Coupled Oscillators - a worked example of the Eigenvalue and Eigenvector analysis when the masses (and springs are unequal).

Step 1 - The formulation of the coupled oscillator equations

Eigenvector/eigenvalue analysis when the masses (and springs are unequal).

In the previous notebook we considered linear chains of equal masses connected by springs that could be of equal or unequal strength (different spring constants). We now wish to look at the more general case where the masses may be unequal. We will make the same steps in the solution as before. The important point to note at the start is that to use the Eigenvalue/Eigenvector analysis we need to ensure that our matrix is both real and symmetric. When the masses were equal this was always true (the K matrix is by definition real and symmetric and remains so with a scalar division by the mass). When the masses are unequal this is no longer the case and we have to employ mathematical trick to recover the simple eigenvalue equation we wish to solve. The method is explained here.

In order to solve this problem we need to apply Newtons second law of motion to each of the masses in turn. We should remember that we will specify the coordinate of each mass m_1, m_2 in terms of their coordinates x_1 and x_2 that are defined as zero when the masses are in their equilibrium positions. Hence we should obtain,

$$m_1\ddot{x}_1 = -k_1x_1 + k_2(x_2 - x_1)$$
$$m_2\ddot{x}_2 = -k_3x_2 + k_2(x_1 - x_2)$$

The trickiest part here is getting the signs correct. For the coupling terms (for example the term including k_2) setting either $x_1 = 0$ or $x_2 = 0$ is often sufficient to check the signs are correct. You should also note by Newton's third law that the force acting between the masses should be equal and opposite for each one. We can write these equations in matrix form as:

$$\begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix} \begin{bmatrix} \ddot{x}_1 \\ \ddot{x}_2 \end{bmatrix} = \begin{bmatrix} -(k_1 + k_2) & k_2 \\ k_2 & -(k_2 + k_3) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \ldots . (1)$$

This is the equation we need to solve. (With the trial solution before without manipulation, it leads to a form of the Generalised Eigenvalue equation.) However, we note we can re-arrange our equations of motion to obtain the recover the simple eigenvalue equationwe wish to solve.

Step 2 - The trial solution

We will use the same trial solution as before.

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} \exp(i\omega t) \quad \dots (2)$$

Hence, using the trial solution on (4) and subsituting into (3) we find

$$\omega^2 \begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} = -k \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} \quad \dots (3)$$

.

In the previous example where $m_1 = m_2 = m$ this naturally led to our eigenvalue equation. However, when $m_1 \neq m_2$ we see this is now no longer the case. However, there is a trick we may employ to return to recover our required form which we follow below. We rewrite (3) as follows:

$$\omega^2 \begin{bmatrix} \sqrt{m_1} & 0 \\ 0 & \sqrt{m_2} \end{bmatrix} \begin{bmatrix} \sqrt{m_1}C_1 \\ \sqrt{m_2}C_2 \end{bmatrix} = \begin{bmatrix} -(k_1 + k_2) & k_2 \\ k_2 & -(k_2 + k_3) \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \end{bmatrix}$$

or going even further

$$\omega^2 \begin{bmatrix} \sqrt{m_1} & 0 \\ 0 & \sqrt{m_2} \end{bmatrix} \begin{bmatrix} \sqrt{m_1}C_1 \\ \sqrt{m_2}C_2 \end{bmatrix} = \begin{bmatrix} \frac{-(k_1+k_2)}{\sqrt{m_1}} & \frac{k_2}{\sqrt{m_2}} \\ \frac{k_2}{\sqrt{m_1}} & \frac{-(k_2+k_3)}{\sqrt{m_2}} \end{bmatrix} \begin{bmatrix} \sqrt{m_1}C_1 \\ \sqrt{m_2}C_2 \end{bmatrix}$$

This looks more complicated but writing $C_1' = \sqrt{m_1}C_1$ and $C_2' = \sqrt{m_2}$ we have

$$\omega^2 \begin{bmatrix} \sqrt{m_1} & 0 \\ 0 & \sqrt{m_2} \end{bmatrix} \begin{bmatrix} C_1' \\ C_2' \end{bmatrix} = \begin{bmatrix} \frac{-(k_1+k_2)}{\sqrt{m_1}} & \frac{k_2}{\sqrt{m_2}} \\ \frac{k_2}{\sqrt{m_1}} & \frac{-(k_2+k_3)}{\sqrt{m_2}} \end{bmatrix} \begin{bmatrix} C_1' \\ C_2' \end{bmatrix} \quad \dots (4)$$

