Computational Linear Algebra: Eigenvalues and Eigenvectors

David Shuman

Eigenvalues and Eigenvectors

Definition (Eigenvector)

If A is an $n \times n$ then an eigenvector for A is a nonzero vector v such that

$$\mathrm{Av} = \lambda \mathrm{v}, \qquad \text{for some } \lambda \in \mathbb{C}$$

The scalar λ is the eigenvalue associated with the eigenvector v.

Two common uses:

- Algorithmic: Reduce coupled system to collection of scalar problems
- Physical: Insights into behavior of systems (e.g., resonance/vibration, stability analysis) (possible topics for TR4)

Definition (Characteristic polynomial)

$$p_A(z) = \det(zI - A)$$

- Theorem: λ is an eigenvalue of A if and only if $p_A(\lambda) = 0$
- One implication: Real matrices may have complex eigenvalues

A Fundamental Difficulty Computing Eigenvalues

- Any monic polynomial can be written as

$$
\begin{aligned}
p(z) &= z^n + a_{n-1}z^{n-1} + \ldots a_1 z + a_0 \\
&= (-1)^n \det \begin{pmatrix}
-z & & & & & -a_0 \\
1 & -z & & & & -a_1 \\
 & 1 & -z & & & -a_2 \\
 & & 1 & -z & & \vdots \\
 & & & \ddots & -z & -a_{n-2} \\
 & & & & 1 & (-z - a_{n-1})
\end{pmatrix}
\end{aligned}
$$

- So the roots of $p(\cdot)$ are equal to the eigenvalues of

$$
A = \begin{pmatrix}
0 & & & & & -a_0 \\
1 & 0 & & & & -a_1 \\
 & 1 & 0 & & & -a_2 \\
 & & 1 & 0 & & \vdots \\
 & & & \ddots & 0 & -a_{n-2} \\
 & & & & 1 & a_{n-1}
\end{pmatrix}
$$

A Fundamental Difficulty Computing Eigenvalues (cont.)

Theorem (Abel, 1824)

For any $n \geq 5$, there is a polynomial $p(z)$ of degree n with rational coefficients that has a real root $p(r) = 0$ with the property that r cannot be written using any expression involving rational numbers, addition, subtraction, multiplication, division, and kth roots.

- No analogue of the quadratic formula for higher order polynomials
- Can't find the exact roots of an arbitrary polynomial in a finite number of steps
- Can't compute eigenvalues of an arbitrary matrix in a finite number of steps
- Eigenvalue computation methods are iterative, not direct

Factorizing Matrices to Compute Eigenvalues

- There are many different algorithms to compute these eigenvalue revealing factorizations
- Still an active area of research
- Best algorithm depends on many of the criteria we've seen before:
  - Structural properties of the matrix (is it normal? is it symmetric? is it sparse? how are the eigenvalues distributed?)
  - What do we want to compute? (all of the eigenvectors and eigenvalues? just the eigenvalues? just a few of the eigenvalues and eigenvectors?)
  - Trade-offs between complexity/speed, memory, numerical stability
- We will not go through most of these algorithms, and the details such as complexity vs. stability trade-offs are beyond the scope of the course
- However, you now have the tools to read about algorithms that you might come across

Power Iteration, Inverse Power Iteration, and Rayleigh Quotient Iteration

See Activity!

Diagonalization

Suppose that A is an $n \times n$ matrix with the following eigeninformation

$$\begin{array}{ccccc} \lambda_1, & \lambda_2, & \lambda_3, & \cdots & \lambda_n \\ \mathsf{v}_1, & \mathsf{v}_2, & \mathsf{v}_3, & & \mathsf{v}_n \end{array}$$

$\mathsf{A}\mathsf{v}_i = \lambda_i \mathsf{v}_i$ implies

$$\mathsf{A} \underbrace{\begin{bmatrix} | & | & \cdots & | \\ \mathsf{v}_1 & \mathsf{v}_2 & \cdots & \mathsf{v}_n \\ | & | & \cdots & | \end{bmatrix}}_{\mathrm{V}} = \begin{bmatrix} | & | & \cdots & | \\ \lambda_1\mathsf{v}_1 & \lambda_2\mathsf{v}_2 & \cdots & \lambda_n\mathsf{v}_n \\ | & | & \cdots & | \end{bmatrix}$$

$$= \underbrace{\begin{bmatrix} | & | & \cdots & | \\ \mathsf{v}_1 & \mathsf{v}_2 & \cdots & \mathsf{v}_n \\ | & | & \cdots & | \end{bmatrix}}_{\mathrm{V}} \underbrace{\begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix}}_{\Lambda}$$

