CS395T Numerical Optimization for Graphics and AI — Linear Algebra

Qixing Huang
The University of Texas at Austin
huangqx@cs.utexas.edu

1 Basic

Linear algebra is widely used in AI and Graphics research. If you have not seriously learned this before, please take a class. At UT, you can take CS 383C NUMERICAL ANLY: LINEAR ALGEBRA. If you have learned it before but forgot the technical details (most likely), I recommend you to go through the following wikipedia page for review:

- Introduction to linear algebra [1].

1.1 Notations

We will use the following convention throughout all the lectures:

- Capital letters denote matrices, e.g., $A, B, C, \cdots$.
- Lowercase bold face letters denote vectors, e.g., $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \cdots$.
- Lowercase letters denote scalars, e.g., $s, t, \cdots$.
- $\boldsymbol{e}_i = (0, \cdots, 1, \cdots, 0)^T$ is the reserved for the canonical basis of $\mathbb{R}^n$.

2 Spectral of Normalized Adjacency Matrix

Eigenvalues and eigenvectors of a square matrix are very fundamental concepts in matrix theory. We are particularly interested in symmetric matrices. Specifically, given a symmetric matrix $A \in \mathbb{R}^{n\times n}$, it has n eigenvalues $\lambda_1(A) \geq \cdots \geq \lambda_n(A)$ and n eigenvectors $\boldsymbol{u}_1(A), \cdots, \boldsymbol{u}_n(A)$. The eigenvalues and eigenvectors are related by the following equality

$$A\boldsymbol{u}_i = \lambda_i \boldsymbol{u}_i, \quad 1 \leq i \leq n.$$

Equivalently, we can write out the eigen-decomposition of A as

$$A = U\Lambda U^T, \tag{1}$$

where

$$U = (\boldsymbol{u}_1, \cdots, \boldsymbol{u}_n), \quad \Lambda = \mathrm{diag}(\lambda_1, \cdots, \lambda_n).$$

In this lecture, we use some basic facts of spectral graph theory to study properties of eigenvalues and eigenvectors of square matrices. Spectral techniques are widely used in Graphics and AI, we will have three lectures on this topic later this semester.

[1] https://en.wikipedia.org/wiki/Linear_algebra

Consider a connected graph of n vertices $\mathcal{G} = (\{1, \cdots, n\}, \mathcal{E})$. With d_i we denote the vertex degree of i-th vertex. Consider so-called normalized adjacency matrix $\overline{A} \in \mathbb{R}^{n \times n}$, whose elements are given by

$$\overline{A}_{ij} = \begin{cases} \frac{1}{\sqrt{d_i d_j}} & (i,j) \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases}$$

We will prove the following facts:

Fact 2.1. *1 is an eigenvalue of $\overline{A}$, and the corresponding eigenvector is $\boldsymbol{u}_1 = (\sqrt{\frac{d_1}{\sum_i d_i}}, \cdots, \sqrt{\frac{d_n}{\sum_i d_i}})$.*

Proof. The proof reviews matrix-vector multiplication. In fact, let $\mathcal{N}(i) \subset \{1, \cdots, n\}$ collects indices of the neighboring vertices of vertex i, then

$$\begin{aligned} \boldsymbol{e}_i^T A \boldsymbol{u}_1 &= \sum_{j \in \mathcal{N}(i)} \frac{1}{\sqrt{d_i d_j}} \cdot \sqrt{\frac{d_j}{\sum_k d_k}} \\ &= \frac{1}{\sqrt{d_i}} \sum_{j \in \mathcal{N}(i)} \sqrt{\frac{1}{\sum_k d_k}} \\ &= \sqrt{\frac{d_i}{\sum_k d_k}}. \end{aligned}$$

$\square$

Fact 2.2. *The eigenvalues of $\overline{A}$ are between -1 and 1.*

The proof of the following fact will use a different definition of eigenvalues for symmetric matrices:

$$\lambda_1(A) = \max_{\boldsymbol{x} \in \mathbb{R}^n} \frac{\boldsymbol{x}^T A \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}}, \tag{2}$$

$$\lambda_n(A) = \min_{\boldsymbol{x} \in \mathbb{R}^n} \frac{\boldsymbol{x}^T A \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}}. \tag{3}$$

