

## Lecture 18: Numerical Linear Algebra #3

### Direct methods for solving Linear Least Squares problems

#### Outline

- 0) Finish LU decomposition
  - Algorithms
  - Operation Counts
  - LU vs  $\text{inv}(A)$
- 1) Linear Least Squares Problems
  - A) Example: Polynomial Fitting
  - B) Fundamental idea: projection onto  $C(A)$
  - C) The Normal Equations  $A^T A x = A^T b$
  - D) Orthogonalization techniques  $A = QR$

### Direct methods for solution of linear systems $Ax=b$

Gaussian elimination and the LU factorization.  
Overview: Given a linear systems  $Ax=b$

## Direct methods for solution of linear systems $A\mathbf{x}=\mathbf{b}$

General Algorithm, LU with partial pivoting (lutx from Moler)

```
function [L,U,p] = lutx(A)
%LUTX Triangular factorization, textbook version
% [L,U,p] = lutx(A) produces a unit lower triangular matrix L,
% an upper triangular matrix U, and a permutation vector p,
% so that L*U = A(p,:.)

[n,n] = size(A);
p = (1:n)';

for k = 1:n-1

    % Find index of largest element below diagonal in k-th column
    [r,m] = max(abs(A(k:n,k)));
    m = m+k-1;

    % Skip elimination if column is zero
    if (A(m,k) ~= 0)

        % Swap pivot row
        if (m ~= k)
            A([k m],:) = A([m k],:);
            p([k m]) = p([m k]);
        end

        % Compute multipliers
        i = k+1:n;
        A(i,k) = A(i,k)/A(k,k);

        % Update the remainder of the matrix
        j = k+1:n;
        A(i,j) = A(i,j) - A(i,k)*A(k,j);
    end
end

% Separate result
L = tril(A,-1) + eye(n,n);
U = triu(A);
```

## Direct methods for solution of linear systems $A\mathbf{x}=\mathbf{b}$

General Algorithm: forward and backward substitution

```
function x = forward(L,x)
% FORWARD. Forward elimination.
% For lower triangular L, x = forward(L,b) solves L*x = b.
[n,n] = size(L);
x(1) = x(1)/L(1,1);
for k = 2:n
    j = 1:k-1;
    x(k) = (x(k) - L(k,j)*x(j))/L(k,k);
end

function x = backsubs(U,x)
% BACKSUBS. Back substitution.
% For upper triangular U, x = backsubs(U,b) solves U*x = b.
[n,n] = size(U);
x(n) = x(n)/U(n,n);
for k = n-1:-1:1
    j = k+1:n;
    x(k) = (x(k) - U(k,j)*x(j))/U(k,k);
end
```

**Direct methods for solution of linear systems  $Ax=b$** 

Matlab's Backslash and blashtx

```
function x = blashtx(A,b)
% BSLASHTX Solve linear system (backslash)
% x = blashtx(A,b) solves  $A*x = b$ 

[n,n] = size(A);
if isequal(triu(A,1),zeros(n,n))
    % Lower triangular
    x = forward(A,b);
    return
elseif isequal(tril(A,-1),zeros(n,n))
    % Upper triangular
    x = backsubs(A,b);
    return
elseif isequal(A,A')
    [R,fail] = chol(A);
    if ~fail
        % Positive definite
        y = forward(R',b);
        x = backsubs(R,y);
        return
    end
end

% Triangular factorization
[L,U,p] = lutz(A);

% Permutation and forward elimination
y = forward(L,b(p));

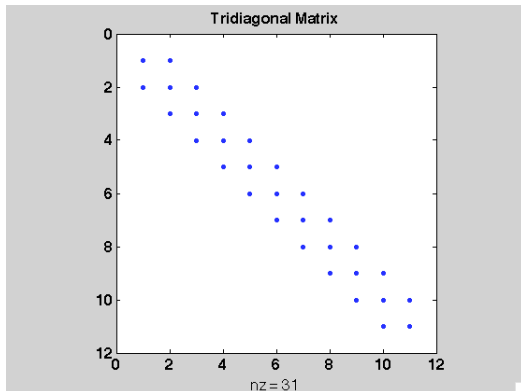
% Back substitution
x = backsubs(U,y);
```

