CS 221

Tuesday 15 November 2011

Agenda

1. Announcements
2. Solving systems of linear equations
3. Measuring time with tic/toc in MATLAB
4. Quiz Coverage
5. Homework Q & A

1. Announcements

- Homework 4 due tomorrow night (Wed 16 November)
- Next in-Class Quiz (#3): Next Week

  Tuesday, 22 November
- Lab Quiz 2 performance: lousy

2. Solving Systems of Equations

- We've seen ways to solve equations of the form:

$$f(x) = 0$$

  - Iterative solutions: bisection, fixed-point, Newton's
- Next we consider problems involving multiple variables: we want to find a set of values x_1, x_2, …, x_n, satisfying:

$$f_1(x_1,x_2,\ldots,x_n) = 0$$
$$f_2(x_1,x_2,\ldots,x_n) = 0$$
$$\ldots$$
$$f_n(x_1,x_2,\ldots,x_n) = 0$$

Solving Systems of Equations

- Such systems can be either linear or nonlinear
  - linear: no higher powers of any x_i
  - General form: n linear equations, n unknowns

$$
\begin{aligned}
a_{11} x_1 + a_{12} x_2 + \dots + a_{1n} x_n &= b_1 \\
a_{21} x_1 + a_{22} x_2 + \dots + a_{2n} x_n &= b_2 \\
&\dots \\
a_{n1} x_1 + a_{n2} x_2 + \dots + a_{nn} x_n &= b_n
\end{aligned}
$$

Solving Linear Systems

- You learned how to solve small sets of linear equations in school
  - Generally: manipulate equations to find one of the unknowns, then plug in to find the other.
  - Example:

$$2x - 3y = 5$$

$$3x - 4y = 12$$

subtract top from bottom, get $x = y+7$, plug in first equation, get $2(y+7) - 3y = 5$, so $y = 9$, so $x = 16$.

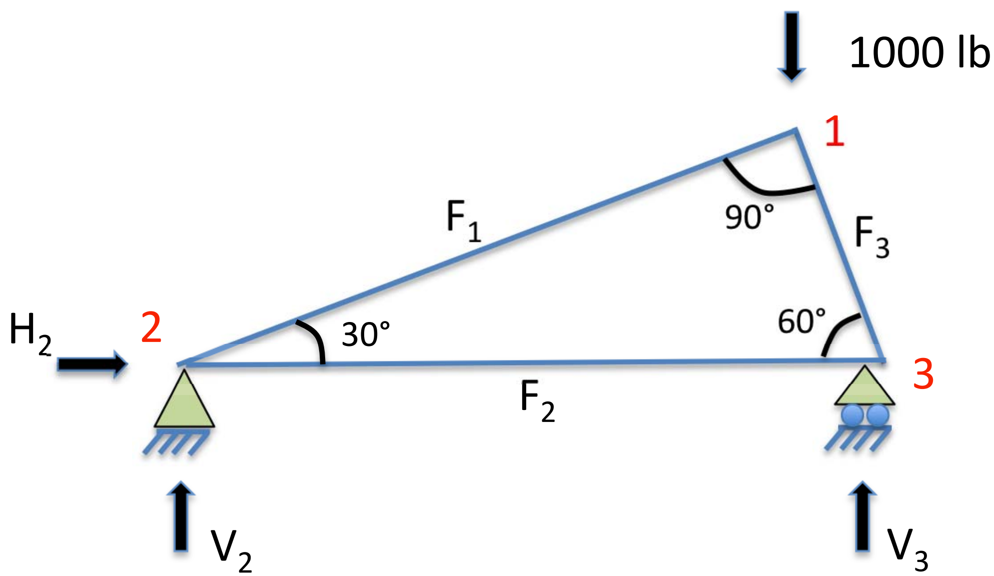
- Unfortunately, such techniques become very difficult for larger systems in general

Tools are Great for Solving Linear Systems!

