Applications of Matrix Functions to Network Analysis Part II: Algorithms

Michele Benzi

Department of Mathematics and Computer Science

Emory University

Atlanta, Georgia, USA

SSMF2013, Lille, France

Outline

Outline

Computational challenges

We saw that many problems in graph analysis lead to the computation of selected entries of $f(A)$, where A is the adjacency matrix and $f(x)$ is an analytic function.

Typically, the diagonal entries of $f(A)$ are wanted, and for large graphs some global averages over subsets of entries of $f(A)$, often expressed in terms of traces or row/column sums.

High accuracy is not always required. For example, in the case of centralities it is important to be able to identify just the top-ranked nodes. In many casess, bounds are often sufficient.

In some instances, only row or column sums of $f(A)$ are required. These can be computed without having to compute any of the entries of $f(A)$.

Computational challenges (cont.)

Many of these computations can be formulated as evaluation of bilinear forms $\mathbf{u}^T f(A)\,\mathbf{v}$ for suitably chosen vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^N$ and for suitable functions $f(x)$:

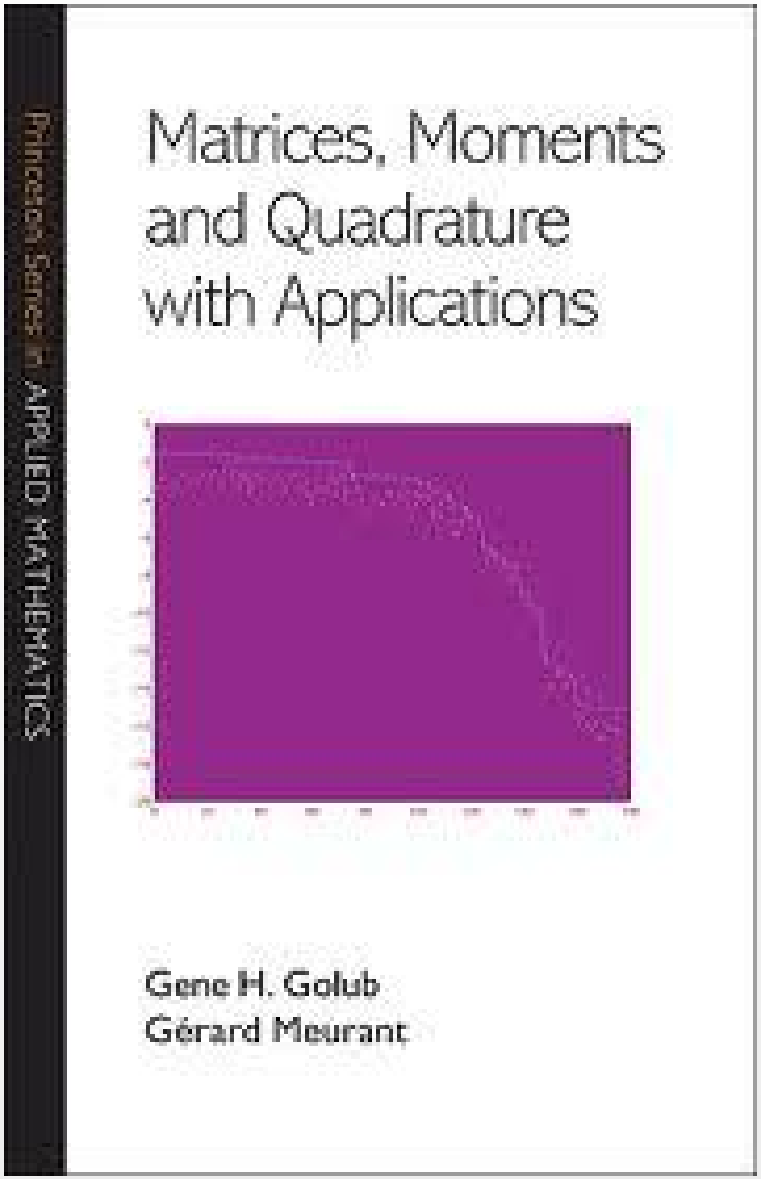
- Subgraph centrality: $SC(i) = \mathbf{e}_i^T f(A)\,\mathbf{e}_i$
- Communicability: $C(i,j) = \mathbf{e}_i^T f(A)\,\mathbf{e}_j$
- Average communicability:

$$\overline{C(i)} = \frac{1}{N-1}\left[\mathbf{1}^T f(A)\,\mathbf{e}_i - \mathbf{e}_i^T f(A)\,\mathbf{e}_i\right]$$

- Total network communicability: $C(G) = \frac{1}{N}\mathbf{1}^T f(A)\,\mathbf{1}$

Question: How do we compute these quantities?

