SAND2013-8728C

Exceptional service in the national interest

Sandia National Laboratories

A computational spectral graph theory tutorial

November 2013

Sandia National Labs: Erik Boman, Karen Devine, *Rich Lehoucq*

Lawrence Livermore National Labs: Van Henson, Geoff Sanders

U.S. DEPARTMENT OF ENERGY | NNSA National Nuclear Security Administration

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Goal

- Gentle introduction to some problems in spectral graph theory
- Developing a computational spectral graph theory capability
  - Also within the greater context of large-scale linear algebra for graphs
- Challenge: Can we leverage existing algorithms and software for the large-scale sparse eigenvalue problem to spectral graph theory?
  - We can go substantially beyond power and inverse iteration
- Challenge: If so, how can such a computational capability be exploited to learn about data?

Outline

- Spectral graph theory:
  - A myriad of graph Laplacians
  - Some problems of interest
  - Some algorithms
- Anasazi package of eigensolvers
  - Anasazi interoperability model (integration into a user's program)
- Some numerical experiments

Notation

- Simple, connected, undirected, graph containing N vertices
  - These assumptions on the graph are not necessary, merely convenient
  - Directed graphs possible
- Adjacency matrix; an edge between vertices i,j if and only if corresponding element of A is one; zero otherwise
- Diagonal degree matrix; row i contains the degree of vertex i

$$\mathcal{G} = (\mathcal{V}, \mathcal{E})$$

$$A = A^T \in \mathbb{R}^{N \times N}$$

$$D \in \mathbb{R}^{N \times N}$$

Graph Laplacians

Sandia National Laboratories

- Combinatorial

$$L = D - A$$

- Normalized

$$\hat{L} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} A D^{-1/2}$$

- Signless

$$K = D + A$$

- Signless normalized

$$\hat{K} = D^{-1/2} K D^{-1/2} = I + D^{-1/2} A D^{-1/2}$$

The spectral problem

- Combinatorial

$$Lx_i = x_i\lambda_i, \quad \lambda_0 = 0 \le \lambda_1 \le \cdots \le \lambda_{N-1}$$

- Normalized

$$\hat{L}\hat{x}_i = \hat{x}_i\hat{\lambda}_i$$

- Signless

$$Ky_i = y_i\nu_i, \quad 0 \le \nu_0 \le \nu_1 \le \cdots \le \nu_{N-1}$$

- Signless normalized

$$\hat{K}\hat{y}_i = \hat{y}_i\hat{\nu}_i$$

Combinatorial & normalized Laplacians

- Combinatorial and normalized graph Laplacians are both generators for a continuous-time Markov chain
- The combinatorial and normalized graph Laplacians are "congruent"
- ½ signless Laplacian is the generator for a "lazy chain" (add a self-loop to each vertex)

$$D^{-1/2}\hat{L}D^{1/2} = I - D^{-1}A$$
$$= D^{-1}L$$

$$D^{1/2}\hat{L}D^{1/2} = L$$

Combinatorial & normalized Laplacians

$$L = D^{1/2}\hat{L}D^{1/2}$$

- Spectral structure the same (up to a simple scaling of the eigenvalues) if and only if the graph is regular (every vertex has the same degree)
- Skewed distribution degree graphs are far from regular and so the combinatorial and normalized graph Laplacians are distinct spectrally
  - The degree of the graph vertices obeys a power law (an example to come)
- Regular, or nearly so, graphs are ubiquitous when solving PDEs numerically

Skewed degree distribution graphs

- Skewed distribution degree graphs are far from regular and so the combinatorial and normalized graph Laplacians are distinct spectrally
  - The degree of the graph vertices obeys a power law
- Not much research into the effect of the degree distribution upon spectral approximation and impact upon numerical algorithms
  - *Maximum principles and decay rates for extremal eigenpairs of scale-free adjacency and modularity matrices,* submitted, G. Sanders, V. E. Henson, T. Jones, J. L. Trask