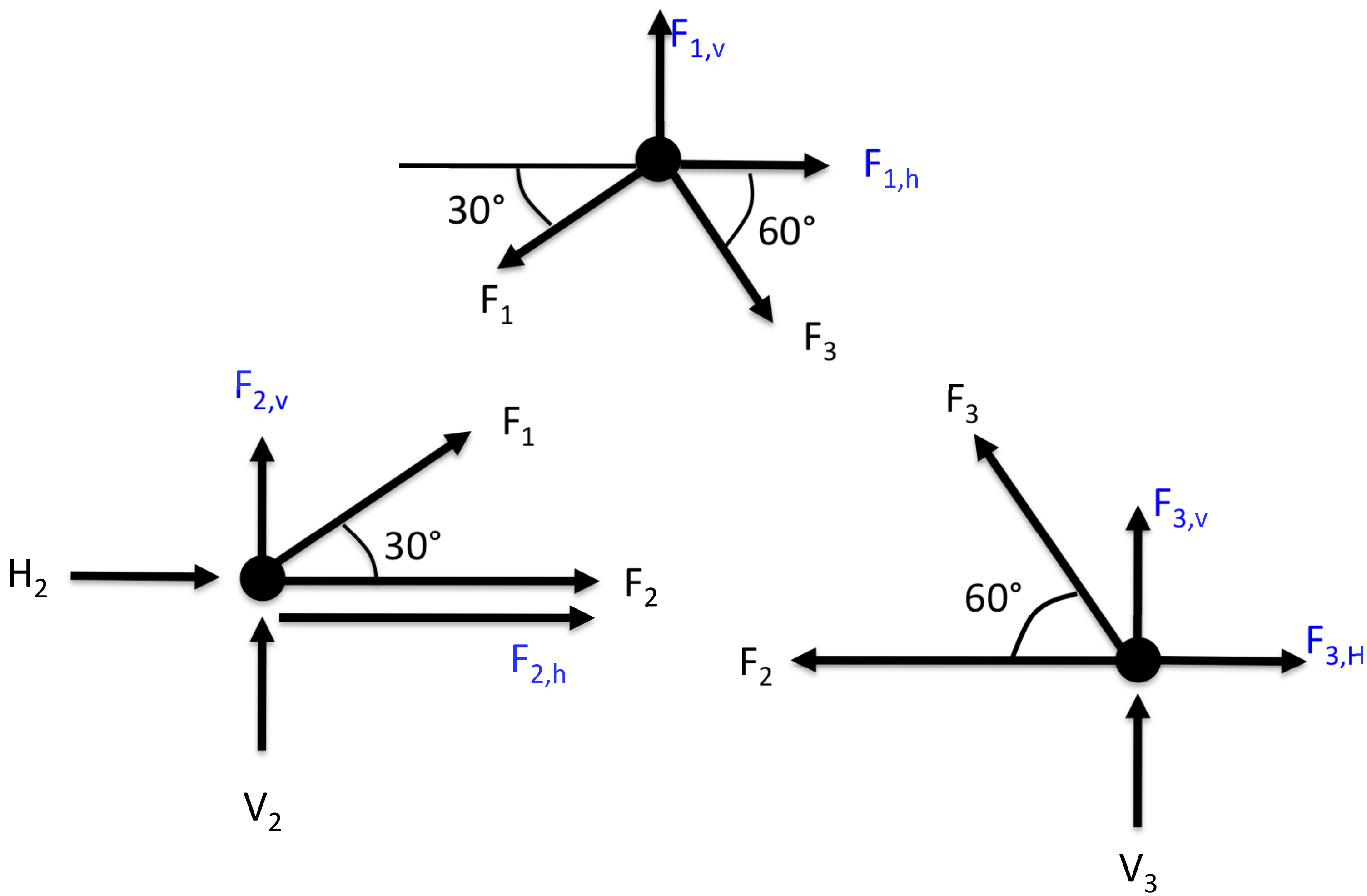
- MATLAB excels at solving “large” systems of linear equations
  - E.g., 800 equations in 800 unknowns

