LINEAR EQATIONS [1.1] + (CONTINUED)

Gaussian Elimination

➤ Back to arbitrary linear systems.

Principle of the method: Since triangular systems are easy to solve, we will transform a linear system into one that is triangular. Main operation: combine rows so that zeros appear in the required locations to make the system triangular.

Recall Notation: Augmented form of a system

$$\begin{cases} 2x_1 + 4x_2 + 4x_3 = 2 \\ x_1 + 3x_2 + 1x_3 = 1 \\ x_1 + 5x_2 + 6x_3 = -6 \end{cases} \quad \text{Notation:} \quad \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 1 & 3 & 1 & 1 \\ 1 & 5 & 6 & -6 \end{array}\right]$$

➤ Main operation used: scaling and adding rows.

- Examples of such operations.

Example: Replace row 2 by: row 2 + row 1:

$$\left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 1 & 3 & 1 & 1 \\ 1 & 5 & 6 & -6 \end{array}\right] \rightarrow \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 3 & 7 & 5 & 3 \\ 1 & 5 & 6 & -6 \end{array}\right]$$

Example: Replace row 3 by: 2 times row 3 - row 1:

$$\left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 3 & 7 & 5 & 3 \\ 1 & 5 & 6 & -6 \end{array}\right] \rightarrow \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 3 & 7 & 5 & 3 \\ 0 & 6 & 8 & -14 \end{array}\right]$$

Example: Replace row 1 by: (0.5 * row 1)

$$\left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 3 & 7 & 5 & 3 \\ 0 & 6 & 8 & -14 \end{array}\right] \rightarrow \left[\begin{array}{ccc|c} 1 & 2 & 2 & 1 \\ 3 & 7 & 5 & 3 \\ 0 & 6 & 8 & -14 \end{array}\right]$$

Gaussian Elimination (cont.)

- Go back to original system. Step 1 must eliminate x_1 from equations 2 and 3, i.e.,
- It must transform:

$$\left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 1 & 3 & 1 & 1 \\ 1 & 5 & 6 & -6 \end{array}\right] \text{ into: } \left[\begin{array}{ccc|c} * & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \end{array}\right]$$

$row_2 := row_2 - \frac{1}{2} \times row_1$:

$$\left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 1 & 5 & 6 & -6 \end{array}\right]$$

$row_3 := row_3 - \frac{1}{2} \times row_1$:

$$\left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 3 & 4 & -7 \end{array}\right]$$

➤ Step 2 must now transform:

$$\left[\begin{array}{ccc|c} \mathbf{2} & \mathbf{4} & \mathbf{4} & \mathbf{2} \\ \mathbf{0} & \mathbf{1} & \mathbf{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{3} & \mathbf{4} & \mathbf{-7} \end{array}\right] \text{ into: } \left[\begin{array}{ccc|c} * & * & * & * \\ \mathbf{0} & * & * & * \\ \mathbf{0} & \mathbf{0} & * & * \end{array}\right]$$

$$\boldsymbol{row_3 := row_3 - 3 \times row_2 :} \rightarrow \left[\begin{array}{ccc|c} \mathbf{2} & \mathbf{4} & \mathbf{4} & \mathbf{2} \\ \mathbf{0} & \mathbf{1} & \mathbf{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{7} & \mathbf{-7} \end{array}\right]$$

