Spring Systems: Oscillation

FUNDAMENTAL MODES. MEANING

I. Suppose a spring system with three masses and many springs oscillates at a fundamental mode with

$$\text{eigenvalue } \lambda = 4 \quad \text{and} \quad \text{eigenvector } \mathbf{v} = \begin{bmatrix} 1 \\ -2 \\ 4 \end{bmatrix}$$

(a) If mass 2 oscillates with amplitude 6, what is the amplitude of oscillation for mass 3?

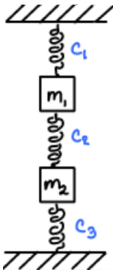
(b) If mass 3 is at maximum height at time $t = 4$, when will it next be at maximum height?

(c) Sketch a rough graph of positions of all masses at time t assuming that mass 1 is at equilibrium position with positive velocity at time $t = 0$ and mass 1 oscillates with amplitude 1.
(For graphing purposes, assume that masses are in a line with their equilibrium positions 8 units apart).

FUNDAMENTAL MODES. COMPUTATION

II. Find the fundamental modes of oscillation of the systems below. *Hint: Only integers required.*

(A)



$c_1 = 4$, $m_1 = 1$, $c_2 = 1$, $m_2 = \frac{1}{4}$, $c_3 = 1$

(B)

$c_1 = 4$, $m_1 = \frac{1}{3}$, $c_2 = 1$, $m_2 = \frac{1}{2}$, $c_3 = 4$

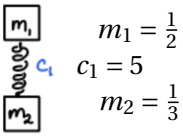
(C)

$c_1 = 5$, $m_1 = 1$, $c_2 = 4$, $m_2 = \frac{1}{5}$

(D)

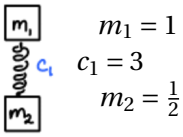
$c_1 = 4$, $m_1 = \frac{1}{2}$, $c_2 = 2$, $m_2 = \frac{1}{4}$

(E)



$m_1 = \frac{1}{2}$, $c_1 = 5$, $m_2 = \frac{1}{3}$

(F)



$m_1 = 1$, $c_1 = 3$, $m_2 = \frac{1}{2}$

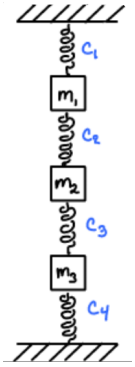
DISPLACEMENT FUNCTIONS

III. For spring systems (A)-(D) in the computational section, write the function $\mathbf{u}(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix}$ giving the displacements of mass 1 and 2 at time t if they begin with initial displacement and velocity $\mathbf{u}(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $\mathbf{u}'(0) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$.

IV. For spring systems (E)-(F) in the computational section, find $\mathbf{u}'(0)$ so that displacement is bounded if $\mathbf{u}(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$.

MATLAB

- Last week we used the `[V,D] = eig(<matrix>)` command to get eigenvalues and eigenvectors of a matrix. We can use this command to solve oscillating spring systems and graph the displacements of the masses. Suppose we have a line of three masses and four springs, fixed at each end with $m_1 = 2$, $m_2 = 4$, $m_3 = 1$ and spring constants $c_1 = 2$, $c_2 = 4$, $c_3 = 1$, $c_4 = 1$.



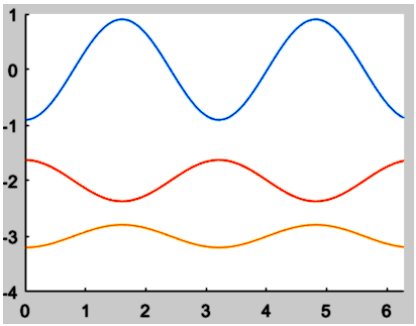

```
>> K = [ 6 -4 0 ; -4 5 -1 ; 0 -1 2]          % stiffness matrix
>> Minv = diag( [ 1/2 1/4 1/1 ] )            % divide by masses
>> [V, D] = eig( Minv * K )                  % eigenvectors and eigenvalues
>> freq = sqrt( diag( D ) )                  % convert eigenvalues to frequencies
```

  Note: `diag( [ 1/2 1/4 1/1 ] )` makes a matrix with `1/2 1/4 1/1` as diagonal elements; but `diag( D )` returns the vector of diagonal elements of the matrix `D`.