Example: Forces on a Truss

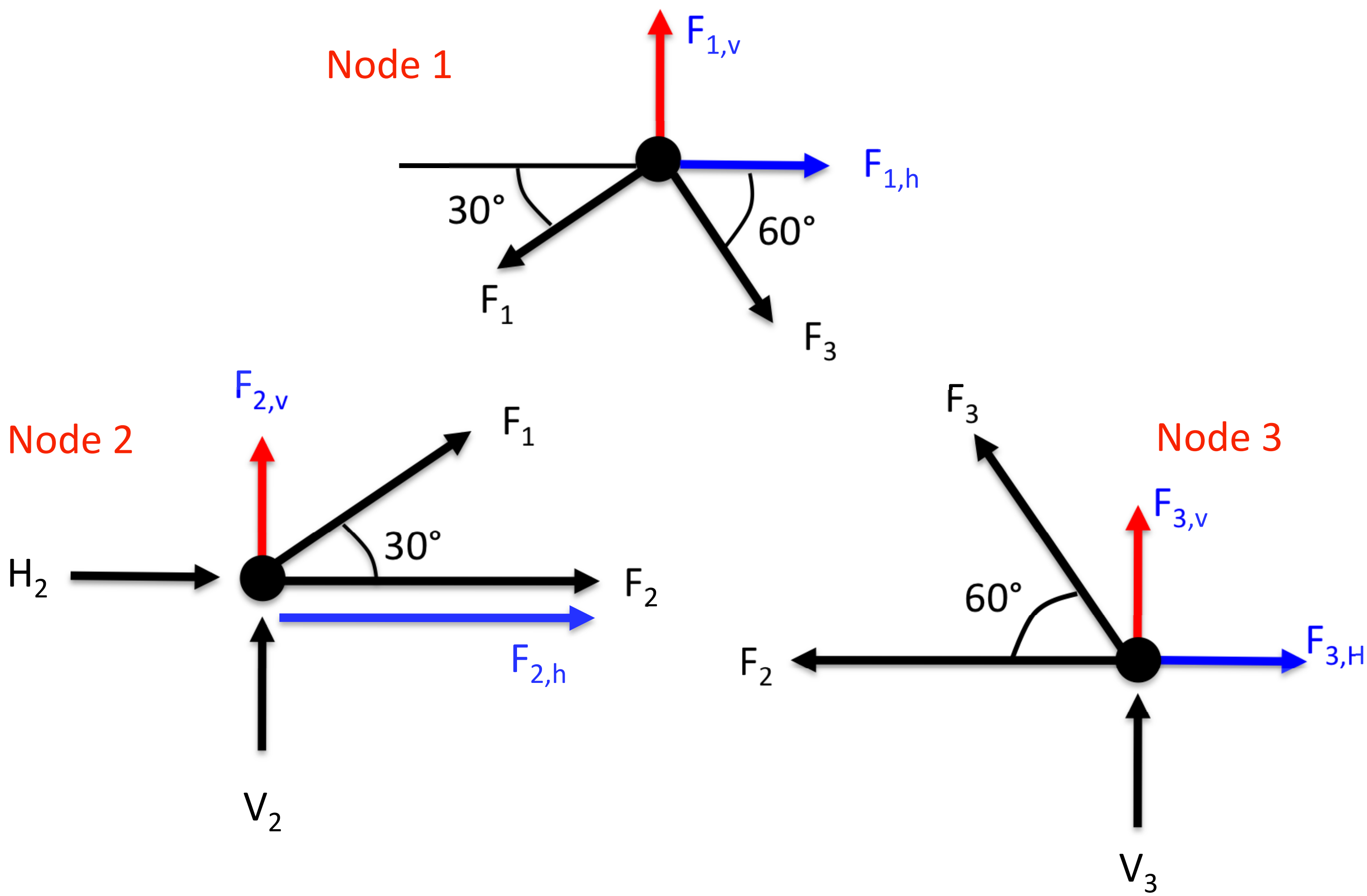
- Force: tension or compression on the members of the truss (F_1, F_2, F_3)
- External reaction: how the truss interacts with its supporting framework



Balancing Forces



Balancing Forces



Hor. and Vert. Forces Sum to Zero!

- Node 1:

  $\Sigma\ F_H = 0 = -F_1 \cos 30° + F_3 \cos 60° + F_{1,h}$

  $\Sigma\ F_V = 0 = -F_1 \sin 30° - F_3 \sin 60° + F_{1,v}$

External forces acting on node

- Node 2:

  $\Sigma\ F_H = 0 = F_2 + F_1 \cos 30° + H_2 + F_{2,h}$

  $\Sigma\ F_V = 0 = F_1 \sin 30° + F_{2,v} + V_2$

- Node 3:

  $\Sigma\ F_H = 0 = -F_2 - F_3 \cos 60° + F_{3,h}$

  $\Sigma\ F_V = 0 = F_3 \sin 60° + F_{3,v} + V_3$

$F_{1,v} = -1000$

all other $F_{i,[v,h]} = 0$

Equations

$\cos 30\, F_1 + 0\, F_2 - \cos 60\, F_3 + 0\, H_2 + 0\, V_2 + 0\, V_3 = F_{1,h}$

$\sin 30\, F_1 + 0\, F_2 + \sin 60\, F_3 + 0\, H_2 + 0\, V_2 + 0\, V_3 = -F_{1,v}$

$-\cos 30\, F_1 + -1\, F_2 + 0\, F_3 + -1\, H_2 + 0\, V_2 + 0\, V_3 = F_{2,h}$

$\sin 30\, F_1 + 0\, F_2 + 0\, F_3 + 0\, H_2 + 1\, V_2 + 0\, V_3 = -F_{2,v}$

$0\, F_1 + 1\, F_2 + \cos 60\, F_3 + 0\, H_2 + 0\, V_2 + 0\, V_3 = F_{3,h}$

$0\, F_1 + 0\, F_2 - \sin 60\, F_3 + 0\, H_2 + 0\, V_2 - 1\, V_3 = F_{3,v}$

Note: every variable should have a nonzero coefficient in <u>some</u> equation!