**Operation Counts for Gaussian Elimination w/ PP**

Dense Matrices

## Operation Counts for Gaussian Elimination w/ PP

Sparse Banded Matrices

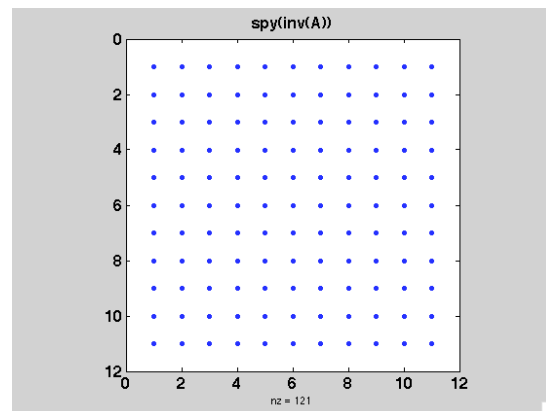
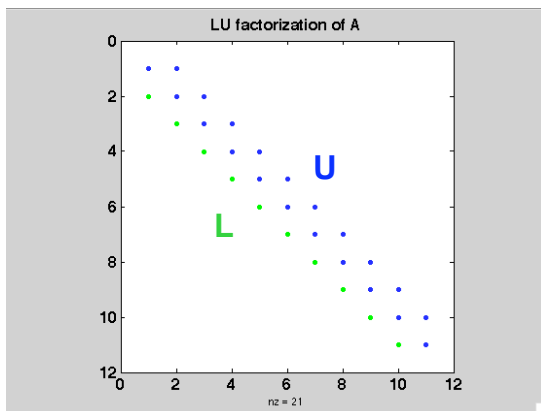


## LU vs. $A^{-1}$

Operation Costs:

*For special matrices can be significantly better than  $n^3/3$*

*Always better than  $\underline{x} = A^{-1}\underline{b}$   $O(n^3)$*



**MATLAB dos and don'ts:**

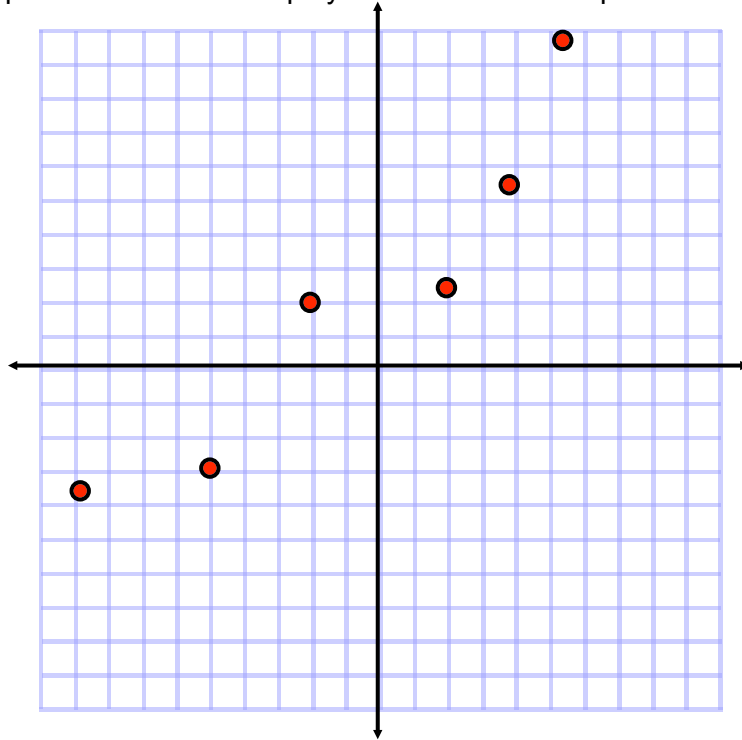
**DON'T** use:  $\underline{x} = \text{inv}(A) * \underline{b}$

**DO** use:  $\underline{x} = A \backslash \underline{b}$

(or for multiple RHS  $[L,U] = \text{lu}(A)$   $\underline{x}_i = U \backslash (L \backslash \underline{b}_i)$ )

