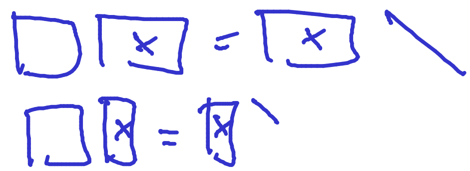
Review & Outline for today

$A = XDX^{-1}$ ← invertible X

↑ diagonal

similarity transformation

$AX = XD$



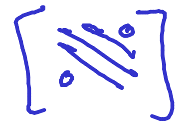
$B = UAU^{-1} \Rightarrow$ A and B have same eigvals

$B = XDX^{-1}$

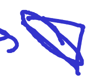
eigvecs of A will be $U^{-1}X$

$AU^{-1}X = U^{-1}XD$

$A = U^{-1}BU = \underline{U^{-1}XDX^{-1}U}$

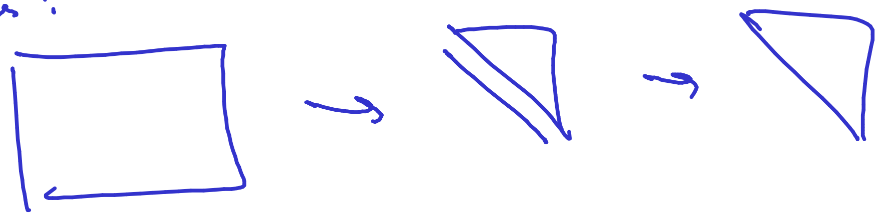
Similarity trans. pave way for algs  → tridiagonal

types of reductions

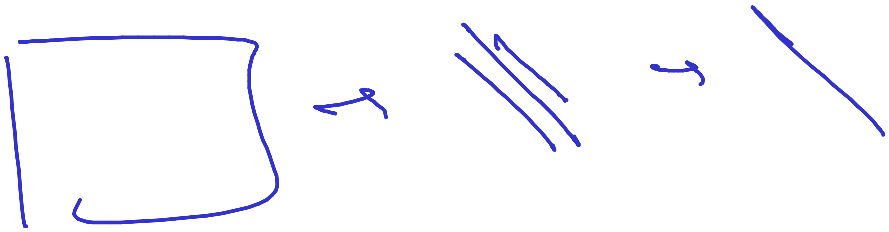
matrix structure	similarity transf	simpler matrix structure
diagonalizable	invertible (X)	diagonal
SPD symmetric pos. def. (real)	real and orthogonal (Q)	diagonal positive
symmetric real	....	diagonal
real matrix	orthogonal	upper-Hessenberg  triangular (Schur form) tridiagonal eigvals →  diagonal

+ symmetry

Algs:



sym



Alt:
don't transform A
directly, only apply A (good if A is sparse
or if A is an operator)

Orthogonal & QR iteration

OI:

X_0 : input $\underline{X_0} \in \mathbb{R}^{n \times k}$

for $i = 0$ to convergence

$Y_i = \underline{A} X_i$

$\underline{X_{i+1} R} = Y_i$ ← orthogonalize

converges to span{ largest k eigvecs of A }

QR iteration

OI with $n = k$

computing OI is equiv. to

$A_0 = A$

for i until conv.

$\hat{Q}_i R_i = A_i$

$A_{i+1} = R_i \hat{Q}_i$

$AX_k = Q_k R_k$

$Q_k = \hat{Q}_k \cdots \hat{Q}_0$

$A_i = \underbrace{Q_k^T}_{\hat{Q}_0 \cdots \hat{Q}_i} A Q_k$

QR Iteration: Incorporating a Shift

How can we accelerate convergence of QR iteration using shifts?

$Q_k R_k = A_k - \sigma_k I$

$A_{k+1} = R_k Q_k + \sigma_k I$

if $A_k =$ ◥ $\Rightarrow$ Schur form

$Q_k R_k =$ ◥ $\quad A_k = Q_k^T A Q_k$

I ◥

σ_k can be chosen as last entry $A_k[n-1, n-1]$ until last column converges

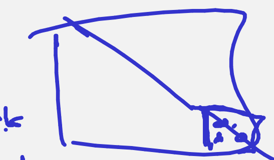
σ_k based on eigvals of $A_k[n-2:n, n-2:n]$

Are A_k and A_{k+1} still similar?