Matrix Representation of the System of Equations

$$\begin{bmatrix} 0.866 & 0 & -0.5 & 0 & 0 & 0 \\ 0.5 & 0 & 0.866 & 0 & 0 & 0 \\ -0.866 & -1 & 0 & -1 & 0 & 0 \\ -0.5 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & -0.866 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ H_2 \\ V_2 \\ V_3 \end{bmatrix} = \begin{bmatrix} F_{1,h} \\ -1000 \\ F_{2,h} \\ F_{2,v} \\ F_{3,h} \\ F_{3,v} \end{bmatrix}$$

System of Equations

$$
\begin{bmatrix}
0.866 & 0 & -0.5 & 0 & 0 & 0 \\
0.5 & 0 & 0.866 & 0 & 0 & 0 \\
-0.866 & -1 & 0 & -1 & 0 & 0 \\
-0.5 & 0 & 0 & 0 & -1 & 0 \\
0 & 1 & 0.5 & 0 & 0 & 0 \\
0 & 0 & -0.866 & 0 & 0 & -1
\end{bmatrix}
\begin{bmatrix}
F_1 \\ F_2 \\ F_3 \\ H_2 \\ V_2 \\ V_3
\end{bmatrix}
=
\begin{bmatrix}
0 \\ -1000 \\ 0 \\ 0 \\ 0 \\ 0
\end{bmatrix}
$$

Coefficient Matrix   Unknowns   Parameters (Loadings)

Solving with MATLAB®

- Create the Coefficient and Constant (Parameter) Matrices
  - A = [cosd(30), 0, -0.5, 0, 0, 0; 0.5, ...];
  - B = [0; -1000; 0; 0; 0; 0;];
    - Note well: B is (must be) a column vector.
- Solve for x in one of two ways:
  - Using “left-division” (matrix operator)

    `x = A\B;`
  - By computing A^{-1} and multiplying B by it:

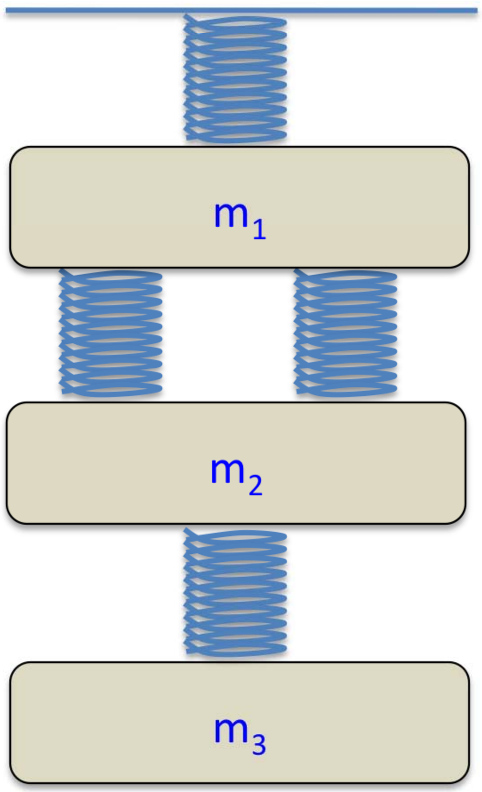
    `x = inv(A)*B;`

Which MATLAB Method to Use?

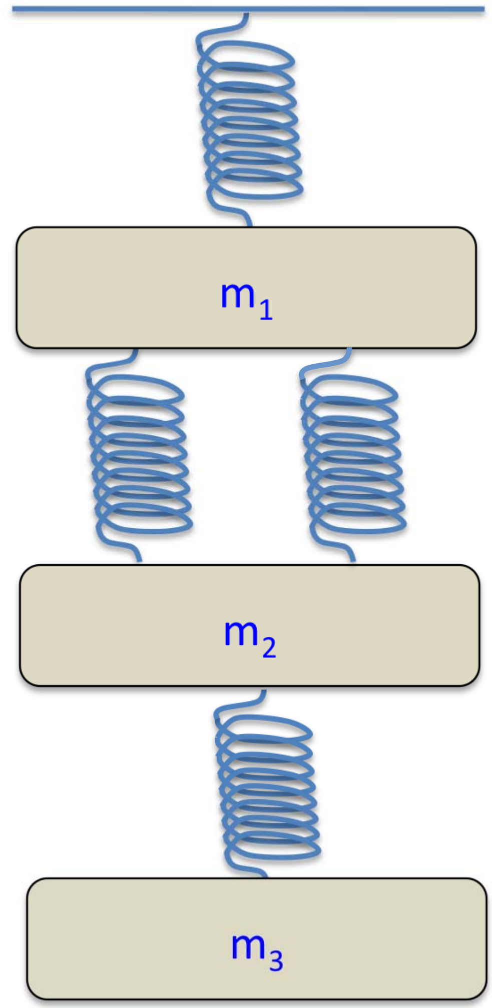
- If you are only going to solve the equation once, use “left division”
- If you are going to re-solve with a different B matrix, compute and save A^{-1}
  - In this example: vary the external forces on the truss.
  - Ainv = inv(A);   x1 = Ainv*B1; x2 = Ainv*B2;
- Why?
  - Matrix operations are expensive; inverting a large matrix is really expensive (read: slow).
  - Left division is faster.
  - 1000x1000 Matrix: Left-division: 0.14 s; invert: 0.31 s
  - 10000x10000:   51.9 s vs. 155.3 s

