Systems of Linear Eqns

CSS455 – Winter 2012
Ch7 of Turner

Systems of Linear Eqns are Common in Scientific Applications

- System of 2 eqns and 2 unknowns is given by:

$$a_{11}x_1 + a_{12}x_2 = b_1$$
$$a_{21}x_1 + a_{22}x_2 = b_2$$

The *a's* and *b's* are known, perhaps from a measurement or from theory.

- Can be generalized to *m* equations, *n* unknowns.
- Can be represented by matrix eqn: **Ax = b**, where

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \text{ and } \quad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

Consider a 2x2 example:

$$\left.\begin{aligned} 2x_1 + 3x_2 &= 8 \text{ (eqn 1)} \\ 5x_1 + 4x_2 &= 13 \text{ (eqn 2)} \end{aligned}\right\} \textit{Example}$$

Activity 5, Part I: With your partner, solve this system and keep track of how you do it.

Consider a 2x2 example:

$$\left.\begin{aligned} 2x_1 + 3x_2 &= 8 \;\; (\text{eqn } 1) \\ 5x_1 + 4x_2 &= 13 \;\; (\text{eqn } 2) \end{aligned}\right\} \textit{Example}$$

First Approach: step-wise solution and substitution:

Solve Eqn 1 for x_2: $x_2 = (8-2x_1)/3$

Substitute x_2 into Eqn 2: $5x_1 + 4\{(8-2x_1)/3\} = 13$

Solve for x_1:

$5x_1 + (32/3) - (8/3)x_1 = 13$

$x_1 = 1$

Consider a 2x2 example:

$$\left.\begin{aligned} 2x_1 + 3x_2 &= 8 \;\; (\text{eqn } 1) \\ 5x_1 + 4x_2 &= 13 \;\; (\text{eqn } 2) \end{aligned}\right\} \textit{Example}$$

First Approach: step-wise solution and substitution:

Substitute x_1 into

Solve for x_2:

$3x_2 = 6$

$x_2 = 2$

It becomes much more complex as the number of unknowns increases. It is still deterministic. The algebra becomes somewhat complex.

Substitute x_1 into Eqn 1

Solve for x_2

Consider a 2x2 example:

$$\left.\begin{aligned} 2x_1 + 3x_2 &= 8 \;\; (\text{eqn } 1) \\ 5x_1 + 4x_2 &= 13 \;\; (\text{eqn } 2) \end{aligned}\right\} \textit{Example}$$

Second approach: manipulate entire equations: adding, subtracting, scaling in order to eliminate variables from the equations.

(3)Eqn 2 – (4) Eqn 1:

$15x_1 - 8x_1 + 12x_2 - 12x_2 = 3(13) - (4)(8) = 7$

$(7)x_1 = 7$

Multiply by (1/7): $x_1 = 1$

Overall transformation: (3/7)Eqn2 – (4/7)Eqn1

Consider a 2x2 example:

$$\left.\begin{aligned} 2x_1 + 3x_2 &= 8 \ \ (\text{eqn } 1) \\ 5x_1 + 4x_2 &= 13 \ \ (\text{eqn } 2) \end{aligned}\right\} \textit{Example}$$

To eliminate x_1, subtract 2 times Eqn2 from 5 times Eqn 1:

(5)Eqn 1 – (2) Eqn 2:

$10x_1 - 10x_1 + 15x_2 - 8x_2 = 5(8) - (2)(13) = 14$

$(7)x_2 = 14$

Multiply by (1/7): $x_2 = 2$

Overall transformation: (5/7)Eqn1 – (2/7)Eqn2

Consider a 2x2 example:

$$\left.\begin{aligned} 2x_1 + 3x_2 &= 8 \ \ (\text{eqn } 1) \\ 5x_1 + 4x_2 &= 13 \ \ (\text{eqn } 2) \end{aligned}\right\} \textit{Example}$$

Form two new equations:

(-4/7)(#1) + (3/7) (#2) and

(5/7)(#1) -(2/7)(#2)

$$\left.\begin{aligned} x_1 &= 1 \, (\text{eqn } 1) \\ x_2 &= 2 \ \ (\text{eqn } 2) \end{aligned}\right\} \textit{Example}$$

Consider a 2x2 example:

- **Ax=b**

$$\left.\begin{aligned} 2x_1 + 3x_2 &= 8 \ \ (\text{eqn } 1) \\ 5x_1 + 4x_2 &= 13 \ \ (\text{eqn } 2) \end{aligned}\right\} \mathbf{A} = \begin{pmatrix} 2 & 3 \\ 5 & 4 \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} 8 \\ 13 \end{pmatrix}$$

Form two new equations:

(-4/7)(#1) + (3/7) (#2) and

(5/7)(#1) -(2/7)(#2)

$$\mathbf{M} = \begin{pmatrix} -4/7 & 3/7 \\ 5/7 & -2/7 \end{pmatrix}$$

Multiply **Ax**=**b** by **M** on both sides

- **MAx = Mb** or

$$\begin{pmatrix} \frac{-4}{7} & \frac{3}{7} \\ \frac{5}{7} & \frac{-2}{7} \end{pmatrix}\begin{pmatrix} 2 & 3 \\ 5 & 4 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \frac{-4}{7} & \frac{3}{7} \\ \frac{5}{7} & \frac{-2}{7} \end{pmatrix}\begin{pmatrix} 8 \\ 13 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

$$(1)x_1 + (0)x_2 = 1$$
$$(0)x_1 + (1)x_2 = 2$$

MA has been transformed to diagonal form (unit matrix)

What are the two equations?

