Eigenproblems II: Computation

CS 205A:
Mathematical Methods for Robotics, Vision, and Graphics

Justin Solomon

Setup

$$A \in \mathbb{R}^{n \times n} \text{ symmetric}$$

$$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^n \text{ eigenvectors}$$

$$|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n| \text{ eigenvalues}$$

Setup

$$A \in \mathbb{R}^{n\times n} \text{ symmetric}$$
$$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^n \text{ eigenvectors}$$
$$|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n| \text{ eigenvalues}$$

Review (Spectral Theorem):
What do we know about the eigenvectors?

Usual Trick

$$\vec{v} \in \mathbb{R}^n$$

$$\Downarrow$$

$$\vec{v} = c_1\vec{x}_1 + \cdots + c_n\vec{x}_n$$

Observation

$$A^k\vec{v} =$$
$$\lambda_1^k\left(c_1\vec{x}_1 + \left(\frac{\lambda_2}{\lambda_1}\right)^k c_1\vec{x}_2 + \cdots + \left(\frac{\lambda_n}{\lambda_1}\right)^k c_n\vec{x}_n\right)$$

For Large k

$$A^k\vec{v} \approx \lambda_1^k \vec{x}_1$$

(assuming $|\lambda_2| < |\lambda_1|$)

Power Iteration

$$\vec{v}_k = A\vec{v}_{k-1}$$

Power Iteration

$$\vec{v}_k = A\vec{v}_{k-1}$$

Question:

What if $|\lambda_1| > 1$?

Normalized Power Iteration

$$\vec{w}_k = A\vec{v}_{k-1}$$

$$\vec{v}_k = \frac{\vec{w}_k}{\|\vec{w}_k\|}$$

Normalized Power Iteration

$$\vec{w}_k = A\vec{v}_{k-1}$$

$$\vec{v}_k = \frac{\vec{w}_k}{\|\vec{w}_k\|}$$

Question: Which norm?

Eigenvalues of Inverse Matrix

$$A\vec{v} = \lambda\vec{v} \implies A^{-1}\vec{v} = \frac{1}{\lambda}\vec{v}$$

Eigenvalues of Inverse Matrix

$$A\vec{v} = \lambda\vec{v} \implies A^{-1}\vec{v} = \frac{1}{\lambda}\vec{v}$$

Question:
What is the largest-magnitude eigenvalue?

Inverse Iteration

$$\vec{w}_k = A^{-1}\vec{v}_{k-1}$$

$$\vec{v}_k = \frac{\vec{w}_k}{\|\vec{w}_k\|}$$

Inverse Iteration

$$\vec{w}_k = A^{-1}\vec{v}_{k-1}$$

$$\vec{v}_k = \frac{\vec{w}_k}{\|\vec{w}_k\|}$$

Question: How to make faster?

Inverse Iteration with LU

$$\text{Solve } L\vec{y}_k = \vec{v}_{k-1}$$
$$\text{Solve } U\vec{w}_k = \vec{y}_k$$
$$\text{Normalize } \vec{v}_k = \frac{\vec{w}_k}{\|\vec{w}_k\|}$$

Eigenvalues of Inverse Matrix

$$A\vec{v} = \lambda\vec{v} \implies (A - \sigma I)\vec{v} = (\lambda - \sigma)\vec{v}$$

Shifted Inverse Iteration

To find eigenvalue closest to σ:

$$\vec{v}_{k+1} = \frac{(A - \sigma I)^{-1}\vec{v}_k}{\|(A - \sigma I)^{-1}\vec{v}_k\|}$$

Heuristic: Convergence Rate

$$A^k \vec{v} =$$

$$\lambda_1^k \left(c_1 \vec{x}_1 + \left(\frac{\lambda_2}{\lambda_1}\right)^k c_1 \vec{x}_2 + \cdots + \left(\frac{\lambda_n}{\lambda_1}\right)^k c_n \vec{x}_n \right)$$

Strategy for Better Convergence

Find σ with

$$\left|\frac{\lambda_2 - \sigma}{\lambda_1 - \sigma}\right| < \left|\frac{\lambda_2}{\lambda_1}\right|$$

Least-Squares Approximation

If $\vec{v}_0$ is *approximately* an eigenvector:

$$\arg\min_{\lambda} \|A\vec{v}_0 - \lambda\vec{v}_0\|^2 = \frac{\vec{v}_0^\top A\vec{v}_0}{\|\vec{v}_0\|_2^2}$$

Rayleigh Quotient Iteration

$$\sigma_k = \frac{\vec{v}_{k-1}^\top A \vec{v}_{k-1}}{\|\vec{v}_{k-1}\|_2^2}$$

$$\vec{w}_k = (A - \sigma_k I)^{-1} \vec{v}_{k-1}$$

$$\vec{v}_k = \frac{\vec{w}_k}{\|\vec{w}_k\|}$$

Rayleigh Quotient Iteration

$$
\begin{aligned}
\sigma_k &= \frac{\vec{v}_{k-1}^\top A \vec{v}_{k-1}}{\|\vec{v}_{k-1}\|_2^2} \\
\vec{w}_k &= (A - \sigma_k I)^{-1} \vec{v}_{k-1} \\
\vec{v}_k &= \frac{\vec{w}_k}{\|\vec{w}_k\|}
\end{aligned}
$$

Efficiency per iteration vs. number of iterations?

Unlikely Failure Mode for Iteration

What is $\vec{v}_0$?

Unlikely Failure Mode for Iteration

What is $\vec{v}_0$?

What happens when $\vec{v}_0 \cdot \vec{x}_1 = 0$?

Bug or Feature?

1. Compute $\vec{x}_0$ via power iteration.
2. Project $\vec{x}_0$ out of $\vec{v}_0$.
3. Compute $\vec{x}_1$ via power iteration.
4. Project $\operatorname{span}\{\vec{x}_0, \vec{x}_1\}$ out of $\vec{v}_0$.
5. $\ldots$

Bug or Feature?

1. Compute $\vec{x}_0$ via power iteration.
2. Project $\vec{x}_0$ out of $\vec{v}_0$.
3. Compute $\vec{x}_1$ via power iteration.
4. Project $\operatorname{span}\{\vec{x}_0, \vec{x}_1\}$ out of $\vec{v}_0$.
5. $\ldots$

Assumption: A is symmetric.

Avoiding Numerical Drift

Do power iteration on $P^\top AP$ where P projects out known eigenvectors.

Deflation

Modify A so that power iteration reveals an eigenvector you have not yet computed.

Similarity Transformations

Similar matrices

Two matrices A and B are *similar* if there exists T with $B = T^{-1}AT$.

Similarity Transformations

Similar matrices

Two matrices A and B are *similar* if there exists T with $B = T^{-1}AT$.

Proposition

Similar matrices have the same eigenvalues.

Householder Asymmetric Deflation

$$
\begin{aligned}
H\vec{x}_1 &= \vec{e}_1 \\
\implies HAH^\top \vec{e}_1 &= HAH\vec{e}_1 \text{ by symmetry} \\
&= HA\vec{x}_1 \text{ since } H^2 = I \\
&= \lambda_1 H\vec{x}_1 \\
&= \lambda_1 \vec{e}_1
\end{aligned}
$$

Householder Asymmetric Deflation

$$HAH^\top = \begin{pmatrix} \lambda_1 & \vec{b}^\top \\ \vec{0} & B \end{pmatrix}$$

Similarity transform of $A \implies$ same eigenvalues.

Householder Asymmetric Deflation

$$HAH^\top = \begin{pmatrix} \lambda_1 & \vec{b}^\top \\ \vec{0} & B \end{pmatrix}$$

Similarity transform of $A \implies$ same eigenvalues.

Do power iteration on B.

<warning>
Justin's favorite algorithm. Ever.
</warning>

Conjugation without Inversion

$$Q^{-1} = Q^\top$$

$$\implies Q^{-1}AQ = Q^\top AQ$$

Conjugation without Inversion

$$
\begin{aligned}
Q^{-1} &= Q^\top \\
\implies Q^{-1}AQ &= Q^\top AQ
\end{aligned}
$$

But which Q?

Should involve matrix structure but be easy to compute.

Experiment

$$
\begin{aligned}
A &= QR \\
Q^{-1}AQ &= \quad ?
\end{aligned}
$$

QR Iteration

$$
\begin{aligned}
A_1 &= A \\
\text{Factor } A_k &= Q_k R_k \\
\text{Multiply } A_{k+1} &= R_k Q_k
\end{aligned}
$$

If QR Iteration Converges

$$A_\infty = Q_\infty R_\infty = R_\infty Q_\infty$$

- Eigenvalues of R_∞ along the diagonal and are eigenvalues of A_∞ $(R_\infty Q_\infty)$
- Eigenvalues of A_∞ are those of A

Next