Answer: Using this book!



Gauss quadrature

Consider the spectral decompositions

$$A = Q\Lambda Q^T, \qquad f(A) = Qf(\Lambda)Q^T.$$

For $\mathbf{u}, \mathbf{v} \in \mathbb{R}^N$ we have

$$\mathbf{u}^T f(A)\mathbf{v} = \mathbf{u}^T Q f(\Lambda) Q^T \mathbf{v} = \mathbf{p}^T f(\Lambda)\mathbf{q} = \sum_{i=1}^{N} f(\lambda_i) p_i q_i,$$

where $\mathbf{p} = Q^T\mathbf{u}$ and $\mathbf{q} = Q^T\mathbf{v}$. Rewrite this as a Riemann-Stieltjes integral:

$$\mathbf{u}^T f(A)\mathbf{v} = \int_a^b f(\lambda) d\mu(\lambda), \quad \mu(\lambda) = \begin{cases} 0 & \lambda < a = \lambda_1 \\ \sum_{j=1}^{i} p_j q_j & \lambda_i \le \lambda < \lambda_{i+1} \\ \sum_{j=1}^{N} p_j q_j & b = \lambda_N \le \lambda. \end{cases}$$

Gauss quadrature (cont.)

The general Gauss-type quadrature rule is

$$\int_a^b f(\lambda) d\mu(\lambda) = \sum_{j=1}^{n} w_j f(t_j) + \sum_{k=1}^{M} v_k f(z_k) + R[f],$$

where the nodes $\{z_k\}$ are prescribed.

- Gauss: $M = 0$,
- Gauss–Radau: $M = 1$, $z_1 = a$ or $z_2 = b$,
- Gauss–Lobatto: $M = 2$, $z_1 = a$ and $z_2 = b$.

The evaluation of these quadrature rules is reduced to

- computation of orthogonal polynomials via three-term recurrence,
- or, equivalently, computation of entries and spectral information of the corresponding tridiagonal matrix (Lanczos).

Gauss quadrature (cont.)

For instance, we have for the Gauss rule:

$$\int_a^b f(\lambda) d\mu(\lambda) = \sum_{j=1}^{n} w_j f(t_j) + R[f]$$

with

$$R[f] = \frac{f^{(2n+M)}(\eta)}{(2n+M)!} \int_a^b \Big[\prod_{j=1}^{n} (\lambda - t_j) \Big]^2 d\mu(\lambda),$$

for some $a < \eta < b$.

Gauss quadrature (cont.)

For instance, we have for the Gauss rule:

$$\int_a^b f(\lambda)d\mu(\lambda) = \sum_{j=1}^{n} w_j f(t_j) + R[f]$$

with

$$R[f] = \frac{f^{(2n+M)}(\eta)}{(2n+M)!} \int_a^b \Big[\prod_{j=1}^{n}(\lambda - t_j)\Big]^2 d\mu(\lambda),$$

for some $a < \eta < b$.

Theorem (Golub-Meurant)

$$\sum_{j=1}^{n} w_j f(t_j) = \mathbf{e}_1^T f(J_n)\mathbf{e}_1 = [f(J_n)]_{11}$$

Gauss quadrature (cont.)

The tridiagonal matrix J_n corresponds to the three-term recurrence relationship satisfied by the set of polynomials orthonormal with respect to $d\mu$.

$$J_n = \begin{pmatrix} \omega_1 & \gamma_1 & & & \\ \gamma_1 & \omega_2 & \gamma_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \gamma_{n-2} & \omega_{n-1} & \gamma_{n-1} \\ & & & \gamma_{n-1} & \omega_n \end{pmatrix}$$

The eigenvalues of J_n are the Gauss nodes, whereas the Gauss weights are given by the squares of the first entries of the normalized eigenvectors of J_n.

The quadrature rule is computed with the Golub–Welsch QR algorithm. Alternatively, the (1,1) entry of $f(J_n)$ can be computed via diagonalization of J_n.

Gauss quadrature (cont.)

Consider the case $\mathbf{u} = \mathbf{v} = \mathbf{e}_i$ (corresp. to the (i, i) entry of $f(A)$).

The entries of J_n are computed using the symmetric Lanczos algorithm:

$$\gamma_j \mathbf{x}_j = \mathbf{r}_j = (A - \omega_j I)\mathbf{x}_{j-1} - \gamma_{j-1}\mathbf{x}_{j-2}, \qquad j = 1, 2, \ldots$$
$$\omega_j = \mathbf{x}_{j-1}^T A \mathbf{x}_{j-1},$$
$$\gamma_j = \|\mathbf{r}_j\|$$

with initial vectors $\mathbf{x}_{-1} = \mathbf{0}$ and $\mathbf{x}_0 = \mathbf{e}_i$.