Multiply **Ax**=**b** by **M** on both sides

- **MAx = Mb** or

$$\begin{pmatrix} \frac{-4}{7} & \frac{3}{7} \\ \frac{5}{7} & \frac{-2}{7} \end{pmatrix}\begin{pmatrix} 2 & 3 \\ 5 & 4 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \frac{-4}{7} & \frac{3}{7} \\ \frac{5}{7} & \frac{-2}{7} \end{pmatrix}\begin{pmatrix} 8 \\ 13 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

MA has been transformed to diagonal form (unit matrix)

In general, premultiplication of **A** by a nonsingular **M** represents scaling, combination, and permutation of the equations. (all valid operations)

Find a matrix **M**, such that the product **MA** is diagonal, and the problem is solved.

Activity 5: Part II

Consider a 2x2 example:

$$\left.\begin{aligned} 2x_1 + 3x_2 &= 8 \;\; (\text{eqn } 1) \\ 5x_1 + 4x_2 &= 13 \;\; (\text{eqn } 2) \end{aligned}\right\} Example$$

In a combination of the two approaches, find a combination of equations that results in a triangular system, and then do routine substitution.

Replace Eqn 2 with a new one that has x_1 eliminated:

(5)(#1) + (-2) (#2)

$$\left.\begin{aligned} 2x_1 + 3x_2 &= 8 \;\; (\text{eqn } 1) \\ (10-10)x_1 + (15-8)x_2 &= 7x_2 = 5(8) - 2(13) = 14 \end{aligned}\right\}$$

$$\left.\begin{aligned} 2x_1 + 3x_2 &= 8 \\ 7x_2 &= 14 \end{aligned}\right\} \quad \begin{pmatrix} 2 & 3 \\ 0 & 7 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 8 \\ 14 \end{pmatrix}$$

Consider a 2x2 example:

$$\left.\begin{aligned} 2x_1 + 3x_2 &= 8 \;\; (\text{eqn } 1) \\ 5x_1 + 4x_2 &= 13 \;\; (\text{eqn } 2) \end{aligned}\right\} Example$$

$$\begin{pmatrix} 2 & 3 \\ 0 & 7 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 8 \\ 14 \end{pmatrix} \quad \left.\begin{aligned} 2x_1 + 3x_2 &= 8 \\ 7x_2 &= 14 \end{aligned}\right\}$$

Upper Triangular

Transform to upper triangular system

Solve eqn 2 for x_2 and substitute into Eqn 1.

Solve Eqn 1 for x_1

Etc.

Consider a 2x2 example:

$$\left.\begin{aligned} 2x_1 + 3x_2 &= 8 \;\; (\text{eqn } 1) \\ 5x_1 + 4x_2 &= 13 \;\; (\text{eqn } 2) \end{aligned}\right\} Example$$

Rewrite the equations:

$$\left.\begin{aligned} x_1 &= \tfrac{1}{2}(8 - 3x_2) = 4 - \tfrac{3}{2}x_2 \;\; (\text{eqn}1) \\ x_2 &= \tfrac{1}{4}(13 - 5x_1) = \tfrac{13}{4} - \tfrac{5}{4}x_1 \;\; (\text{eqn}2) \end{aligned}\right\}$$

Consider an iteratived approach:

$$\left.\begin{aligned} x_1^{[n+1]} &= 4 - \tfrac{3}{2}x_2^{[n]} \;\; (\text{eqn } 1) \\ x_2^{[n+1]} &= \tfrac{13}{4} - \tfrac{5}{4}x_1^{[n]} \;\; (\text{eqn } 2) \end{aligned}\right\}$$

Under what conditions would you expect this to be most likely to succeed?

P_{11}	$P_{12}=0$	$P_{13}=0$	$P_{14}=0$	$P_{15}=0$	$P_{16}=0$	P_{17}
$P_{21}=1$	P_{22}	P_{23}	P_{24}	P_{25}	P_{26}	$P_{27}=0$
$P_{31}=1$	P_{32}	P_{33}	P_{34}	P_{35}	P_{36}	$P_{37}=0$
$P_{41}=1$	P_{42}	P_{43}	P_{44}	P_{45}	P_{46}	$P_{47}=0$
$P_{51}=1$	P_{52}	P_{53}	P_{54}	P_{55}	P_{56}	$P_{57}=0$
$P_{61}=1$	P_{62}	P_{63}	P_{64}	P_{65}	P_{66}	$P_{67}=0$
$P_{71}=1$	P_{72}	P_{73}	P_{74}	P_{75}	P_{76}	$P_{77}=0$
P_{81}	$P_{82}=0$	$P_{83}=0$	$P_{84}=0$	$P_{85}=0$	$P_{86}=0$	P_{87}

Rats in a Maze

- Green cells have food
- Red cells do not
- Yellow cells cannot be reached.
- Rats cannot return from red or green cells.
- For any white cell, the probability of finding food is the average of the four neighbors (why?)

P_{11}	$P_{12}=0$	$P_{13}=0$	$P_{14}=0$	$P_{15}=0$	$P_{16}=0$	P_{17}
$P_{21}=1$	P_{22}	P_{23}	P_{24}	P_{25}	P_{26}	$P_{27}=0$
$P_{31}=1$	P_{32}	P_{33}	P_{34}	P_{35}	P_{36}	$P_{37}=0$
$P_{41}=1$	P_{42}	P_{43}	P_{44}	P_{45}	P_{46}	$P_{47}=0$
$P_{51}=1$	P_{52}	P_{53}	P_{54}	P_{55}	P_{56}	$P_{57}=0$
$P_{61}=1$	P_{62}	P_{63}	P_{64}	P_{65}	P_{66}	$P_{67}=0$
$P_{71}=1$	P_{72}	P_{73}	P_{74}	P_{75}	P_{76}	$P_{77}=0$
P_{81}	$P_{82}=0$	$P_{83}=0$	$P_{84}=0$	$P_{85}=0$	P =	

Rats in a Maze

For white cells:

$$P_{i,j} = \left(\frac{1}{4}\right)\left(P_{i-1,j} + P_{i+1,j} + P_{i,j-1} + P_{i,j+1}\right)$$

