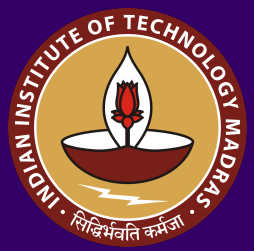
Scientific Computing & Advanced Programming Lab

SAURAV SAMANTARAY

Department of Mathematics
IIT Madras

Lecture-Slides-7



sauravsray@iitm.ac.in

Eigenvalue Decomposition

- For a square matrix $A \in \mathbb{R}^{n \times n}$, an eigenvector $\mathbf{v} \neq \mathbf{0}$ satisfies

$$\boxed{A\mathbf{v} = \lambda \mathbf{v}}$$

where λ is the corresponding eigenvalue.

- If A has n linearly independent eigenvectors, we can write $\boxed{A = V \Lambda V^{-1}}$ where

$$V = \begin{bmatrix} | & | & & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \\ | & | & & | \end{bmatrix}, \qquad \Lambda = \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{bmatrix}.$$

Interpretation

The eigenvectors provide special directions that are only **scaled** by the action of A.

Why Eigen decomposition?

- If $A = V\Lambda V^{-1}$, then many matrix operations become simpler.
- For example, $A^k = V\Lambda^k V^{-1}$, since Λ is diagonal,

$$\Lambda^k = \begin{bmatrix} \lambda_1^k & & 0 \\ & \ddots & \\ 0 & & \lambda_n^k \end{bmatrix}.$$

- This is useful for studying: powers of matrices, dynamical systems, stability, differential equations, iterative numerical methods, etc. .
- If A is symmetric i.e., $A = A^T$, the decomposition has the particularly nice form

$$\boxed{A = Q\Lambda Q^T}$$

where Q is orthogonal:$Q^TQ = I$.

Computing eigenvalues exactly is ”impossible.”

- There is no direct method for computing eigenvalues for matrices of size five or higher in general
- One may wonder if some day we will discover a direct method.
- It can be proved that such a method cannot exist for general matrices of size five or higher.
- The Abel-Ruffini theorem (also known as Abel’s impossibility theorem) says that there is no general algebraic solution to polynomial equations of degree five or higher.
- no direct method exists for exactly finding roots of polynomials of degree five or higher.

Methods for computing a single eigenvalue: Power Iteration

- Suppose that A is a square diagonalizable matrix.
- Then A has an eigenvalue decomposition $A = VAV^{-1}$
- Assume that the eigenvectors form a basis. Write an arbitrary initial vector x_0 as:
$$x_0 = c_1 v_1 + c_2 v_2 + \ldots + c_n v_n.$$
- Applying A:
$$Ax_0 = c_1 \lambda_1 v_1 + c_2 \lambda_2 v_2 + \ldots + c_n \lambda_n v_n.$$
- Similarly, applying A k times we have:
$$A^k x_0 = c_1 \lambda_1^k v_1 + c_2 \lambda_2^k v_2 + \ldots + c_n \lambda_n^k v_n.$$
- Suppose: $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \ldots \geq |\lambda_n|$.

Methods for computing a single eigenvalue: Power Iteration

- Then:

$$A^k x_0 = \lambda_1 \left[c_1 v_1 + c_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k v_2 + \ldots + c_n \left(\frac{\lambda_n}{\lambda_1} \right)^k v_n \right]$$

- As $k \to \infty$:

$$\frac{|\lambda_i|}{|\lambda_1|} \to 0, \quad i > 1$$

- Therefore, the first term in the sum dominates the sum as $k \to \infty$
- Consequently,

$$A^k x_0 \approx \lambda_1 c_1 v_1$$

Power Iteration: Algorithm

Given: $A \in \mathbb{R}^{n \times n}$, tolerance $\varepsilon > 0$

1. Choose a nonzero initial vector x_0 and normalize:

$$x_0 \leftarrow \frac{x_0}{\|x_0\|_2}.$$

2. For $k = 0, 1, 2, \ldots$:

$$y_k = Ax_k,$$

$$x_{k+1} = \frac{y_k}{\|y_k\|_2}.$$

3. Check convergence:

$$\|x_{k+1} - x_k\|_2 < \varepsilon.$$

Power Iteration: Algorithm

- The last algorithm only gives the dominant eigenvector.
- What about the eigenvalue?

Rayleigh quotient iteration

- if x is an eigenvector with $Ax = \lambda x$
- Then $x^T Ax = \lambda x^T x$
- Define $r(x)$ as: $$r(x) = \frac{x^T Ax}{x^T x}, \qquad x \neq 0$$
- Every iteration on the power iteration compute:

$$r_k = \frac{x_k^T Ax_k}{x_k^T x_k}$$

Rayleigh Quotient

- For $x \neq 0$, the Rayleigh quotient is $r(x) = \frac{x^T A x}{x^T x}$.
- For a symmetric matrix $A = A^T$,

$$\nabla r(x) = \frac{2}{x^T x} \left(Ax - r(x)x \right).$$

- If x_i is an eigenvector, $Ax_i = \lambda_i x_i$,, then

$$r(x_i) = \lambda_i$$

- and therefore $\boxed{\nabla r(x_i) = 0.}$
- Thus, eigenvectors are stationary points of the Rayleigh quotient.