Each additional Lanczos step amounts to adding another node to the Gauss-type quadrature rule, resulting in tighter and tighter bounds (implementation based on MMQ Matlab toolbox by G. Meurant). Note that the vectors $\mathbf{x}_i$ are initially very sparse.

Off-diagonal entries $[f(A)]_{ij} = \mathbf{e}_i^T f(A)\mathbf{e}_j$ can be estimated using the nonsymmetric Lanczos algorithm.

Gauss quadrature (cont.)

For $f(A) = \mathrm{e}^A$ and $f(A) = (I - cA)^{-1}$ we obtain:

- bounds on $[f(A)]_{ii}$ from symmetric Lanczos,
- bounds on $[f(A)]_{ii} + [f(A)]_{ij}$ from unsymmetric Lanczos,
- lower bounds from the Gauss and the Gauss-Radau rules,
- upper bounds from the Gauss-Radau and Gauss-Lobatto rules.

In computations we often use the simple (Gerschgorin) estimates

$$\lambda_{\mathsf{min}} \approx - \max_{1 \le i \le N} \{\mathsf{deg}(i)\}, \qquad \lambda_{\mathsf{max}} \approx \max_{1 \le i \le N} \{\mathsf{deg}(i)\}.$$

where $\mathsf{deg}(i)$ is the degree of node v_i.

Using more accurate estimates of the extreme eigenvalues of A generally leads to improved results, especially for scale-free graphs.

Computing row and column sums

Recall that the computations of row/column sums of matrix functions amounts to computing the action of $f(A)$ or $f(A^T)$ on the vector $\mathbf{1}$ of all ones.

For instance, computing the total communicability of a graph with adjacency matrix A,

$$C(G) = \frac{1}{N}\mathbf{1}^T \mathrm{e}^A \mathbf{1},$$

amounts to taking the arithmetic mean of the entries of the vector $\mathrm{e}^A\mathbf{1}$.

This can be accomplished very fast using Krylov subspace methods for evaluating the action of the matrix exponential on a vector.

A very efficient Matlab tolbox has been developed by Stefan Güttel:

`http://www.mathe.tu-freiberg.de/~guettels/funm_kryl`

Outline

Summary of results

Numerical experiments were performed on a number of networks, both synthetic (generated with the CONTEST Toolbox for Matlab, Taylor & Higham 2010) and from real-world applications, including PPI, social, collaboration, and transportation networks.

- A few (3-5) Lanczos steps per estimate are usually enough
- Independent of N for graphs with bounded max degree
- Almost as good for power-law degree distributions
- Very efficient approach for computing global averages

M. Benzi and P. Boito, *Quadrature rule-based bounds for functions of adjacency matrices*, Linear Algebra Appl., 433 (2010), pp. 637-652.

Summary of results (cont.)

Example: Range-dependent network from CONTEST, $N = 100$.

- MMQ approximations of the Estrada index $EE(G) = 425.0661$

# it	1	2	3	4	5
Gauss	348.9706	416.3091	424.4671	425.0413	425.0655
Radau (lower)	378.9460	420.6532	424.8102	425.0570	425.0659
Radau (upper)	652.8555	437.7018	425.6054	425.0828	425.0664
Lobatto	2117.9233	531.1509	430.3970	425.2707	425.0718

- MMQ approximations of $[\mathrm{e}^A]_{1,5} = 0.396425$

# it	1	2	3	4	5
Radau (lower)	-2.37728	0.213316	0.388791	0.396141	0.396431
Radau (upper)	4.35461	0.595155	0.404905	0.396626	0.396420