Differential and integral operators

- Discretization of self-adjoint elliptic PDEs lead to (weighted or non-simple) nearly regular graphs
  - Boundary conditions typically prevent the graph from being regular
- What is the continuum operator associated with a dense limit of graphs (setting aside what this means)?
- Limits of Markov chains, and dense graph limits lead to an integral operator instead of a differential operator

Algebraic eigenvalue problem

- All four spectral problems are examples of symmetric eigenvalue problems–this is about the only good news!
- All four graph Laplacians are symmetric positive semi-definite
  - Combinatorial, normalized: constant vector is associated with a zero eigenvalue; the connected component of the graph
  - Signless variants: the vector of plus, minus ones is associated with the bipartition of a graph and then has a zero eigenvalue

$$Qc = \underbrace{\begin{bmatrix} I_M & 0 \\ 0 & -I_{N-M} \end{bmatrix}}_{Q} \underbrace{\begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix}}_{c} = \begin{bmatrix} \mathbf{1} \\ -\mathbf{1} \end{bmatrix}$$

$$Lc = 0$$

$$KQc = 0$$

$$\Leftrightarrow$$

$$\mathcal{G} \text{ is bipartite}$$

Outline

- Spectral graph theory:
  - A myriad of graph Laplacians
  - Some problems of interest
  - Some algorithms
- Anasazi package of eigensolvers
  - Anasazi interoperability model (integration into a user's program)
- Some numerical experiments

What are the eigenvectors used for?

Sandia National Laboratories

- Graph partitioning: useful for community detection
  - Determine the algebraic connectivity
  - Determine the algebraic bipartivity
- Deterministic approach for Markov chains
  - Asymptotic (stationary) distribution
  - Approximate various expected values
- Model reduction
  - A spectral, or eigen, basis for expressing quantities of interest on the graph (e.g., expected values), Fouss, Pirotte, Renders, Saerens IEEETKDE 2007.

Algebraic connectivity

Sandia National Laboratories

Combinatorial and normalized graph Laplacians:

- The smallest positive eigenvalue represents the "algebraic connectivity" of a graph
- The corresponding eigenvector gives the partition of the graph into two subgraphs with a small number of edges
- This spectral partition solves the relaxed version of a combinatorial optimization problem

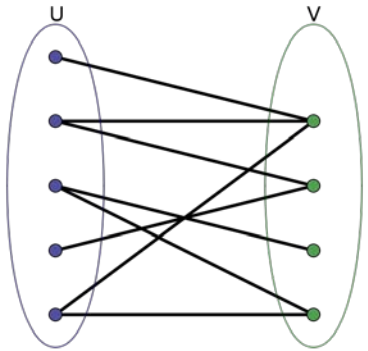
$$\frac{x_1 \bullet L x_1}{x_1 \bullet x_1} = \lambda_1,$$

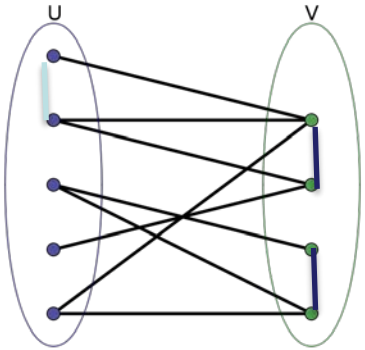
$$\frac{\hat{x}_1 \bullet \hat{L} \hat{x}_1}{\hat{x}_1 \bullet \hat{x}_1} = \hat{\lambda}_1,$$

$$\min_{q \in \{0,1\}^{N \times N}} \frac{q \bullet (D - A) q}{q \bullet D q} \geq \hat{\lambda}_1, \quad \min_{q \in \{0,1\}^{N \times N}} \frac{q \bullet (D - A) q}{q \bullet q} \geq \lambda_1$$

Bipartite and near bipartite graphs

- Bipartite graph: partition the graph into two subgraphs with no internal edges
- Near bipartite graph: small number of internal edges
- More likely is that there are numerous near bipartite communities