Example: Spring Systems

Initial State



At Quiescence



Spring Forces

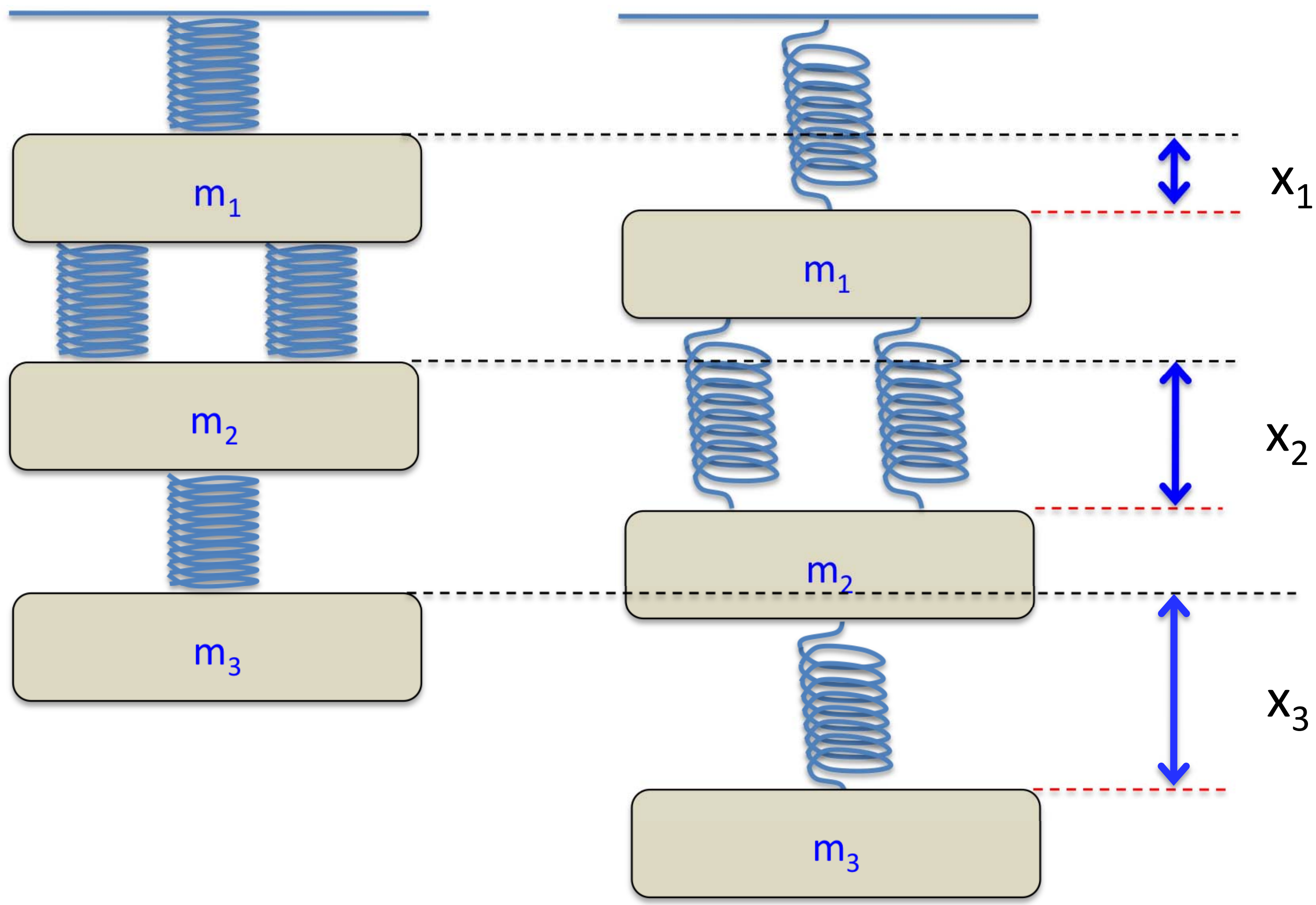
- Springs exert force proportional to the amount they are "stretched"