- We can plot the fundamental modes of oscillation. MatLab has multiple ways of writing functions, but the standard way is to use *anonymous functions* written as "`@(<var>) <function>`" For example we can record the displacements in the first fundamental mode of oscillation (with $\mathbf{u}(0) = \mathbf{v}_1$ and $\mathbf{u}'(0) = \mathbf{0}$) using

```
>> u1 = @(t)  V(1,1) * cos( freq(1) * t )     % mass #1
>> u2 = @(t)  V(2,1) * cos( freq(1) * t )     % mass #2
>> u3 = @(t)  V(3,1) * cos( freq(1) * t )     % mass #3
```

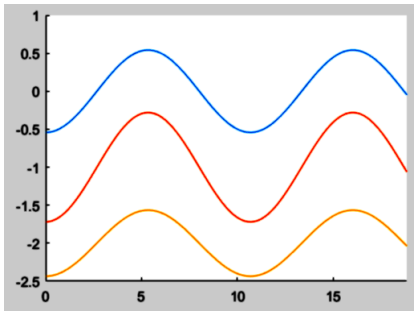
  We can plot these using `fplot(<function>, <range>)`

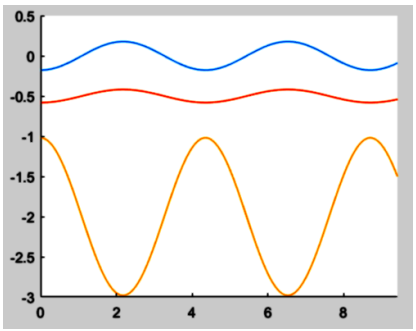


```
>> hold on
>> fplot( @(t)  u1(t),     [0, 2*pi] )
>> fplot( @(t)  u2(t) - 2, [0, 2*pi] )
>> fplot( @(t)  u3(t) - 3, [0, 2*pi] )
```

  The command "`hold on`" in the code above tells MatLab to plot multiple functions in the same plot window. Otherwise each function is given its own window. Note that we shift each u_i because **u** is **displacement** from from equilibrium, rather than **position**.

  Plotting the other two fundamental modes similarly gives the following graphs.





  [For cosmetic reasons, we used different equilibrium positions and ranges when drawing the plots above.]

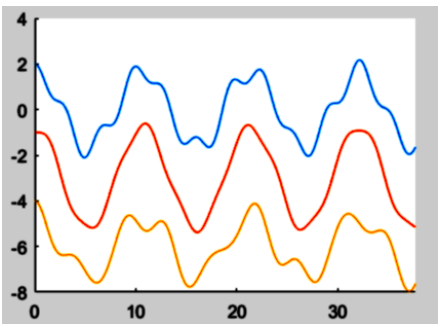
  It would have been slightly more clever to define u using multiple variables.

```
>> U = @(n,i,t)  V(i,n) * cos( freq(n) * t )     % n = mode of oscillation, i = mass #
>> fplot( @(t)  U(1,1,t),     [0, 2*pi] )        % first mode, mass #1
>> fplot( @(t)  U(1,2,t) - 2, [0, 2*pi] )        % first mode, mass #2
```

  etc...

- For a heterogeneous system you must first write the initial conditions as eigenvectors. For example, suppose that the masses all began with displacement 2 and velocity 0 (so that $\sin(\omega_n t)$ coefficients are 0).

```
>> c = V \ [ 2 2 2 ]'
>> u = @(i,t)  c(1) * U(1,i,t)  +  ...
               c(2) * U(2,i,t)  +  ...
               c(3) * U(3,i,t)
>> hold on
>> fplot( @(t)  u(1,t),     [0, 12*pi] )
>> fplot( @(t)  u(2,t) - 3, [0, 12*pi] )
>> fplot( @(t)  u(3,t) - 6, [0, 12*pi] )
```



  The "..." in lines 16 and 17 above tell MatLab that the formula is continued on the next line. Since the middle mass is heaviest, it oscillates cleanly and the other masses get jerked around by it.