College of Computer & Information Science — Spring 2012
Northeastern University — 17 January 2012
CS 7880: Network Algorithms and Analysis — Scribe: Rajmohan Rajaraman

- Spectra of common graphs
- Computation of eigenvalues and eigenvectors

These notes and those of the next few lectures on spectral graph theory are largely based on the excellent lecture notes of Jon Kelner and Dan Spielman, who have taught related courses at MIT and Yale, respectively.

1 Spectra of common graphs

We consider spectra of common graphs.

Theorem 1. *Complete graph The Laplacian of the complete graph K_n has the eigenvector $\mathbf{1}$ with eigenvalue 0 and $n-1$ orthogonal eigenvectors with the eigenvalue n.*

Proof: We know that the null space of K_n has dimension 1. So consider any eigenvector v that is orthogonal to $\mathbf{1}$. We thus have $\sum_i v(i) = 0$. Since v is an eigenvector with, say, eigenvalue λ, it follows that

$$(n-1)v(i) - \sum_{j \neq i} v(j) = \lambda v(i)$$

for all i. Hence $(n-\lambda)v(i) = 0$ for all i. We thus have $\lambda = n$, which implies that all the remaining $n-1$ eigenvalues equal n. □

Theorem 2. *Ring The Laplacian of the ring R_n on n vertices has the eigenvectors*

$$\begin{aligned} x_k(i) &= \sin(2\pi ki/n),\ 1 \leq k < n/2 \\ y_k(i) &= \cos(2\pi ki/n),\ 0 \leq k \leq n/2 \end{aligned}$$

both with eigenvalue $2 - 2\cos(2\pi k/n)$.

Proof: If v is an eigenvector of the Laplacian of R_n, then we have

$$(2-\lambda)v(i) = v((i-1) \bmod n) + v((i+1) \bmod n).$$

for $0 \leq i < n$. It is not too hard to verify that the above vectors x_k and y_k satisfy the above equation. In particular, the following calculation shows the claim for a given k.

$$\sin(2\pi k(i-1)/n) + \sin(2\pi k(i+1)/n) = 2\sin(2\pi ki/n)cos(2\pi k/n).$$

We verify their orthogonality

$$\begin{aligned} \sum_i x_j(i)x_k(i) &= \sum_i \sin(2\pi ji/n)\sin(2\pi ki/n) \\ &= \sum_i \frac{1}{2}\left(\cos(2\pi(j+k)i/n) - \cos(2\pi(j-k)i/n)\right) \\ &= 0. \end{aligned}$$

since $\sum_i \cos(2\pi \ell i/n)$ equals 0 for every positive integer ℓ. □

2 Computation of eigenvalues and eigenvectors

One can compute the eigendecomposition of a matrix M by considering the equation

$$(M - \lambda I)v = 0,$$

where the eigenvector v is required to be non-zero. This implies that the matrix $M - \lambda I$ is non-singular, thus $\det(M - \lambda I) = 0$. Clearly, $\det(M - \lambda I)$ is a polynomial in λ; it is referred to as the characteristic polynomial of M. The roots of this polynomial are precisely the eigenvalues of M. For small matrices, one can attempt to find the roots symbolically, but this is impossible for large matrices. One way around this is to use a numerical algorithm such as Newton's method to approximate the roots of the characteristic polynomial, and then compute the eigenvectors using, say, Gaussian elimination. This is not a common method, owing to the impact of numerical errors on the accuracy of eigenvalues and eigenvectors.

The largest eigenvalue can be approximated using what is called the *power method*. Let $v_1, \ldots, v_n$ denote the n eigenvectors of M with corresponding eigenvalues $\lambda_1 \leq \ldots \leq \lambda_n$. Let v be any vector that is not orthogonal to the eigenvector v_n. Since $v_1, \ldots, v_n$ form an orthonormal basis of R^n, we have $v = \sum_i \alpha_i v_i$, where $\alpha_n \neq 0$. Then,

$$Mv = \sum_i \alpha_i \lambda_i v_i$$

with $\|Mv\| = \sqrt{\sum_i \alpha_i^2 \lambda_i^2}$. Similarly, we can compute

$$M^j v = \sum_i \alpha_i \lambda_i^j v_i$$

with $\|M^j v\| = \sqrt{\sum_i \alpha_i^2 \lambda_i^(2j)}$. If λ_n is the unique maximum eigenvalue, then

$$\lim_{j \to \infty} \frac{M^j v}{\|M^j v\|} = v_n.$$

Thus, the iterative power method, in the limit, yields the eigenvector corresponding to the largest eigenvalue. Refinements of the power method are used in the much more popular and practical QR factorization algorithm for computing the eigendecomposition of a matrix. The power method is also used to find a stationary distribution for a Markov chain, and is the basis of Google's PageRank algorithm.