$Q_k R_k = A_k - \sigma_k I \Rightarrow R_k Q_k = Q_k^T(A_k - \sigma_k I)Q_k$

$A_{k+1} - \sigma_k I = R_k Q_k$

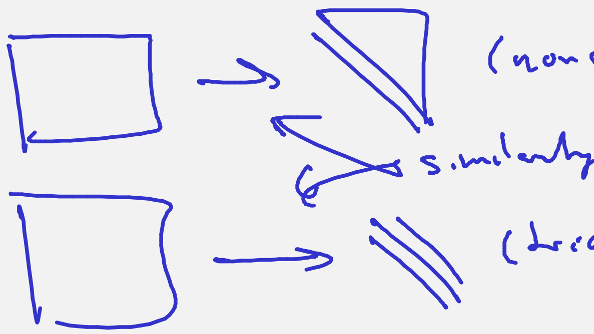
similar similar $A_{k+1} - \sigma_k I \Rightarrow A_k$ and A_{k+1} are similar



QR Iteration: Computational Expense

A full QR factorization at each iteration costs $O(n^3)$–can we make that cheaper?

of QR iteration



(nonsym.) upper-Hessenberg form

(tridiagonal) $B = QAQ^T$

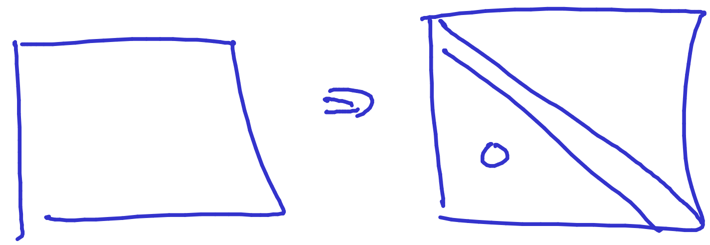
$A^T = A$ $\quad B^T = Q^{TT} A^T Q^T = QAQ^T = B$

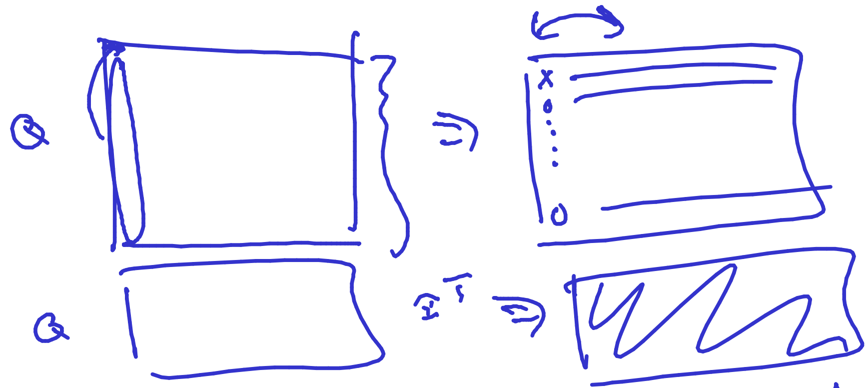
QR factorization of [tridiagonal] can be done with $n-1$ Givens rotations

↳ QR cost of $O(n^2)$ is $O(n)$ for sym.

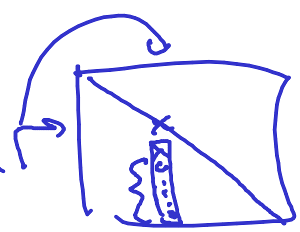
Demo: Householder Similarity Transforms [cleared]