The proof is easy — using (1), we have

$$\begin{aligned} \max_{\boldsymbol{x} \in \mathbb{R}^n} \frac{\boldsymbol{x}^T A \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}} &= \max_{\boldsymbol{x} \in \mathbb{R}^n} \frac{(U^T \boldsymbol{x})^T \Lambda (U^T \boldsymbol{x})}{(U^T \boldsymbol{x})^T (U^T \boldsymbol{x})} \\ &= \max_{\boldsymbol{y} \in \mathbb{R}^n} \frac{\boldsymbol{y}^T \Lambda \boldsymbol{y}}{\boldsymbol{y}^T \boldsymbol{y}} \\ &= \max_{\boldsymbol{y} \in \mathbb{R}^n} \frac{\sum_{i=1}^n \lambda_i y_i^2}{\sum_{i=1}^n y_i^2} \\ &= \lambda_1. \end{aligned} \tag{4}$$

We can generalize (2) and (3) to other eigenvalues. For example,

$$\lambda_i(A) = \max_{\boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{u}_1^T \boldsymbol{x} = 0, \cdots, \boldsymbol{u}_{i-1}^T \boldsymbol{x} = 0} \frac{\boldsymbol{x}^T A \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}},$$

$$\lambda_i(A) = \min_{\boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{u}_n^T \boldsymbol{x} = 0, \cdots, \boldsymbol{u}_{i+1}^T \boldsymbol{x} = 0} \frac{\boldsymbol{x}^T A \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}}. \tag{5}$$

Alternatively, we have

$$\lambda_i(A) = \min_{U,dim(U)=n-i} \max_{\boldsymbol{x}\in U} \frac{\boldsymbol{x}^T A\boldsymbol{x}}{\boldsymbol{x}^T\boldsymbol{x}},$$
$$\lambda_i(A) = \max_{U,dim(U)=i} \min_{\boldsymbol{x}\in U} \frac{\boldsymbol{x}^T A\boldsymbol{x}}{\boldsymbol{x}^T\boldsymbol{x}}. \tag{6}$$

The proofs are more details can be found at Rayleigh quotient (`https://en.wikipedia.org/wiki/Rayleigh_quotient`) and min-max theorem (`https://en.wikipedia.org/wiki/Min-max_theorem`). Now we give the proof of Fact 2.2.

Proof of Fact 2.2. First of all,

$$\begin{aligned}
\boldsymbol{x}^T\overline{A}\boldsymbol{x} &= \sum_{(i,j)\in\mathcal{E}} \frac{x_i x_j}{\sqrt{d_i d_j}} \\
&= \frac{1}{2}\sum_{(i,j)\in\mathcal{E}} \Big((\frac{x_i}{\sqrt{d_i}} + \frac{x_j}{\sqrt{d_j}})^2 - \frac{x_i^2}{d_i} - \frac{x_j^2}{d_j} \Big) \\
&\geq -\frac{1}{2}\sum_{(i,j)\in\mathcal{E}} \Big(\frac{x_i^2}{d_i} + \frac{x_j^2}{d_j} \Big) \\
&= -\frac{1}{2}(\sum_{i=1}^n \sum_{j\in\mathcal{N}(i)} \frac{x_i^2}{d_i} + \sum_{j=1}^n \sum_{i\in\mathcal{N}(j)} \frac{x_j^2}{d_j}) \\
&= -\frac{1}{2}(2\sum_{i=1}^n x_i^2) \\
&= -\sum_{i=1}^n x_i^2.
\end{aligned} \tag{7}$$

In other words,

$$\lambda_n(\overline{A}) \geq -1.$$

In the other direction,

$$\begin{aligned}
\boldsymbol{x}^T\overline{A}\boldsymbol{x} &= \sum_{(i,j)\in\mathcal{E}} \frac{x_i x_j}{\sqrt{d_i d_j}} \\
&= \frac{1}{2}\sum_{(i,j)\in\mathcal{E}} \Big(-(\frac{x_i}{\sqrt{d_i}} - \frac{x_j}{\sqrt{d_j}})^2 + \frac{x_i^2}{d_i} + \frac{x_j^2}{d_j} \Big) \\
&\leq \frac{1}{2}\sum_{(i,j)\in\mathcal{E}} \Big(\frac{x_i^2}{d_i} + \frac{x_j^2}{d_j} \Big) \\
&= \sum_{i=1}^n x_i^2,
\end{aligned} \tag{8}$$

which means

$$\lambda_1(\overline{A}) \leq 1.$$

□

3 Rotation Matrices

Rodrigues' rotation formula, named after Olinde Rodrigues, is an efficient algorithm for rotating a Euclidean vector, given a rotation axis and an angle of rotation. In other words, Rodrigues' formula provides an algorithm to compute the exponential map from $\mathfrak{so}(3)$ to $SO(3)$ without computing the full matrix exponential.

