Engineering Analysis ENG 3420 Fall 2009

Dan C. Marinescu

Office: HEC 439 B

Office hours: Tu-Th 11:00-12:00

Lecture 18

- Last time:
  - Linear algebra functions in Matlab
  - Vector products
  - Tensor product of two matrices
  - Norm and matrix condition number
  - Characteristic equation, eigenvectors, eigenvalues
- Today:
  - Midterm: solutions and discussions
  - More on characteristic equation, eigenvectors, eigenvalues
  - The inverse of a matrix
  - More on
    - LU Factorization
    - Cholesky decomposition
- Next Time (Chapter 12)
  - Gauss-Siedel

The inverse of a square

- If [*A*] is a square matrix, there is another matrix $[A]^{-1}$, called the inverse of [*A*], for which $[A][A]^{-1}=[A]^{-1}[A]=[I]$
- The inverse can be computed in a column by column fashion by generating solutions with unit vectors as the right-hand-side constants:

$$[A]\{x_1\}=\begin{Bmatrix}1\\0\\0\end{Bmatrix} \quad [A]\{x_2\}=\begin{Bmatrix}0\\1\\0\end{Bmatrix} \quad [A]\{x_3\}=\begin{Bmatrix}0\\0\\1\end{Bmatrix}$$

$$[A]^{-1}=\begin{bmatrix}x_1 & x_2 & x_3\end{bmatrix}$$

Canonical base of an n-dimensional vector space

100……000

010……000

001……000

…………….

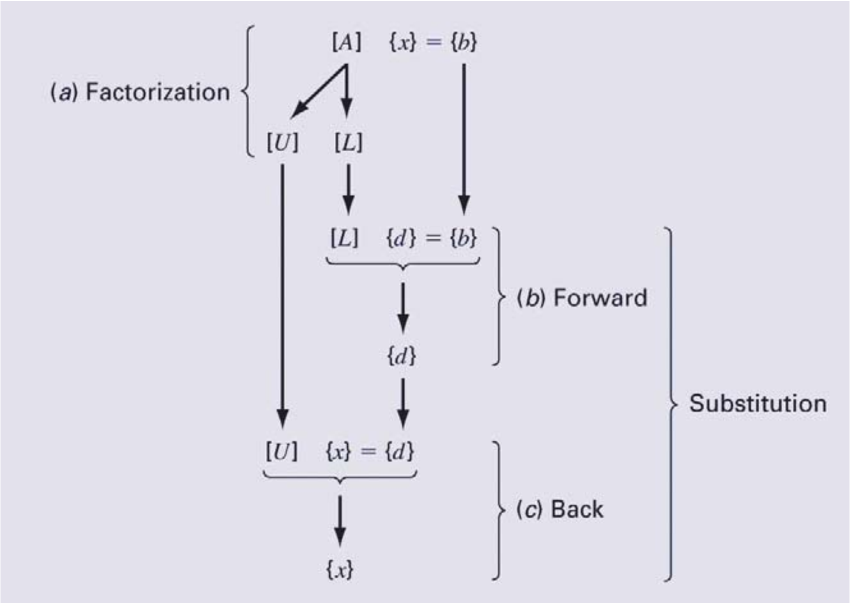
000…….100

000…….010

000…….001

Matrix Inverse (cont)

- *LU* factorization can be used to efficiently evaluate a system for multiple right-hand-side vectors - thus, it is ideal for evaluating the multiple unit vectors needed to compute the inverse.



The response of a linear system

- The response of a linear system to some stimuli can be found using the matrix inverse.

$$[\text{Interactions}]\{\text{response}\} = \{\text{stimuli}\}$$

$$Ar = s$$

$$A^{-1}Ar = A^{-1}s$$

$$A^{-1}A = I$$

$$r = A^{-1}s$$

LU Factorization

- *LU* factorization involves two steps:
  - Decompose the [*A*] matrix into a product of:
    - a lower triangular matrix [*L*] with 1 for each entry on the diagonal.
    - and an upper triangular matrix [*U*
  - Substitution to solve for {*x*}
- Gauss elimination can be implemented using *LU* factorization
- The forward-elimination step of Gauss elimination comprises the bulk of the computational effort.
- *LU* factorization methods separate the time-consuming elimination of the matrix [*A*] from the manipulations of the right-hand-side [*b*].

Gauss Elimination as *LU* Factorization

- To solve [*A*]{*x*}={*b*}, first decompose [*A*] to get [*L*][*U*]{*x*}={*b*}
- MATLAB's lu function can be used to generate the [*L*] and [*U*] matrices:
  [*L*, *U*] = lu(*A*)
- Step 1 → solve [*L*]{*y*}={*b*}; {*y*} can be found using *forward* substitution.
- Step 2→ solve [*U*]{*x*}={*y*}, {*x*} can be found using *backward* substitution.
- In MATLAB:
  ```
  [L, U] = lu(A)
  d = L\b
  x = U\d
  ```
- *LU* factorization → requires the same number of floating point operations (flops) as for Gauss elimination.
- Advantage → once [*A*] is decomposed, the same [*L*] and [*U*] can be used for multiple {*b*} vectors.

Cholesky Factorization

- A symmetric matrix → a square matrix, A, that is equal to its transpose:

  $A = A^T$ (T stands for transpose).
- The *Cholesky factorization* → based on the fact that a symmetric matrix can be decomposed as:

  $[A] = [U]^T[U]$
- The rest of the process is similar to *LU* decomposition and Gauss elimination, except only one matrix, [*U*], needs to be stored.
- Cholesky factorization with the built-in chol command:

  U = chol(A)
- MATLAB's left division operator \ examines the system to see which method will most efficiently solve the problem. This includes trying banded solvers, back and forth substitutions, Cholesky factorization for symmetric systems. If these do not work and the system is square, Gauss elimination with partial pivoting is used.