

QR iteration II and Power Method

Friday, November 22, 2024 10:57 AM

Class 38: November 22, 2024

Recall: Last time, we covered an introduction to a series of important results relevant to the problem of finding an orthonormal basis of eigenvectors $\{u_1, u_2, \dots, u_n\}$ and the corresponding eigenvalues $\{\lambda_1, \lambda_2, \dots, \lambda_n\}$. This is equivalent to computing the eigendecomposition $A = U D U^*$:

- We defined the concepts of eigenvalue, eigenvector and eigenpair: (λ, x) is an eigenpair if x is a non-zero solution to $(A - \lambda I)x = 0$. λ is an eigenvalue if $(A - \lambda I)$ is singular, which is equivalent to $\det(A - \lambda I) = 0$.
- We defined two important tools: the characteristic polynomial $p_A(\lambda) = \det(A - \lambda I)$ (eigenvalues are the roots of p_A) and the eigenspace E_{λ} and its dimension $\dim(E_{\lambda})$.
- We then introduced the concepts of **similarity** and **unitary similarity**. Eigenvalues (and eigenvectors modulo a change of basis) are preserved under similarity, which makes unitary similarity transformations a crucial tool to find our eigendecomposition.

We then discussed that an eigendecomposition doesn't always exist, but a Schur decomposition always does. That is: given A $n \times n$ with complex (or real) entries, you can always find U unitary and T triangular such that $T = U^* A U$ (and so, $A = U T U^*$). The eigenvalues (spectrum) of A is then the diagonal of T .

We used Schur to sketch a proof of the "spectral theorems":

- If A is $n \times n$ and **Hermitian (symmetric)**, then all its eigenvalues are real and T is diagonal, meaning $D = U^* A U$ for D diagonal!
- If A is $n \times n$ and **Normal**, then there are complex matrix U and complex diagonal matrix D (T is again diagonal) such that $D = U^* A U$.

This told us very important information: **An algorithm to approximately find a Schur decomposition (compute U and T) will leave us in the best possible shape: If A is eigendecomposable, then T will be diagonal and we will get our decomposition. If A is not eigendecomposable, we can "read off" the eigenvalues from $\text{diag}(T)$ and find whatever eigenvectors we can find.**

Finally, we introduced the QR iteration as an algorithm based on the QR decomposition that produces iterates A_k and Q_k such that $A_k \rightarrow T$ and $Q_k \rightarrow U$. Today, we will discuss the performance of this algorithm (it is painfully slow) and how to accelerate it so it is much faster.

QR iteration:

$$\text{Start } A_0 = A$$

while (algo has not converged)

$$[Q_k, R_k] = \text{qr}(A_k)$$

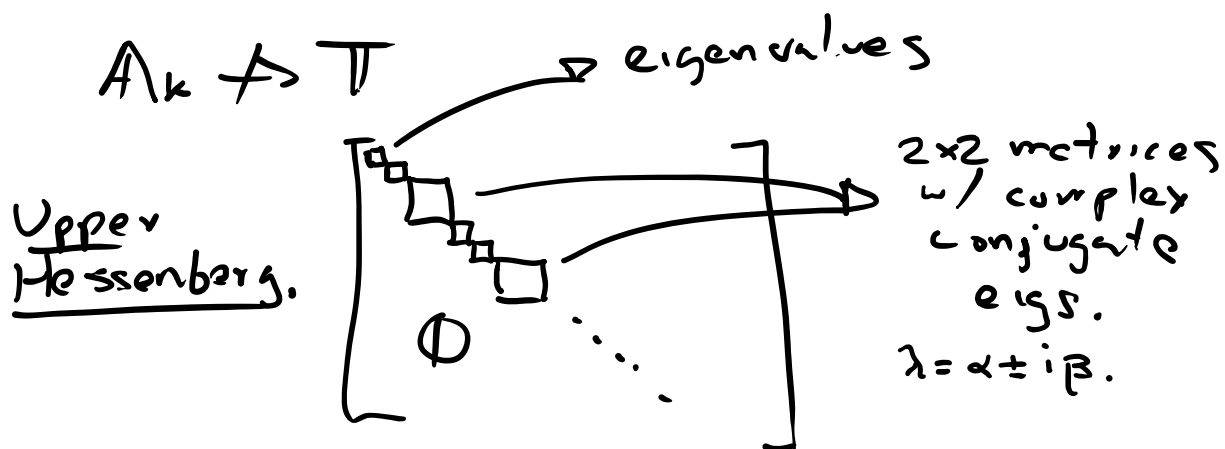
$$A_{k+1} = \underbrace{R_k Q_k}_{= Q_k^* A_k Q_k}$$

$$\dots \lim_{k \rightarrow \infty} A_k = \underline{\underline{T}}, \quad \lim_{k \rightarrow \infty} (Q_1 \dots Q_k) = \underline{\underline{U}}$$

- Convergence QR iter is linear
(ratios of $\lambda_i / \lambda_{i+1}$ Order by abs value).
- Seems to order them.

- Seems to order them.

~~Non~~ Non-symmetric real matrices



Convergence / Stopping criteria

QR iter: Cost per iter $O(n^3)$
Large # (linear, slow)

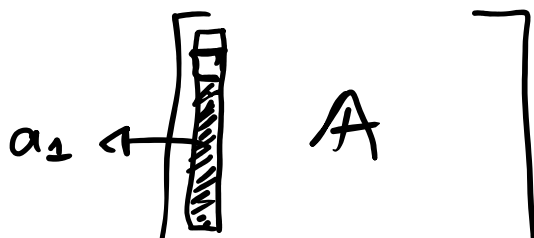
Accelerations:

\rightarrow Cost per iter $- O(n^3)$.

- Maybe $A_0 = A$ is not a good idea.

\hookrightarrow Pick $A_0 \sim A$ but $qr(A_0)$ is cheap?

Use householder reflectors, but \ll similarity transf.



Do H_1, \dots, H_k from Householder QR, it does not work. $\rightarrow A$ is not triangular (has no zeros).

\rightarrow Make A Upper-Hessenberg.

$$H_1 = \begin{bmatrix} 1 & & 0 \\ & H_{w_1} & \\ 0 & & \end{bmatrix} \quad H_{w_1} = I - 2w_1w_1^*$$

$$A_0 = (H_{n-2} \dots H_2 H_1) A (H_1 H_2 \dots H_{n-2})$$

$$A_0 = \begin{bmatrix} \text{Upper Triangular} \\ \text{⓪} \end{bmatrix} \rightarrow \text{Cost}_{qr} \underline{O(n^2)}$$

What if A symmetric? $A_0 = \begin{bmatrix} \text{Upper Triangular} & \text{⓪} \\ \text{⓪} & \end{bmatrix}$
 $\hookrightarrow \text{Cost}_{qr} \rightarrow \underline{O(n)}$

Modified algo:

$$A_0 = \text{UpperHess}(A) \quad O(n^3)$$

$$\text{QR_iter}(A_0) \rightarrow O(n^2) \text{ p/step}$$

$(A \text{ sym} - O(n) \text{ p/st})$