Local Error of the Rayleigh Quotient

- Let x_i be an eigenvector and let y be close to x_i.
- Taylor expansion:

$$r(y) = r(x_i) + \nabla r(x_i)^T (y - x_i) + \frac{1}{2}(y - x_i)^T H(x_i)(y - x_i) + \mathcal{O}(\|y - x_i\|^3).$$

- Since $\nabla r(x_i) = 0, \qquad r(x_i) = \lambda_i,$ we obtain

$$r(y) - \lambda_i = \frac{1}{2}(y - x_i)^T H(x_i)(y - x_i) + \mathcal{O}(\|y - x_i\|^3).$$

- Hence, for $A = A^T$ $\boxed{|r(y) - \lambda_i| = \mathcal{O}(\|y - x_i\|^2)}$
- In other words, the Rayleigh quotient has second-order eigenvalue accuracy.

Numerical Verification of Second-Order Accuracy

- $$d_k \approx C e_k^2, \qquad e_k = \|x_k - x_i\|, \qquad d_k = |r(x_k) - \lambda_i|,$$

- $$\log d_k \approx \log C + 2 \log e_k.$$

- For two successive iterations,

$$\log d_{k+1} - \log d_k \approx 2\big(\log e_{k+1} - \log e_k\big).$$

- Hence define the observed order

$$\boxed{p_k = \frac{\log d_{k+1} - \log d_k}{\log e_{k+1} - \log e_k} = \frac{\log(d_{k+1}/d_k)}{\log(e_{k+1}/e_k)} \longrightarrow 2.}$$

What Changes for a Nonsymmetric Matrix?

- For general A, $\boxed{r(x) = \dfrac{x^T A x}{x^T x}}$ still makes sense.
- However, $\nabla(x^T A x) = (A + A^T)x$, so

$$\boxed{\nabla r(x) = \frac{(A + A^T)x - 2r(x)x}{x^T x}.}$$

- If $Ax_i = \lambda_i x_i$, then $\nabla r(x_i) = \frac{(A^T - A)x_i}{x_i^T x_i}$, which is generally non-zero.
- Therefore, the stationary-point argument is special to $A = A^T$.
- The Rayleigh quotient can still estimate eigenvalues, but its second-order error result does not generally hold for nonsymmetric matrices.

QR Decomposition: Motivation

- $$A = \begin{pmatrix} | & & | \\ a_1 & \cdots & a_n \\ | & & | \end{pmatrix}$$

  a matrix whose columns are not necessarily orthogonal.

- We would like to replace the columns of A by an **orthonormal basis** spanning the same space.

  $$\boxed{\{a_1, \ldots, a_n\} \quad \longrightarrow \quad \{q_1, \ldots, q_n\}}$$

  where

  $$q_i^T q_j = \delta_{ij}.$$

- This leads to the factorization

  $$\boxed{A = QR}$$

  where Q contains the orthonormal basis and R contains the

Orthogonal and Orthonormal Vectors

- Two vectors q_i and q_j are orthogonal if

$$q_i^T q_j = 0, \qquad i \neq j.$$

- They are orthonormal if, in addition,

$$\|q_i\| = 1.$$

- $$Q = \begin{pmatrix} | & & | \\ q_1 & \cdots & q_n \\ | & & | \end{pmatrix},$$

  then orthonormality of the columns is equivalent to

$$\boxed{Q^T Q = I} \implies \boxed{Q^{-1} = Q^T}$$

- An important consequence is $\|Qx\| = \|x\|$. Thus orthogonal matrices preserve lengths and angles.

Gram–Schmidt: First Step

Given linearly independent vectors $a_1, a_2, \ldots, a_n$, we construct an orthonormal set $q_1, q_2, \ldots, q_n$.

1. Start with the first vector:

$$\boxed{q_1 = \frac{a_1}{\|a_1\|}} \implies \|q_1\| = 1.$$

2. The second vector a_2 generally has a component in the q_1 direction. Its projection onto q_1 is

$$\text{proj}_{q_1}(a_2) = (q_1^T a_2) q_1.$$

3. Remove this component: $u_2 = a_2 - (q_1^T a_2) q_1.$

4. Then

$$q_1^T u_2 = 0.$$

Gram–Schmidt: Normalization

- We now normalize the orthogonalized vector: $\boxed{q_2 = \dfrac{u_2}{\|u_2\|}}$
- Therefore, $q_1^T q_2 = 0, \qquad \|q_1\| = \|q_2\| = 1.$
- For the third vector, remove its components in both previous directions:

$$u_3 = a_3 - (q_1^T a_3) q_1 - (q_2^T a_3) q_2.$$

- Then $q_3 = \frac{u_3}{\|u_3\|}$.

