Computational Methods

CMSC/AMSC/MAPL 460

Eigenvalues and Eigenvectors

Ramani Duraiswami,

Dept. of Computer Science

Eigen Values of a Matrix

- Definition:
- A $N \times N$ matrix **A** has an eigenvector **x** (non-zero) with corresponding eigenvalue λ if

$$\mathbf{Ax}=\lambda\ \mathbf{x}$$

- This means

$$\mathbf{Ax}-\lambda\ \mathbf{x}=0 \qquad\qquad (\mathbf{A}-\lambda \mathrm{I})\ \mathbf{x}=0$$

- If a matrix vector product gives a zero vector, then either the vector is zero, or the matrix has zero determinant (is singular).
- Here this means $\det(A-\lambda\ I) =0$

Left and Right Eigenvectors

- Right eigenvector of a matrix A is

$$Ax=\lambda x$$

- For a $N\times N$ matrix we can also define a left matrix product

$$y^t A=v^t$$

- So if we have

$$y^t A=\lambda y^t$$

then y is a left eigenvector of A

- If A is symmetric $A=A^t$
- $(Ax)^t = x^t A^t = x^t A = (\lambda x)^t = \lambda x^t$
- So left and right eigenvectors of a symmetric matrix are the same

Symmetric Matrices

- A matrix is symmetric if its transpose is equal to itself
- **A** is symmetric if $\mathbf{A}^t = \mathbf{A}$ For a complex matrix $\mathbf{A}^H = \mathbf{A}$
- Eigenvalues and Eigenvectors of a real symmetric (complex hermitian) matrix are real and eigenvectors are orthogonal.

Characteristic Equation

- A**x** = λ **x** can be written as

  $$(A- \lambda I)\mathbf{x} = \mathbf{0}$$

  which holds for x ≠ 0, so (A- λ I) is singular and

  $$\det(A- \lambda I) = 0$$

- This is called the characteristic polynomial. If A is n × n the polynomial is of degree n and so A has n eigenvalues, counting multiplicities.

Example

$$A = \begin{pmatrix} 4 & 3 \\ 1 & 2 \end{pmatrix} \qquad A - \lambda I = \begin{pmatrix} 4-\lambda & 3 \\ 1 & 2-\lambda \end{pmatrix}$$

$$\det(A - \lambda I) = 0 \quad \Rightarrow \quad (4-\lambda)(2-\lambda) - (1)(3) = 0$$

$$\lambda^2 - 6\lambda + 5 = 0 \quad \Rightarrow \quad (\lambda - 5)(\lambda - 1) = 0$$

- Hence the two eigenvalues are 1 and 5.

Example (continued)

- Once we have the eigenvalues, the eigenvectors can be obtained by substituting back into $(A- \lambda\ I)\mathbf{x} = \mathbf{0}$.
- This gives eigenvectors $(1\ -1)^T$ and $(1\ 1/3)^T$
- Note that we can scale the eigenvectors any way we want.
- Determinant are not used for finding the eigenvalues of large matrices.

Positive Definite Matrices

- A complex matrix A is *positive definite* if for every nonzero complex vector $\mathbf{x}$ the quadratic form $\mathbf{x}^H A \mathbf{x}$ is real and:

$$\mathbf{x}^H A \mathbf{x} > 0$$

  where $\mathbf{x}^H$ denotes the conjugate transpose of $\mathbf{x}$ (i.e., change the sign of the imaginary part of each component of $\mathbf{x}$ and then transpose).

Eigenvalues of Positive Definite Matrices

- If A is positive definite and λ and $\mathbf{x}$ are an eigenvalue/eigenvector pair, then:

$$A\mathbf{x} = \lambda\,\mathbf{x} \qquad \mathbf{x}^H A\mathbf{x} = \lambda\,\mathbf{x}^H\mathbf{x}$$

- Since $\mathbf{x}^H A\mathbf{x}$ and $\mathbf{x}^H\mathbf{x}$ are both real and positive it follows that λ is real and positive.

Properties of Positive Definite Matrices

- If A is a positive definite matrix then:
  - A is nonsingular.
  - The inverse of A is positive definite.
  - Gaussian elimination can be performed on A without pivoting.
  - The eigenvalues of A are positive.

Hermitian Matrices

- A square matrix for which $A = A^H$ is said to be an *Hermitian* matrix.
- If A is real and Hermitian it is said to be *symmetric*, and $A = A^T$.
- Every Hermitian matrix is positive definite.
- Every eigenvalue of an Hermitian matrix is real.
- Different eigenvectors of an Hermitian matrix are orthogonal to each other, i.e., their scalar product is zero.