P_{11}	$P_{12}=0$	$P_{13}=0$	$P_{14}=0$	$P_{15}=0$	$P_{16}=0$	P_{17}
$P_{21}=1$	P_{22}	P_{23}	P_{24}	P_{25}	P_{26}	$P_{27}=0$
$P_{31}=1$	P_{32}	P_{33}	P_{34}	P_{35}	P_{36}	$P_{37}=0$
$P_{41}=1$	P_{42}	P_{43}	P_{44}	P_{45}	P_{46}	$P_{47}=0$
$P_{51}=1$	P_{52}	P_{53}	P_{54}	P_{55}	P_{56}	$P_{57}=0$
$P_{61}=1$	P_{62}	P_{63}	P_{64}	P_{65}	P_{66}	$P_{67}=0$
$P_{71}=1$	P_{72}	P_{73}	P_{74}	P_{75}	P_{76}	$P_{77}=0$
P_{81}	$P_{82}=0$	$P_{83}=0$	$P_{84}=0$	$P_{85}=0$	$P_{86}=0$	

Rats in a Maze

For cell (5,4)

$$P_{5,4} = \left(\frac{1}{4}\right)\left(P_{4,4} + P_{6,4} + P_{5,3} + P_{5,5}\right)$$

Gauss Seidel Method for finding $P_{i,j}$

- Assign those P-values that are known.
- Guess at the initial values for the other P's
- Do an iterative updating of the internal P's until they converge (if they do):

$$\textit{for} \quad \textit{each } i \textit{ and } j$$
$$P_{i,j}^{new} = \left(\frac{1}{4}\right)\left(P_{i-1,j}^{old} + P_{i+1,j}^{old} + P_{i,j-1}^{old} + P_{i,j+1}^{old}\right)$$
$$\textit{update} \quad P_{i,j}^{old} = P_{i,j}^{new}$$

Gauss Seidel Method for finding $P_{i,j}$

- For the (8 x 7) maze, there would be 56 such equations
- 26 of them are constants (4 being corners)
- 30 must be iterated as below (30 eqs, 30 unks)

$$\textit{for} \quad \textit{each } i \textit{ and } j$$
$$P_{i,j}^{new} = \left(\frac{1}{4}\right)\left(P_{i-1,j}^{old} + P_{i+1,j}^{old} + P_{i,j-1}^{old} + P_{i,j+1}^{old}\right)$$
$$\textit{update} \quad P_{i,j}^{old} = P_{i,j}^{new}$$

Jacobi Method for finding $P_{i,j}$

- Assign those P-values that are known.
- Guess at the initial values for the other P's
- Do an iterative updating of the internal P's until they converge (if they do):

$$\textit{for} \quad \textit{each } i \textit{ and } j$$
$$P_{i,j}^{new} = \left(\frac{1}{4}\right)\left(P_{i-1,j}^{old} + P_{i+1,j}^{old} + P_{i,j-1}^{old} + P_{i,j+1}^{old}\right)$$
$$\text{update at end of cycle} \quad P^{old} = P^{new}$$

Is this a system of linear equations?

$$\mathbf{Ap} = \mathbf{b} \qquad \mathbf{Ap} = \mathbf{b}$$

$$p = \begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_N \end{pmatrix} = \begin{pmatrix} \vdots \\ p_{i,j} \\ \vdots \\ \vdots \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{pmatrix} = \begin{pmatrix} \vdots \\ b_{i,j} \\ \vdots \\ \vdots \end{pmatrix}$$

$$P_{5,4} - \left(\frac{1}{4}\right)\left(P_{4,4} + P_{6,4} + P_{5,3} + P_{5,5}\right) = 0$$

$$\mathbf{Ap} = \mathbf{b}$$

$$\begin{pmatrix} \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & & & & & & & \cdots \\ \cdots & & & & & & & \cdots \\ 0 & -.25 & 0 & 1 & 0 & -.25 & -.25 & -.25 \\ \cdots & & & & & & & \cdots \\ \cdots & & & & & & & \cdots \\ \cdots & & & & & & & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{pmatrix} \begin{pmatrix} \vdots \\ p_{4,4} \\ \vdots \\ p_{5,4} \\ \vdots \\ p_{6,4} \\ p_{5,5} \\ p_{5,3} \end{pmatrix} = \begin{pmatrix} \vdots \\ \vdots \\ 0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{pmatrix}$$

$$P_{1,4} = 1$$

$$\mathbf{Ap} = \mathbf{b}$$

$$\begin{pmatrix} \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & & & & & & & \cdots \\ \cdots & & & & & & & \cdots \\ \cdots & & & & & & & \cdots \\ \cdots & & & & & & & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{pmatrix} \begin{pmatrix} \vdots \\ \vdots \\ p_{1,4} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ 1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{pmatrix}$$

$\mathbf{Ap} = \mathbf{b}$

$$\begin{pmatrix} \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & & & & & & & \cdots \\ \cdots & & & & & & & \cdots \\ 0 & -.25 & 0 & 1 & 0 & -.25 & -.25 & -.25 \\ \cdots & & & & & & & \cdots \\ \cdots & & & & & & & \cdots \\ \cdots & & & & & & & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{pmatrix} \begin{pmatrix} \vdots \\ p_{4,4} \\ \vdots \\ p_{5,4} \\ \vdots \\ p_{6,4} \\ p_{5,5} \\ p_{5,3} \end{pmatrix} = \begin{pmatrix} \vdots \\ \vdots \\ 0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{pmatrix}$$

There are 56 eqs for the $P_{i,j}$
Rows of A have only 5 nonzero elements at most: sparse
26 rows are diagonal and 30 are more general.
Can be reduced from start to 30 equations-30 unknowns

Two MP2 Issues

- Convergence criterion changed to absolute rather than relative. Why?
- Turner gives a nice algorithm for Gauss Seidel in matrix notation on page 236-37. Whereas I have suggested an algorithm where each iteration explicitly loops over the rows, he suggests one in which this work is in a matrix/vector product.
- Why might we favor one rather than other?