Finally we recognise that

$$\begin{bmatrix} \sqrt{m_1} & 0 \\ 0 & \sqrt{m_2} \end{bmatrix}^{-1} = \begin{bmatrix} \frac{1}{\sqrt{m_1}} & 0 \\ 0 & \frac{1}{\sqrt{m_2}} \end{bmatrix}$$

so that multiplying the left hand and right hand side of (4) by the inverse mass matrix we get

$$\omega^2 \begin{bmatrix} C_1' \\ C_2' \end{bmatrix} = \begin{bmatrix} \frac{-(k_1+k_2)}{m_1} & \frac{k_2}{\sqrt{m_1 m_2}} \\ \frac{k_2}{\sqrt{m_1 m_2}} & \frac{-(k_2+k_3)}{m_2} \end{bmatrix} \begin{bmatrix} C_1' \\ C_2' \end{bmatrix}$$

We can recognise this again as our eigenvalue equation

$$(A - \lambda I)\,\vec{v} = 0$$

for a real and symmetric matrix for which we know that we have real eigenvalues and orthonormal eigenvectors.

Step 3 - Finding the Eigenvalues and Eigenvectors.

We will now use python, as before, to calculate the eigenvalues and vectors for the matrix. To make things clear for this example we will take $m_2 = 4m_1 = 4m$ and we will take $k_1 = k_2 = k_3 = k$ but we can substitute any values of m or k we wish. Hence we need to find the eigenvalues and vectors of,

$$\begin{bmatrix} 2 & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

Here is the code we need to do the calculation,

In [1]:

```
### The first part imports the linear algebra library so it recognises the function nee
ded to calculate the eigenvalues and vectors
import numpy as NP
from scipy import linalg as LA
###
### Now we'll enter the array
###
A = NP.array([[2,-1/2],[-1/2,1/2]])
print (A)
e_vals,e_vecs = LA.eig(A)
print ("Eigenvalues")
print(e_vals)
print ("Eigenvectors")
print(e_vecs)
```

```
[[ 2.  -0.5]
 [-0.5  0.5]]
Eigenvalues
[2.15138782+0.j 0.34861218+0.j]
Eigenvectors
[[ 0.95709203  0.28978415]
 [-0.28978415  0.95709203]]
```

From the result we see from this case (slightly different to the lecture) we have two eigenvalues. $\omega^2 = 2.15$ and $\omega^2 = 0.349$. We've lost the simple $\omega = 1$ result when the masses were equal.

The eigenvectors are returned as a matrix of normalised column vectors so the first eigenvector is $\begin{bmatrix} 0.957 \\ -0.290 \end{bmatrix}$ and the second eigenvector is $\begin{bmatrix} 0.290 \\ 0.957 \end{bmatrix}$. We note we still have 'in phase' and in 'anti-phase' motions but the relative amplitudes of C_1' and C_2' have changed. However, the physical amplitudes of the two masses are determined by C_1 and C_2 in our trial solution so to obtain the physical ratio of the amplitudes we need to write $C_1 = \frac{C_1'}{\sqrt{m_1}}$ and $C_2 = \frac{C_2'}{\sqrt{m_2}}$ in our trial solution, hence

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = A \begin{bmatrix} 0.957 \\ -0.145 \end{bmatrix} \exp(i(1.46t)) + B \begin{bmatrix} 0.145 \\ 0.957 \end{bmatrix} \exp(i(0.591t))$$

where A and B are constants to be determined from the initial conditions.

Exercise

The python code from the example above is copied below. Go back to Step 1 above and find the matrix when you enter different values of m_1 and m_2 while keeping k constant. Use the code below to calculate the eigenvalues and eigenvectors and try to understand how the nature of the coupled oscillations has changed. The code has been modified to make it easy to change the values of m_1 and m_2. Look in particular when $m_1 \simeq m_2$ and $m_1 << m_2$. Try to interpret physically what is going on. What would you expect the limiting values of the eigenvalues and eigenvectors to be?