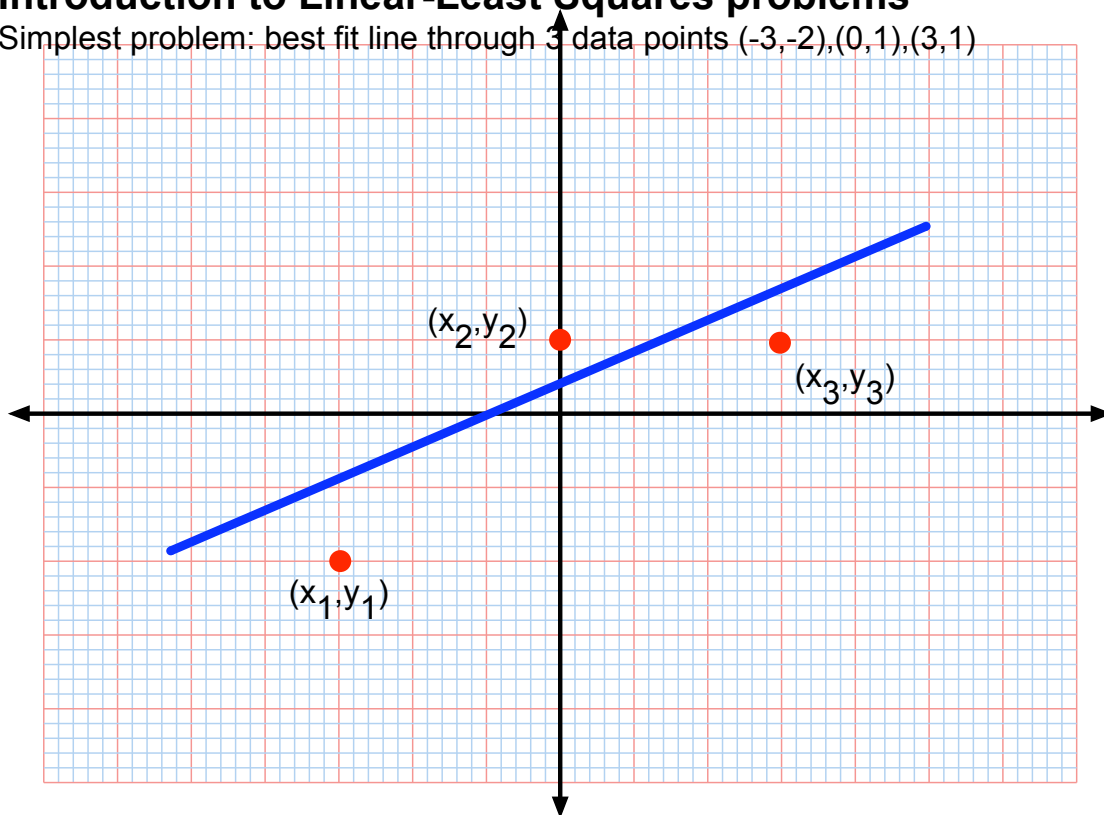
## Introduction to Linear-Least Squares problems

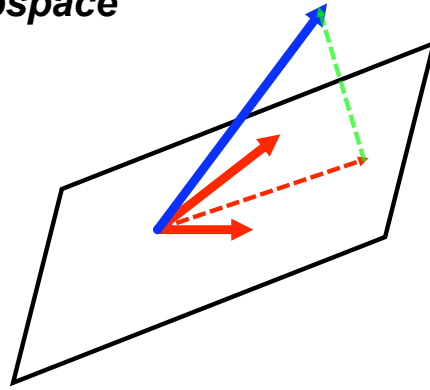
Example: best fit n'th order polynomial to  $m > n$  data points



## Introduction to Linear-Least Squares problems

Simplest problem: best fit line through 3 data points  $(-3, -2), (0, 1), (3, 1)$



***Orthogonal Projection onto a Subspace******Solution of General Linear Least-Squares by the normal equations***

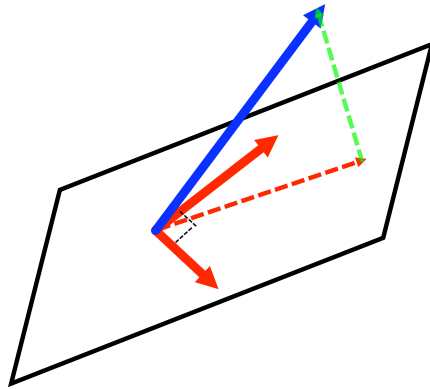
***Normal Equations and Condition number squaring***

The SVD and the  $\text{cond}(A)_2$

***Normal Equations and Condition number squaring***

Example:  $A = \begin{bmatrix} 1 & a & 0 \\ 1 & 0 & a \end{bmatrix}$

***Solution of Least Squares by Orthogonalization  
(QR factorization)***



***Solution of Least Squares by Orthogonalization  
(QR factorization)***

**QR Factorization Algorithms:  
#1 Modified Gram-Schmidt Orthogonalization**

Idea: Take  $A \rightarrow Q$  by repeated projections, get  $R=Q^T A$

**QR Factorization Algorithms:  
#1 Modified Gram-Schmidt Orthogonalization**

Algorithm:

***QR Factorization Algorithms:  
#2 Orthogonalization by Householder transformation***

Idea: Take  $A \rightarrow R$  directly by repeated applications of Q matrices  
(similar algorithm to LU factorization)