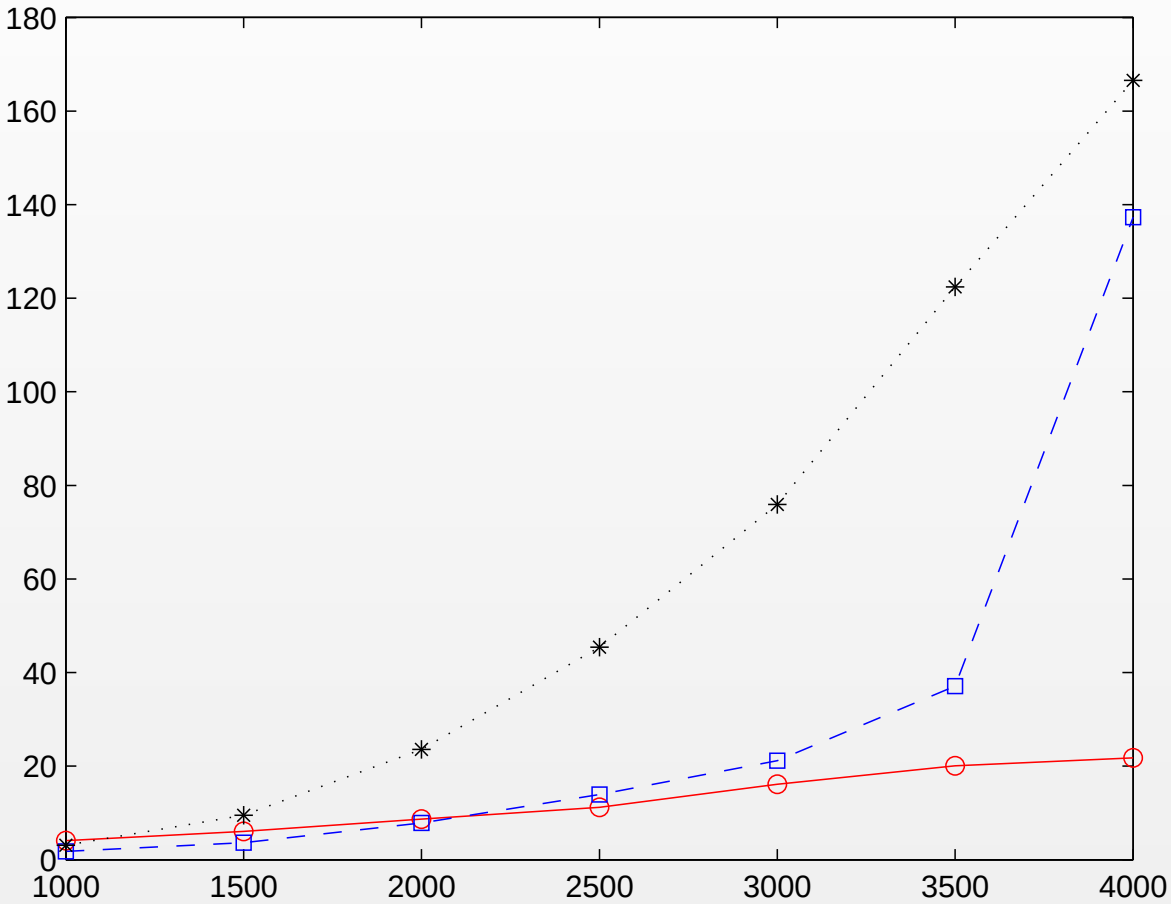
Summary of results (cont.)

Bounds for the Estrada index of small-world matrices of increasing size. Watts–Strogatz model with parameters $(4, 0.1)$; five Lanczos iterations.

N	Gauss	Gauss-Radau (l)	Gauss-Radau (u)	Gauss-Lobatto
1000	1.64e5	1.63e5	1.64e5	1.66e5
2500	4.09e5	4.10e5	4.11e5	4.12e5
5000	8.18e5	8.18e5	8.20e5	8.33e5
7500	1.23e6	1.22e6	1.22e6	1.25e6
10000	1.63e6	1.63e6	1.64e6	1.66e6

Five iterations suffice to reach a small relative error, independent of N.

Summary of results (cont.)



Timings (in seconds) for computing the diagonal of the exponential of small-world matrices of increasing order N. Blue: time for the Matlab `expm` function. Black: time using `eig`. Red: using 5 iterations of the Lanczos algorithm for each node.

Summary of results (cont.)

Experiments were performed on scale-free networks (Barabási– Albert model) of increasing size, to see the effect of increasing maximum degree.

In this case, the Gerschgorin estimate $\lambda_{\mathsf{max}} \approx d_{\mathsf{max}}$ gives poor results. Indeed, whereas the max degree increases with N, the spread of the eigenvalues tend to grow much more slowly.

A better strategy is to estimate λ_{min}, λ_{max} using Lanczos.

The number of Lanczos steps per estimate (either diagonal elements or off-diagonal ones) increases very slowly with N, for a fixed accuracy. Typically, 9-11 Lanczos steps suffice to achieve relative errors around 10^{-6}.

Results for real world networks

Network	N	NNZ	λ_1	λ_2
Zachary Karate Club	34	156	6.726	4.977
Drug Users	616	4024	18.010	14.234
Yeast PPI	2224	13218	19.486	16.134
Pajek/Erdos971	472	2628	16.710	10.199
Pajek/Erdos972	5488	14170	14.448	11.886
Pajek/Erdos982	5822	14750	14.819	12.005
Pajek/Erdos992	6100	15030	15.131	12.092
SNAP/ca-GrQc	5242	28980	45.617	38.122
SNAP/ca-HepTh	9877	51971	31.035	23.004
SNAP/as-735	7716	26467	46.893	27.823
Gleich/Minnesota	2642	6606	3.2324	3.2319

Characteristics of selected real world networks. All networks are undirected.