Eigen Decomposition

- Let $\lambda_1, \lambda_2, \ldots, \lambda_n$ be the eigenvalues of the $n \times n$ matrix A and $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ the corresponding eigenvectors.
- Let Λ be the diagonal matrix with $\lambda_1, \lambda_2, \ldots, \lambda_n$ on the main diagonal.
- Let X be the $n \times n$ matrix whose jth column is $\mathbf{x}_j$.
- Then $AX = X \Lambda$, and so we have the *eigen decomposition* of A:

$$A = X^t \Lambda X^{-1}$$

- This requires X to be invertible, thus the eigenvectors of A must be linearly independent.

Powers of Matrices

- If $A = X^t \Lambda X^{-1}$ then:

  $$A^2 = (X^t \Lambda X^{-1})(X^t \Lambda X^{-1}) = X^t \Lambda (X^{-1}X) \Lambda X^{-1} = X^t \Lambda^2 X^{-1}$$

  Hence we have:

  $$A^p = X^t \Lambda^p X^{-1}$$

- Thus, A^p has the same eigenvectors as A, and its eigenvalues are $\lambda_1^p, \lambda_2^p, \ldots, \lambda_n^p$.
- We can use these results as the basis of an iterative algorithm for finding the eigenvalues of a matrix.

The Power Method

- Label the eigenvalues in order of decreasing absolute value so $|\lambda_1|>|\lambda_2|>\ldots|\lambda_n|$.
- Consider the iteration formula:

$$\mathbf{y}_{k+1} = A\mathbf{y}_k$$

where we start with some initial $\mathbf{y}_0$, so that:

$$\mathbf{y}_k = A^k\mathbf{y}_0$$

- Then $\mathbf{y}_k$ converges to the eigenvector $\mathbf{x}_1$ corresponding the eigenvalue λ_1.

Proof

- We know that $A^k = X \Lambda^k X^{-1}$, so:

$$\mathbf{y}_k = A^k \mathbf{y}_0 = X \Lambda^k X^{-1} \mathbf{y}_0$$

- Now we have:

$$\Lambda^k = \begin{pmatrix} \lambda_1^k & & & \\ & \lambda_2^k & & \\ & & \ddots & \\ & & & \lambda_n^k \end{pmatrix} = \lambda_1^k \begin{pmatrix} 1 & & & \\ & \lambda_2^k / \lambda_1^k & & \\ & & \ddots & \\ & & & \lambda_n^k / \lambda_1^k \end{pmatrix}$$

- The terms on the diagonal get smaller in absolute value as k increases, since λ_1 is the dominant eigenvalue.

Proof (continued)

- So we have

$$y_k = \lambda_1^k \begin{pmatrix} \vdots & & \vdots \\ x_1 & \cdots & x_n \\ \vdots & & \vdots \end{pmatrix} \begin{pmatrix} 1 & & & \\ & 0 & & \\ & & \ddots & \\ & & & 0 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix} = \lambda_1^k c_1 x_1$$

- Since $\lambda_1^k c_1 \mathbf{x}_1$ is just a constant times $\mathbf{x}_1$ then we have the required result.

Example

- Let A = [2 -12; 1 -5] and $\mathbf{y}_0$=[1 1]’
- $\mathbf{y}_1$ = -4[2.50 1.00]’
- $\mathbf{y}_2$ = 10[2.80 1.00]’
- $\mathbf{y}_3$ = -22[2.91 1.00]’
- $\mathbf{y}_4$ = 46[2.96 1.00]’
- $\mathbf{y}_5$ = -94[2.98 1.00]’
- $\mathbf{y}_6$ = -190[2.99 1.00]’
- The iteration is converging on a scalar multiple of [3 1]’, which is the correct dominant eigenvector.

Rayleigh Quotient

- Note that once we have the eigenvector, the corresponding eigenvalue can be obtained from the *Rayleigh quotient*:

  dot(A**x**,**x**)/dot(**x**,**x**)

  where dot(**a**,**b**) is the scalar product of vectors **a** and **b** defined by:

  $\mathrm{dot}(\mathbf{a},\mathbf{b}) = a_1b_1+a_2b_2+\ldots+a_nb_n$
- So for our example, $\lambda_1 = -2$.

Scaling

- The λ_1^k can cause problems as it may become very large as the iteration progresses.
- To avoid this problem we scale the iteration formula:

$$\mathbf{y}_{k+1} = A(\mathbf{y}_k / r_{k+1})$$

where r_{k+1} is the component of $A\mathbf{y}_k$ with largest absolute value.