Algebraic bipartivity

Signless graph Laplacians

- Zero when the graph is bipartite; vector *p* consisting of 1 or -1 gives the bipartition
- *Near bipartite* then defined to be the smallest positive value
- Relax the optimization problem and use the eigenvector; smallest eigenvalue is the “algebraic bipartivity”

$$\frac{y_0 \bullet K y_0}{y_0 \bullet y_0} = \nu_0,$$

$$\frac{\hat{y}_0 \bullet \hat{K} \hat{y}_0}{\hat{y}_0 \bullet \hat{y}_0} = \hat{\nu}_0,$$

$$\min_{p \in \{-1,1\}^{N \times N}} \frac{p \bullet (D + A) p}{p \bullet D p} \geq \hat{\nu}_0, \quad \min_{p \in \{-1,1\}^{N \times N}} \frac{p \bullet (D + A) p}{p \bullet p} \geq \nu_0$$

Expected values on graphs

- Hitting time: probability of reaching a vertex from a vertex
- Mean hitting time: expected value of the time to reach a vertex from a vertex
- Commute time: the mean hitting time to and from a vertex

$$T_j = \inf\left\{n \geq 0 : X_n \in \mathcal{V}_j\right\}$$
$$P_i(T_j < \infty)$$

$$E_i T_j = \sum_{n \leq \infty} n P_i(T_j = n)$$

$$E_i T_j + E_j T_i$$

Model reduction

Sandia National Laboratories

- Expected values on graphs
  - Hitting time: probability of reaching a vertex from a vertex
  - Mean hitting time: expected value of the time to reach a vertex from a vertex
  - Commute time: the mean hitting time to and from a vertex
- The above problems can be posed as linear systems involving the normalized graph Laplacian
- If many expected values are desired (i.e., numerous vertices) then using a sub-basis of eigenvectors to approximate may be expedient
  - Sub-basis: “small” number of eigenvectors

Outline

- Spectral graph theory:
  - A myriad of graph Laplacians
  - Some problems of interest
  - Some algorithms
- Anasazi package of eigensolvers
  - Anasazi interoperability model (integration into a user's program)
- Some numerical experiments

Eigenvalue algorithms

- Brief overview
  - Power & Lanczos iterations, restarting, preconditioned iteration
- Constraints on computation
  - Only matrix vector products with the Laplacian L is available
  - Fixed storage requirements
  - Cannot solve linear systems $Lu=b$
  - Preconditioner M for the Laplacian available where $M \approx L$ and linear systems $Mv=b$
- Focus on compute the smallest eigenvalues and eigenvectors of the Laplacian of interest

Power iteration

$$u^{(i+1)} = (L + \frac{1}{N} cc^T - \sigma I)u^{(i)}, \quad \sigma > \frac{\lambda_{N-1}}{2}$$

$$= Lu^{(i)} + \frac{c \bullet u^{(i)}}{N} c - \sigma u^{(i)}$$

$$u^{(i)} \to x_1, \quad \theta^{(i)} = \underbrace{\frac{u^{(i)} \bullet Lu^{(i)}}{u^{(i)} \bullet u^{(i)}}}_{\text{Rayleigh quotient}} \to \lambda_1$$

- "The power method is no longer a serious technique for computing eigenvectors", B. N. Parlett, The symmetric eigenvalue problem
- But the start for serious techniques

Lanczos iteration

$$\text{Span}\left\{u^{(0)}, Lu^{(0)}, \cdots, L^{m-1}u^{(0)}\right\}$$

- Use the sequence of iterates for the above Krylov space and then estimate the smallest eigenvector
- The Lanczos iteration is a three-term recurrence for computing an orthogonal representation of the Krylov space
  - Maintaining a numerically orthogonal set of vectors is non-trivial (but can be done)
  - An alternative is to restart the iteration; this also enables storage requirements to be fixed in advance
- A preconditioner M cannot be used (except within an inner iteration)