In [2]:

```
### The first part imports the linear algebra library so it recognises the function nee
ded to calculate the eigenvalues and vectors
import numpy as NP
from scipy import linalg as LA
###
### Enter values into the array below for the equations of motion you have determined.
###
m1=1
m2=4
A = NP.array([[2/m1,-1/NP.sqrt(m1*m2)],[-1/NP.sqrt(m1*m2),2/m2]])
print (A)
e_vals,e_vecs = LA.eig(A)
print ("Eigenvalues")
print(e_vals)
print ("Eigenvectors")
print(e_vecs)
```

```
[[ 2.  -0.5]
 [-0.5  0.5]]
Eigenvalues
[2.15138782+0.j 0.34861218+0.j]
Eigenvectors
[[ 0.95709203  0.28978415]
 [-0.28978415  0.95709203]]
```

Practical example - the Carbon Dioxide molecule?

If we ignore the quantum nature of the problem we may consider a carbon dioxide molecule as linear (in reality we would also need to consider 'bending' modes) and comprising a central carbon atom of mass 12 connected to two oxygen atoms of mass 16. We can approximate the interaction between the atoms as spring-like with equal spring constants (k) on either side.

The equation of motion for the system may be written as

$$m_{O1}\ddot{x}_{O1} = -k(x_{O1} - x_C)$$
$$m_C\ddot{x}_C = k(x_{O1} - x_C) + k(x_{O2} - x_C)$$
$$m_{O2}\ddot{x}_{O2} = -k(x_{O2} - x_C)$$

where m_C, m_{O1} and m_{O2} are the masses of the carbon and oxygen atoms and x_C, x_{O1} and x_{O2} are their coorindates. Using our trial solution as before we obtain,

$$-\omega^2 \begin{bmatrix} m_{O1} & 0 & 0 \\ 0 & m_C & 0 \\ 0 & 0 & m_{O2} \end{bmatrix} \begin{bmatrix} C_{O1} \\ C_C \\ C_{O2} \end{bmatrix} = -k \begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} C_{O1} \\ C_C \\ C_{O2} \end{bmatrix}$$

.

As we did in the example above we can manipulate the mass matrix to obtain the following

$$-\omega^2 \begin{bmatrix} \sqrt{m_{O1}} & 0 & 0 \\ 0 & \sqrt{m_C} & 0 \\ 0 & 0 & \sqrt{m_{O2}} \end{bmatrix} \begin{bmatrix} \sqrt{m_{O1}}C_{O1} \\ \sqrt{m_C}C_C \\ \sqrt{m_{O2}}C_{O2} \end{bmatrix} = -k \begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} C_{O1} \\ C_C \\ C_{O2} \end{bmatrix}$$

.

After some manipulation this can be expressed as,

$$\omega^2 \begin{bmatrix} C'_{O1} \\ C'_C \\ C'_{O2} \end{bmatrix} = k \begin{bmatrix} \frac{1}{m_{O1}} & -\frac{1}{\sqrt{m_{O1}m_C}} & 0 \\ -\frac{1}{\sqrt{m_{O1}m_C}} & \frac{2}{m_C} & -\frac{1}{\sqrt{m_C m_{O2}}} \\ 0 & -\frac{1}{\sqrt{m_C m_{O2}}} & \frac{1}{m_{O2}} \end{bmatrix} \begin{bmatrix} C'_{O1} \\ C'_C \\ C'_{O2} \end{bmatrix}$$

which we see is again a real and symmetric matrix for which the eigenvalues are real and the eigenvectors orthonormal.

So let's take $k = 1$ and let $m_C = 12$ and $m_{O1} = m_{O2} = 16$. The eigenvalues and vectors for these values are generated in the python code below.

In [3]:

```
### The first part imports the linear algebra library so it recognises the function nee
ded to calculate the eigenvalues and vectors
import numpy as NP
from scipy import linalg as LA
m_O1=16
m_C=12
m_O2=16
A = NP.array([[1/m_O1,-1/NP.sqrt(m_O1*m_C),0],
              [-1/NP.sqrt(m_O1*m_C),2/m_C,-1/NP.sqrt(m_O2*m_C)],
              [0, -1/NP.sqrt(m_O2*m_C), 1/m_O2]
             ])
print (A)
e_vals,e_vecs = LA.eig(A)
print ("Eigenvalues")
print(e_vals)
print ("Eigenvectors")
print(e_vecs)
```

```
[[ 0.0625     -0.07216878  0.        ]
 [-0.07216878  0.16666667 -0.07216878]
 [ 0.         -0.07216878  0.0625    ]]
Eigenvalues
[ 2.29166667e-01+0.j  6.25000000e-02+0.j -1.29611271e-17+0.j]
Eigenvectors
[[-3.69274473e-01 -7.07106781e-01  6.03022689e-01]
 [ 8.52802865e-01 -1.07262183e-16  5.22232968e-01]
 [-3.69274473e-01  7.07106781e-01  6.03022689e-01]]
```

