Eigenvalue Problems

Last Time …

- Social Network Graphs
- Betweenness
  - Girvan-Newman Algorithm
- Graph Laplacian
  - Spectral Bisection
  - λ_2, w_2

Today …

Small deviation into eigenvalue problems …

Formulation

- Standard eigenvalue problem: Given a $n \times n$ matrix A, find scalar λ and a nonzero vector x such that

$$Ax = \lambda x$$

- λ is a eigenvalue, and x is the corresponding eigenvector

- Spectrum = $\lambda(A)$ = set of eigenvalues of A
- Spectral radius = $\rho(A) = \max\{ |\lambda| : \lambda \in \lambda(A)\}$

Characteristic Polynomial

- Equation $Ax = \lambda x$ is equivalent to
$$(A - \lambda I)x = 0$$
- Eigenvalues of A are roots of the characteristic polynomial
$$\det(A - \lambda I) = 0$$

- The characteristic polynomial is a powerful theoretical tool but usually not useful computationally

Considerations

- Properties of eigenvalue problem affecting choice of algorithm
  - Are all eigenvalues needed, or only a few?
  - Are only eigenvalues needed, or are corresponding eigenvectors also needed?
  - Is matrix real or complex?
  - Is matrix relatively small and dense, or large and sparse?
  - Does matrix have any special properties, such as symmetry?

Problem Transformations

- *Shift*: If $Ax = \lambda x$ and σ is any scalar, then $(A - \sigma I)x = (\lambda - \sigma)x$
- *Inversion*: If A is nonsingular and $Ax = \lambda x$, then $\lambda \neq 0$ and $A^{-1}x = \left(\frac{1}{\lambda}\right)x$
- Powers: If $Ax = \lambda x$, then $A^k x = \lambda^k x$
- Polynomial: If $Ax = \lambda x$ and $p(t)$ is a polynomial, then $p(A)x = p(\lambda)x$

Similarity Transforms

- B is similar to A if there is a nonsingular matrix T, such that

$$B = T^{-1}AT$$

- Then,

$$By = \lambda y \Rightarrow T^{-1}ATy = \lambda y \Rightarrow A(Ty) = \lambda(Ty)$$

- Similarity transformations preserve eigenvalues and eigenvectors are easily recovered

Diagonal form

- Eigenvalues of a diagonal matrix are the diagonal entries and the eigenvectors are columns of the Identity matrix
- The diagonal form is highly desirable in simplifying eigenvalue problems for general matrices by similarity transformations
- But not all matrices are diagonalizable by similarity transformations

Triangular form

- Any matrix can be transformed into a triangular form by similarity
- The eigenvalues are simply the diagonal values

- Eigenvectors are not as obvious, but still easy to compute

Power iteration

- Simplest method for computing one eigenvalue-eigenvector pair

$$x_k = Ax_{k-1}$$

- Converges to multiple of eigenvector corresponding to dominant eigenvalue
- We have seen this before while computing the Page Rank
- Proof of convergence ?

Convergence of Power iteration

Express starting vector x_0 in terms of the eigenvectors of A

$$x_0 = \sum_{i=1}^{n} \alpha_i \boldsymbol{v}_i$$

Then,

$$x_k = Ax_{k-1} = A^2 x_{k-2} = \cdots = A^k x_0$$

$$\sum_{i=1}^{n} \lambda_i^k \alpha_i \boldsymbol{v}_i = \lambda_n^k \left(\alpha_n \boldsymbol{v}_n + \sum_{i=1}^{n-1} \left(\frac{\lambda_i}{\lambda_n} \right)^k \alpha_i \boldsymbol{v}_i \right)$$

Since $\left| \frac{\lambda_i}{\lambda_n} \right| < 1$, successively higher powers go to zero

