Gaussian Elimination – Example 1

Consider a full 2×2 linear system to start with:

$$\begin{aligned} x_1 &+ 3x_2 = 5 \\ 2x_1 &+ 4x_2 = 6 \end{aligned} \qquad (Ax = b)$$

Step 1: **Reduce $Ax = b$ to an upper triangular system $Ux = z$.**

- **Subtract 2 times equation 1 from equation 2:** $r_2 - 2r_1 \rightarrow r_2$ (r_i = "row i")

  $r_2 - 2r_1 \Rightarrow r_2 \Rightarrow 0 - 2x_2 = -4$

  $x_2 = 2$

- **Notice that only the second equation changed.**
- **Forget what you learned in other courses about Gaussian Elimination:**
  - * **DO NOT eliminate the upper triangular part!**
  - * **DO NOT scale the last equation by $-\frac{1}{2}$!**
  - * **DO NOT perform any other row/column ops!** (even if you know they're valid)

Step 2: **Once system is in upper triangular form, solve using backward substitution:**

$x_2 = 2$

$x_1 = -1$

Gaussian Elimination – Example 2

This time, use augmented matrix notation to solve $Ax = b$ where $A = \begin{bmatrix} -3 & 2 & -1 \\ 6 & -6 & 7 \\ 3 & -4 & 4 \end{bmatrix}$ and $b = [-1,\ -7,\ -6]^T$.

$m_{21} r_1 \to 6 \quad -4 \quad 2 \quad 1 \quad 2$

Step 1: Reduce to upper triangular form:

- Write as an equivalent augmented system:

$$[\,A \,\vdots\, b\,] = \left[\begin{array}{ccc|c} -3 & 2 & -1 & -1 \\ 6 & -6 & 7 & -7 \\ 3 & -4 & 4 & -6 \end{array}\right]$$

(−3 is called the pivot element)

- Use row $i = 1$ to eliminate (zero out) the sub-diagonal entries in column $j = 1$:
  (a) compute multipliers $m_{ij} = \frac{a_{ij}}{a_{jj}}$: $\quad m_{21} = \frac{6}{-3} = -2 \quad , \quad m_{31} = \frac{3}{-3} = -1$
  (b) perform elementary row operations: $\quad r_i - m_{ij} r_j \to r_i \quad$ (for $j = i+1, \ldots, n$)

$r_2 - m_{21} r_1 \to r_2$
$r_3 - m_{31} r_1 \to r_3$

$$\left[\begin{array}{ccc|c} -3 & 2 & -1 & -1 \\ 0 & -2 & 5 & -9 \\ 0 & -2 & 3 & -7 \end{array}\right]$$

- For column $j = 2$, use row 2 and multiplier $m_{32} = \frac{-2}{-2} = 1$ to zero out the last remaining sub-diagonal entry:

$$r_3 - m_{32} r_2 \to r_3 \qquad \left[\begin{array}{ccc|c} -3 & 2 & -1 & -1 \\ 0 & -2 & 5 & -9 \\ 0 & 0 & -2 & 2 \end{array}\right] \Longrightarrow \text{upper triangle Matrix}$$

Step 2: Perform back substitution to solve $Ux = z$:

$$x_3 = \frac{2}{-2} = -1$$

$$x_2 = \frac{[-9 - 5(-1)]}{-2} = 2 \implies x = \begin{bmatrix} 2 \\ 2 \\ -1 \end{bmatrix}$$

Check $Ax=b$ to verify solution

$$x_1 = \frac{-1 + (-1) - 2(2)}{-3} = 2$$

Recall: Outcomes of Row Reduction

- **When row-reducing an augmented matrix there are three possible outcomes:**

1. $\left[\begin{array}{ccc|c} -2 & -3 & 1 & -1 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 8 & -5 \end{array}\right]$

det $= -16 \neq 0$

Original system is solvable
$\Longrightarrow$ **unique solution**

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

det $= 0$
Last row: "$0 =$ nonzero"

Matrix is singular and RHS is inconsistent
$\Longrightarrow$ **no solution**

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

det $= 0$
Last row: "$0 = 0$"

Matrix is singular and RHS is consistent
$\Longrightarrow$ **infinitely many solutions**

- **In MACM 316, we're mainly interested in "uniquely solvable" problems (like 1).**
- **But any GE algorithm should still "gracefully" handle special cases (like 2,3).**

Gaussian Elimination – Example 3

Another 3×3 example with $A = \begin{bmatrix} 2 & 1 & 4 \\ 3 & 2 & 1 \\ -1 & 4 & -2 \end{bmatrix}$ and $b = \begin{bmatrix} 7 \\ 1 \\ 1 \end{bmatrix}$

Step 1: Row-reduce to upper triangular form.

- Rewrite as an augmented system:

Pivot

$$[A \mid b] = \left[\begin{array}{ccc|c} 2 & 1 & 4 & 7 \\ 3 & 2 & 1 & 1 \\ -1 & 4 & -2 & 1 \end{array}\right] \quad \begin{array}{l} m_{21} = \frac{3}{2} \\ m_{31} = -\frac{1}{2} \end{array}$$

- Eliminate column 1 entries **BUT** this time store multipliers in place of the sub-diagonal zeroes (why? later ...)