The general step is therefore $\boxed{u_j = a_j - \sum_{i=1}^{j-1} (q_i^T a_j) q_i}$

followed by

$$\boxed{q_j = \frac{u_j}{\|u_j\|}.}$$

From Gram-Schmidt $\rightarrow$ QR

- Rearranging the Gram-Schmidt equations gives

$$a_1 = (q_1^T a_1) q_1,$$
$$a_2 = (q_1^T a_2) q_1 + (q_2^T a_2) q_2,$$

- and, in general, $\boxed{a_j = \sum_{i=1}^{j} (q_i^T a_j) q_i.}$
- Thus every column of A can be expressed in the orthonormal basis $q_1, \ldots, q_n$.
- Collecting the q_i into $Q = \begin{pmatrix} | & & | \\ q_1 & \cdots & q_n \\ | & & | \end{pmatrix}$, we obtain

$$\boxed{A = QR}.$$

Structure of R

- The coefficients in the expansion of a_j are $R_{ij} = q_i^T a_j, \qquad i \leq j$ and for $i > j$, $R_{ij} = 0$.
- Therefore $\boxed{R = \begin{pmatrix} * & * & * & \cdots \\ 0 & * & * & \cdots \\ 0 & 0 & * & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}}$ is upper triangular.
- Hence $\boxed{A = QR, \qquad Q^T Q = I, \qquad R \text{ upper triangular.}}$

QR Decomposition: Example

Consider $A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$.

Its columns are $a_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$, $a_2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$.

First,

$$q_1 = \frac{a_1}{\|a_1\|} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

The projection of a_2 onto q_1 is

$$(q_1^T a_2) q_1 = \frac{1}{2} \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Therefore

$$u_2 = a_2 - (q_1^T a_2) q_1 = \frac{1}{2} \begin{pmatrix} 1 \\ -1 \end{pmatrix}.$$

QR Decomposition: Example Continued

Normalize u_2: $q_2 = \frac{u_2}{\|u_2\|} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$.

Therefore

$$\boxed{Q = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.}$$

Now compute

$$R_{ij} = q_i^T a_j.$$

Hence

$$\boxed{R = \begin{pmatrix} \sqrt{2} & 1/\sqrt{2} \\ 0 & 1/\sqrt{2} \end{pmatrix}.}$$

QR Decomposition: Pseudocode

Input: $A = [a_1, \ldots, a_n] \in \mathbb{R}^{m \times n}$

Output: $Q = [q_1, \ldots, q_n]$ and R

1. For $j = 1, \ldots, n$:
2. Start with the current column: $u_j \leftarrow a_j$
3. Remove its components in the previous directions:
   For $i = 1, \ldots, j-1$:

$$R_{ij} \leftarrow q_i^T a_j, \qquad u_j \leftarrow u_j - R_{ij} q_i$$

4. Normalize the remaining component:

$$R_{jj} \leftarrow \|u_j\|, \qquad q_j \leftarrow \frac{u_j}{R_{jj}}$$

From QR Decomposition to Eigenvalues

- We now have a way to factor any matrix A as $\boxed{A = QR}$, where Q is orthogonal and R is upper triangular.
- $\boxed{\text{Can repeated QR factorizations help us find eigenvalues?}}$
- Recall the power iteration:

$$Ax_k \longrightarrow \lambda_1 x_k.$$

  It repeatedly applies A and extracts the dominant eigendirection.

- To find *all* eigenvalues, we need to apply the same idea to several directions simultaneously.

From One Vector to a Matrix

- Power iteration starts with one vector: x_0.
- Instead, start with an orthonormal basis:

$$X_0 = \begin{pmatrix} | & & | \\ x_1 & \cdots & x_n \\ | & & | \end{pmatrix}, \qquad X_0^T X_0 = I.$$

- Apply A to every column: $Y_1 = AX_0$. After repeated multiplication,

$$Y_k = A^k X_0.$$

- This is essentially power iteration applied to **all directions simultaneously**.
- But there is a problem: $\boxed{X_0^T X_0 = I \quad \not\Rightarrow \quad (AX_0)^T (AX_0) = I.}$
- The columns lose orthogonality.

Orthogonal Iteration

- After applying A, restore orthonormality using QR decomposition.
- Starting with X_k, $AX_k = Q_k R_k$. Set $\boxed{X_{k+1} = Q_k}$.
- Thus one iteration is $\boxed{X_k \xrightarrow{A} AX_k \xrightarrow{\text{QR}} X_{k+1}.}$
- This gives the **orthogonal iteration**.
- It is the matrix analogue of power iteration:

$$\boxed{\text{Power iteration} = \text{multiply + normalize}}$$

$$\boxed{\text{Orthogonal iteration} = \text{multiply + orthogonalize.}}$$