CS 598 EVS: Tensor Computations

Matrix Computations Background

Edgar Solomonik

University of Illinois at Urbana-Champaign

Conditioning

- **Absolute Condition Number:**

- **(Relative) Condition Number:**

Posedness and Conditioning

- **What is the condition number of an ill-posed problem?**

Matrix Condition Number

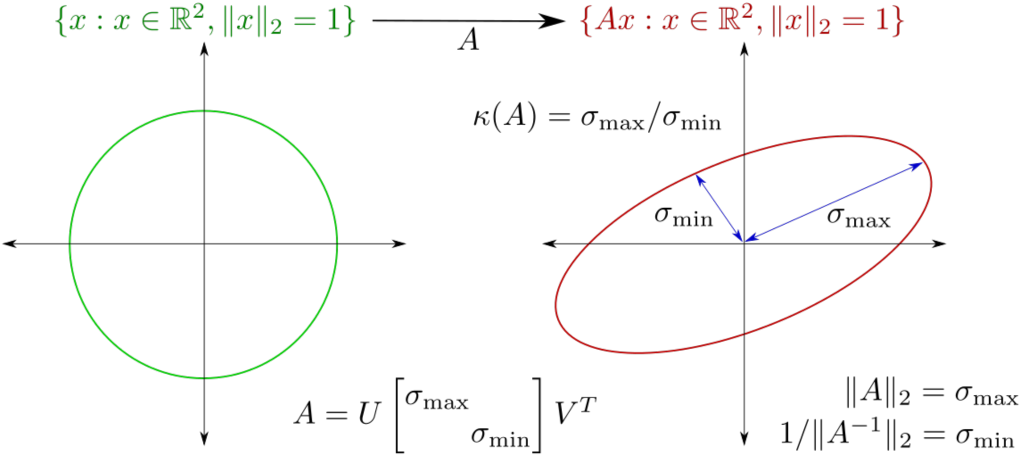
- The matrix condition number $\kappa(\boldsymbol{A})$ is the ratio between the max and min distance from the surface to the center of the unit ball (norm-1 vectors) transformed by $\boldsymbol{A}$:

- The matrix condition number bounds the worst-case amplification of error in a matrix-vector product:

Singular Value Decomposition

- The singular value decomposition (SVD)
- Condition number in terms of singular values

Visualization of Matrix Conditioning



$$\{x : x \in \mathbb{R}^2, \|x\|_2 = 1\} \xrightarrow[A]{} \{Ax : x \in \mathbb{R}^2, \|x\|_2 = 1\}$$

$$\kappa(A) = \sigma_{\max}/\sigma_{\min}$$

$$A = U \begin{bmatrix} \sigma_{\max} & \\ & \sigma_{\min} \end{bmatrix} V^T$$

$$\|A\|_2 = \sigma_{\max}$$

$$1/\|A^{-1}\|_2 = \sigma_{\min}$$

Linear Least Squares

- Find $\boldsymbol{x}^\star = \operatorname{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} ||\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}||_2$ where $\boldsymbol{A} \in \mathbb{R}^{m\times n}$:

- Given the SVD $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ we have $\boldsymbol{x}^\star = \underbrace{\boldsymbol{V}\boldsymbol{\Sigma}^\dagger\boldsymbol{U}^T}_{\boldsymbol{A}^\dagger}\boldsymbol{b}$, where $\boldsymbol{\Sigma}^\dagger$ contains the reciprocal of all nonzeros in $\boldsymbol{\Sigma}$, and more generally $\dagger$ denotes pseudoinverse:

Demo: *Normal equations vs Pseudoinverse*
Demo: *Issues with the normal equations*

Normal Equations

- *Normal equations* are given by solving $\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}^T\boldsymbol{b}$:

- However, solving the normal equations is a more ill-conditioned problem then the original least squares algorithm

Solving the Normal Equations

- If $\boldsymbol{A}$ is full-rank, then $\boldsymbol{A}^T\boldsymbol{A}$ is symmetric positive definite (SPD):