- System is now triangular

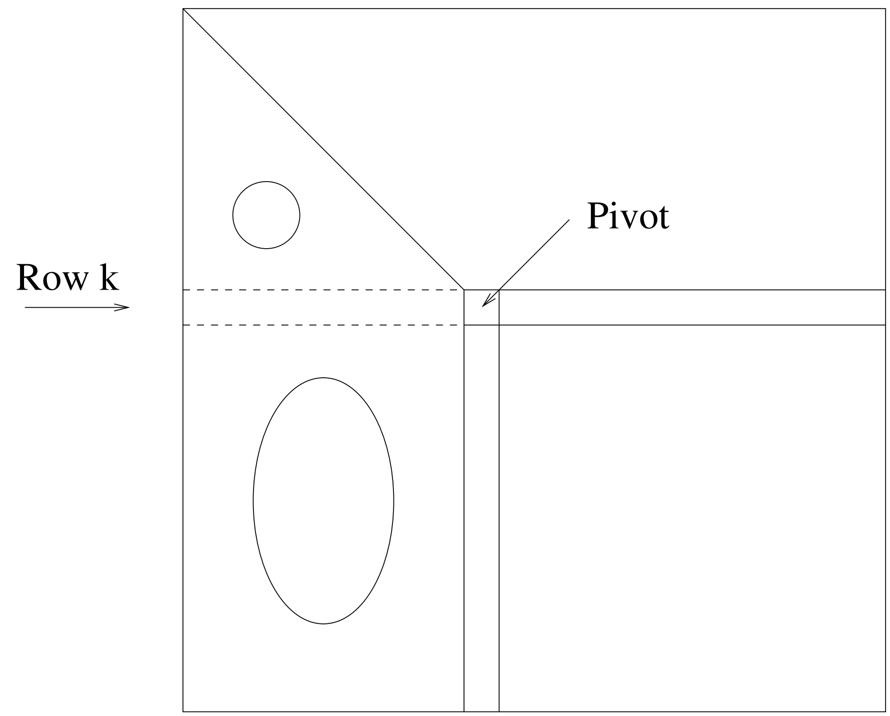
$$\boxed{\left\{\begin{array}{rcl} \boldsymbol{2x_1 + 4x_2 + 4x_3} & \boldsymbol{=} & \mathbf{2} \\ \boldsymbol{x_2 - x_3} & \boldsymbol{=} & \mathbf{0} \\ \boldsymbol{7x_3} & \boldsymbol{=} & \mathbf{-7} \end{array}\right.} \rightarrow \text{Solve}$$

✍ Find the solution of the above triangular system and verify that it is a solution of the original system

Gaussian Elimination: The algorithm

Recall: an algorithm is a sequence of operations (a 'recipe') to be performed by a computer.

- General step of Gaussian elimination :
- At step k subtract multiples of row k from rows $k+1, k+2, \cdots, n$ in order to zero-out entries below a_{kk} in column k.
- Repeat this step for $k = 1, 2, ..., n-1$



$\boxed{\text{Step } k \text{ in words:}}$

for each row i where i runs from $i = k + 1$ to $i = n$ do:
subtract piv * row k from row i (where $piv = a_{ik}/a_{kk}$).

ALGORITHM : 1. *Gaussian Elimination*

1. *For* $k = 1 : n - 1$ *Do:*
2. *For* $i = k + 1 : n$ *Do:*
3. $piv := a_{ik}/a_{kk}$
4. *For* $j := k + 1 : n + 1$ *Do :*
5. $a_{ij} := a_{ij} - piv * a_{kj}$
6. *End*
6. *End*
7. *End*

Matlab Script:

```
  function [x]  = gauss (A, b)
% function [x]  = gauss (A, b)
% solves A x  = b by Gaussian elimination
 n = size(A,1) ;
 A = [A,b];
 for k=1:n-1
    for i=k+1:n
      piv = A(i,k) / A(k,k) ;
      A(i,k+1:n+1)=A(i,k+1:n+1)-piv*A(k,k+1:n+1);
    end
 end
 x = backsolv(A,A(:,n+1));
```

- Input: matrix $\boldsymbol{A}$ and right-hand side $\boldsymbol{b}$. Output: solution $\boldsymbol{x}$.
- Invokes `backsolv.m` to solve final triangular system.