One approach to solving the system

- **Ax=b**
- Find the inverse $\mathbf{A}^{-1}$ of the matrix **A** and multiply both sides by it.
  $\mathbf{A}^{-1}\mathbf{A}\mathbf{x}=\mathbf{A}^{-1}\mathbf{b}$ or $\mathbf{I}\mathbf{x} = \mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$
- In the example, $\mathbf{A}^{-1}$ is found by the *inv* function of Matlab to be:

$$\begin{pmatrix} -.5714 & .4286 \\ .7143 & -.2857 \end{pmatrix}$$

The same matrix we found by elementary row manipulations before.

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

This approach transforms **A** into diagonal (unit matrix) form.

Issues with the inverse

If A^{-1} is inverse of A *(inv(A))*,

$$\mathbf{AA}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The explicit formation of A^{-1} is computationally expensive and numerically problematic. In most cases, we don't need this explicit inverse of the matrix.

Triangular Linear Systems

- Via the inverse of **A** , **A** was transformed to diagonal form. This is computationally difficult and not necessary.
- Consider an **A** that is upper triangular.

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

- No further transformation is necessary.
- ***Backward substitution***.
  - Start with Eq 3: $a_{33}x_3 = b_3$
  - Solve 3d equation : $x_3 = b_3/a_{33}$.
  - Substitute the value of x_3 into the 2d eqn:

$$a_{22}x_2 + a_{23}x_3 = b_2$$
$$x_2 = (b_2 - a_{23}x_3)/a_{22}$$
$$x_2 = \left(b_2 - a_{23}\left(\tfrac{b_3}{a_{33}}\right)\right)/a_{22}$$

- Now we know x_3 and x_2.
  - Solve Eq 1 for x_1:

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1$$
$$a_{11}x_1 = (b_1 - a_{12}x_2 - a_{13}x_3)$$
$$x_1 = (b_1 - a_{12}x_2 - a_{13}x_3)/a_{11}$$

Substitute the previously obtained values of x_2 and x_3

Starting from the last equation, we solve for each x_i *in terms of the previously found values.*

$$x_i = \left(b_i - \sum_{j=i+1}^{n} a_{ij}x_j\right)\Big/ a_{ii}$$

for $i = n\text{-}1,1$

$x_n = b_n / a_{nn}$

Matlab code for backward substitution

- First find the last value: `x(n) = b(n)/a(n,n)`

$$x_i = \left(b_i - \sum_{j=i+1}^{n} a_{ij}x_j\right)\Big/ a_{ii}$$

- Loop over the remaining rows of the matrix:

```
for i = n-1:-1:1
    x(i) = (b(i) - a(i,i+1:n)*x(i+1:n))/a(i,i)
end
```

- Organized as *dot product* between the partial row of **A** and partial vector **x**.

What is the (big *O*) complexity here?

Consider column approach

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1$$
$$a_{22}x_2 + a_{23}x_3 = b_2$$
$$a_{33}x_3 = b_3$$

Solve eqn 3 for x_3

Substitute into Eqns 2 1 and 1, accumulating on the right hand side.

$$a_{11}x_1 + a_{12}x_2 = b_1 - a_{13}x_3 = b_1 - a_{13}(b_3/a_{33})$$
$$a_{22}x_2 = b_2 - a_{23}x_3 = b_2 - a_{23}(b_3/a_{33})$$

$$x_3 = b_3 / a_{33}$$

Solve Eqn 2 for x_2 and substitute into each equation above it.

Saxpy algorithm for backward substitution

- Loop over all except first row of the matrix:

```
for j = n:-1:2
    x(j) = b(j)/a(j,j)
    b(1:j-1) = b(1:j-1) - x(j) *a(1:j-1,j)
end
x(1) = b(1)/a(1,1)
```

- As each x_j is found, it is then subtracted from the rows above j in the matrix, with the sum being accumulated in b_i (a vector operation)

Forward Substitution

- If **A** is *lower* triangular, start with equation 1, solving for x_1 and substitute into the subsequent row in the matrix. *Similar to back subst algorithms.*

$$\mathbf{A} = \begin{pmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

Activity 6, Part I.

How to transform general system **A** to upper triangular

- Begin with 1st column, design an algorithm that zeros out all the elements below A_{11}
- Proceed to 2nd column and zero out all elements beneath A_{22}, *without further changing the 1st column*.
- Continue across all columns of matrix **A**
- Consider the 2nd step, where the first column has been transformed already.

How to transform to triangular form

- Consider the matrix

$$\mathbf{A} = \begin{pmatrix} 9 & 8 & 8 & 9 \\ 0 & 7 & 4 & 7 \\ 0 & 4 & 6 & 1 \\ 0 & 2 & 7 & 4 \end{pmatrix}$$

- To zero out A_{32}, multiply the 2^{nd} row by (-4/7) and add to the third.
- To zero out A_{42}, multiply the 2^{nd} row by (-2/7) and add to the fourth.

These multipliers combine the 2d equation with the 3d and 4th to create zeros in the 2d column.

How to transform to triangular form

- Consider the matrix

$$\mathbf{A} = \begin{pmatrix} 9 & 8 & 8 & 9 \\ 0 & 7 & 4 & 7 \\ 0 & 4 & 6 & 1 \\ 0 & 2 & 7 & 4 \end{pmatrix}$$

- Premultiply by **M**

$$\mathbf{M} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & \frac{-4}{7} & 1 & 0 \\ 0 & \frac{-2}{7} & 0 & 1 \end{pmatrix}$$

These multipliers combine the 2d equation with the 3d and 4th to create zeros in the 2d column.

