Investigation into Numerical Integration

School of Physics and Astronomy
University of Manchester

Amir El-hamdy

March 1, 2016

1 Solving a set of coupled equations

In order to see the practicality of matrix methods we will consider a system of springs. For a large enough system, say 100 particles, it would become cumbersome to solve without a computer therefore we should look to solve the problem with matrix methods.

1.1 Find equilibrium for particles connected by random springs in 1D

In this problem we will consider a system of N unit-mass particles in between two fixed points with each particle connected by a spring to its two neighbours. Figure 1 displays the problem for N = 5.

We can simplify the problem by considering the fixed points to lie at $x = 0$ and $x = X$. Ultimately the start and end points of the fixed points do not matter as the entire system is relative and is therefore translation invariant.

If we then write out the equations of motion for each mass and collect terms we can obtain the matrix equation

$$\begin{bmatrix} -k_1 - k_0 & k_1 & 0 & 0 & 0 \\ k_1 & -k_2 - k_1 & k_2 & 0 & 0 \\ 0 & k_2 & -k_3 - k_2 & k_3 & 0 \\ 0 & 0 & k_3 & -k_4 - k_3 & k_4 \\ 0 & 0 & 0 & k_4 & -k_5 - k_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -Xk_5 \end{bmatrix}$$

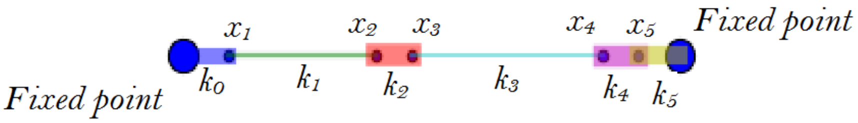


Figure 1: Spring system for N=5

This problem can be easily generalised for $N \geq 0$ and then solved using standard matrix techniques.

1.2 Expand the energy about the equilibrium position

We now wish to find the frequencies of normal mode oscillations about the equilibrium position. We can find this by finding the eigenvalues of the matrix containing the second derivatives of the potential

$$\frac{\partial^2}{\partial \vec{r_i} \partial \vec{r_j}} V \tag{1}$$

We can obtain V by summing the energy stored within each spring

$$V = \frac{1}{2}\left(k_0 \left[x_1 - 0\right]^2 + k_1 \left[x_2 - x_1\right]^2 + ... + k_5 \left[X - x_5\right]^2\right) \tag{2}$$