Gaussian Elimination: Pivoting

Consider again Gaussian Elimination for the linear system

$$\begin{cases} 2x_1 + 2x_2 + 4x_3 = 2 \\ x_1 + x_2 + x_3 = 1 \\ x_1 + 4x_2 + 6x_3 = -5 \end{cases} \quad \text{Or:} \quad \left[\begin{array}{ccc|c} 2 & 2 & 4 & 2 \\ 1 & 1 & 1 & 1 \\ 1 & 4 & 6 & -5 \end{array}\right]$$

$row_2 := row_2 - \frac{1}{2} \times row_1$: $\qquad$ $row_3 := row_3 - \frac{1}{2} \times row_1$:

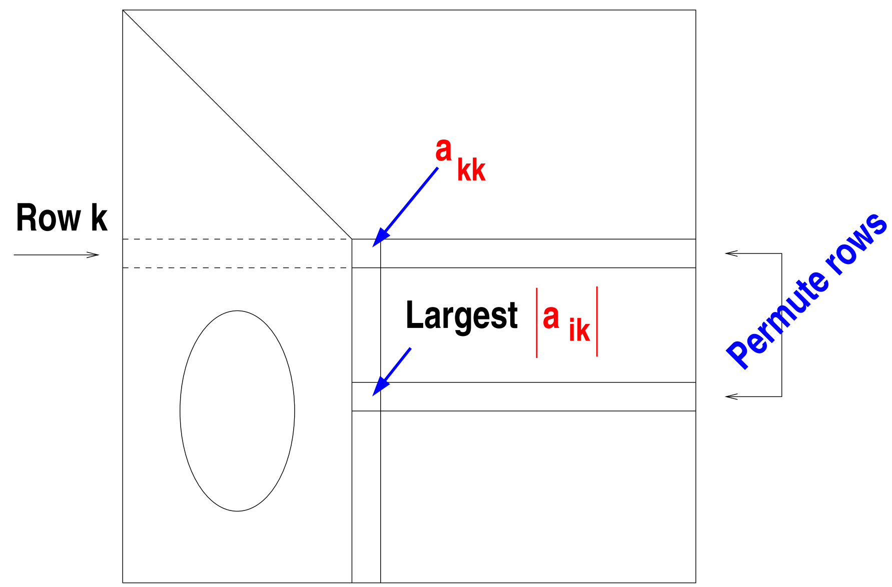
$$\left[\begin{array}{ccc|c} 2 & 2 & 4 & 2 \\ 0 & 0 & -1 & 0 \\ 1 & 4 & 6 & -5 \end{array}\right] \qquad \left[\begin{array}{ccc|c} 2 & 2 & 4 & 2 \\ 0 & 0 & -1 & 0 \\ 0 & 3 & 4 & -6 \end{array}\right]$$

➤ Pivot a_{22} is zero. Solution : permute rows 2 and 3 $\longrightarrow$

$$\left[\begin{array}{ccc|c} 2 & 2 & 4 & 2 \\ 0 & 3 & 4 & -6 \\ 0 & 0 & -1 & 0 \end{array}\right]$$

Gaussian Elimination: Partial Pivoting

General situation



- Partial Pivoting: *Always* Permute row k with row l such that

$$|a_{lk}| = \max_{i=k,\ldots,n} |a_{ik}|$$

- More 'stable' algorithm.

Gauss-Jordan Elimination

Principle of the method: We will now transform the system into one that is even easier to solve than a triangular system, namely a diagonal system. The method is very similar to Gaussian Elimination. It is just a bit more expensive.

Back to original system (P. 2-2). Step 1 must transform:

$$\left[\begin{array}{ccc|c} \mathbf{2} & \mathbf{4} & \mathbf{4} & \mathbf{2} \\ \mathbf{1} & \mathbf{3} & \mathbf{1} & \mathbf{1} \\ \mathbf{1} & \mathbf{5} & \mathbf{6} & \mathbf{-6} \end{array}\right] \text{ into: } \left[\begin{array}{ccc|c} \boldsymbol{x} & \boldsymbol{x} & \boldsymbol{x} & \boldsymbol{x} \\ \mathbf{0} & \boldsymbol{x} & \boldsymbol{x} & \boldsymbol{x} \\ \mathbf{0} & \boldsymbol{x} & \boldsymbol{x} & \boldsymbol{x} \end{array}\right]$$

- Same step as 1st step of Gaussian Elimination.