Activity 6: Part II

- Zeroes out the lower part of column 2.

$$\mathbf{M}\ \mathbf{A} = \begin{pmatrix} 9 & 8 & 8 & 9 \\ 0 & 7 & 4 & 7 \\ 0 & 0 & 3.7143 & -3 \\ 0 & 0 & 5.8571 & 2 \end{pmatrix}$$

- The transformation matrix **M** was a unit matrix,except for elements

  $M_{32}= -A_{32}/A_{22}$

  $M_{42} = -A_{42}/A_{22}$

- Subtract Eqn 2 from Eqns 3 and 4 so as to produce desired zeroes.

Gaussian Elimination

- Proceed across the matrix column by column, and the overall transformation matrix would be the product of those for each column:
- $\mathbf{M} = \mathbf{M_3 M_2 M_1}$
  (M_1 zeroes col 1; M_2 zeroes col 2; M_3 zeroes col 3)
- Multiply both sides of the equation
  $\mathbf{M_3 M_2 M_1 A x} = \mathbf{M_3 M_2 M_1\, b}$
- $\mathbf{M_3 M_2 M_1 A}$ is upper triangular and the system can be solved by backward substitution.

Choice of Pivot elements

- In determining the elimination matrix, elements of the form A_{jk}/A_{kk} were used. A_{kk} is known as the *pivot*.
- If the diagonal element is zero, then the rows of the matrix must be permuted to bring a nonzero pivot into place. (order of the eqns is irrelevant)
- If the pivot is small, the rows should still be permuted to obtain a larger pivot, which reduces rounding error and increases stability.

Rows 2-4 could have been permuted here before transformation

- Consider the matrix **A**

$$\mathbf{A} = \begin{pmatrix} 9 & 8 & 8 & 9 \\ 0 & 7 & 4 & 7 \\ 0 & 4 & 6 & 1 \\ 0 & 2 & 7 & 4 \end{pmatrix} \Rightarrow \mathbf{PA} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 9 & 8 & 8 & 9 \\ 0 & 7 & 4 & 7 \\ 0 & 4 & 6 & 1 \\ 0 & 2 & 7 & 4 \end{pmatrix} = \begin{pmatrix} 9 & 8 & 8 & 9 \\ 0 & 4 & 6 & 1 \\ 0 & 7 & 4 & 7 \\ 0 & 2 & 7 & 4 \end{pmatrix}$$

- In general, stable algorithms employ pivoting, where the column of the matrix is scanned for the largest remaining element to use as the pivot.

Note that the permutation matrix can also be represented by a vector giving the final sequence of rows. Call this vector *piv*.

$$piv = [1 \quad 3 \quad 2 \quad 4]$$
$$\mathbf{PA} = \mathbf{A}(piv,:)$$
$$\mathbf{Pb} = \mathbf{b}(piv)$$

Formally this looks like inversion

- $\mathbf{Ax} = \mathbf{b}$
- $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$, (we have solved for **x**)
- (the inverse is not explicitly formed, however. We form the product $\mathbf{A}^{-1}\mathbf{b}$, *which is a vector*.)
- The explicit inverse is seldom needed.
- $\mathbf{AX} = \mathbf{B}$ (*generalized to several columns*)
- $\mathbf{X} = \mathbf{A}^{-1}\mathbf{B}$
- If the product $\mathbf{A}^{-1}\mathbf{B}$ appears in a formula, do not form $\mathbf{A}^{-1}$ explicitly. Instead, solve the system $\mathbf{AX} = \mathbf{B}$ for $\mathbf{X}$ ($= \mathbf{A}^{-1}\mathbf{B}$) column by column.

LU Factorization

- Suppose **A** can be factored into the product of lower and upper triangular matrices.
- $\mathbf{A} = \mathbf{LU}$ (defer until later how to do this)
- We can write the system to be solved as $\mathbf{LUx} = \mathbf{b}$
- The product **Ux** is a vector and can be denoted **y**: $\mathbf{Ux} = \mathbf{y}$

Solution using L and U

- **LUx= L(Ux) = Ly = b** where **Ux=y**
- Solve **Ly = b** by forward substitution. **y** is in intermediate solution.
- Solve **Ux=y** by backward substitution. **x** is the desired final solution**.**

One possible **LU** Factorization

- Let **MA** be upper triangular by design (we already know how to define **M)**.
- Define **U = MA**
- **LU = LMA=A**, and the process can be viewed as factoring **A** into lower and upper triangular parts.
- **LMA = A (L is inverse of M by definition)**
- We can write the system to be solved as **LUx = b**

How do we find the factors **L** and **U**?

- We have already seen how to transform the matrix from full form to upper triangular.
- $\mathbf{M = M_3M_2M_1}$
- Multiply both sides of the equation $\mathbf{M_3M_2M_1Ax = M_3M_2M_1\,b}$
- Consider the inverse of that transformation.

Inverse Transformations

- Each transformation $\mathbf{M_i}$ has an inverse $\mathbf{M_i^{-1}}$.
- The form of the transformation matrix is

$$\mathbf{M_i} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & n_3 & 1 & 0 & 0 \\ 0 & n_4 & 0 & 1 & 0 \\ 0 & n_5 & 0 & 0 & 1 \end{pmatrix}$$

The multipliers n_i have been chosen to create zeros in the 2d column of **A** by subtracting multiples of the 2d equation from the other ones.