Results for real world networks (cont.)

Network	`expm`	`mmq`	`funm_kryl`
Zachary Karate Club	0.062	0.138	0.120
Drug Users	0.746	2.416	0.363
Yeast PPI	47.794	9.341	0.402
Pajek/Erdos971	0.542	2.447	0.317
Pajek/Erdos972	579.214	35.674	0.410
Pajek/Erdos982	612.920	39.242	0.393
Pajek/Erdos992	656.270	53.019	0.325
SNAP/ca-GrQc	281.814	23.603	0.465
SNAP/ca-HepTh	2710.802	58.377	0.435
SNAP/as-735	2041.439	75.619	0.498
Gleich/Minnesota	1.956	10.955	0.329

Timings (in seconds) to compute centrality rankings based on the diagonal and row sum of e^A for various test problems using different codes.

A large example: the Wikipedia graph

The Wikipedia graph is a directed graph with $N = 4,189,503$ nodes and $67,197,636$ edges (as of June 6, 2011).

The row/columns of the exponential of the adjacency matrix can be used to rank the hubs and authorities in Wikipedia in order of importance.

Using `funm_kryl` we can compute the hub and authority rankings for all the nodes in about 216s on a parallel system comprising 24 Intel(R) Xeon(R) E5-2630 2.30GHz CPUs.

Global communicability measures

Recall that the (normalized) total communicability of a graph G with N nodes and adjacency matrix A is defined as $C(G) = \frac{1}{N}\mathbf{1}^T \mathrm{e}^A \mathbf{1}$.

This quantity yields a global measure of how "easily" information flows over the network.

Thus, highly connected networks, such as small-world networks without bottlenecks, can be expected to have a high total communicability.

Conversely, large-diameter networks with a high degree of locality (such as regular grids, road networks, etc.) or networks containing bottlenecks are likely to display a low value of $C(G)$.

Global communicability measures (cont.)

Note that $C(G)$ can be easily bounded from below and from above:

$$\frac{1}{N} EE(G) \leq C(G) \leq \mathrm{e}^{\lambda_1}$$

where $EE(G) = \mathrm{Tr}(\mathrm{e}^{A})$ is the Estrada index of the graph. These bounds are sharp, as can be seen from trivial examples.

In the next Table we present the results of some calculations of the normalized Estrada index, glocal communicability and e^{λ_1} for various real-world networks.

M. Benzi and C. Klymko, *Total communicability as a centrality measure*, J. Complex Networks, in press (2013).

Global communicability measures (cont.)

Network	$\frac{1}{N}EE(G)$	$C(G)$	e^{λ_1}
Zachary Karate Club	30.62	608.79	833.81
Drug Users	1.12e05	1.15e07	6.63e07
Yeast PPI	1.37e05	3.97e07	2.90e08
Pajek/Erdos971	3.84e04	4.20e06	1.81e07
Pajek/Erdos972	408.23	1.53e05	1.88e06
Pajek/Erdos982	538.58	2.07e05	2.73e06
Pajek/Erdos992	678.87	2.50e05	3.73e06
SNAP/ca-GrQc	1.24e16	8.80e17	6.47e19
SNAP/ca-HepTh	3.05e09	1.06e11	3.01e13
SNAP/as-735	3.00e16	3.64e19	2.32e20
Gleich/Minnesota	2.86	14.13	35.34

Normalized Estrada index, total communicability, and e^{λ_1} for various networks.

Summary

- When matrix functions are applied to network analysis, one is faced with the problem of evaluating bilinear expressions of the form $\mathbf{u}^T f(A)\mathbf{v}$ or to evaluate products like $f(A)\mathbf{v}$; sometimes traces $\mathrm{Tr}[f(A)]$ are also wanted.
- Gauss–type quadrature rules combined with the Lanczos process provide an elegant method for evaluating bilinear forms; these methods, however, cost approximately $O(N)$ per estimate and therefore are not practical for truly large-scale networks.
- The evaluation of $f(A)\mathbf{v}$ can be done efficiently even for rather large network using current Lanczos- or Arnoldi-based software.
- Improved quadrature rule-based methods are being developed by Lothar Reichel and co-workers, but the basic limitations remain.

D. Martin and L. Reichel, *Block Gauss and anti-Gauss quadrature with applications to networks*, preprint, 2012.

C. Fenu, D. Martin, L. Reichel, and G. Rodriguez, *Network analysis via partial spectral factorization and Gauss quadrature*, preprint, 2013.

Outline

Hubs and authorities in digraphs

Suppose now that $G = (V, E)$ is a directed graph. In such graphs each node can play two different roles: that of a hub, and that of a authority.

