

Lecture 17: Numerical Linear Algebra #2

Direct methods for solving $A\mathbf{x}=\mathbf{b}$

Outline

- 0) Review: condition number and error estimation in $A\mathbf{x}=\mathbf{b}$
- 1) The condition number and residual error
- 2) Gaussian Elimination and the LU decomposition
- 3) Gaussian Elimination with partial pivoting $PA=LU$
- 4) Algorithms: (lutx, bslashtx, A\b)

The condition number of a matrix A

Definition: the condition number of a matrix A , $\text{cond}(A)$ (or sometimes $\kappa(A)$) is $\|A\| \|A^{-1}\|$

Condition number of special matrices

1) $\text{cond}(I) =$

2) $\text{cond}(D) =$

3) $\text{cond}(A)$ for A singular:

4) $\text{cond}(\alpha A) =$

The condition number of a matrix A

Example: $A = \begin{bmatrix} 1 & 2 \\ 1+\epsilon & 2 \end{bmatrix}$ $\epsilon \geq \epsilon_{\text{mach}}$

The condition number and error analysis of $A\mathbf{x}=\mathbf{b}$

The condition number and error analysis of $A\underline{x}=\underline{b}$

Relative error and the residual

The condition number and error analysis of $A\underline{x}=\underline{b}$

Example: $A = \begin{bmatrix} 1 & 2 \\ 1+\epsilon & 2 \end{bmatrix}$, $b = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$ ($\epsilon = \epsilon_{\text{mach}}$)

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

Gaussian elimination and the LU factorization.

Overview: Given a linear systems $Ax=b$

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

Gaussian elimination and the LU factorization.

Example: $A = \begin{bmatrix} 1 & -1 & 2 \\ -1 & -1 & 0 \\ 2 & -6 & 9 \end{bmatrix}$

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

Failure of Gaussian Elimination: the zero pivot

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

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

Gaussian Elimination with Partial Pivoting

Example: $A = \begin{bmatrix} \varepsilon & 1 \\ 1 & 1 \end{bmatrix}$, $b = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$

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

Gaussian Elimination with Partial Pivoting: $PA=LU$
 Permutation matrices P and permutation vectors

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 $Ax=b$

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 bslashtx

```
function x = bslashtx(A,b)
% BSLASHTX Solve linear system (backslash)
% x = bslashtx(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] = lutx(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