  $$F_{spring\ i} = k_i\, x_i$$

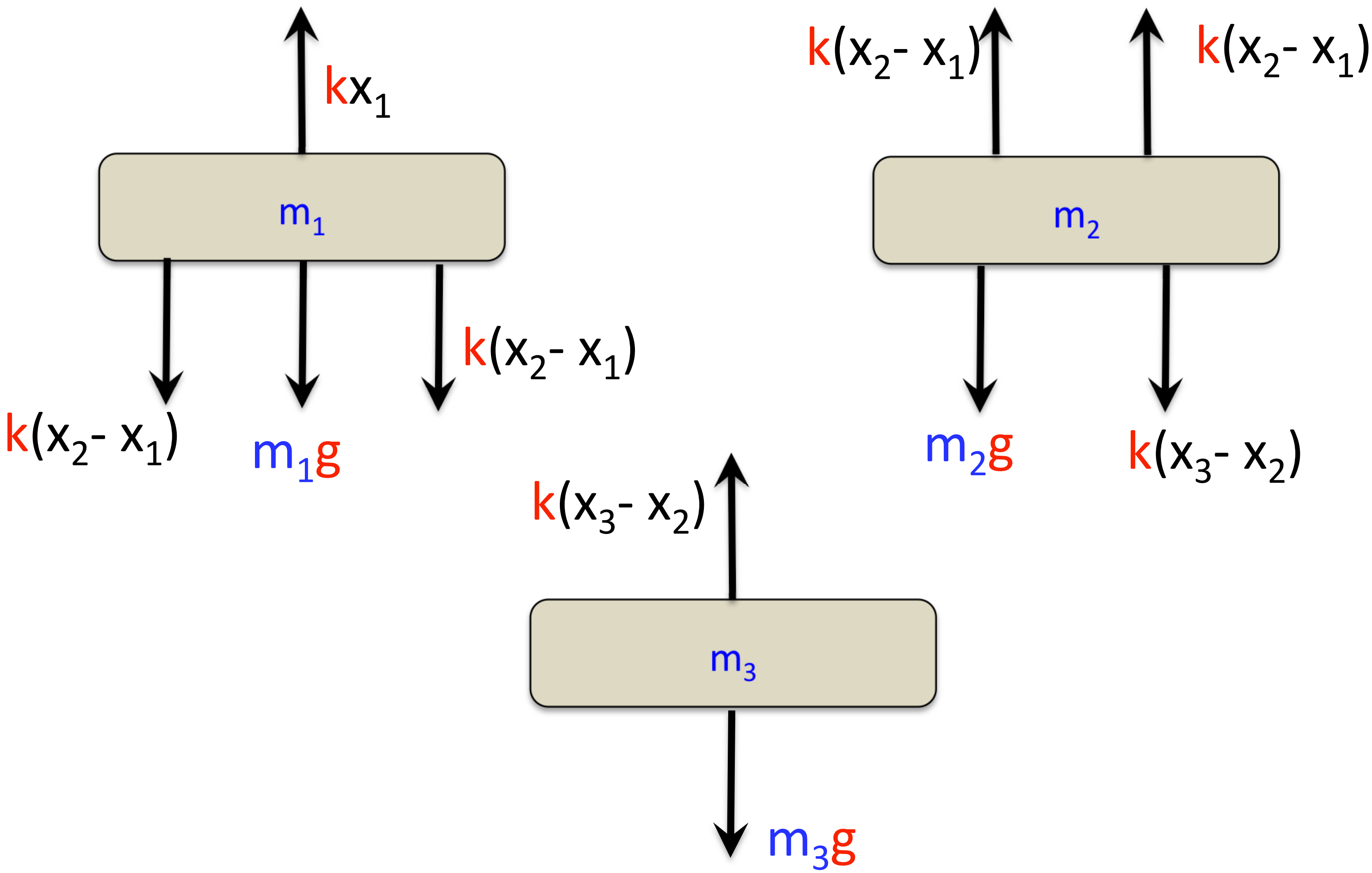
  For this problem: assume all k_i's = k

- In steady-state, all masses are at rest, and all forces are balanced
  - Spring force up = gravity + spring force down

Example: Spring Systems



Free-Body Diagrams



Equations

Mass 1: $kx_1 = 2k(x_2 - x_1) + m_1g$

Mass 2: $2k(x_2 - x_1) = m_2g + k(x_3 - x_2)$

Mass 3: $k(x_3 - x_2) \quad = m_3g$

Rewrite to get:

$$3k\, x_1 - 2k\, x_2 + 0\, x_3 = m_1g$$
$$-2k\, x_1 + 3k\, x_2 - k\, x_3 = m_2g$$
$$0\, x_1 - k\, x_2 + k\, x_3 = m_3g$$

Matrix Equation

- [K] [X] = [W]
  - [K] is called the <u>stiffness matrix</u>

$$\begin{bmatrix} 3k & -2k & 0 \\ -2k & 3k & -k \\ 0 & -k & k \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \begin{matrix} = \\ = \\ = \end{matrix} \begin{bmatrix} m_1 g \\ m_2 g \\ m_3 g \end{bmatrix}$$

Matrix Equation

- [K] [X] = [W]
  - [K] is called the stiffness matrix

$$\begin{bmatrix} 3(10) & -2(10) & 0 \\ -2(10) & 3(10) & -(10) \\ 0 & -(10) & (10) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \begin{matrix} = \\ = \\ = \end{matrix} \begin{bmatrix} (2)(9.8) \\ (3)(9.8) \\ (2.5)(9.8) \end{bmatrix}$$