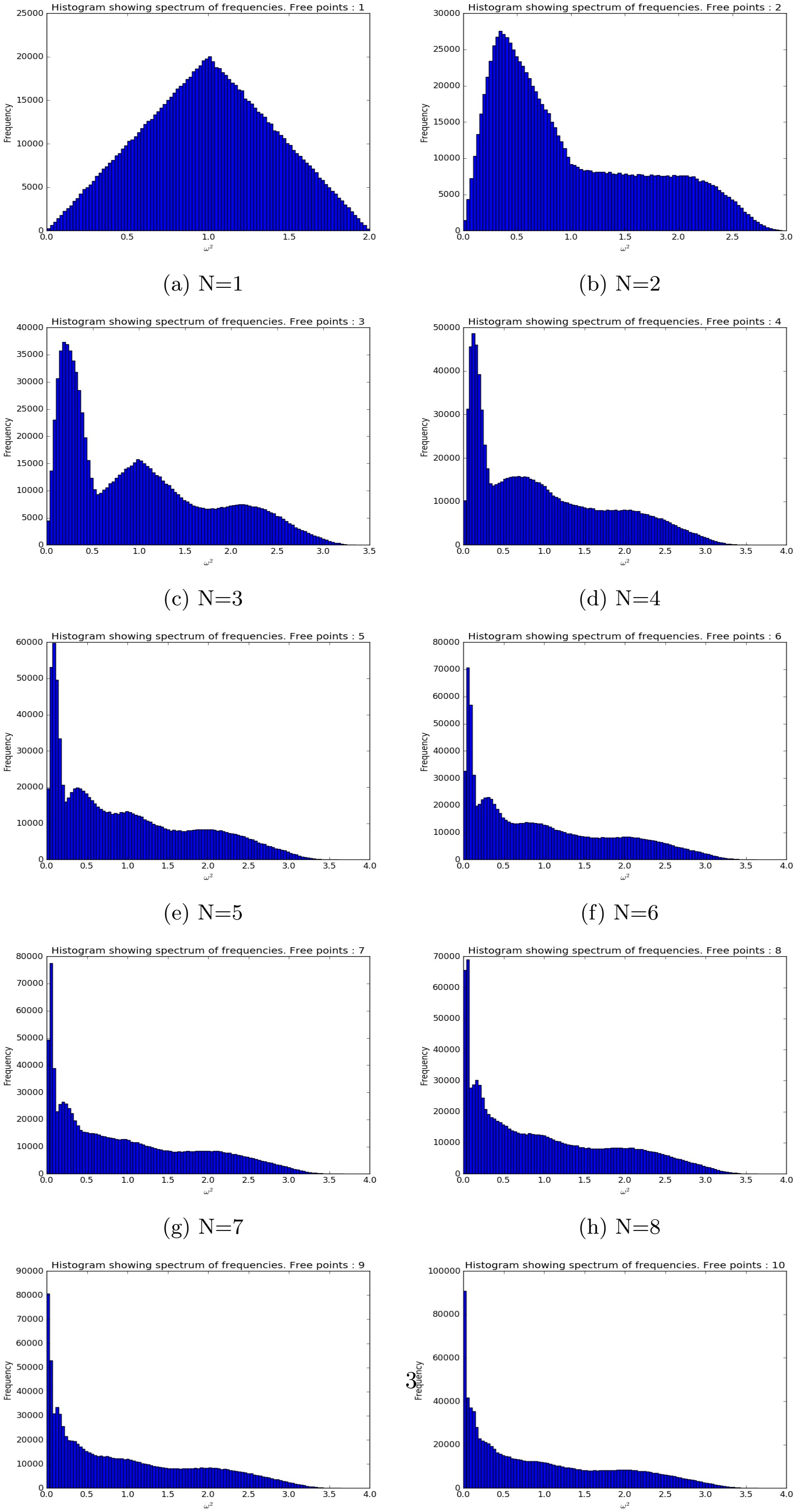
We can then write down the matrix of derivatives as

$$\frac{\partial^2}{\partial \vec{r_i} \partial \vec{r_j}} V = \begin{bmatrix} k_0 + k_1 & -k_1 & 0 & 0 & 0 \\ -k_1 & k_1 + k_2 & -k_2 & 0 & 0 \\ 0 & -k_2 & k_2 + k_3 & -k_3 & 0 \\ 0 & 0 & -k_3 & k_3 + k_4 & -k_4 \\ 0 & 0 & 0 & -k_4 & k_4 + k_5 \end{bmatrix} \tag{3}$$

The eigenvalues of this matrix can then be found using standard matrix techniques.

1.3 Analyse the spectrum of normal mode frequencies

Now that we have a method to obtain the frequency of normal mode oscillations we can apply this to networks of different sizes.
Figure 2shows the frequency distribution of networks size N=1-10. The total for each plot is 1,000,000 values.



(a) N=1

(b) N=2

(c) N=3

(d) N=4

(e) N=5

(f) N=6

(g) N=7

(h) N=8

(i) N=9

(j) N=10

Figure 2: Histogram plots displaying frequency spectrums for different networks

Finally we can observe the frequency spectrum obtained when we sum the results displayed in Figure 2. This spectrum is shown in Figure 3

