms all of the nearest neighbor trials from previous figure.

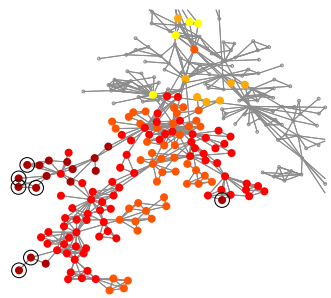
Pictorial illustration



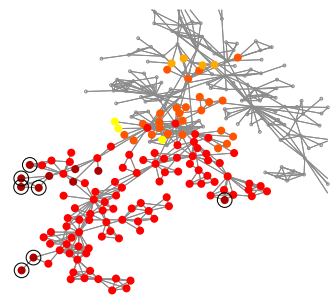
(g) Zhou on sparse graph



(h) Zhou on dense graph



(i) Zhou+Push on sparse graph



(j) Zhou+Push on dense graph

We artificially densify this graph to A_k (for $k = 2, 3, 4, 5$) to compare sparse and dense diffusions and implicit ACL regularization. (Unavoidable errors are caused by a mislabeled node.) Things to note:

- On dense graphs, regularizing diffusions has smaller effect (b vs. d).
- In sparse graphs, regularizing diffusions has bigger effect (a vs. c).
- Regularized diffusions less sensitive to density changes than unregularized diffusions (c vs. d).

Conclusion

- Many real data graphs are very not nice:
  - data analysis design decisions are often reasonable but somewhat arbitrary
  - "noise" from label errors, node/edge errors, arbitrary decisions hurt diffusion-bsed algorithms in different ways
- Many preprocessing design decisions make data more nice:
  - "good" when algorithm users do it—it helps algorithms return something meaningful
  - "bad" when algorithm developers do it—algorithms don't get stress-tested on not nice data
- Local and locally-biased spectral algorithms:
  - have very nice algorithmic and statistical properties
  - can also be used to robustify the graph construction step to arbitrariness of data preprocessing decisions