The Hyperlink-Induced Topic Search (HITS) algorithm (Kleinberg, 1999) provides a way of ranking hubs and authorities in a digraph. The main application at the time was to WWW information retrieval.

In a nutshell:

Hubs and authorities in digraphs

Suppose now that $G = (V, E)$ is a directed graph. In such graphs each node can play two different roles: that of a hub, and that of a authority.

The Hyperlink-Induced Topic Search (HITS) algorithm (Kleinberg, 1999) provides a way of ranking hubs and authorities in a digraph. The main application at the time was to WWW information retrieval.

In a nutshell:

- A *hub* is a webpage that points to many good authorities;

Hubs and authorities in digraphs

Suppose now that $G = (V, E)$ is a directed graph. In such graphs each node can play two different roles: that of a hub, and that of a authority.

The Hyperlink-Induced Topic Search (HITS) algorithm (Kleinberg, 1999) provides a way of ranking hubs and authorities in a digraph. The main application at the time was to WWW information retrieval.

In a nutshell:

- A *hub* is a webpage that points to many good authorities;
- An *authority* is a webpage that is pointed to by many good hubs;

Hubs and authorities in digraphs

Suppose now that $G = (V, E)$ is a directed graph. In such graphs each node can play two different roles: that of a hub, and that of a authority.

The Hyperlink-Induced Topic Search (HITS) algorithm (Kleinberg, 1999) provides a way of ranking hubs and authorities in a digraph. The main application at the time was to WWW information retrieval.

In a nutshell:

- A *hub* is a webpage that points to many good authorities;
- An *authority* is a webpage that is pointed to by many good hubs;
- Every webpage i is given an authority score x_i and a hub score y_i, based on an iterative scheme.

Calculation of HITS scores

The HITS ranking scheme is an iterative method converging to a stationary solution.

- Initially, each x_i and y_i is set equal to 1.

Calculation of HITS scores

The HITS ranking scheme is an iterative method converging to a stationary solution.

- Initially, each x_i and y_i is set equal to 1.
- Weights are updated in the following way:

$$x_i^{(k)} = \sum_{j:(j,i)\in E} y_j^{(k-1)} \quad \text{and} \quad y_i^{(k)} = \sum_{j:(i,j)\in E} x_j^{(k)}$$

for $k = 1, 2, 3, \ldots$

Calculation of HITS scores

The HITS ranking scheme is an iterative method converging to a stationary solution.

- Initially, each x_i and y_i is set equal to 1.
- Weights are updated in the following way:

$$x_i^{(k)} = \sum_{j:(j,i)\in E} y_j^{(k-1)} \quad \text{and} \quad y_i^{(k)} = \sum_{j:(i,j)\in E} x_j^{(k)}$$

  for $k = 1, 2, 3, \ldots$
- Weights are normalized so that $\sum_j (x_j^{(k)})^2 = 1$ and $\sum_j (y_j^{(k)})^2 = 1$.

Calculation of HITS scores

The HITS ranking scheme is an iterative method converging to a stationary solution.

- Initially, each x_i and y_i is set equal to 1.
- Weights are updated in the following way:

$$x_i^{(k)} = \sum_{j:(j,i)\in E} y_j^{(k-1)} \quad \text{and} \quad y_i^{(k)} = \sum_{j:(i,j)\in E} x_j^{(k)}$$

  for $k = 1, 2, 3, \ldots$
- Weights are normalized so that $\sum_j (x_j^{(k)})^2 = 1$ and $\sum_j (y_j^{(k)})^2 = 1$.
- Under mild conditions, the sequences of vectors $\{\mathbf{x}^{(k)}\}$ and $\{\mathbf{y}^{(k)}\}$ converge as $k \to \infty$.

HITS as the power method

When written in terms of matrices, the HITS process becomes

$$\mathbf{x}^{(k)} = A^T \mathbf{y}^{(k-1)} \quad \text{and} \quad \mathbf{y}^{(k)} = A\mathbf{x}^{(k)}.$$

This can be rewritten as

$$\mathbf{x}^{(k)} = A^T A\mathbf{x}^{(k-1)} \quad \text{and} \quad \mathbf{y}^{(k)} = AA^T \mathbf{y}^{(k-1)}.$$

- HITS is just an iterative power method to compute the dominant eigenvector for $A^T A$ and AA^T.