Restart the Lanczos iteration

$$\begin{aligned}
&\text{Span}\left\{u^{(0)}, Lu^{(0)}, \cdots, L^{m-1}u^{(0)}\right\} \rightarrow \text{Span}\left\{u^{(0)}, Lu^{(0)}, \cdots, L^{\ell-1}u^{(0)}\right\},\ m > \ell \\
&\quad \rightarrow \text{Span}\left\{u^{(0)}, Lu^{(0)}, \cdots, L^{m-1}u^{(0)}\right\} \rightarrow \text{Span}\left\{u^{(0)}, Lu^{(0)}, \cdots, L^{\ell-1}u^{(0)}\right\} \\
&\qquad \ddots \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \ddots
\end{aligned}$$

- Restarting is an accordion like process
  - Leads to the class of implicit restart methods
- Enables storage requirements to be fixed in advance
  - User prescribes m
- Use of a preconditioner M not possible (except within an inner iteration)

Preconditioned iteration

$$u^{(i+1)} = (L + \frac{1}{N} cc^T - \sigma I)u^{(i)}, \quad \sigma > \frac{\lambda_{N-1}}{2}$$

$$w^{(i+1)} = w^{(i)} - M^{-1}(L + \frac{1}{N} cc^T - \theta^{(i)} I)w^{(i)}$$

$$w^{(i)} \to x_1, \quad \theta^{(i)} = \underbrace{\frac{w^{(i)} \bullet Lw^{(i)}}{w^{(i)} \bullet w^{(i)}}}_{\text{Rayleigh quotient}} \to \lambda_1$$

- Preconditioned iteration is a minor variation on power iteration
- Preconditioner $N=I$ and selecting σ appropriately leads to equivalence with the power iteration
- Preconditioner $N=L$ leads to inverse iteration (or the power iteration on the inverse of L).

Eigenvalue algorithms

Sandia National Laboratories

- The basic ideas presented lead to sophisticated algorithms exploiting subspace projection
  - Block-Krylov Schur and implicit restarted Arnoldi methods
  - Gradient based algorithms such as Davidson, IRTR, LOBPCG, tracemin
  - Newton-based approaches (Jacobi-Davidson)
- Important extensions to all the algorithms are:
  - Incorporating a preconditioner
  - Incorporating blocking so that the Laplacian is applied to a collection of vectors so to improve the floating point performance; the details are platform specific
  - Efficient but a stable scheme for maintaining numerical orthogonality of the vectors used to represent the subspace

Outline

- Spectral graph theory:
  - A myriad of graph Laplacians
  - Some problems of interest
  - Some algorithms
- Anasazi package of eigensolvers
  - Anasazi interoperability model (integration into a user's program)
- Some numerical experiments

Sandia National Laboratories

- Collection of algorithms for the large-scale solution of the algebraic eigenvalue problems $AX = X\Lambda$ or $AX = BX\Lambda$ where A is a large sparse matrix
  - Think of computing 1-100 eigenvectors for a matrix of order 100,000+; the limit depends upon computational resources; 1.75 trillion achieved
  - Developers: Baker, Hetmaniuk, *Lehoucq*, Thornquist

- Block-based eigensolvers:
  - Improve reliability for clustered eigenvalues
  - Achieve better data locality for linear algebra operations

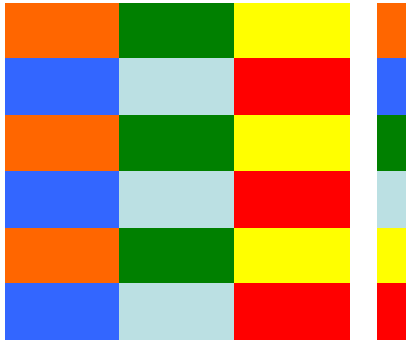
- Four algorithms that go beyond the power iteration
  - *LOBPCG Locally Optimal Block Preconditioned Conjugate Gradient (Knyasev, 2002; Hetmaniuk & Lehoucq, 2006)*
  - Block Krylov-Schur (block extension of Stewart, 2000)
  - Block Davidson (Arbenz, Hetmaniuk, Lehoucq, Tuminaro, 2005)
  - IRTR Implicit Riemannian Trust Region (Absil, Baker, Gallivan, 2006)