Upper-Hessenberg reduction





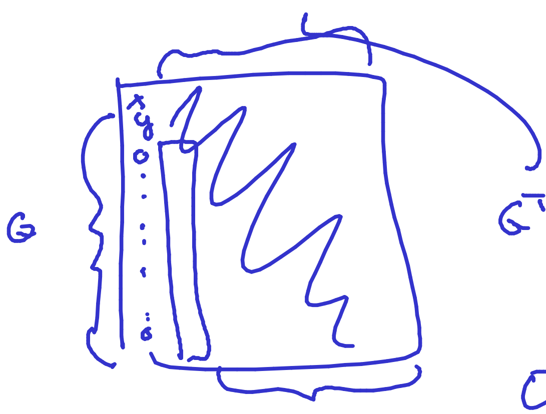
ith



$A_0 = A$

for $i = 0$ to $n-1$

$Q_i = \text{House}(A_i[i+1:n, i])$



$O(n)$

$O(n^2)$

$A_{i+1} = \underbrace{Q_i^T A_i Q_i}_{\text{s.m. transf.}}$

acts on $n-i-1$ last rows

acts on $n-i-1$ last cols

overall $O(n^3)$

A_0 is U-H

$Q_0 R_0 = A_0$

$A_1 = R_0 Q_0$

U-H * ▷ = U-H

$Q_0 = A_0 R_0^{-1} =$

$$Q_i = \begin{array}{c} \overbrace{\hphantom{\quad I \quad}}^{i+1 \text{ cols}} \\ \begin{bmatrix} I & \\ & I - 2\dfrac{vv^T}{v^T v} \end{bmatrix} \end{array}$$

QR/Hessenberg: Overall procedure

Overall procedure:

1. Reduce matrix to Hessenberg form
2. Apply QR iteration using Givens QR to obtain Schur form

Why does QR iteration *stay* in Hessenberg form?

What does this process look like for symmetric matrices?

Krylov space methods: Intro

What subspaces can we use to look for eigenvectors?

power iteration: $x_k = Ax_{k-1}$

nice in that we only 'apply' A to vectors

how well can we do with k applications of A

"Krylov subspace" $S_k = \text{span}\{x_0, Ax_0, \dots, A^{k-1}x_0\}$

find best $x \in S_k$ to e.g., $\boxed{\underset{\min}{\overset{\max}{\updownarrow}} \frac{x^T A x}{x^T x}}$ | $\min_x x^T A x - x^T b$

$\min_x \|Ax - b\|$

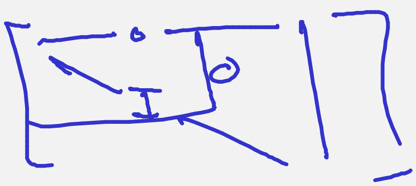
Krylov for Matrix Factorization

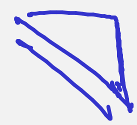
What matrix factorization is obtained through Krylov space methods?

$k = n$ 　 lets assume K_n is invertible

$$K_n = \begin{bmatrix} x_0 & Ax_0 & \cdots & A^{n-1}x_0 \end{bmatrix}$$

$$AK_n = \begin{bmatrix} Ax_0 & \cdots & A^{n-1}x_0 & A^n x_0 \end{bmatrix}$$

$K_n^{-1} A K_n =$ 

u-H 

Conditioning in Krylov Space Methods/Arnoldi Iteration (I)

What is a problem with Krylov space methods? How can we fix it?

columns of K_n $A^{n-2}x$ $A^{n-1}x$ become linearly dependent with large n

$K_k \in \mathbb{R}^{n\times k}$

$= \left[x\ Ax \dots A^{k-1}x \right]$

$= Q_k R_k$

$\operatorname{span}(Q_k) = \operatorname{span}(K_k) = S_k$

$Q_n A Q_n^T =$

$K_n^{-1} A K_n =$

$K_n = Q_n R_n$

$K_n^{-1} = R_n^{-1} Q_n^T$

$\underbrace{R_n K_n^{-1}}_{Q_n^T} A \underbrace{K_n R_n^{-1}}_{Q_n} =$

$\underbrace{Q_k A Q_k^T}_{H_k} =$ $\}k$

k

Conditioning in Krylov Space Methods/Arnoldi Iteration (II)

$Q_k^T A Q_k = H_k$

$AQ_n = Q_n H_n$

$Q_n = \begin{bmatrix} q_1 & \cdots & q_n \end{bmatrix}$

$Aq_k = \underline{h_{1k}} q_1 + \underline{h_{2k}} q_2 + \dots \underline{h_{k+1,k}} \underbrace{q_{k+1}}$

having determined $q_1 \dots q_k$, we can find q_{k+1}

$H_{ij} = q_i^T A q_j$

Demo: Arnoldi Iteration [cleared] (Part 1)