•The inverse of this is of the form

$$\mathbf{M}_i^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & -n_3 & 1 & 0 & 0 \\ 0 & -n_4 & 0 & 1 & 0 \\ 0 & -n_5 & 0 & 0 & 1 \end{pmatrix}$$

Each transformation has a **lower triangular** inverse. $\mathbf{M}_i^{-1} = \mathbf{L}_i$

Inverse Transformations

- Activity 6: Part III

Inverse Transformations

- The overall transformation **M** is the product of the column transformation matrices
- The overall inverse **L** (lower triangular) is the product of individual inverses.

$$\mathbf{M} = \mathbf{M_3 M_2 M_1}$$

$$\mathbf{M^{-1}} = [\mathbf{M_3 M_2 M_1}]^{-1} = \mathbf{M_1^{-1} M_2^{-1} M_3^{-1}} = \mathbf{L_1 L_2 L_3} \equiv \mathbf{L}$$

$$\mathbf{L} = \mathbf{L_1 L_2 L_3}$$

Since **MA** is upper triangular (call it **U**), and **L** is lower triangular and the inverse of **M**: **LMA** = **LU** is the LU factorization of **A**.

- In general, stable algorithms employ pivoting, where the column of the matrix is scanned for the largest remaining element to use as the pivot.
- The effect of the permutations must be accounted for. The permutation matrix **P** is also returned, along with **L** and **U.**

$$\mathbf{Ax} = \mathbf{b}$$
$$\mathbf{LU} = \mathbf{PA}$$
$$\mathbf{LUx} = \mathbf{Pb}$$
$$\mathbf{Ux} = \mathbf{y}$$
$$\mathbf{Ly} = \mathbf{Pb}$$

Note that the permutation matrix can also be represented by a vector giving the final sequence of rows. Call this vector *piv*.

$$\mathbf{PA} = \mathbf{A}(piv,:)$$
$$\mathbf{Pb} = \mathbf{b}(piv)$$

Complexity

- For a single step (i.e. column) in the elimination process, there are approx n elements to be zeroed. But the remainder of each row is also multiplied. Number of multiplications for one column goes like n^2
- There are n columns, so overall work goes something like n^3
- Since we only work on the lower triangular part of the matrix, the actual work goes like $(1/3)n^3$.

Number of operations ~ $(1/3)n^3$

$$\mathbf{M} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & \frac{-4}{7} & 1 & 0 \\ 0 & \frac{-2}{7} & 0 & 1 \end{pmatrix} \quad \mathbf{M\ A} = \begin{pmatrix} 9 & 8 & 8 & 9 \\ 0 & 7 & 4 & 7 \\ 0 & 0 & 3.7143 & -3 \\ 0 & 0 & 5.8571 & 2 \end{pmatrix}$$

- Backward and forward substitutions each require about n^2 operations.
- Total is about $(1/3)n^3 + n^2$. If there are multiple right hand sides (**b** vectors), only the n^2 step is repeated for each one.
- Explicit calculation of the inverse requires the same as performing n of these repetitions. Therefore, calculation of $\mathbf{A}^{-1}$ explicitly requires at least n^3 operations, a factor of three slower.

- **Banded systems.**

  Greatly simplified if the band is not wide (lower dimensionality for complexity of calculation).
- Tridiagonal matrices are particularly useful.
- **Sparse systems.** There are special methods applicable to very, very large matrices.
- **Iterative methods** for systems in which a reasonable guess can be made initially.

Example using Matlab - define **A** and **b.**

- » A = [1,2,3,4;2,1,4,1;3,4,1,5;2,3,5,2]
- A =

```
1 2 3 4
2 1 4 1
3 4 1 5
2 3 5 2
```

```
» b = [5;2;6;3]

b =
   5
   2
   6
   3
```

Solve **Ax=b** using the \ operator (invokes Gaussian elimination with full square A)

- » x = A\b
- x =

  0.2113
  -0.1549
  0.1408
  1.1690

- check the condition number:
  » cond(A,1)
  ans =
  17.3944

Solve by Explicit LU Factorization

(note that L is actually permuted)

» [L,U] = lu(A)

L =

0.3333	-0.4000	0.8000	1.0000
0.6667	1.0000	0	0
1.0000	0	0	0
0.6667	-0.2000	1.0000	0

U =

3.0000	4.0000	1.0000	5.0000
0	-1.6667	3.3333	-2.3333
0	0	5.0000	-1.8000
0	0	0	2.8400

» y=L\b

y =

6.0000
-2.0000
-1.4000
3.3200

\ does substitution here.

» x = U\y

x =

0.2113
-0.1549
0.1408
1.1690

Solve by Explicit LU Factorization

(note that L is actually permuted)

The pivot information is stored in **L,** which is not actually lower triangular in this case.

Using the call **[L,U,P] =** lu**(A),** the function returns a truly lower triangular **L** and the matrix **P** necessary to get good pivoting. Now **LU** = **PA** and we must use **P** in the solution:

y = L\(P*b)

x = U\y

Comparison of methods for efficiency

Run the demo *MethodComparison.m* for different size systems.

Iterative Refinement

$\mathbf{Ax} = \mathbf{b}$

$\mathbf{r} = \mathbf{b} - \mathbf{Ax} = 0$ for exact solution

$\mathbf{r} = \mathbf{b} - \mathbf{A}\hat{\mathbf{x}}$ where $\hat{x}$ is an approximate solution

$\mathbf{Ay} = \mathbf{r}$ solve for $\mathbf{y}$ with usual methods

$\mathbf{A}(\hat{\mathbf{x}} + \mathbf{y}) = \mathbf{b} - \mathbf{r} + \mathbf{r} = \mathbf{b},$ *and* $\hat{\mathbf{x}} + \mathbf{y} = \mathbf{x}$

- If you can solve **Ay=r** approximately to get an approximate solution $\hat{\mathbf{y}}$
- Then there is an iterative procedure that may converge to the exact solution **x**.
- This might be useful if for numeric reasons the usual algorithms yield inaccurate solutions.

Iterative Refinement

The **LU** procedure is useful here because the factoring into **L** and **U** is only done once (n^3), where as the successive substitutions are n^2. It is the same **A** and same **L** and **U**.

Run demo *IterRefinement.m* to see how this works.

Iterative Refinement

The **LU** procedure is useful here because the factoring into **L** and **U** is only done once (n^3), where as the successive substitutions are n^2. It is the same **A** and same **L** and **U**.