Efficient matrix-vector products are important for skewed degree graphs

Sandia National Laboratories

- A. Yoo, A. H. Baker, R. Pearce, and V. E. Henson *A scalable eigensolver for large scale-free graphs using 2D graph partitioning* SC '11
- Erik G. Boman, Karen D. Devine, and Sivasankaran Rajamanickam *Scalable Matrix Computations on Large Scale-Free Graphs Using 2D Graph Partitioning*

1D row-wise matrix distribution; 6 processes



2D matrix distribution; 6 processes

Anasazi interoperability model

- Templated C++; user provides implementations of sparse matrix vector products and other large-scale linear algebra operations
  - Leverage user's data structures and software investment
  - Memory system neutral (distributed/shared memory)
  - Assume Fortran BLAS/LAPACK libraries available for dense matrix calculations

Method Name	Description
`Apply(A,X,Y)`	Applies the operator A to the multivector X, placing the result in the multivector Y

Method Name	Description
`Clone(X,numvecs)`	Creates a new multivector from X with *numvecs* vectors
`CloneCopy(X,index)`	Creates a new multivector with a copy of the contents of a subset of the multivector X (deep copy)
`CloneView(X,index)`	Creates a new multivector that shares the selected contents of a subset of the multivector X (shallow copy)
`GetVecLength(X)`	Returns the vector length of the multivector X
`GetNumberVecs(X)`	Returns the number of vectors in the multivector X
`MvTimesMatAddMv(alpha,X, D,beta,Y)`	Applies a dense matrix D to multivector X and accumulates the result into multivector Y: $Y \leftarrow \alpha X D + \beta Y$
`MvAddMv(alpha,X,beta,Y)`	Performs multivector AXPBY: $Y \leftarrow \alpha X + \beta Y$
`MvTransMv(alpha,X,Y,D)`	Computes the dense matrix $D \leftarrow \alpha X^H Y$
`MvDot(X,Y,d)`	Computes the corresponding dot products: $d[i] \leftarrow \bar{x}_i y_i$
`MvScale(X,d)`	Scales the ith column of a multivector X by $d[i]$
`MvNorm(X,d)`	Computes the 2-norm of each vector of X: $d[i] \leftarrow \|x_i\|_2$
`SetBlock(X,Y,index)`	Copies the vectors in X to a subset of vectors in Y
`MvInit(X,alpha)`	Replaces each entry in the multivector X with a scalar α
`MvRandom(X)`	Replaces the entries in the multivector X by random scalars
`MvPrint(X)`	Print the multivector X

Multivector is a collection, or block, of vectors

Outline

- Spectral graph theory:
  - A myriad of graph Laplacians
  - Some problems of interest
  - Some algorithms
- Anasazi package of eigensolvers
  - Anasazi interoperability model (integration into a user's program)
- Some numerical experiments