If $\boldsymbol{v}$ is a vector in R^3 and $\boldsymbol{e}$ is a unit vector rooted at the origin describing an axis of rotation about which $\boldsymbol{v}$ is rotated by an angle θ, Rodrigues' rotation formula to obtain the rotated vector is

$$\mathbf{v}_{\mathrm{rot}} = (\cos\theta)\mathbf{v} + (\sin\theta)(\mathbf{e}\times\mathbf{v}) + (1-\cos\theta)(\mathbf{e}\cdot\mathbf{v})\mathbf{e}\,.$$

For the rotation of a single vector it may be more efficient than converting $\boldsymbol{e}$ and θ into a rotation matrix to rotate the vector.

Exponential map. The exponential map effects a transformation from the axis-angle representation of rotations to rotation matrices,

$$\exp\colon \mathfrak{so}(3) \to \mathrm{SO}(3)\,.$$

Essentially, by using a Taylor expansion one derives a closed-form relation between these two representations. Given a unit vector $\omega \in \mathfrak{so}(3) = \mathbb{R}^3$ representing the unit rotation axis, and an angle, $\theta \in \mathbb{R}$, an equivalent rotation matrix R is given as follows, where K is the cross product matrix of ω, that is, $K\boldsymbol{v} = \omega\times\boldsymbol{v}$ for all vectors $\boldsymbol{v} \in \mathbb{R}^3$,

$$R = \exp(\theta\mathbf{K}) = \sum_{k=0}^{\infty}\frac{(\theta\mathbf{K})^k}{k!} = I + \theta\mathbf{K} + \frac{1}{2!}(\theta\mathbf{K})^2 + \frac{1}{3!}(\theta\mathbf{K})^3 + \cdots$$

Because K is skew-symmetric, and the sum of the squares of its above-diagonal entries is 1, the characteristic polynomial $P(t)$ of K is $P(t) = det(K - tI) = -(t^3+t)$. Since, by the Cayley-Hamilton theorem, $P(K) = 0$, this implies that $K^3 = -K$.

As a result, $K^4 = -K^2, K^5 = K, K^6 = K^2, K^7 = -K$. This cyclic pattern continues indefinitely, and so all higher powers of K can be expressed in terms of K and K^2. Thus, from the above equation, it follows that

$$R = I + \left(\theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \cdots\right)\mathbf{K} + \left(\frac{\theta^2}{2!} - \frac{\theta^4}{4!} + \frac{\theta^6}{6!} - \cdots\right)\mathbf{K}^2\,,$$

that is,

$$R = I + (\sin\theta)\mathbf{K} + (1-\cos\theta)\mathbf{K}^2\,.$$

4 Quaternion

Besides the standard matrix representation of matrices, Quaternion is another widely used representation of matrices. Here we provide a concise introduction, and please refer to

- `https://en.wikipedia.org/wiki/Quaternion`

for more details.

One can think that quaternions generalize complex numbers. A quaternion is generally represented in the form:

$$a + b\boldsymbol{i} + c\boldsymbol{j} + d\boldsymbol{k}$$

where a, b, c, and d are real numbers, and $\boldsymbol{i}$, $\boldsymbol{j}$, and $\boldsymbol{k}$ are the fundamental quaternion units.

Quaternions give a simple way to encode this axis-angle representation in four numbers, and can be used to apply the corresponding rotation to a position vector, representing a point relative to the origin in $\mathbb{R}^3$.