In problem 3-4: … by the iterative refinement method and by the forward/backward substitution methods.
Should read:
… by the forward/backward substitution methods and then by iterative refinement

Least Squares Fitting Problem

CSS 455
Chapter 7 of Turner

Experimental Situation

- A measurement of some property y is made at a value of some variable t.
- t is the *independent* variable
- y is the *dependent* variable
- *t and y* both have experimental uncertainties
- Measurements at the set $\{t_1,t_2,...,t_m\}$ yields the set $\{y_1,y_2,...,y_m\}$ observations.

Model Fitting of Data (Regression)

- Often desirable to describe the relationship between {*t*} and {*y*} in functional form.
  - Useful for tabulation, interpolation, extrapolation, etc.,
  - Useful for comparison to fundamental theory that often is expressed in terms of analytical functions.
- **Common Problem:** find a function *f(t)* that will reproduce the experimental values *y(t)*

Functional Form of *f(t)*

- Select terms in *t* either systematically or intuitively. Examples:
  - polynomial in *t*: $1, t, t^2, t^3, \ldots, t^n$
  - trigonometric in *t*: $1, \sin(t), \sin(2t), \ldots \sin(nt)$
    $1, \cos(t), \cos(2t), \ldots \cos(nt)$

- Combine the selected terms in order to get the "best" fit to the experimental data.
- If the function is written as a *linear combination* of the terms, this problem becomes a *linear optimization* problem to determine the values of the *coefficients x:*

$$f(t) = x_1(1) + x_2 t + x_3 t^2 + \ldots + x_n t^{n-1}$$

- The function is linear in coefficients, but not in the independent variable.

The x's are now the coefficients.

What is the "best" value for the x's?

- Common Choice: Minimize the error between values of y_i and the corresponding values of $f(t_i)$, *where the set {x} is now treated as a variable to be optimized.* *(i.e. find the 'best' set of {x} coefficients.)*
- Linear Least Squares minimizes the sum of the errors in the Euclidean (2-norm) sense:

$$\min_{\mathbf{x}} \sum_{i=1}^{m} (y_i - f(t_i, \mathbf{x}))^2$$

Corresponding to each observation is an equation:

- $f(t_i, \mathbf{x}) \approx y_i$, $i = 1 \ldots m$ *(# of observations)*
- With the terms written explicitly:
  $x_1 + x_2 t_i + x_3 t_i^2 + \ldots + x_n t_i^{n-1} \approx y_i$, $i = 1 \ldots m$

There are m equations with n terms in each equation. The values of t_i and y_i are experimental data, and the values of x_i are to be chosen by the least squares procedure.

- An exact fit is obtained when $y_i = f(t_i, \mathbf{x})$ for all i. *(i.e. the experimental values of the independent variable are reproduced exactly by the model function.)* Interpolant – Ch4
- In general exact fits are obtained when the number of terms in f is at least as great as the number of observations ($n \geq m$).
- **Most often, however, $m > n$,** *and only an approximate fit to the data is to be obtained.*

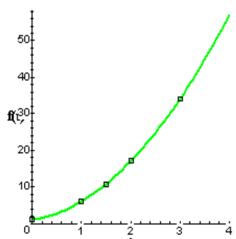
Matlab code for least squares polynomial fit to data.

```
%Demo program for ch 7
%
close all
clear all
clc
% set up 5 data points for least squares fit
t = [0 1 1.5 2 3];
y = [1.1 7 10 10.7 31];
%Fit with quartic poly (exact)
n = 4;
P4 = polyfit(t,y,n);
%Evaluate polynomial for plotting
z = linspace(0,4,100);
F4 = polyval(P4,z);
% fit with quadratic poly
n = 2;
P2 = polyfit(t,y,n);
F2 = polyval(P2,z);
plot(z,F2,'-k',z,F4,'-r',t,y,'b*')
```

When the degree of the polynomial is insufficient for an exact fit, *polyfit* uses the least squares criterion for an optimum fit to data.

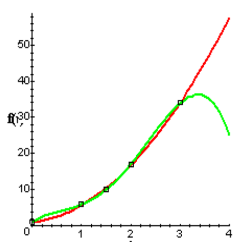
5 data sets - exact fit to: $y = 1 + 2t + 3t^2$

- Data:
  tval := [0,1,1.5, 2, 3];
  yval := [1,6,10.75,17,34];
- Two Least Sqr Fits:
  Green Line (5 terms)
  Red Line (3 terms)
- Both are exact fits and fall on top of each other and the data.



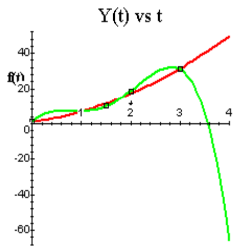
5 data sets - approx fit to : $y \approx 1 + 2t + 3t^2$

- **Data:**
  tval := [0,1,1.5, 2, 3];
  yval := [1,6,10. ,17,34];
- **Two Least Sqr Fits:**
  Green Line (5 terms)
  Red Line (3 terms)
- **Green is exact fit and red is best 3-term fit.**
- **"Exact" fit does not lie close to desired curve between data points.**



5 data sets - approx fit to : $y \approx 1 + 2t + 3t^2$

- **Data:**
  xval := [0, 1, 1.5, 2, 3];
  yval := [1.1, 7, 10,18.7,31]
- **Two Least Sqr Fits:**
  Green Line - exact with 5 terms
  Red Line - best with 3 terms
- **"Exact" fit is not even qualitatively correct.**



Formulate in Matrix Notation

- Let **A** be a matrix of the *t* values. Each row of **A** corresponds to one equation or one measurement.
- Let **b** be a vector of the observations *y* in each equation.
- Let **x** be the vector of the **x** coefficients to be determined.