Firstly we note, that within the numerical errors the third eigenvalue reported is zero. This is as we would expect - it corresponds to a linear translation of the centre of mass of the molecule. Remember that the eigenvectors refer to C' values. Calculate the corresponding values for the C terms. What do you notice happens the third eigenvector? For the first eigenvalue we see that two oxygen atoms are moving with the same amplitude but in the opposite direction to the central carbon atom and we have $\omega^2 = 0.229, \omega = 0.48$. For the second eigenvalue, we see that the carbon atom (to within numerical error) is stationary and the oxygen atoms oscillate around it with a frequency $\omega^2 = 0.0626, \omega = 0.25$ with equal and opposite amplitudes. The first two eigenvectors are the asymmetric and symmetric (breathing) modes of vibration for the molecule. Without calculation can you see why the frequency of the breathing mode is $\omega = 0.25$?

You may like to check that the eigenvectors above and the ones you calculate below are orthonormal ($\vec{a}.\vec{b} = 0$).

Careful spectroscopic measurements might reveal other frequencies to these due to isotopic variations in the Carbon (^{13}C has 1 % abundance) and Oxygen (^{18}O has 0.2% abundance) atoms.

The python code above is copied below. By changing the masses, use this to calculate the eigenvalues and eigenvectors for the following molecules:

$^{16}O -^{12} C -^{18} O$

$^{18}O -^{12} C -^{18} O$

$^{16}O -^{13} C -^{16} O$

$^{16}O -^{13} C -^{18} O$

$^{18}O -^{13} C -^{18} O$

Try to intrepret what this means in terms of the eigenvalues and eigenvectors you obtain. What happens when the three masses are all different? Can you understand what is happening to the third eigenvector and how it can be reconciled to be equivalent to the translation of the whole molecule?

In [4]:

```
### The first part imports the linear algebra library so it recognises the function nee
ded to calculate the eigenvalues and vectors
import numpy as NP
from scipy import linalg as LA
m_O1=18
m_C=12
m_O2=16
A = NP.array([[1/m_O1,-1/NP.sqrt(m_O1*m_C),0],
              [-1/NP.sqrt(m_O1*m_C),2/m_C,-1/NP.sqrt(m_O2*m_C)],
              [0, -1/NP.sqrt(m_O2*m_C), 1/m_O2]
             ])
print (A)
e_vals,e_vecs = LA.eig(A)
print ("Eigenvalues")
print(e_vals)
print ("Eigenvectors")
print(e_vecs)
```

```
[[ 0.05555556 -0.06804138  0.        ]
 [-0.06804138  0.16666667 -0.07216878]
 [ 0.         -0.07216878  0.0625    ]]
Eigenvalues
[ 2.25766751e-01+0.j  5.89554712e-02+0.j -2.95187402e-18+0.j]
Eigenvectors
[[-0.3433881  -0.70055711  0.62554324]
 [ 0.85901399  0.03500568  0.51075392]
 [-0.37970986  0.71273721  0.58976782]]
```

Conclusion

Hopefully with these examples you can see how the effect of different masses changes our eigenvalues and eigenvectors. We have not considered the case of varying k but if you allow for different ks it is easy to see that the K matrix is always symmetrical and this technique still works. The problem with varying masses is that simple manipulation leads to an asymmetric matrix for which the eigenvalues may not necessarily be real and for which the eigenvectors are not orthonormal. The manipulation we used to mass weight the coefficients C in our initial solution to get the C' values is a standard method in thess problems and allows us to recover the symmetric matrix we need to solve our problem. Different ks would be the case with different bonds in our molecule. You may wish to try varying k as well in the case of our triatomic molecule. A chemical example might be hydrogen cyanide (H-C-N). Why not give it a try with $m_H = 1, 2$, $m_C = 12, 13$ and $m_N = 14, 15$ and varying the relative strength of the H-C bond and the C-N triple bond.

In [ ]:

```
```