Our Anasazi Testing Platform

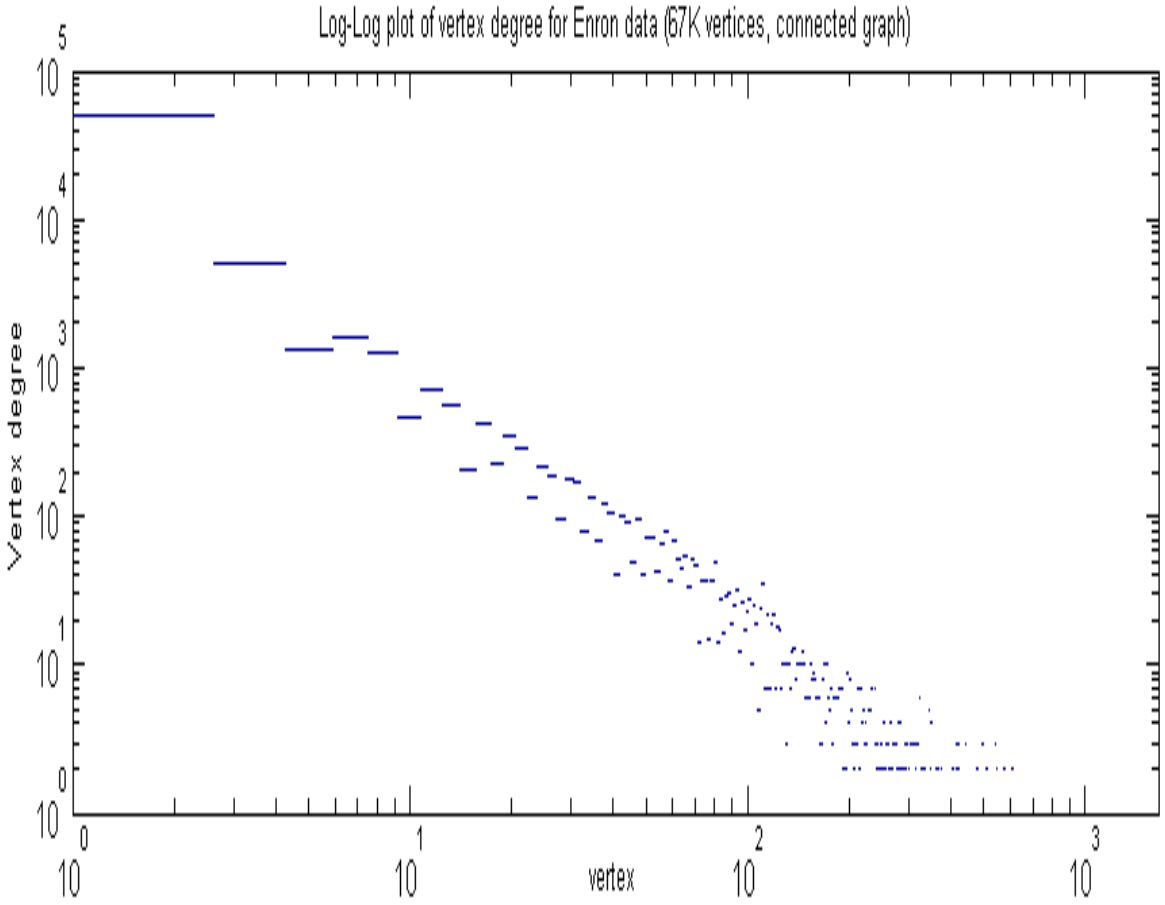
Sandia National Laboratories

- C++ driver program for testing/evaluating various eigensolvers, parameters, inputs, parallel distributions, scalability
- Enables use of all Anasazi eigensolvers: BKS, BD, LOBPCG, IRTR
- Uses Trilinos' IFPACK preconditioners: Jacobi, SGS, IC, ILU, KLU, Support graph (MST)
- Finds smallest or largest eigenvalues and corresponding eigenvectors.
- Constructs matrices from Matrix-Market input:
  - Combinatorial Laplacian
  - Normalized Laplacian
  - Signless, normalized Laplacian
  - Adjacency Matrix
- Creates 1D and 2D matrix distributions.
  - Built on Trilinos' Epetra matrix/vector classes.
  - Current effort to adopt Epetra64 to solve problems with > 2 billion edges or vertices
- Runs in parallel (distributed memory with MPI) or serial.
  - Similar program available for shared memory (e.g., XMT, UV) using MEGRAPHS.
- Options set through command-line arguments:

```
anasazi.exe --file=big.mtx --use2D --matrix=Laplacian
--normalize --nev=25 --tol=0.001 --method=LOBPCG
```

MATLAB driver also available

Test graph: Enron data set



- 67K vertices
- 1.6K is the max degree
- 507K edges

Termination depends upon residuals

- Combinatorial

$$\left\| Lx_i - x_i \lambda_i \right\|_2$$

- Normalized

$$\left\| \hat{L}\hat{x}_i - \hat{x}_i \hat{\lambda}_i \right\|_2$$

- Signless

$$\left\| Ky_i - y_i \nu_i \right\|_2$$

- Signless normalized

$$\left\| \hat{K}\hat{y}_i - \hat{y}_i \hat{\nu}_i \right\|_2$$

Combinatorial Laplacian

Solver=LOBPCG, blocksize = 5, residual tolerance 1e-5, approximate 5 smallest eigenpairs