Example with Scaling

- Let A = [2 -12; 1 -5] and $\mathbf{y}_0$=[1 1]'
- A$\mathbf{y}_0$ = [-10 -4]' so r_1=-10 and $\mathbf{y}_1$=[1.00 0.40]'.
- A$\mathbf{y}_1$ = [-2.8 -1.0]' so r_2=-2.8 and $\mathbf{y}_2$=[1.0 0.3571]'.
- A$\mathbf{y}_2$ = [-2.2857 -0.7857]' so r_3=-2.2857 and $\mathbf{y}_3$=[1.0 0.3437]'.
- A$\mathbf{y}_3$ = [-2.1250 -0.7187]' so r_4=-2.1250 and $\mathbf{y}_4$=[1.0 0.3382]'.
- A$\mathbf{y}_4$ = [-2.0588 -0.6912]' so r_5=-2.0588 and $\mathbf{y}_5$=[1.0 0.3357]'.
- A$\mathbf{y}_5$ = [-2.0286 -0.6786]' so r_6=-2.0286 and $\mathbf{y}_6$=[1.0 0.3345]'.
- r is converging to the correct eigenvector -2.

Scaling Factor

- At step k+1, the scaling factor r_{k+1} is the component with largest absolute value is $A\mathbf{y}_k$.
- When k is sufficiently large $A\mathbf{y}_k \simeq \lambda_1 \mathbf{y}_k$.
- The component with largest absolute value in $\lambda_1 \mathbf{y}_k$ is λ_1 (since $\mathbf{y}_k$ was scaled in the previous step to have largest component 1).
- Hence, $r_{k+1} \to \lambda_1$ as $k \to \infty$.

MATLAB Code

```
function [lambda,y]=powerMethod(A,y,n)
for (i=1:n)
    y = A*y;
    [c j] = max(abs(y));
    lambda = y(j);
    y = y/lambda;
end
```

Convergence

- The Power Method relies on us being able to ignore terms of the form $(\lambda_j / \lambda_1)^k$ when k is large enough.
- Thus, the convergence of the Power Method depends on $|\lambda_2|/|\lambda_1|$.
- If $|\lambda_2|/|\lambda_1|=1$ the method will not converge.
- If $|\lambda_2|/|\lambda_1|$ is close to 1 the method will converge slowly.

Orthonormal Vectors

- A set S of nonzero vectors are *orthonormal* if, for every **x** and **y** in S, we have dot(**x**,**y**)=0 (orthogonality) and for every **x** in S we have $||\mathbf{x}||_2=1$ (length is 1).

The QR Algorithm

- The QR algorithm for finding eigenvalues is based on the QR factorisation that represents a matrix A as:

$$A = QR$$

  where Q is a matrix whose columns are orthonormal, and R is an upper triangular matrix.
- Note that $Q^H Q = I$ and $Q^{-1} = Q^H$.
- Q is termed a *unitary* matrix.

QR Algorithm without Shifts

$A_0 = A$

for k=1,2,…

$Q_k R_k = A_k$

$A_{k+1} = R_k Q_k$

end

Since:

$$A_{k+1} = R_k Q_k = Q_k^{-1} A_k Q_k$$

then A_k and A_{k+1} are similar and so have the same eigenvalues.

A_{k+1} tends to an upper triangular matrix with the same eigenvalues as A. These eigenvalues lie along the main diagonal of A_{k+1}.

QR Algorithm with Shift

$A_0 = A$
for k=1,2,…
    $s = A_k(n,n)$
    $Q_k R_k = A_k - sI$
    $A_{k+1} = R_k Q_k + sI$
end

Since:

$$A_{k+1} = R_k Q_k + sI$$

$$= Q_k^{-1}(A_k - sI)Q_k + sI$$

$$= Q_k^{-1} A_k Q_k$$

so once again A_k and A_{k+1} are similar and so have the same eigenvalues.

The shift operation subtracts s from each eigenvalue of A, and speeds up convergence.

MATLAB Code for QR Algorithm

- Let A be an $n \times n$ matrix

  ```
  n = size(A,1);
  I = eye(n,n);
  s = A(n,n); [Q,R] = qr(A-s*I); A = R*Q+s*I
  ```

- Use the up arrow key in MATLAB to iterate or put a loop round the last line.

Deflation

- The eigenvalue at A(n,n) will converge first.
- Then we set s=A(n-1,n-1) and continue the iteration until the eigenvalue at A(n-1,n-1) converges.
- Then set s=A(n-2,n-2) and continue the iteration until the eigenvalue at A(n-2,n-2) converges, and so on.
- This process is called *deflation*.