Lecture 06: Systems of Linear Equations #5: Everything else

Outline:

1) The LU factorization: final comments
   - Row swaps: PA=LU
   - Permutation Matrices and an Example
   - A "real" problem: SpringDemo and Matlab
   - Sparse, Tridiagonal matrices
   - LU vs inv(A);

2) One last Operation: The matrix transpose $A^T$
   - symmetric matrices $A^T=A$
   - product Rule $(AB)^T=B^TA^T$
   - symmetry of $R^T R$ and $RR^T$

LU Factorization and row exchanges: PA=LU

The Problem:
A full description of Gaussian Elimination includes Permutation matrices

The Fix:
In General you can't know the order of permutations before you begin,
but you can track permutations as you proceed such that at the end,
you can permute $A$ once such that $PA=LU$.

Two Matlab approaches (version 7+)

$[L,U,P]=	ext{lu}(A);$ % such that $P*A=L*U$

- or -

$[L,U,P]=	ext{lu}(A,	ext{vector})$ % such that $A(p,:)=L*U$

LU Factorization and row exchanges: PA=LU

For small problems, we'll just permute first then find the LU (and let Matlab
handle the hard stuff)

Example: $A = \begin{bmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$ (several choices for $P$)

LU Factorization and row exchanges: PA=LU

Solving $Ax=b$ using PA=LU

The LU in action: the SpringBlock Demo using Matlab

The Problem: given $N$ masses (each of mass $m_i$) connected by $N$
Springs (with spring constants $k_i$) under gravity,
find the equilibrium positions of the masses. i.e.

The LU in action: the SpringBlock Demo using Matlab

Force Balance on each block, leads to a system of linear equations

Equilibrium implies:
Force Balance on each block, leads to a system of linear equations

\[-k_i x_{i-1} + (k_i + k_{i+1})x_i - k_{i+1} x_{i+1} = m_i g\]

Systems of linear equations can always be written \(Ax = b\) (or \(Kx = m\))

Special Case, \(k_i = m_i = 1\) (all masses and springs are equal)

Solve using Matlab:

\(N=5;\)
\(k=\text{ones}(N,1);\)
\(m=\text{ones}(N,1);\)
\([x,A]=\text{springDemo}(k,m);\)
\(\text{seeLUvsInv}(A);\)  % compare LU(A) to inv(A)

Themes and variations:

\(N=100;\)  % large numbers
\(k=\text{rand}(N,1);\)  % random spring values
\(m=\text{zeros}(N,1);\)  % can lead to near singular matrices
\(m(i)=1;\)  % single point mass

Big Points:

- Coupled Dynamics often lead to large sparse linear systems
- LU of a tri-diagonal system is still tri-diagonal (fill in)
- \(A^{-1}\) for a tri-diagonal system can be dense (why?)

Operation costs for tri-diagonal A

- \(x=A\backslash b\) (LU decomposition) \(O(N)\)
- \(x=\text{inv}(A)\backslash b\) (inverse) \(O(N^3)\) for the inverse and \(O(N^2)\) for \(A^{-1}b\)

Multi-dimensional problems (2-D, 3-D) still sparse but much more expensive by “direct methods”...but there are many other ways to solve \(Ax = b\) (but not in this class)
Final Detail before the big picture:

The Matrix Transpose $A^T$

Vector transpose $x^T$ transforms a column vector into a row vector (and vice versa).

Vectors are just skinny matrices $(m \times 1)$, therefore...

The Matrix Transpose $A^T$ transforms columns of $A$ into ...?

Examples:

Symmetric Matrices:

Definition: A square Matrix $A$ is symmetric if $A^T \cdot A$ (i.e. the rows are the same as the columns)

Examples:

It's not obvious yet but symmetric matrices have important and special properties.

Rules of the Matrix Transpose:

Addition:

Repeated Transpose:

Product Rule: (very important!)

Proof: start by showing $(Ax)^T = x^TA^T$...

Onwards to the big ideas:

Vector Spaces and subspaces:

Linear Independence

Bases and Dimension:

The deeper meaning of $Ax = b$: the 4 fundamental subspaces of a matrix $A$

More Fun Facts using the matrix transpose:

Inverse of $A^T$: prove that $(A^T)^{-1} = (A^{-1})^T$

Show $A^T \cdot A$ and $A A^T$ are square symmetric matrices (but not equal)

Symmetric Matrices can be factored as $A = LDL^T$ (where $D$ is a diagonal matrix of pivots)