CS 450: Numerical Anlaysis[1]

Linear Systems

University of Illinois at Urbana-Champaign

[1] *These slides have been drafted by Edgar Solomonik as lecture templates and supplementary material for the book "Scientific Computing: An Introductory Survey" by Michael T. Heath (slides).*

Vector Norms

- **Properties of vector norms**

- **A norm is uniquely defined by its unit sphere:**

- p**-norms**

Inner-Product Spaces

- **Properties of inner-product spaces**: Inner products $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ must satisfy

$$
\begin{aligned}
\langle \boldsymbol{x}, \boldsymbol{x} \rangle &\geq 0 \\
\langle \boldsymbol{x}, \boldsymbol{x} \rangle &= 0 \quad \Leftrightarrow \quad \boldsymbol{x} = \boldsymbol{0} \\
\langle \boldsymbol{x}, \boldsymbol{y} \rangle &= \langle \boldsymbol{y}, \boldsymbol{x} \rangle \\
\langle \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{z} \rangle &= \langle \boldsymbol{x}, \boldsymbol{y} \rangle + \langle \boldsymbol{x}, \boldsymbol{z} \rangle \\
\langle \alpha \boldsymbol{x}, \boldsymbol{y} \rangle &= \alpha \langle \boldsymbol{x}, \boldsymbol{y} \rangle
\end{aligned}
$$

- **Inner-product-based vector norms**

Matrix Norms

- **Properties of matrix norms:**

$$||\boldsymbol{A}|| \geq 0$$
$$||\boldsymbol{A}|| = 0 \quad \Leftrightarrow \quad \boldsymbol{A} = \boldsymbol{0}$$
$$||\alpha \boldsymbol{A}|| = |\alpha| \cdot ||\boldsymbol{A}||$$
$$||\boldsymbol{A} + \boldsymbol{B}|| \leq ||\boldsymbol{A}|| + ||\boldsymbol{B}|| \quad (\textit{triangle inequality})$$

- **Frobenius norm:**

- **Operator/induced/subordinate matrix norms:**

Induced Matrix Norms

- **Interpreting induced matrix norms**:

- **General induced matrix norms**:

Matrix Condition Number

Demo: Conditioning of 2x2 Matrices
Demo: Condition number visualized

- **Definition**: $\kappa(\boldsymbol{A}) = ||\boldsymbol{A}|| \cdot ||\boldsymbol{A}^{-1}||$ is the ratio between the shortest/longest distances from the unit-ball center to any point on the surface.

- **Intuitive derivation**:

$$\kappa(\boldsymbol{A}) = \max_{\text{inputs}} \max_{\text{perturbations in input}} \left| \frac{\text{relative perturbation in output}}{\text{relative perturbation in input}} \right|$$

since a matrix is a linear operator, we can decouple its action on the input $\boldsymbol{x}$ and the perturbation $\boldsymbol{\delta x}$ since $\boldsymbol{A}(\boldsymbol{x} + \boldsymbol{\delta x}) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{A}\boldsymbol{\delta x}$, so

$$\kappa(\boldsymbol{A}) = \left| \frac{\overbrace{\max_{\text{perturbations in input}} \text{relative perturbation growth}}^{||\boldsymbol{A}||}}{\underbrace{\max_{\text{inputs}} \text{relative input reduction}}_{1/||\boldsymbol{A}^{-1}||}} \right|$$

Matrix Conditioning

- 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 transformed by $\kappa(\boldsymbol{A})$:
- The matrix condition number bounds the worst-case amplification of error in a matrix-vector product:

Norms and Conditioning of Orthogonal Matrices

- **Orthogonal matrices**:

- **Norm and condition number of orthogonal matrices**:

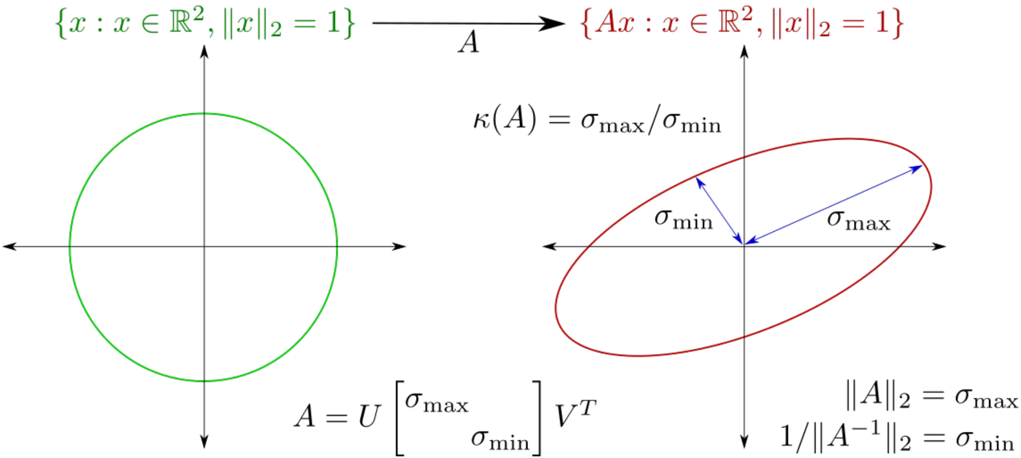
Singular Value Decomposition

- **The singular value decomposition (SVD)**:

Norms and Conditioning via SVD

- **Norm and condition number in terms of singular values**:

Visualization of Matrix Conditioning



Conditioning of Linear Systems

- **Lets now return to formally deriving the conditioning of solving $Ax = b$:**

Conditioning of Linear Systems II

- **Consider perturbations to the input coefficients $\hat{A} = A + \delta A$:**

Solving Basic Linear Systems

- Solve $\boldsymbol{D}\boldsymbol{x} = \boldsymbol{b}$ if $\boldsymbol{D}$ is diagonal

- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$ if $\boldsymbol{Q}$ is orthogonal

- Given SVD $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$, solve $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$

Solving Triangular Systems

Demo: *Coding back-substitution*

- **$Lx = b$ if L is lower-triangular is solved by forward substitution:**

$$\begin{aligned} l_{11}x_1 &= b_1 & & x_1 = \\ l_{21}x_1 + l_{22}x_2 &= b_2 & \Rightarrow \quad & x_2 = \\ l_{31}x_1 + l_{32}x_2 + l_{33}x_3 &= b_3 & & x_3 = \\ &\vdots & & \vdots \end{aligned}$$

- **Algorithm can also be formulated recursively by blocks:**

Solving Triangular Systems

- **Existence of solution to $Lx = b$:**

- **Uniqueness of solution:**

- **Computational complexity of forward/backward substitution:**

Properties of Triangular Matrices

- $Z = XY$ **is lower triangular is** X **and** Y **are both lower triangular:**

- L^{-1} **is lower triangular if it exists:**

LU Factorization

- An ***LU factorization*** consists of a unit-diagonal lower-triangular ***factor*** L and upper-triangular factor U such that $A = LU$:

- Given an LU factorization of A, we can solve the linear system $Ax = b$:

Gaussian Elimination Algorithm

- **Algorithm for factorization is derived from equations given by $A = LU$:**

- **The computational complexity of LU is $O(n^3)$:**

Existence of LU Factorization

$A = LU$

- **The LU factorization may not exist:** Consider matrix $\begin{bmatrix} 3 & 2 \\ 6 & 4 \\ 0 & 3 \end{bmatrix}$.

- **Permutation of rows enables us to transform the matrix so the LU factorization does exist:**

$PA = LU$ always exist if A is full rank

Gaussian Elimination with Partial Pivoting

- ***Partial pivoting*** **permutes rows to make divisor u_{ii} is maximal at each step:**

- **A row permutation corresponds to an application of a *row permutation matrix* $\boldsymbol{P}_{jk} = \boldsymbol{I} - (\boldsymbol{e}_j - \boldsymbol{e}_k)(\boldsymbol{e}_j - \boldsymbol{e}_k)^T$:**

Partial Pivoting Example

- Lets consider again the matrix $\boldsymbol{A} = \begin{bmatrix} 3 & 2 \\ 6 & 4 \\ 0 & 3 \end{bmatrix}$.

$$P_1 = \begin{bmatrix} & 1 & \\ 1 & & \\ & & 1 \end{bmatrix} \Rightarrow P_1 A = \begin{bmatrix} 6 & 4 \\ 3 & 2 \\ 0 & 3 \end{bmatrix}$$

$$P_1 A = \begin{bmatrix} 1 \\ 1/2 \\ 0 \end{bmatrix} \begin{bmatrix} 6 & 4 \end{bmatrix} + \begin{bmatrix} & \\ & 0 \\ & 3 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$P_2 \begin{bmatrix} 0 \\ 3 \end{bmatrix} = 3\begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & \\ & P_2 \end{bmatrix} P_1 A = \begin{bmatrix} 1 & \\ 0 & 1 \\ 1/2 & 0 \end{bmatrix} \begin{bmatrix} 6 & 4 \\ & 3 \end{bmatrix}$$

Complete Pivoting

$P_1 A P_2 x = b$ $\quad$ $PA = LU$ or $P_1 A P_2^T = LU$

- ***Complete pivoting*** **permutes rows and columns to make divisor u_{ii} is maximal at each step:**