$\boldsymbol{row_2 := row_2 - 0.5 \times row_1}$: $\boldsymbol{row_3 := row_3 - 0.5 \times row_1}$:

Step 1:

$$\left[\begin{array}{rrr|r} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 1 & 5 & 6 & -6 \end{array}\right] \qquad \left[\begin{array}{rrr|r} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 3 & 4 & -7 \end{array}\right]$$

Must now transform: $\left[\begin{array}{rrr|r} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 3 & 4 & -7 \end{array}\right]$ into: $\left[\begin{array}{ccc|c} x & 0 & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \end{array}\right]$

$\boldsymbol{row_1 := row_1 - 4 \times row_2}$: $\boldsymbol{row_3 := row_3 - 3 \times row_2}$:

Step 2:

$$\left[\begin{array}{rrr|r} 2 & 0 & 8 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 3 & 4 & -7 \end{array}\right] \qquad \left[\begin{array}{rrr|r} 2 & 0 & 8 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 7 & -7 \end{array}\right]$$

There is now a third step:

To transform: $\left[\begin{array}{ccc|c} 2 & 0 & 8 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 7 & -7 \end{array}\right]$ into: $\left[\begin{array}{ccc|c} x & 0 & 0 & x \\ 0 & x & 0 & x \\ 0 & 0 & x & x \end{array}\right]$

$row_1 := row_1 - \frac{8}{7} \times row_3$: $\quad row_2 := row_2 - \frac{-1}{7} \times row_3$:

Step 3:

$\left[\begin{array}{ccc|c} 2 & 0 & 0 & 10 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 7 & -7 \end{array}\right]$ $\quad$ $\left[\begin{array}{ccc|c} 2 & 0 & 0 & 10 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 7 & -7 \end{array}\right]$

Final System:

$$\begin{cases} 2x_1 & = 10 \\ x_2 & = -1 \\ 7x_3 & = -7 \end{cases}$$

Solution:

$$x_1 = 5, \quad x_2 = -1, \quad x_3 = -1$$

Gauss-Jordan - variants

Common variant: Before an elimination step is started divide the row by diagonal entry a_{kk}

➤ At the end all diagonal entries are ones $\longrightarrow$ solution $=$ rhs

✍ Redo the previous example with this variant.

✍ Is this more or less costly than the original method?

> NOTE: unless otherwise specified Gauss-Jordan will refer to this scaled version.

➤ Also: Pivoting can be implemented just like Gaussian elimination.

Important: Never swap a pivot row with a row above it! (destroys structure)

```
 function x = gaussj (A, b)
%---------------------------------------------
% function x = gaussj (A, b)
% solves A x  = b by Gauss-Jordan elimination
% this version scales rows.
%---------------------------------------------
n = size(A,1) ;
A = [A,b] ;
for k=1:n
   A(k,k:n+1) = A(k,k:n+1)/A(k,k);
   for i=1:n
       if (i ~= k)
           piv = A(i,k) ;
           A(i,k:n+1)=A(i,k:n+1)-piv*A(k,k:n+1);
       end
   end
end
x = A(:,n+1);
```

Linear systems – summary of complexity results

➤ The number of operations needed to solve a triangular linear system with n unknowns is

$$C_T(n) = n^2$$

➤ The number of operations required to solve a linear system with n unknowns by Gaussian elimination is

$$C_G(n) \approx \frac{2}{3}n^3$$

➤ The number of operations required to solve a linear system with n unknowns by Gauss-Jordan elimination is

$$C_{GJ}(n) \approx n^3$$

➤ Note: remember that Gauss-Jordan costs 50% more than Gauss.