A Euclidean vector such as $(2,3,4)$ or (a_x, a_y, a_z) can be rewritten as $2\boldsymbol{i}+3\boldsymbol{j}+4\boldsymbol{k}$ or $a_x\boldsymbol{i}+a_y\boldsymbol{j}+a_z\boldsymbol{k}$, where $\boldsymbol{i},\boldsymbol{j},\boldsymbol{k}$ are unit vectors representing the three Cartesian axes. A rotation through an angle of θ around the axis defined by a unit vector

$$\mathbf{u} = (u_x, u_y, u_z) = u_x\boldsymbol{i} + u_y\boldsymbol{j} + u_z\boldsymbol{k}.$$

can be represented by a quaternion. This can be done using an extension of Euler's formula:

$$\boldsymbol{q} = e^{\frac{\theta}{2}(u_x\boldsymbol{i}+u_y\boldsymbol{j}+u_z\boldsymbol{k})} = \cos(\frac{\theta}{2}) + (u_x\boldsymbol{i} + u_y\boldsymbol{j} + u_z\boldsymbol{k})\sin(\frac{\theta}{2}).$$

It can be shown that the desired rotation can be applied to an ordinary vector $\boldsymbol{p} = (p_x, p_y, p_z) = p_x\boldsymbol{i} + p_y\boldsymbol{j} + p_z\boldsymbol{k}$ in 3-dimensional space, considered as a quaternion with a real coordinate equal to zero, by evaluating the conjugation of $\boldsymbol{p}$ by $\boldsymbol{q}$:

$$\boldsymbol{p}' = \boldsymbol{q}\boldsymbol{p}\boldsymbol{q}^{-1}$$

using the Hamilton product, where $\boldsymbol{p}' = (p'_x, p'_y, p'_z)$ is the new position vector of the point after the rotation. In this instance,

$$\boldsymbol{q}^{-1} = e^{-\frac{\theta}{2}(u_x\mathbf{i}+u_y\mathbf{j}+u_z\mathbf{k})} = \cos\frac{\theta}{2} - (u_x\mathbf{i} + u_y\mathbf{j} + u_z\mathbf{k})\sin\frac{\theta}{2}.$$

We will leave the proof as a homework.

It follows that conjugation by the product of two quaternions is the composition of conjugations by these quaternions: If $\boldsymbol{p}$ and $\boldsymbol{q}$ are unit quaternions, then rotation (conjugation) by $\boldsymbol{pq}$ is

$$\boldsymbol{pq}\mathbf{v}(\boldsymbol{pq})^{-1} = \mathbf{pqvq}^{-1}\mathbf{p}^{-1} = \mathbf{p}(\mathbf{qvq}^{-1})\mathbf{p}^{-1},$$

which is the same as rotating (conjugating) by $\boldsymbol{q}$ and then by $\boldsymbol{p}$. The scalar component of the result is necessarily zero.

Quaternion-derived rotation matrix. A quaternion rotation $\mathbf{p}' = \mathbf{qpq}^{-1}$ (with $\mathbf{q} = q_r + q_i\mathbf{i} + q_j\mathbf{j} + q_k\mathbf{k}$) can be algebraically manipulated into a matrix rotation $\mathbf{p}' = \mathbf{Rp}$, where R is the rotation matrix given by

$$\mathbf{R} = \begin{bmatrix} 1 - 2s(q_j^2 + q_k^2) & 2s(q_iq_j - q_kq_r) & 2s(q_iq_k + q_jq_r) \\ 2s(q_iq_j + q_kq_r) & 1 - 2s(q_i^2 + q_k^2) & 2s(q_jq_k - q_iq_r) \\ 2s(q_iq_k - q_jq_r) & 2s(q_jq_k + q_iq_r) & 1 - 2s(q_i^2 + q_j^2) \end{bmatrix}$$

Here $s = \|q\|^{-2}$ and if $\boldsymbol{q}$ is a unit quaternion, $s = 1$.

Recovering the axis-angle representation. The expression $\mathbf{qpq}^{-1}$ rotates any vector quaternion $\mathbf{p}$ around an axis given by the vector $\mathbf{a}$ by the angle θ, where $\mathbf{a}$ and θ depends on the quaternion

$$(a_x, a_y, a_z) \;=\; \frac{(q_i, q_j, q_k)}{\sqrt{q_i^2 + q_j^2 + q_k^2}}$$

$$\theta \;=\; 2\,\mathrm{atan2}\left(\sqrt{q_i^2 + q_j^2 + q_k^2}, q_r\right),$$

where atan2 is the two-argument arctangent. Care should be taken when the quaternion approaches a scalar, since due to degeneracy the axis of an identity rotation is not well-defined.

References