- Since $\boldsymbol{A}^T\boldsymbol{A}$ is SPD we can use Cholesky factorization, to factorize it and solve linear systems:

QR Factorization

- If $\boldsymbol{A}$ is full-rank there exists an orthogonal matrix $\boldsymbol{Q}$ and a unique upper-triangular matrix $\boldsymbol{R}$ with a positive diagonal such that $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}$

- A reduced QR factorization (unique part of general QR) is defined so that $\boldsymbol{Q} \in \mathbb{R}^{m \times n}$ has orthonormal columns and $\boldsymbol{R}$ is square and upper-triangular

- We can solve the normal equations (and consequently the linear least squares problem) via reduced QR as follows

Computing the QR Factorization

- The Cholesky-QR algorithm uses the normal equations to obtain the QR factorization
- Orthogonalization-based methods are most efficient and stable for QR factorization of dense matrices

Eigenvalue Decomposition

- If a matrix $\boldsymbol{A}$ is diagonalizable, it has an *eigenvalue decomposition*

- $\boldsymbol{A}$ and $\boldsymbol{B}$ are *similar*, if there exist $\boldsymbol{Z}$ such that $\boldsymbol{A} = \boldsymbol{Z}\boldsymbol{B}\boldsymbol{Z}^{-1}$

Similarity of Matrices

matrix	*similarity*	*reduced form*
SPD		
real symmetric		
Hermitian		
normal		
real		
diagonalizable		
arbitrary		

Rayleigh Quotient

- For any vector $\boldsymbol{x}$ that is close to an eigenvector, the *Rayleigh quotient* provides an estimate of the associated eigenvalue of $\boldsymbol{A}$:

Introduction to Krylov Subspace Methods

- *Krylov subspace methods* work with information contained in the $n \times k$ matrix

$$\boldsymbol{K}_k = \underline{\begin{bmatrix} \overset{\text{starting vector}}{\boldsymbol{x}_0} & \boldsymbol{A}\boldsymbol{x}_0 & \cdots & \boldsymbol{A}^{k-1}\boldsymbol{x}_0 \end{bmatrix}}$$

$\operatorname{span}(K_k)$

$$\min_{x \in \operatorname{span}(K_k)} \left\{ \frac{x^T A x}{x^T x}, \|Ax-b\|_2 \right\}$$

- $\boldsymbol{A}$ is similar to *companion matrix* $\boldsymbol{C} = \boldsymbol{K}_n^{-1}\boldsymbol{A}\boldsymbol{K}_n$:

Krylov Subspaces

- Given $\underline{\boldsymbol{Q}}_k \boldsymbol{R}_k = \boldsymbol{K}_k$, we obtain an orthonormal basis for the Krylov subspace,

$$\underline{\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{x}_0)} = span(\boldsymbol{Q}_k) = \{p(\boldsymbol{A})\boldsymbol{x}_0 : deg(p) < k\},$$

  where p is any polynomial of degree less than k.

- The Krylov subspace includes the $k-1$ approximate dominant eigenvectors generated by $k-1$ steps of power iteration:

Power iteration

$$\underline{x}^{(k)} = A\underline{x}^{(k-1)}$$

compute $\rho = \dfrac{x^{(k)T} A x^{(k)}}{x^{(k)T} x^{(k)}}$

$$x^{(k)} = x^{(k)} / \rho$$

Krylov Subspace Methods

$AX = XD$

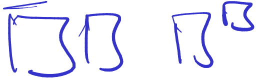
- The $k \times k$ matrix $\boldsymbol{H}_k = \boldsymbol{Q}_k^T \boldsymbol{A}\boldsymbol{Q}_k$ minimizes $||\boldsymbol{A}\boldsymbol{Q}_k - \boldsymbol{Q}_k\boldsymbol{H}_k||_2$:

Ritz vectors/values

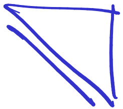
$\underbrace{\text{(eigvec/eigvals}}$ of H_k)

approximate of A

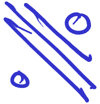
$$AQ_k \approx Q_k \overset{?}{H_k}$$



- $\boldsymbol{H}_k$ is upper-Hessenberg, because the companion matrix $\boldsymbol{C}_n$ is upper-Hessenberg:



if A is symmetric



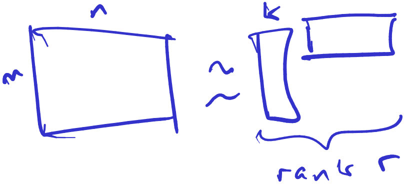
Rayleigh-Ritz Procedure

- The eigenvalues/eigenvectors of $\boldsymbol{H}_k$ are the *Ritz values/vectors*:

- The Ritz vectors and values are the *ideal approximations* of the actual eigenvalues and eigenvectors based on only $\boldsymbol{H}_k$ and $\boldsymbol{Q}_k$:

Low Rank Matrix Approximation

- Given a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ seek rank $r < m, n$ approximation



$f_A(x) = Ax$

$\text{rank}(\tilde{A}) = r$

$\underline{\tilde{A} - A}$

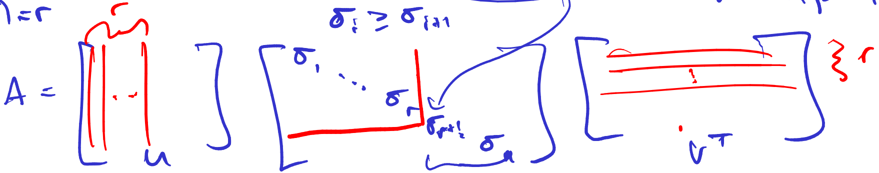
- Eckart-Young (optimal low-rank approximation by SVD) theorem

$$\min_{\substack{\tilde{A} \\ \text{rank}(\tilde{A}) = r}} \|\tilde{A} - A\|_F \quad \text{or} \quad \|\tilde{A} - A\|_2$$