$$\Rightarrow \quad \mathsf{A} = \mathrm{V}\Lambda\mathrm{V}^{-1}$$

Similar Matrices

- On the previous slide we have factored our matrix in the form
$$\mathsf{A} = \mathsf{V}\Lambda\mathsf{V}^{-1}$$
- This is an example of similar matrices
- We say that two $n \times n$ matrices A and B are similar if there exists an invertible $n \times n$ matrix T such that
$$\mathsf{A} = T\mathsf{B}T^{-1}$$
- The similar matrices A and B are very much alike. They have the same:
  - rank and nullity
  - determinant
  - characteristic polynomial
  - eigenvalues (and multiplicities of eigenvalues)
- In fact, they are the matrices of the linear transformation being described in two different bases. The matrix T is the change of basis matrix

Diagonalizability

- Q: Can I diagonalize any $n \times n$ matrix A?
- A: No, a *diagonalization* exists if and only if A is nondefective
- Nondefective $\Leftrightarrow$ algebraic multiplicity of each eigenvalue of A is equal to its geometric multiplicity
- Algebraic multiplicity: how many times the eigenvalue is repeated as a root of the characteristic polynomial
- Geometric multiplicity: number of linearly independent eigenvectors associated with that eigenvalue; i.e., the dimension of its eigenspace. The eigenspace corresponding to λ is the set of all eigenvectors associated with eigenvalue λ. It is a subspace of $\mathbb{R}^n$.

$$E_\lambda = \{\, \mathrm{v} \,|\, \mathrm{Av} = \lambda \mathrm{v} \,\}.$$

- General fact: algebraic multiplicity $\geq$ geometric multiplicity
- Example: $\mathrm{B} = \begin{pmatrix} 2 & 1 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{pmatrix}$
  - Algebraic multiplicity of $\lambda = 2$ is 3
  - Geometric multiplicity of $\lambda = 2$ is 1

Eigenvalue Revealing Factorizations

- The main technique to compute eigenvalues is to factor the matrix A into an eigenvalue revealing factorization
- We have already seen the first such factorization:

$$\mathsf{A} = \mathsf{V}\Lambda\mathsf{V}^{-1},$$

  which can be done when A is nondefective
- When A has an additional property, we can factorize it so that the eigenvector matrix is unitary:

$$\mathsf{A} = \mathsf{Q}\Lambda\mathsf{Q}^*$$

  - The asterisk here is conjugate transpose. When Q is real, this is just the transpose and Q is an orthonormal matrix. When Q has complex values, the definition of unitary is that $\mathsf{QQ}^* = \mathsf{Q}^*\mathsf{Q} = I$
  - The extra property required is that A is normal: $\mathsf{A}^*\mathsf{A} = \mathsf{AA}^*$ (or in the case that A is real, $\mathsf{A}^T\mathsf{A} = \mathsf{AA}^T$)
  - Symmetric matrices and circulant matrices are all normal matrices

Reminder: Spectral Theorem for Real Symmetric Matrices

If A is a real, symmetric $n \times n$ matrix ($\mathsf{A}^T = \mathsf{A}$), then

- A has real eigenvalues
- If $\mathsf{v}_1, \mathsf{v}_2$ are eigenvectors corresponding to distinct eigenvalues $\lambda_1 \neq \lambda_2$. Then

$$\begin{aligned} \lambda_1 \langle \mathsf{v}_1, \mathsf{v}_2 \rangle &= \langle \lambda_1 \mathsf{v}_1, \mathsf{v}_2 \rangle = \langle \mathsf{A}\mathsf{v}_1, \mathsf{v}_2 \rangle = \langle \mathsf{v}_1, \mathsf{A}^T \mathsf{v}_2 \rangle \\ &= \langle \mathsf{v}_1, \mathsf{A}\mathsf{v}_2 \rangle = \langle \mathsf{v}_1, \lambda_2 \mathsf{v}_2 \rangle = \lambda_2 \langle \mathsf{v}_1, \mathsf{v}_2 \rangle, \end{aligned}$$

  so $\langle \mathsf{v}_1, \mathsf{v}_2 \rangle = 0$ and the eigenvectors of A are orthogonal

- So for a real symmetric matrix, you can choose real, orthonormal eigenvectors such that $\mathsf{A} = \mathsf{Q}\Lambda\mathsf{Q}^T$

Eigenvalue Revealing Factorizations (cont.)