$\mathbf{Ax} = \mathbf{b}$

$$\underbrace{\begin{pmatrix} 1 & t_1 & t_1^2 & \cdots & t_1^{n-1} \\ 1 & t_2 & t_2^2 & \cdots & t_2^{n-1} \\ 1 & t_3 & t_3^2 & \cdots & t_3^{n-1} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & t_m & t_m^2 & \cdots & t_m^{n-1} \end{pmatrix}}_{m \text{ x } n} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \cdots \\ x_n \end{pmatrix}}_{n \text{ x } 1} = \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \cdots \\ y_m \end{pmatrix}}_{m \text{ x } 1}$$

Consider the case with 8 data points to be fit with a 3-term polynomial.

m = 8 and n = 3.

A is 8 x 3.

x is 3 x 1

b is 8 x 1

$$\begin{pmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ 1 & t_3 & t_3^2 \\ 1 & t_4 & t_4^2 \\ 1 & t_5 & t_5^2 \\ 1 & t_6 & t_6^2 \\ 1 & t_7 & t_7^2 \\ 1 & t_8 & t_8^2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \end{pmatrix}$$

Strategy to find the best fit...

- The residual will not be zero because the fit to the data is approximate. Minimize the magnitude of the residual:

$$\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x} \qquad (\mathbf{r} \text{ is } m \text{ x } 1)$$

$$|\mathbf{r}|^2 = \mathbf{r}^\mathrm{T}\mathbf{r} = (\mathbf{b} - \mathbf{A}\mathbf{x})^\mathrm{T}(\mathbf{b} - \mathbf{A}\mathbf{x})$$

To minimize, differentiate with respect to the vector x (or $\mathbf{x}^\mathrm{T}$) and set equal to zero:

Strategy to find the best fit...

- The residual will not be zero because the fit to the data is approximate. Minimize the magnitude of the residual:

$$\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x} \qquad (\mathbf{r} \text{ is } m \text{ x } 1)$$

$$|\mathbf{r}|^2 = \mathbf{r}^\mathrm{T}\mathbf{r} = (\mathbf{b} - \mathbf{A}\mathbf{x})^\mathrm{T}(\mathbf{b} - \mathbf{A}\mathbf{x})$$

$$\frac{\partial}{\partial \mathbf{x}}\left(|\mathbf{r}|^2\right) = \frac{\partial}{\partial \mathbf{x}}\left(\mathbf{b}^T\mathbf{b} + \mathbf{x}^\mathrm{T}\mathbf{A}^\mathrm{T}\mathbf{A}\mathbf{x} - 2\mathbf{x}^\mathrm{T}\mathbf{A}^\mathrm{T}\mathbf{b}\right) = 0$$

$$|\mathbf{r}|^2 = \mathbf{b}^T\mathbf{b} + (\mathbf{Ax})^{\mathbf{T}}(\mathbf{Ax}) - \mathbf{b}^{\mathbf{T}}\mathbf{Ax} - (\mathbf{Ax})^{\mathbf{T}}\mathbf{b}$$

$$\xrightarrow{(\mathbf{Ax})^{\mathbf{T}} = \mathbf{x}^{\mathbf{T}}\mathbf{A}^{\mathbf{T}}} \mathbf{b}^T\mathbf{b} + \mathbf{x}^{\mathbf{T}}\mathbf{A}^{\mathbf{T}}\mathbf{Ax} - \mathbf{x}^{\mathbf{T}}\mathbf{A}^{\mathbf{T}}\mathbf{b} - \mathbf{b}^{\mathbf{T}}\mathbf{Ax}$$

$$\xrightarrow{(\mathbf{b}^{\mathbf{T}}\mathbf{Ax})^{\mathbf{T}} = \mathbf{x}^{\mathbf{T}}\mathbf{A}^{\mathbf{T}}\mathbf{b}} \mathbf{b}^T\mathbf{b} + \mathbf{x}^{\mathbf{T}}\mathbf{A}^{\mathbf{T}}\mathbf{Ax} - 2\mathbf{x}^{\mathbf{T}}\mathbf{A}^{\mathbf{T}}\mathbf{b}$$

To minimize, differentiate with respect to the vector x (or $\mathbf{x}^{\mathbf{T}}$) and set equal to zero:

$$\frac{\partial}{\partial \mathbf{x}}\left(|\mathbf{r}|^2\right) = \frac{\partial}{\partial \mathbf{x}}\left(\mathbf{b}^T\mathbf{b} + \mathbf{x}^{\mathbf{T}}\mathbf{A}^{\mathbf{T}}\mathbf{Ax} - 2\mathbf{x}^{\mathbf{T}}\mathbf{A}^{\mathbf{T}}\mathbf{b}\right) = 0$$

$$\frac{\partial}{\partial \mathbf{x}}\left(|\mathbf{r}|^2\right) = \left(2\mathbf{A}^{\mathbf{T}}\mathbf{Ax} - 2\mathbf{A}^{\mathbf{T}}\mathbf{b}\right) = 0$$

$$\mathbf{A}^{\mathbf{T}}\mathbf{Ax} - \mathbf{A}^{\mathbf{T}}\mathbf{b} = 0$$

or

$$\mathbf{A}^{\mathbf{T}}\mathbf{Ax} = \mathbf{A}^{\mathbf{T}}\mathbf{b}$$

$\mathbf{A}^{\mathbf{T}}\mathbf{A}$ is *(n x m)(m x n) or (n x n)*

x *is (n x 1)*

$\mathbf{A}^{\mathbf{T}}\mathbf{b}$ is *(n x m)(m x 1) or (n x 1)*

This is now a "standard" linear equation problem of dimension *n*.

- Finding the best approximate solution for the linear least squares equations is equivalent to finding the exact solution to this square system (*normal equations).*
- *The solution to this exact problem is the value of the* **x** *vector that <u>minimizes</u> the magnitude of the residual of the least squares problem.*