Do Not Scale r_2 by 2

$$\begin{array}{l} r_2 - \frac{3}{2} r_1 \longrightarrow r_2 \\ r_3 - (-\frac{1}{2}) r_1 \longrightarrow r_3 \end{array} \quad \left[\begin{array}{ccc|c} 2 & 1 & 4 & 7 \\ (\frac{3}{2}) & \frac{1}{2} & -5 & -\frac{19}{2} \\ (-\frac{1}{2}) & \frac{9}{2} & 0 & \frac{9}{2} \end{array}\right] \quad m_{32} = \frac{9/2}{1/2} = 9$$

– Storing the multiplier m_{ij} for later, do not use for back substitution.

- Eliminate remaining entry in column 2:

Do Not Scale r_3 by $\frac{1}{45}$

$$r_3 - 9 r_2 \longrightarrow r_3 \quad \left[\begin{array}{ccc|c} 2 & 1 & 4 & 7 \\ (\frac{3}{2}) & \frac{1}{2} & -5 & -\frac{19}{2} \\ (-\frac{1}{2}) & (9) & 45 & 90 \end{array}\right]$$

Step 2: **Finish with backward substitution, ignoring the (m_{ij})**

$x_3 = 2$

$x_2 = 2\left(-\frac{19}{2} + 5(2)\right) = 1$

$x_1 = -1$

$x = \begin{bmatrix} -1 \\ 1 \\ 2 \end{bmatrix}$

. . . this is looking more and more like an algorithm!

Gaussian Elimination (GE) Algorithm

Pseudo-code for row reduction step:

```
for j = 1 : n-1 do
  if A(j,j) = 0 then
    ERROR: Division by zero!
  endif

  % Zero out sub-diagonal entries in col j
  for i = j+1 : n do
    m(i,j) = A(i,j) / A(j,j)    % multiplier

    % Row operation r_i - m_ij*r_j -> r_i
    A(i,j) = 0
    for p = j+1 : n do
      A(i,p) = A(i,p) - m(i,j) * A(j,p)
    enddo
    b(i) = b(i) - m(i,j) * b(j)
  enddo
enddo
if A(n,n) = 0 then
  ERROR: Division by zero!
endif
```

Input:

- `n` : number of unknowns
- `A(1:n,1:n)` : matrix entries
- `b(1:n)` : RHS vector

Output:

- `x(1:n)` : solution vector

Backward substitution step:

```
x(n) = b(n) / A(n,n)
for i = n-1 : -1 : 1 do
  x(i) = b(i)
  for j = i+1 : n do
    x(i) = x(i) - A(i,j) * x(j)
  enddo
  x(i) = x(i) / A(i,i)
enddo
return x
```

Note: **Could replace** `m(i,j)` **with** `A(i,j)`**! (store multipliers in** `A`**)**

Operation Count for GE

Count total number of multiply / divide operations (neglect $+/-$):

[Note: Textbook counts **all** operations: $+, -, \times, \div$]

- **Cost of row reduction phase, $\mathcal{R}(n)$:**

$$\mathcal{R}(n) = 2\sum_{j=1}^{n-1} j + \sum_{j=1}^{n-1} j^2$$

$$= \frac{2n(n-1)}{2} + \frac{n(n-1)(2n-1)}{6}$$

$$= \frac{2n^3+3n^2-5n}{6} \quad = O(n^3)$$

Column j	Multipliers and updating RHS	Updating row entries
1	$n-1$	$(n-1)^2$
2	$n-2$	$(n-2)^2$
$\vdots$	$\vdots$	$\vdots$
$n-2$	2	2^2
$n-1$	1	1^2
Total	$2\sum_{j=1}^{n-1} j$	$\sum_{j=1}^{n-1} j^2$

- **Cost of backward substitution, $\mathcal{B}(n)$:**

$$\mathcal{B}(n) = 1 + (n-1) + \sum_{j=1}^{n-1} j = \frac{n^2+n}{2} \quad = O(n^2)$$

- **Total operation count:**

$$\mathcal{T}(n) = \mathcal{R}(n) + \mathcal{B}(n) = \frac{n^3+3n^2-n}{3} \quad = O(n^3)$$

Question: What fraction of total time is spent on row reduction?

ASIDE: Big-O Notation and Computational Complexity

- When comparing algorithms, big-O notation becomes extremely useful
- Suppose a problem has some size n (for n large) then the total cost or complexity of an algorithm may be nicely represented by an expression like:

$$O(n), \quad O(n^p), \quad O(n^2 \log(n)), \quad \text{etc.}$$

- These rates of growth in cost are studied in the next set of lectures (3c)
- Here are a few references on big-O notation for large-n growth:
  - * http://youtu.be/__vX2sjlpXU : a 5-minute introduction to $O(n^p)$ type costs for simple algorithm components (warning: n and N are often swapped)
  - * http://bigocheatsheet.com : common CS algorithms compared