- Every A (even those that are defective or not normal) has a Schur factorization:

$$A = QTQ^*$$

- Here, Q is unitary, and T is an upper triangular matrix
- Q: Why is this an eigenvalue revealing factorization?
- A: Because the eigenvalues of an upper triangular matrix are just the diagonal elements!
- Another factorization when A is real is the real Schur factorization, which guarantees Q is also real, but then T is allowed to have 2×2 blocks along the diagonal

The QR Algorithm

- We'll briefly examine one general purpose method, of which there are different variations/extensions
- The goal of this method is to find the eigenvectors and eigenvalues of A all at once

Pseudocode:

$$
\begin{aligned}
& A_0 = A \\
& \text{for } k = 1, 2, \ldots \\
& \qquad Q_k R_k = A_{k-1} && \text{QR factorization of } A_{k-1} \\
& \qquad A_k = R_k Q_k && \text{Recombine factors in reverse order}
\end{aligned}
$$

QR Algorithm: Convergence

- Let $A_0 = A$, and we have

$$\begin{aligned} \mathrm{A}_0 &:= \mathrm{Q}_1\mathrm{R}_1 \\ \mathrm{A}_1 &:= \mathrm{R}_1\mathrm{Q}_1 = \mathrm{Q}_2\mathrm{R}_2 \\ \mathrm{A}_2 &:= \mathrm{R}_2\mathrm{Q}_2 = \mathrm{Q}_3\mathrm{R}_3 \dots \end{aligned}$$

- Then

$$\begin{aligned} &\mathrm{A}_{k-1} = \mathrm{Q}_k\mathrm{R}_k = \mathrm{Q}_k\mathrm{R}_k\mathrm{Q}_k\mathrm{Q}_k^T = \mathrm{Q}_k\mathrm{A}_k\mathrm{Q}_k^T \\ &\Rightarrow \mathrm{A}_k = \mathrm{Q}_k^T\mathrm{A}_{k-1}\mathrm{Q}_k \\ &\Rightarrow \mathrm{A}_k = \mathrm{Q}_k^T \dots \mathrm{Q}_2^T\mathrm{Q}_1^T\mathrm{A}\mathrm{Q}_1\mathrm{Q}_2 \dots \mathrm{Q}_k \\ &\Rightarrow \mathrm{A} = \mathrm{Q}_1\mathrm{Q}_2 \dots \mathrm{Q}_k\mathrm{A}_k\mathrm{Q}_k^T \dots \mathrm{Q}_2^T\mathrm{Q}_1^T \end{aligned}$$

- A is similar to A_k
- A_k converges to a diagonal matrix under appropriate technical conditions (e.g., for symmetric matrices, if the magnitudes of the eigenvalues are all distinct - see Thm 12.4 in book)

The QR Algorithm: Intuition

- **Block power iteration:** Suppose that you have a guess for the n orthogonal vectors $x_1, x_2, \dots, x_n$. Put them in a matrix X and multiply:

$$A \cdot \begin{bmatrix} | & | & \cdots & | \\ x_1 & x_2 & \cdots & x_n \\ | & | & \cdots & | \end{bmatrix} = \begin{bmatrix} | & | & \cdots & | \\ A \cdot x_1 & A \cdot x_2 & \cdots & A \cdot x_n \\ | & | & \cdots & | \end{bmatrix}$$

- The new vectors $Ax_1, Ax_2, \dots, Ax_n$
  - are no longer orthogonal
  - have moved towards the dominant eigenvector
- Main idea: before applying A again, let's orthonormalize the current estimate of eigenvectors: $AX = Q_1 R_1$
- Then repeat this process with the new set of vectors in Q_1:

$$AQ_1 = Q_2 R_2$$
$$AQ_2 = Q_3 R_3 \dots$$

- This is commonly called *simultaneous iteration*
- Can show that if you start with $X = I$, simultaneous iteration is equivalent to the QR algorithm

The QR Algorithm: Towards a Practical Version

- This is yet another example of a major theme of the algorithms we've seen throughout the semester: putting zeros into matrices to generate factorizations (see also LU, Cholesky, QR with Householder reflectors)
- Some extensions to get to the algorithms used in practical eigenvalue solvers:
  - Preprocess A: If it is symmetric, reduce it to a tridiagonal form, and if it is not symmetric reduce it to an upper Hessenberg form (upper triagonal plus first subdiagonal)
  - Use shifts at each step like we did with the inverse power iteration, and perform QR factorization on $\mathrm{A}_{k-1} - s_k I$
  - When off-diagonal elements get close to 0, split into submatrices (this is called *deflation*)
- QR with these extensions was THE method of choice to compute eigenvalues from 1960s to 1990s
  - New tweaks still being made in software such as LAPACK
  - New algorithms created for solving symmetric eigenvalue problems, but variants of QR are still mainstays for non-symmetric eigenvalue problems