$P_1 A P_2^T = LU$ $\quad$ $|l_{ij}| \le 1$

$|l_{21}| \cdot \|u_{12}\|_\infty \le |a_{21}|$

$$P_1 A P_2^T = \begin{bmatrix} l_{11} & \\ l_{21} & L_{22} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} \\ & U_{22} \end{bmatrix}$$

- **Complete pivoting is noticeably more expensive than partial pivoting:**

P-P $O(n)$ comparisons for each column

C-P complete pivoting, need $O(n^2)$ comparisons per column

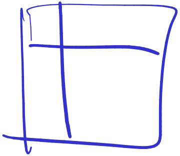
Round-off Error in LU

- **Lets consider factorization of** $\begin{bmatrix} \epsilon & 1 \\ 1 & 1 \end{bmatrix}$ **where** $\epsilon < \epsilon_{\textbf{mach}}$**:**

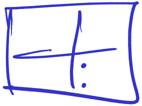
- **Permuting the rows of** A **in partial pivoting gives** $PA = \begin{bmatrix} 1 & 1 \\ \epsilon & 1 \end{bmatrix}$

Error Analysis of LU

- **The main source of round-off error in LU is in the computation of the Schur complement:**



$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ & U_{22} \end{bmatrix}$$



$$S_{22} = A_{22} - L_{21}U_{12} \qquad S_{22} = L_{22}U_{22}$$

- **When computed in floating point, absolute backward error δA in LU (so $\hat{L}\hat{U} = A + \delta A$) is $|\delta a_{ij}| \leq \epsilon_{\text{mach}}(|\hat{L}| \cdot |\hat{U}|)_{ij}$**

$$L = \begin{bmatrix} l_1 \\ l_2 \\ \vdots \\ l_n \end{bmatrix} \qquad U = \begin{bmatrix} u_1 & \cdots & u_n \end{bmatrix}$$

$$a_{ij} - \langle l_i, u_j \rangle$$

$$fl(a_{ij} - \langle l_i, u_j \rangle) - \text{true} = \epsilon \langle |l_i|, |u_j| \rangle$$

Helpful Matrix Properties

$\sum_{j\neq i} |a_{ij}| < |a_{ii}|$ & $\sum_{i \neq j} |a_{ij}| < |a_{jj}|$

- **Matrix is *diagonally dominant*, so** $\sum_{i\neq j} |a_{ij}| \leq |a_{ii}|$**:**

  strict ($<$)

- **Matrix is *symmetric positive definite (SPD)*, so** $\forall_{\boldsymbol{x}\neq 0}, \boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} > 0$**:**

  Cholesky $\lambda_i(A) > 0$

  $A = LL^T$ ← not unit diagonal

  A =  $A = D$, $d_{ii} > 0$

  $x^T D x = \sum_i x_i^2 d_{ii}$

- **Matrix is symmetric but indefinite:**

  $A = A^T$

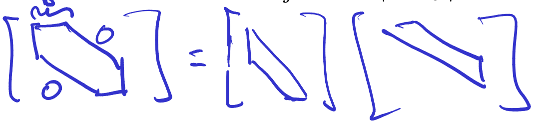
  $PAP^T = LDL^T$ ← unit-diagonal

  $A = UDU^T$

  $x^T A x = x^T U D U^T x$

  $= y^T D y$, $y = U^T x$

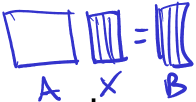
- **Matrix is *banded*,** $a_{ij} = 0$ **if** $|i-j| > b$**:**



$O(nb^2)$

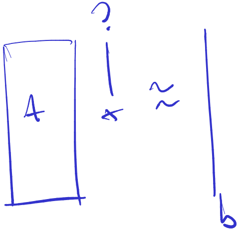
Solving Many Linear Systems

- **Suppose we have computed $A = LU$ and want to solve $AX = B$ where B is $n \times k$ with $k < n$:**



- **Suppose we have computed $A = LU$ and now want to solve a perturbed system $(\boldsymbol{A} - \boldsymbol{u}\boldsymbol{v}^T)\boldsymbol{x} = \boldsymbol{b}$:**
  Can use the *Sherman-Morrison-Woodbury* formula

$$(\boldsymbol{A} - \boldsymbol{u}\boldsymbol{v}^T)^{-1} = \boldsymbol{A}^{-1} + \frac{\boldsymbol{A}^{-1}\boldsymbol{u}\boldsymbol{v}^T\boldsymbol{A}^{-1}}{1 - \boldsymbol{v}^T\boldsymbol{A}^{-1}\boldsymbol{u}}$$



$A^T A x = A^T b$

Cholesky $(A^T A)$

$\kappa(A)^2 = \kappa(A^T A)$

$\underbrace{x^T A^T}_{y^T} \underbrace{A x}_{y} = y^T y > 0$