Power iteration with shift

- Convergence rate of power iteration depends on the ratio $\left|\frac{\lambda_{n-1}}{\lambda_n}\right|$
- It is possible to choose a shift, $A - \sigma I$, such that

$$\left|\frac{\lambda_{n-1} - \sigma}{\lambda_n - \sigma}\right| < \left|\frac{\lambda_{n-1}}{\lambda_n}\right|$$

so convergence is accelerated

- Shift must be added back to result to obtain eigenvalue of original matrix

Inverse iteration

- If the smallest eigenvalues are required rather than the largest, we can make use of the fact that the eigenvalues of A^{-1} are reciprocals of those of A, so the smallest eigenvalue of A is the reciprocal of the largest eigenvalue of A^{-1}

- This leads to the inverse iteration scheme

$$Ay_k = x_{k-1}$$
$$x_k = y_k / \|y_k\|_\infty$$

- Inverse of A is not computed explicitly, but some factorization of A is used to solve the system at each iteration

Shifted inverse iteration

- As before, the shifting strategy using a scalar σ can greatly improve convergence
- It is particularly useful for computing the eigenvector corresponding to the approximate eigenvalue
- Inverse iteration is also useful for computing the eigenvalue closest to a given value β, since if β is used as the shift, then the desired eigenvalue corresponds to the smallest eigenvalue of the shifted matrix

Deflation

- Once the dominant eigenvalue and eigenvector (λ_n, w_n) have been computed, the remaining eigenvalues can be computed using deflation, which effectively removes the known eigenvalue
- Let H be any nonsingular matrix such that $Hx = \alpha e_1$
- Then the similarity transform determined by H transforms A into,

$$HAH^{-1} = \begin{bmatrix} \lambda_n & b^T \\ 0 & B \end{bmatrix}$$

- Can now work with B to compute the next eigenvalue,
- Process can be repeated to find additional eigenvalues and eigenvectors

Deflation

- Alternate approach: let u_n be any vector such that $u_n^T w_n = \lambda_n$
- Then $A - w_n u_n^T$ has eigenvalues $\lambda_{n-1}, \ldots, \lambda_1, 0$

- Possible choices for u_n
  - $u_n = \lambda_n w_n$, if A is symmetric and w_n is normalized so that $\|w_n\|_2 = 1$
  - $u_n = \lambda_n y_n$, where y_n is the corresponding left eigenvector ($A^T y_n = \lambda_n y_n$)
  - $u_n = A^T e_k$, if w_n is normalized such that $\|w_n\|_\infty = 1$ and the k^{th} component of w_n is 1

QR Iteration

- Iteratively converges to a triangular or block-triangular form, yielding all eigenvalues of A
- Starting with $A_0 = A$, at iteration k compute QR factorization,

$$Q_k R_k = A_{k-1}$$

- And form the reverse product,

$$A_k = R_k Q_k$$

- Product of orthogonal matrices Q_k converges to matrix of corresponding eigenvectors
- If A is symmetric, then symmetry is preserved by the QR iteration, so A_k converges to a matrix that is both symmetric and triangular
  → diagonal

Preliminary reductions

- Efficiency of QR iteration can be enhanced by first transforming the matrix to be as close to triangular form as possible
- Hessenberg matrix is triangular except for one additional nonzero diagonal immediately adjacent to the main diagonal
- Symmetric Hessenberg matrix is tridiagonal
- Any matrix can be reduced to Hessenberg form in finite number of steps using Householder transformations

- Work per iteration is reduced from $\mathcal{O}(n^3)$ to $\mathcal{O}(n^2)$ for general matrices and $\mathcal{O}(n)$ for symmetric matrices

Krylov subspace methods

- Reduces matrix to Hessenberg or tridiagonal form using only matrix-vector products
- For arbitrary starting vector x_0, if

$$K_k = [x_0 \quad Ax_0 \quad \cdots \quad A^{k-1}x_0]$$

then

$$K_n^{-1} A K_n = C_n$$

where C_n is upper Hessenberg

- To obtain a better conditioned basis for span(K_n), compute the QR factorization,

$$Q_n R_n = K_n$$

so that

$$Q_n^H A Q_n = R_n C_n R_n^{-1} \equiv H$$

with H is upper Hessenberg

Krylov subspace methods

- Equating k^{th} columns on each side of the equation $AQ_n = Q_n H$ yields

$$Aq_k = h_{1k}q_1 + \cdots + h_{kk}q_k + h_{k+1,k}q_{k+1}$$

relating q_{k+1} to preceding vectors $q_1, \ldots, q_k$

- Premultiplying by q_j^H and using orthonormality

$$h_{jk} = q_j^H A q_k, \qquad j = 1, \ldots, k$$

- These relationships yield the Arnoldi iteration

Arnoldi iteration

x_0 : arbitrary nonzero starting vector

$q_1 = x_0 / \|x_0\|_2$

for $k = 1,2, \ldots$

$u_k = A q_k$

for $j = 1$ to k

$h_{jk} = q_j^H u_k$

$u_k = u_k - h_{jk} q_j$

$h_{k+1,k} = \|u_k\|_2$

if $h_{k+1,k} = 0$ then stop

$q_{k+1} = u_k / h_{k+1,k}$

Arnoldi iteration

- If

$$Q_k = [q_1 \quad \cdots \quad q_k]$$

then

$$H_k = Q_k^H A Q_k$$

is a upper Hessenberg matrix

- Eigenvalues of H_k, called Ritz values, are approximate eigenvalues of A, and Ritz vectors given by $Q_k y$, where y is an eigenvector of H_k, are corresponding approximate eigenvectors of A
- Eigenvalues of H_k must be computed by another method, such as QR iteration, but this is an easier problem as $k \ll n$

Arnoldi iteration

- Is fairly expensive in both work and storage because each new vector q_k must be orthogonalized against all previous columns of Q_k, and all must be stored for that purpose
- Is usually restarted periodically with a carefully chosen starting vector
- Ritz vectors and values produced are often good approximations to eigenvalues and eigenvectors of A after relatively few iterations

Lanczos iteration

Work and storage costs drop dramatically if the matrix is symmetric or Hermitian, since the recurrence has only three terms and H_k is tridiagonal

$q_0, \beta_0 = 0$ and x_0 = arbitrary nonzero starting vector

$q_1 = x_0 / \|x_0\|_2$

for $k = 1,2, \ldots$

$u_k = A q_k$

$\alpha_k = q_k^H u_k$

$u_k = u_k - \beta_{k-1} q_{k-1} - \alpha_k q_k$

$\beta_k = \|u_k\|_2$

if $\beta_k = 0$ then stop

$q_{k+1} = u_k / \beta_k$

Lanczos iteration

- α_k and β_k are diagonal and subdiagonal entries of symmetric tridiagonal matrix T_k
- As with Arnoldi, Lanczos does not produce eigenvalues and eigenvectors directly, but only the tridiagonal matrix T_k, whose eigenvalues and eigenvectors must be computed by another method to obtain Ritz values and vectors
- If $\beta = 0$, then the invariant subspace has already been identified, i.e., the Ritz values and vectors are already exact at that point

Lanczos iteration

- In principle, if we let Lanczos run until $k = n$, the resulting tridiagonal matrix would be orthogonally similar to A
- In practice, rounding errors cause loss of orthogonality
- Problem can be overcome by reorthogonalizing vectors
- In practice, this is usually ignored. The resulting approximations are still good

Krylov subspace methods

- Great advantage of Arnoldi and Lanczos is their ability to produce good approximations to extreme eigenvalues for $k \ll n$
- They only require one matrix-vector product per step and little auxiliary storage, so are ideally suited for large sparse matrices
- If eigenvalues are needed in the middle of the spectrum, say near σ, then the algorithms can be applied to $(A - \sigma I)^{-1}$, assuming it is practical to solve systems of the form $(A - \sigma I)x = y$