Preconditioner	Iterations	Matvecs	Setup time	Iteration time	Total time
None	6896	34505		564.8	564.8
Jacobi	466	2335	0.0	46.9	46.9
IC(0)	476	2385	0.2	43.7	43.9
KLU	12	65	23.2	2.6	25.8
SGS	155	780	0.0	21.4	21.4
MST(1)	125	630	1.9	13.2	15.1
MST(2)	53	270	3.3	6.2	9.5

Results: Normalized Laplacian

Solver=LOBPCG, blocksize = 5, residual tolerance 1e-5, approximate 5 smallest eigenpairs

Preconditioner	Iterations	Matvecs	Setup time	Iterate time	Total time
KLU	11	70	22.9	2.5	25.4
IC(0)	270	1355	0.2	22.8	23.0
None	250	1255	0	20.0	20.0
MST(2)	42	215	7.9	6.2	14.1
MST(1)	102	515	2.0	10.7	12.7
SGS	87	440	0.0	11.8	11.8

Comparison of Eigensolvers

blocksize = 5 except for BKS that used a blocksize=1, residual tolerance 1e-5 , compute the smallest five eigenpairs

Method	Laplacian	Precond itioner	Iterations	Matvecs	Total time
BD	Combintor	Jacobi	>50000	--	--
BKS	Combintor		>50000	--	--
LOBPCG	Combintor	Jacobi	466	2335	46.7
IRTR	Combintor	Jacobi	14	492	27.1
BD	Normalized		4397	43980	200.6
BKS	Normalized		865	865	22.6
LOBPCG	Normalized		250	1255	20.4
IRTR	Normalized		12	380	13.6

Comparison of Eigensolvers

blocksize = 5 except for BKS that used a blocksize=1, residual tolerance 1e-5, approximate the smallest five eigenpairs

Method	Laplacian	Preconditioner	Iterations	Matvecs	Total time
BKS	Signless		>10000	--	--
BD	Signless	Jacobi	>10000	--	--
LOBPCG	Signless	Jacobi	457	2330	42.9
IRTR	Signless	Jacobi	16	777	40.9
BD	N Signless		5001	50035	246.1
BKS	N Signless		1935	1935	29.5
IRTR	N Signless		14	785	27.9
LOBPCG	N Signless		282	1455	23.5

Enron graph: Eigenvalues

Sandia National Laboratories

The 5 smallest eigenvalues of the Laplacians.

Combinatorial	Normalized	Signless	N Signless
0	0	5.2e-3	1.4e-3
5.2e-3	1.4e-3	5.3e-3	2.6e-3
5.3e-3	2.6e-3	5.8e-3	2.9e-3
5.8e-3	2.9e-3	8.8e-3	4.4e-3
8.1e-3	4.0e-3	1.0e-2	5.2e-3

$$\left(\min_k D_k\right)\lambda_i \le \hat{\lambda}_i \le \left(\max_k D_k\right)\lambda_i$$

$$\lambda_i \le \hat{\lambda}_i \le 1634\lambda_i$$

Mind the relative gap!

- The combinatorial Laplacian eigenvalues are more disparate in size
- This affects the convergence rate of the solvers
  - Relative gap, roughly, determines the convergence rate
- Normalized Laplacians are better behaved; the convergence rate is never slower

$$\left(\min_k D_k\right)\lambda_i \le \hat{\lambda}_i \le \left(\max_k D_k\right)\lambda_i$$

$$\lambda_i \le \hat{\lambda}_i \le 1634\lambda_i$$

$$\underbrace{\frac{\lambda_{j+1}-\lambda_j}{\lambda_N-\lambda_j}}_{\lambda_j \text{ gap}} \lessapprox \underbrace{\frac{\hat{\lambda}_{j+1}-\hat{\lambda}_j}{\hat{\lambda}_N-\hat{\lambda}_j}}_{\hat{\lambda}_j \text{ gap}}$$