HITS as the power method

When written in terms of matrices, the HITS process becomes

$$\mathbf{x}^{(k)} = A^T \mathbf{y}^{(k-1)} \quad \text{and} \quad \mathbf{y}^{(k)} = A\mathbf{x}^{(k)}.$$

This can be rewritten as

$$\mathbf{x}^{(k)} = A^T A\mathbf{x}^{(k-1)} \quad \text{and} \quad \mathbf{y}^{(k)} = AA^T \mathbf{y}^{(k-1)}.$$

- HITS is just an iterative power method to compute the dominant eigenvector for $A^T A$ and AA^T.
  - $A^T A$ is the *authority matrix*.

HITS as the power method

When written in terms of matrices, the HITS process becomes

$$\mathbf{x}^{(k)} = A^T \mathbf{y}^{(k-1)} \quad \text{and} \quad \mathbf{y}^{(k)} = A\mathbf{x}^{(k)}.$$

This can be rewritten as

$$\mathbf{x}^{(k)} = A^T A\mathbf{x}^{(k-1)} \quad \text{and} \quad \mathbf{y}^{(k)} = AA^T \mathbf{y}^{(k-1)}.$$

- HITS is just an iterative power method to compute the dominant eigenvector for $A^T A$ and AA^T.
  - $A^T A$ is the *authority matrix*.
  - AA^T is the *hub matrix*.

HITS as the power method

When written in terms of matrices, the HITS process becomes

$$\mathbf{x}^{(k)} = A^T \mathbf{y}^{(k-1)} \quad \text{and} \quad \mathbf{y}^{(k)} = A\mathbf{x}^{(k)}.$$

This can be rewritten as

$$\mathbf{x}^{(k)} = A^T A\mathbf{x}^{(k-1)} \quad \text{and} \quad \mathbf{y}^{(k)} = AA^T \mathbf{y}^{(k-1)}.$$

- HITS is just an iterative power method to compute the dominant eigenvector for $A^T A$ and AA^T.
  - $A^T A$ is the *authority matrix*.
  - AA^T is the *hub matrix*.
  - Note that the Perron–Frobenius Theorem applies to both matrices.

HITS reformulation (cont.)

Letting

$$\mathcal{A} = \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix}$$

we obtain a symmetric matrix.

Taking powers of this matrix, we get:

$$\mathcal{A}^{2k} = \begin{bmatrix} (AA^T)^k & 0 \\ 0 & (A^TA)^k \end{bmatrix}$$

$$\mathcal{A}^{2k+1} = \begin{bmatrix} 0 & A(A^TA)^k \\ (A^TA)^kA^T & 0 \end{bmatrix}.$$

Additionally, $\mathcal{A}^T\mathcal{A} = \mathcal{A}\mathcal{A}^T = \mathcal{A}^2$.

HITS reformulation (cont.)

- Let $\mathbf{u}^{(k)} = \begin{bmatrix} \mathbf{y}^{(k)} \\ \mathbf{x}^{(k)} \end{bmatrix}$

HITS reformulation (cont.)

- Let $\mathbf{u}^{(k)} = \begin{bmatrix} \mathbf{y}^{(k)} \\ \mathbf{x}^{(k)} \end{bmatrix}$
- The HITS iteration becomes

$$\mathbf{u}^{(k)} = \mathcal{A}^2 \mathbf{u}^{(k)} = \begin{bmatrix} AA^T & 0 \\ 0 & A^T A \end{bmatrix} \mathbf{u}^{(k-1)}.$$

HITS reformulation (cont.)

- Let $\mathbf{u}^{(k)} = \begin{bmatrix} \mathbf{y}^{(k)} \\ \mathbf{x}^{(k)} \end{bmatrix}$
- The HITS iteration becomes

$$\mathbf{u}^{(k)} = \mathcal{A}^2 \mathbf{u}^{(k)} = \begin{bmatrix} AA^T & 0 \\ 0 & A^T A \end{bmatrix} \mathbf{u}^{(k-1)}.$$

- The first N entries of $\mathbf{u}^{(k)}$ are the hub rankings on the nodes. The last N are the authority rankings.

HITS reformulation (cont.)

- Let $\mathbf{u}^{(k)} = \begin{bmatrix} \mathbf{y}^{(k)} \\ \mathbf{x}^{(k)} \end{bmatrix}$
- The HITS iteration becomes

$$\mathbf{u}^{(k)} = \mathcal{A}^2 \mathbf{u}^{(k)} = \begin{bmatrix} AA^T & 0 \\ 0 & A^T A \end{bmatrix} \mathbf{u}^{(k-1)}.$$

- The first N entries of $\mathbf{u}^{(k)}$ are the hub rankings on the nodes. The last N are the authority rankings.
- The HITS rankings use only the information contained in the dominant eigenvector of $\mathcal{A}$.

HITS reformulation (cont.)