Matrix Equation

- [K] [X] = [W]
  - [K] is called the stiffness matrix

$$\begin{bmatrix} 30 & -20 & 0 \\ -20 & 30 & -10 \\ 0 & -10 & 10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \begin{matrix} = \\ = \\ = \end{matrix} \begin{bmatrix} 19.6 \\ 29.4 \\ 24.5 \end{bmatrix}$$

The Solution

$$\text{inv}(K) = \begin{bmatrix} 0.10 & 0.10 & 0.10 \\ 0.10 & 0.15 & 0.15 \\ 0.10 & 0.15 & 0.25 \end{bmatrix}$$

$$\text{inv}(K)*W = \begin{bmatrix} 7.35 \\ 10.045 \\ 12.495 \end{bmatrix}$$

Solving with Excel

- Set up coefficient and parameter matrices (K and W) in the spreadsheet
- Compute inverse of K (using the MINVERSE function), call it Kinv
- Multiply Kinv times W (using the MMULT function) to get the solution

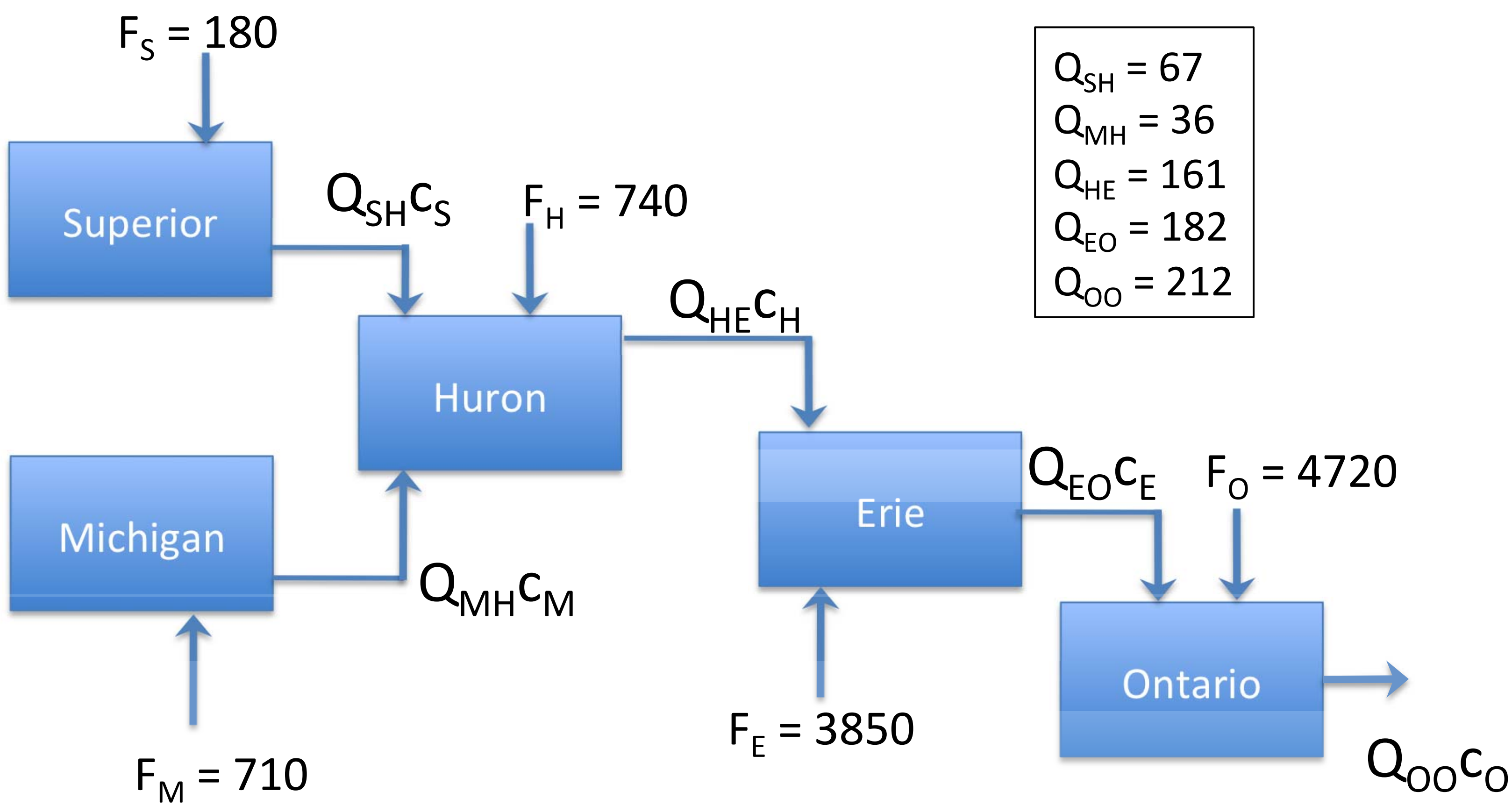
Changing the Parameters

- What if m_2 is now 1 kg?
  - Simply change W and re-compute inv(K)*W!