$m \geq n$

$\|X\|_F^2 = \sum_i \sigma_i(X)^2$

$\sigma_i \geq \sigma_{i+1}$



Rank Revealing Matrix Factorizations

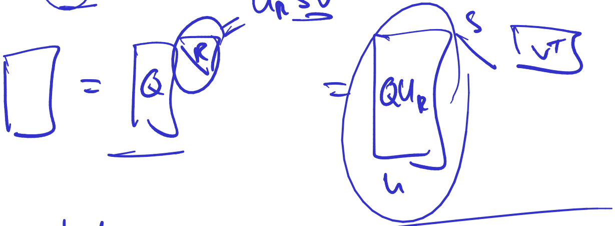
- Computing the SVD

diagonalize $\underline{A^TA}$ → eigvecs of A^TA = singvecs of A

$AV_{:k} = U_1 S_1$

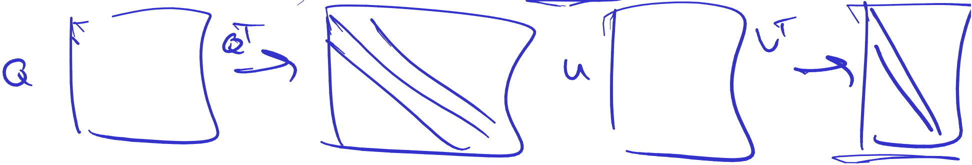
USV^T

$U_R S V^T$

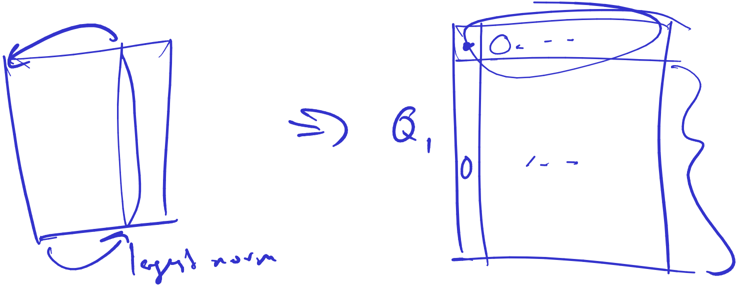


- QR with column pivoting

Golub-Kahan bidiagonalization $O(n^3)$

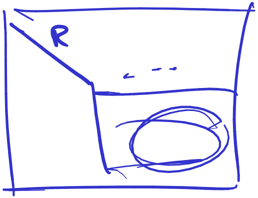


QR with column pivoting



after k steps

$Q_1 \cdots Q_k$



Orthogonal Iteration

- For sparse matrices, QR factorization creates fill, so must revert to iterative methods

R = L factor of A^TA

A is square

$B_k = AQ_k$ span(Q_∞) = span(of leading R singular vectors of A)

$Q_{k+1}R = B_k$ A^TA $A^T(AQ_k)$

- Orthogonal iteration interleaves deflation and power iteration

Randomized SVD

QR-col-piv $O(mn^2)$

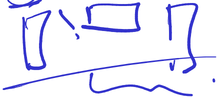
- Orthogonal iteration for SVD can also be viewed as a randomized algorithm

pick Q_0 to be random & orthogonal

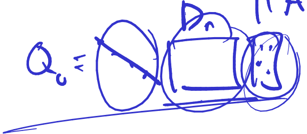
AQ_0 if A is low rank (rank r)

$O(mn \cdot \log n)$

$U S V^T Q_0$



$Q_0 =$



$\|A - \tilde{A}\|_F$

$A^T(AQ_0)$

$V^T Q_0$ as a set of random normalized linear combinations of cols of US

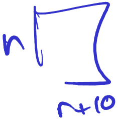
r

$u_1\sigma_1 \ldots u_r\sigma_r$

$Q_1 R = QR(AQ_0)$

$\text{span}(Q_1) = \text{span}(U)$

$(USU^{\dagger} + \boxed{\underrightarrow{E}})\, Q_0$



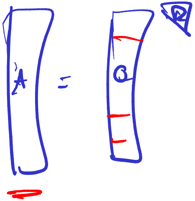
Generalized Nyström Algorithm

- The generalized Nyström algorithm provides an efficient way of computing a sketched low-rank factorization

$$\tilde{A} = A S_1 (S_2^T A S_1)^+ S_2^T A$$

where S_1 and S_2 are sketching matrices

- Gaussian random
- FFT structure SRFT
- leverage-score sampling



Multidimensional Optimization

- Minimize $f(\boldsymbol{x})$

- Quadratic optimization $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x}$

Basic Multidimensional Optimization Methods

- Steepest descent: minimize f in the direction of the negative gradient:

- Given quadratic optimization problem $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}^T\boldsymbol{x}$ where $\boldsymbol{A}$ is symmetric positive definite, the error $\boldsymbol{e}_k = \boldsymbol{x}_k - \boldsymbol{x}^*$ satisfies

$$||\boldsymbol{e}_{k+1}||_{\boldsymbol{A}} =$$

  - When sufficiently close to a local minima, general nonlinear optimization problems are described by such an SPD quadratic problem.
  - Convergence rate depends on the conditioning of $\boldsymbol{A}$, since

Gradient Methods with Extrapolation

- We can improve the constant in the linear rate of convergence of steepest descent by leveraging *extrapolation methods*, which consider two previous iterates (maintain *momentum* in the direction $\boldsymbol{x}_k - \boldsymbol{x}_{k-1}$):

- The *heavy ball method*, which uses constant $\alpha_k = \alpha$ and $\beta_k = \beta$, achieves better convergence than steepest descent:

Conjugate Gradient Method

- The *conjugate gradient method* is capable of making the optimal (for a quadratic objective) choice of α_k and β_k at each iteration of an extrapolation method:
- *Parallel tangents* implementation of the method proceeds as follows

Krylov Optimization

- Conjugate gradient (CG) finds the minimizer of $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x}$ (which satisfies optimality condition $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$) within the Krylov subspace of $\boldsymbol{A}$:

CG and Krylov Optimization

The solution at the kth step, $\boldsymbol{y}_k = ||\boldsymbol{b}||_2 \boldsymbol{T}_k^{-1} \boldsymbol{e}_1$ is obtained by CG from $\boldsymbol{y}_{k+1}$ with a single matrix-vector product with $\boldsymbol{A}$ and vector operations with $O(n)$ cost

Preconditioning

- Convergence of iterative methods for $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ depends on $\kappa(\boldsymbol{A})$, the goal of a preconditioner $\boldsymbol{M}$ is to obtain $\boldsymbol{x}$ by solving

$$\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{M}^{-1}\boldsymbol{b}$$

  with $\kappa(\boldsymbol{M}^{-1}\boldsymbol{A}) < \kappa(\boldsymbol{A})$

- Common preconditioners select parts of $\boldsymbol{A}$ or perform inexact factorization

Conjugate Gradient Convergence Analysis

- In previous discussion, we assumed $\boldsymbol{K}_n$ is invertible, which may not be the case if $\boldsymbol{A}$ has $m < n$ distinct eigenvalues, however, in exact arithmetic CG converges in $m - 1$ iterations[1]

[1] This derivation follows *Applied Numerical Linear Algebra* by James Demmel, Section 6.6.4

Conjugate Gradient Convergence Analysis (II)

- Using $\boldsymbol{z} = \boldsymbol{\rho}_{k-1}(\boldsymbol{A})\boldsymbol{A}\boldsymbol{x}$, we can simplify $\phi(\boldsymbol{z}) = (\boldsymbol{x} - \boldsymbol{z})^T\boldsymbol{A}(\boldsymbol{x} - \boldsymbol{z})$ as

- We can bound the objective based on the eigenvalues of $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$ using the identity $\boldsymbol{p}(\boldsymbol{A}) = \boldsymbol{Q}\boldsymbol{p}(\boldsymbol{\Lambda})\boldsymbol{Q}^T$,

Conjugate Gradient Convergence Analysis (III)

- Using our bound on the square of the residual norm $\phi(\boldsymbol{z})$, we can see why CG converges after $m-1$ iterations if there are only $m < n$ distinct eigenvalues

- To see that the residual goes to 0, we find a suitable polynomial in $\mathcal{Q}_m$ (the set of polynomials q_m of degree m with $q_m(0) = 1$)

Round-off Error in Conjugate Gradient

- CG provides strong convergence guarantees for SPD matrices in exact arithmetic

- Due to round-off CG may stagnate / have plateaus in convergence

Graph and Matrix Duality

- graphs have have a natural correspondence with sparse matrices

- matrix-based representations of graphs can be used to devise algorithms

Graph Partitioning from Eigenvectors

- The Laplacian matrix provides a model of interactions on a graph that is useful in many contexts

- The second-smallest-eigenvalue eigenvector of the Laplacian (the Fiedler vector), gives a good partitioning of the graph

Newton’s Method

- Newton’s method in n dimensions is given by finding minima of n-dimensional quadratic approximation using the gradient and Hessian of f:

Nonlinear Least Squares

- An important special case of multidimensional optimization is *nonlinear least squares*, the problem of fitting a nonlinear function $f_{\boldsymbol{x}}(t)$ so that $f_{\boldsymbol{x}}(t_i) \approx y_i$:

- We can cast nonlinear least squares as an optimization problem to minimize residual error and solve it by Newton's method:

Gauss-Newton Method

- The Hessian for nonlinear least squares problems has the form:

- The *Gauss-Newton* method is Newton iteration with an approximate Hessian:

Constrained Optimization Problems

- We now return to the general case of *constrained* optimization problems:

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{g}(\boldsymbol{x}) = \mathbf{0} \quad \text{and} \quad \boldsymbol{h}(\boldsymbol{x}) \leqslant \mathbf{0}$$

- Generally, we will seek to reduce constrained optimization problems to a series of simpler optimization problems: