March 31, 2020

Numerical Analysis

Last class: Power method for computing eigenvalues of matrices.

$\Rightarrow$ only allows for the computation of the eigenvalue with the largest absolute value.

Start with random $\vec{w}_0$. with $\|\vec{w}_0\| = 1$.

compute powers of A applied to $\vec{w}_0$

Since $\vec{w}_0 = c_1 \vec{v}_1 + \dots + c_n \vec{v}_n$

$A^k \vec{w}_0 = c_1 \lambda_1^k \vec{v}_1 + \dots + c_n \lambda_n^k \vec{v}_n$

If $|\lambda_1| > |\lambda_j|$ for all $j \neq 1$, then

$$A^k \vec{w}_0 \approx c_1 \lambda_1^k \vec{v}_1$$

Normalize: $\vec{w}_k = \dfrac{A^k \vec{w}_0}{\|A^k \vec{w}_0\|} \approx \vec{v}_1$

to compute λ_1, take $\underbrace{(A\vec{w}_k)_i / w_{ki}}_{\text{ratio of components}} \approx \lambda_1$

Since $A\vec{w}_k \approx \lambda_1 \vec{w}_k$ for k large enough.

Convergence relies on $|\lambda_1|$ being larger than all other $|\lambda_i|$.

Ex: $\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \Rightarrow \vec{v}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \lambda_1 = 1$ $\quad |\lambda_1| = |\lambda_2|$

$\vec{v}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \lambda_2 = -1$

Let $\vec{w}_0 = \begin{pmatrix} a \\ b \end{pmatrix}$, then

$$\left.\begin{aligned} A\vec{w}_0 &= \begin{pmatrix} a \\ -b \end{pmatrix} \\ A^2\vec{w}_0 &= \begin{pmatrix} a \\ b \end{pmatrix} \\ &\vdots \end{aligned}\right\}$$

doesn't converge to anything

Power method fails for this example.

If $\vec{w}_k = \dfrac{A^k \vec{w}_0}{\| A^k \vec{w}_0 \|}$, then

$$\| \vec{w}_k - \vec{v}_1 \| \approx O\left(\left| \frac{\lambda_2}{\lambda_1} \right|^k \right)$$

How do we accelerate this convergence?

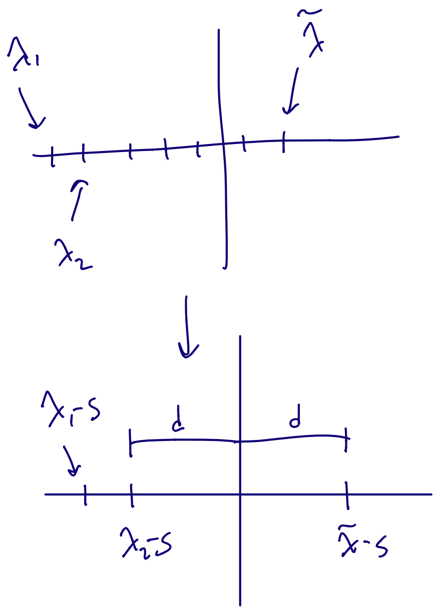
<u>Idea one</u> Power method with shift

If a matrix A has eigenvalues $\lambda_1 \dots \lambda_n$, then $A - sI$ has eigenvalues $\lambda_i - s$.

<u>Pf:</u>

$$\begin{aligned} (A - sI)\vec{v} &= A\vec{v} - s\vec{v} \\ &= \lambda\vec{v} - s\vec{v} \\ &= (\lambda - s)\vec{v} \end{aligned}$$

<u>Choose s</u> to increase the convergence rate:



to minimize the ratio $\left| \dfrac{\lambda_2}{\lambda_1} \right|$,

choose shift such that $|\lambda_2| = |\tilde{\lambda}|$.

Ie. choose s such that

$$\left| \frac{\lambda_2 - s}{\lambda_1 - s} \right| = \left| \frac{\tilde{\lambda} - s}{\lambda_1 - s} \right|.$$

Power method with shift allows for computing the most negative <u>or</u> the most positive eigenvalue.

How do we compute eigenvalues in the middle?

I.e., if $\lambda_1 > \lambda_2 > \lambda_3 > \dots > \lambda_{n-1} > \lambda_n$

then Power Method w/ Shift can compute either λ_1 or λ_n.

To compute $\lambda_2 \dots \lambda_{n-1}$, we need a different idea.

<u>Idea Two</u> Apply the power method to find the eigenvalues of $(A-sI)^{-1}$. This is called the <u>Inverse Power Method with Shift.</u>

If A has eigenvalue λ, then A^{-1} has eigenvalue $\frac{1}{\lambda}$.

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

Furthermore: $(A-sI)^{-1}$ has eigenvalue $\dfrac{1}{\lambda - s}$.

If we choose s properly to make $\frac{1}{\lambda-s}$ large, then the Inverse Power Method with shift can converge <u>very</u> rapidly.

Choosing s close to λ_ℓ causes $\dfrac{1}{\lambda_\ell - s}$ to become <u>very</u> large in absolute value, while $\dfrac{1}{\lambda_j - s}$ for $j \neq \ell$ remains bounded.

This scheme is of course much more expensive since "applying" A^{-1} requires solving a linear system ($O(n^3)$ vs. $O(n^2)$ flops).

<u>The algorithm</u>

1. Set $\vec{w}_0$ to be random.
2. Solve $(A-sI)\vec{y}_1 = \vec{w}_0$.
   $\iff \vec{y}_1 = (A-sI)^{-1}\vec{w}_0$
3. Set $\vec{w}_1 = \vec{y}_1 / \|\vec{y}_1\|$
4. Proceed as in the Power Method.

The hard part is knowing what to choose for s. You need estimates for the eigenvalues.

Both schemes only compute <u>one</u> eigenvalue/vector at a time.

Jacobi's Method

Can we compute all eigenvalues and vectors at the same time.

If A were diagonal, then we immediately know the eigenvalues. Can we make A diagonal?

Recall Similarity transform: $B = M^{-1}AM$ then B is similar to A, i.e. they have the same eigenvalues.

Proof: Look at their characteristic polynomials:

$$p_A(\lambda) = \det(A - \lambda I) \quad \text{degree } n \text{ polynomial}$$

$$\begin{aligned} p_B(\lambda) &= \det(B - \lambda I) \\ &= \det(M^{-1}AM - \lambda I) \\ &= \det(M^{-1}AM - \lambda M^{-1}M) \\ &= \det(M^{-1}(A - \lambda I)M) \\ &= \det(M^{-1})\det(A - \lambda I)\det(M) \\ &= p_A(\lambda). \end{aligned}$$

If M were chosen to be the matrix of eigenvector of A, then $A = M\underset{\uparrow \text{ eigenvalues}}{D}M^{-1} \Rightarrow \underbrace{D = M^{-1}AM}_{\text{Diagonalization of } A.}$

Ex: Let A be a real symmetric 2×2 matrix.

$A = \begin{pmatrix} a & b \\ b & d \end{pmatrix} \Rightarrow$ Eigenvalues are real, and it is diagonalized by an orthogonal matrix V.

All 2×2 orthogonal matrices can be parameterized as:

$$V = \begin{pmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{pmatrix} \qquad (2\times 2 \text{ rotation matrix}).$$

We want

$$\underbrace{V^T}_{=V^{-1}} A V = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$$

Write out components for 12, 21 entries of V^TAV:

$$\begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix} \underbrace{\begin{pmatrix} a & b \\ b & d \end{pmatrix} \begin{pmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{pmatrix}} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$$

$$\begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix} \begin{pmatrix} a\cos\varphi - b\sin\varphi & a\sin\varphi + b\cos\varphi \\ b\cos\varphi - d\sin\varphi & b\sin\varphi + d\cos\varphi \end{pmatrix} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$$

$$\Rightarrow \quad a\cos\varphi\sin\varphi + b\cos^2\varphi - b\sin^2\varphi - d\cos\varphi\sin\varphi = 0$$

$$a\cos\varphi\sin\varphi - b\sin^2\varphi + b\cos^2\varphi - d\cos\varphi\sin\varphi = 0$$

Next: Find φ.

<u>Add equations:</u> $(a-d)\cos\varphi\sin\varphi + b(\cos^2\varphi - \sin^2\varphi) = 0$

$$\Rightarrow (a-d)\frac{1}{2}\sin 2\varphi + b\cos 2\varphi = 0$$

$$\Rightarrow \tan 2\varphi = \frac{2b}{d-a} \qquad \Rightarrow \varphi = \frac{1}{2}\operatorname{atan}\left(\frac{2b}{d-a}\right).$$

If $d-a=0$, then $\frac{2b}{d-a} = \infty$,

$\Rightarrow$ in C or Fortran use

$$\varphi = \frac{1}{2}\operatorname{atan2}(d-a, 2b).$$

So we found φ such that $V^TAV = D = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$.

The Jacobi Method for an $n \times n$ matrix

Define $$R^{pq}(\varphi) = \begin{array}{cl} \begin{pmatrix} 1 &&&&&& \\ & 1 &&&&& \\ && 1 &&&& \\ &&& \cos\varphi && \sin\varphi & \\ &&&& \ddots\ 1 && \\ &&& -\sin\varphi && \cos\varphi & \\ &&&&&& \ddots\ 1 \end{pmatrix} & \begin{array}{l} \\ \\ \\ \text{row } p \\ \\ \text{row } q \\ \\ \end{array} \\ \qquad\qquad \text{column } p \qquad \text{column } q. & \end{array}$$

$R^{pq}(\varphi)$ can be used to set the pq and qp elements of a real symmetric matrix A to zero.

$R^{pq}(\varphi)^T A \, R^{pq}(\varphi)$ leaves all rows and columns unchanged except for row/column p and row/column q.

The algorithm:

① Set $A^{(0)} = A$

② Find pq element in $A^{(k)}$ with maximum absolute value.

③ Compute $$\varphi_k = \frac{1}{2} \operatorname{atan}\left(\frac{2 a_{pq}^{(k)}}{a_{qq}^{(k)} - a_{pp}^{(k)}} \right)$$

④ Set $$A^{(k+1)} = R^{pq}(\varphi_k)^T A^{(k)} R^{pq}(\varphi_k)$$

Continue this algorithm until all $|a_{pq}^{(k)}| < \epsilon$, $p \neq q$.

Then $$A^{(k)} \to \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix} \quad \text{as } k \to \infty$$

and furthermore: $$R^{(k)} = R(\varphi_1) R(\varphi_2) \cdots R(\varphi_k) \to \overbrace{\left(\vec{v}_1 \ldots \vec{v}_n \right)}^{\text{matrix of eigenvectors.}}$$