- Let $\mathbf{u}^{(k)} = \begin{bmatrix} \mathbf{y}^{(k)} \\ \mathbf{x}^{(k)} \end{bmatrix}$
- The HITS iteration becomes

$$\mathbf{u}^{(k)} = \mathcal{A}^2 \mathbf{u}^{(k)} = \begin{bmatrix} AA^T & 0 \\ 0 & A^T A \end{bmatrix} \mathbf{u}^{(k-1)}.$$

- The first N entries of $\mathbf{u}^{(k)}$ are the hub rankings on the nodes. The last N are the authority rankings.
- The HITS rankings use only the information contained in the dominant eigenvector of $\mathcal{A}$.
- Can we do better than HITS?

Subgraph centrality for digraphs

Recall that the subgraph centrality of a node v_i in an undirected graph is

$$SC(i) = [\mathrm{e}^A]_{ii},$$

and that the communicability between two nodes v_i, v_j is

$$C(i,j) = [\mathrm{e}^A]_{ij}.$$

These definitions apply primarily to undirected graphs.

In analogy with HITS, we can calculate these quantities for the augmented symmetric matrix

$$\mathcal{A} = \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix}.$$

Subgraph centrality for digraphs (cont.)

As we saw in Lecture I, this matrix is the adjacency matrix of the bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ associated with the original directed graph G.

The network behind $\mathcal{A}$ can be thought of as follows:

Subgraph centrality for digraphs (cont.)

As we saw in Lecture I, this matrix is the adjacency matrix of the bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ associated with the original directed graph G.

The network behind $\mathcal{A}$ can be thought of as follows:

- Take the vertices from the original network A and make two copies V and V'.

Subgraph centrality for digraphs (cont.)

As we saw in Lecture I, this matrix is the adjacency matrix of the bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ associated with the original directed graph G.

The network behind $\mathcal{A}$ can be thought of as follows:

- Take the vertices from the original network A and make two copies V and V'.
- Undirected edges exist between the two sets based on the following rule: $E' = \{(v_i, v'_j)|$ there is a directed edge from v_i to v_j in the original network$\}$.

Subgraph centrality for digraphs (cont.)

As we saw in Lecture I, this matrix is the adjacency matrix of the bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ associated with the original directed graph G.

The network behind $\mathcal{A}$ can be thought of as follows:

- Take the vertices from the original network A and make two copies V and V'.
- Undirected edges exist between the two sets based on the following rule: $E' = \{(v_i, v'_j) |$ there is a directed edge from v_i to v_j in the original network$\}$.
- This creates a graph with $2N$ nodes.

Subgraph centrality for digraphs (cont.)

As we saw in Lecture I, this matrix is the adjacency matrix of the bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ associated with the original directed graph G.

The network behind $\mathcal{A}$ can be thought of as follows:

- Take the vertices from the original network A and make two copies V and V'.
- Undirected edges exist between the two sets based on the following rule: $E' = \{(v_i, v'_j) |$ there is a directed edge from v_i to v_j in the original network$\}$.
- This creates a graph with $2N$ nodes.
- The nodes $v_i \in V_1 \subset \mathcal{V}$ represent the original nodes in their role as hubs, while the nodes $v'_i \in V_2 \subset \mathcal{V}$ represent the original nodes in their role as authorities.

Subgraph centrality for digraphs (cont.)

Using the SVD of A, we easily find

$$e^{\mathcal{A}} = \begin{bmatrix} \cosh\left(\sqrt{AA^T}\right) & A\left(\sqrt{A^TA}\right)^{\dagger} \sinh\left(\sqrt{A^TA}\right) \\ \sinh\left(\sqrt{A^TA}\right)\left(\sqrt{A^TA}\right)^{\dagger} A^T & \cosh\left(\sqrt{A^TA}\right) \end{bmatrix}.$$

Here $A^{\dagger}$ denotes the pseudoinverse of A.

The diagonal entries of the top left block, $\cosh\left(\sqrt{AA^T}\right)$, can be used to rank the nodes of G in their role as hubs; those of the bottom right block, $\cosh\left(\sqrt{A^TA}\right)$, can be used to rank the nodes in their role as authorities.

The interpretation of the off-diagonal entries is a bit more involved, but they too provide useful information on the network.

Subgraph centrality for digraphs (cont.)

When the gap between the largest and second largest singular value of A is large enough, the resulting rankings tend to agree with those obtained by HITS.

However, if the gap is relatively small the new rankings can be significantly different, and arguably more meaningful, than those obtained with HITS, since more singular value/vector information is taken into account.

This method is an alternative to the Katz-like approach based on the row and column sums of $f(A)$. For details and comparisons, see

M. Benzi, E. Estrada and C. Klymko, *Ranking hubs and authorities using matrix functions*, Linear Algebra Appl., 438 (2013), pp. 2447-2474.