$$W = \begin{bmatrix} 19.6 \\ 9.8 \\ 24.5 \end{bmatrix}$$

$$\text{inv}(K) * W = \begin{bmatrix} 5.4 \\ 7.1 \\ 9.6 \end{bmatrix}$$

Example:
Great Lakes Chloride Concentration



Flow Balance Problems

- Basic principle: conservation of mass
- Flow in = flow out (assuming no chemical changes)

Flow Balance Equations

- $Q_{SH}c_S = F_S$
- $Q_{MH}c_M = F_M$
- $Q_{SH}c_S + Q_{MH}c_M - Q_{HE}c_H = -F_H$
- $Q_{HE}c_H - Q_{EO}c_E = -F_E$
- $Q_{EO}c_E - Q_{OO}c_O = -F_O$

Matrix Equations

$$\begin{bmatrix} Q_{SH} & 0 & 0 & 0 & 0 \\ 0 & Q_{MH} & 0 & 0 & 0 \\ Q_{SH} & Q_{MH} & -Q_{HE} & 0 & 0 \\ 0 & 0 & Q_{HE} & -Q_{EO} & 0 \\ 0 & 0 & 0 & Q_{EO} & -Q_{OO} \end{bmatrix} \begin{bmatrix} c_S \\ c_M \\ c_H \\ c_E \\ c_O \end{bmatrix} = \begin{bmatrix} F_S \\ F_M \\ -F_H \\ -F_E \\ -F_O \end{bmatrix}$$

Matrix Equations

$$\begin{bmatrix} 67 & 0 & 0 & 0 & 0 \\ 0 & 36 & 0 & 0 & 0 \\ 67 & 36 & -161 & 0 & 0 \\ 0 & 0 & 161 & -182 & 0 \\ 0 & 0 & 0 & 182 & -212 \end{bmatrix} \begin{bmatrix} c_S \\ c_M \\ c_H \\ c_E \\ c_O \end{bmatrix} = \begin{bmatrix} 180 \\ 710 \\ -740 \\ -3850 \\ -4720 \end{bmatrix}$$

Solution

$c_S = 2.69$

$c_M = 19.7$

$c_H = 10.1$

$c_E = 30.1$

$c_O = 48.1$

(Note significant digits.)

Summary on Linear Equations

- The "geometry" of the problem dictates the coefficient matrix through a balance principle
  - Conservation of mass
  - Forces balanced at equilibrium
- The "inputs" (RHS vector – external forces, e.g.) can be changed (to get a new solution) without changing the coefficient matrix
- The MATLAB "left division" operator is faster than inv() if you don't need to use it more than once
  - Computing the inverse is computationally more expensive than just getting the answer (Gauss elimination)

3. Timing Operations with “tic” and “toc”

- MATLAB has built-in functions to time operations.
- Use like this:

  tic; <operation to be timed>; toc
- Prints elapsed time in seconds.
  - To save elapsed time, do: var = toc;
- More elaborate timing structures (e.g., nested calls) are possible.
- How long does it take to invert a 2000-by-2000 matrix?

4. Quiz Coverage

- Plotting
  - plot() and fplot() in MATLAB, plot types in Excel
  - E.g., "when would you use fplot() instead of plot()?"

    (when you want to plot the curve of a function, not data)
  - What different kinds of plots/graphs are for
- Equation classification: linear, nonlinear polynomial, nonlinear general
- Finding roots:
  - fzero() and roots() in MATLAB
  - Goal-Seeking in Excel

Quiz Coverage

- Matrix mathematics
  - Operators in MATLAB (including .^ and .*)
  - Functions in Excel
- Function Handles
  - What they are, what they are for
- Everything about loops and conditionals
  - Know how to interpret and simulate execution of code!
- Note: Curve-fitting and solving systems of linear equations will be on the final

Example Problem

- What is the value of v after this sequence of statements is executed?

```
v = 13;
while v > 0
    v = v/2;
    v = v + 3;
end
```

Example Problem

- What's wrong with this?

```
A = [ 1, 2, 3; 4, 5, 6];
B = [  -1, -2, -3; -4, -5, -6];
C = A*B;
```

Example Problem

- Consider the function $f(x) = 5x^3 - 3x^2 - 4x + 3$
- Suppose you a bisection root-finding function with @f, a lower bound of -1, and an upper bound of -0.8. How many iterations will it take until the error bound is less than 10^{-5}?
  - Simulate the bisection method!

5. Homework Q & A