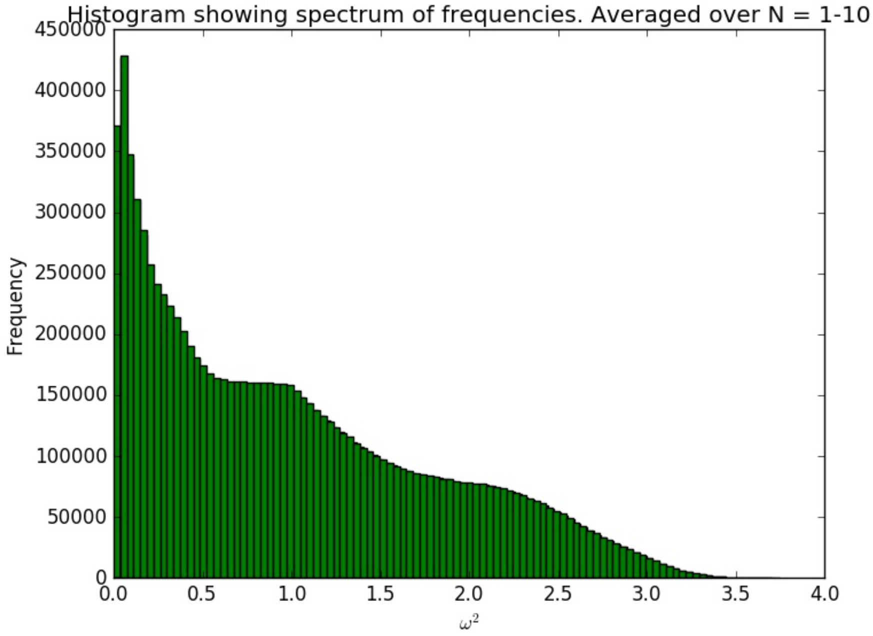


Figure 3: Cumulative frequency spectrum for N=1-10

2 Applying matrix methods to the quartic oscillator in Quantum Mechanics

2.1 Separating odd from even states

The example we shall look at is the eigenstates of the quartic oscillator (we shall use $\hbar = m = \omega = 1$)

$$\hat{H} = \frac{\hat{p}^2}{2} + \frac{1}{2}x^2 + \frac{\lambda}{4}x^4 \tag{4}$$

We use $|n\rangle$ for the eigenstates of the Harmonic oscillator.

$\left[\frac{\hat{p}^2}{2} + \frac{1}{2}x^2\right]|n\rangle = (n + \frac{1}{2}|n\rangle)$

It can be shown that

$$
\begin{aligned}
4x^4|n\rangle =& \sqrt{(n+1)(n+2)(n+3)(n+4)}|n+4\rangle \\
&+ 2(2n+3)\sqrt{(n+1)(n+2)}|n+2\rangle \\
&+ 3(2n^2+2n+1)|n\rangle \\
&+ 2(2n-1)\sqrt{n(n-1)}|n-2\rangle \\
&+ \sqrt{n(n-1)(n-2)(n-3)}|n-4\rangle
\end{aligned}
$$

Thus we can produce a matrix A_{mn} with components given by the function h defined below.

```
def h(m,n,lam):
    fac=lam/16.0
    if m-n==4:
        return fac*np.sqrt(1.0*(n+1)*(n+2)*(n+3)*(n+4))
    elif m-n==2:
        return fac*2*(3+2*n)*np.sqrt(1.0*(n+1)*(n+2))
    elif n==m:
        return fac*(3+6*n+6*n*n)+n+0.5
    elif m-n==-2:
        return fac*2*(2*n-1)*np.sqrt(1.0*n*(n-1))
    elif m-n==-4:
        return fac*np.sqrt(1.0*n*(n-1)*(n-2)*(n-3))
    else:
        return 0
```

Clearly we can see odd and even states are independent as in order to produce an even number by the result of $m - n$ both numbers must be even or odd.
This simplifies the problem as we can now separate the large NxN matrix into two smaller AxA and BxB matrix. Where A and B are given by ceil(N/2) and floor(N/2) representing odd and even states respectively.
Since the two sets of linear equations are independent the eigenvalues for each matrix can be solved separately. Due to the great reduction in size of the matrix it is easier to calculate the eigenvalues like this and compile the eigenvalues at the end.

2.2 Plotting lowest eigenenergy against λ

A plot of the lowest eigenenergy vs λ is shown in Figure 4

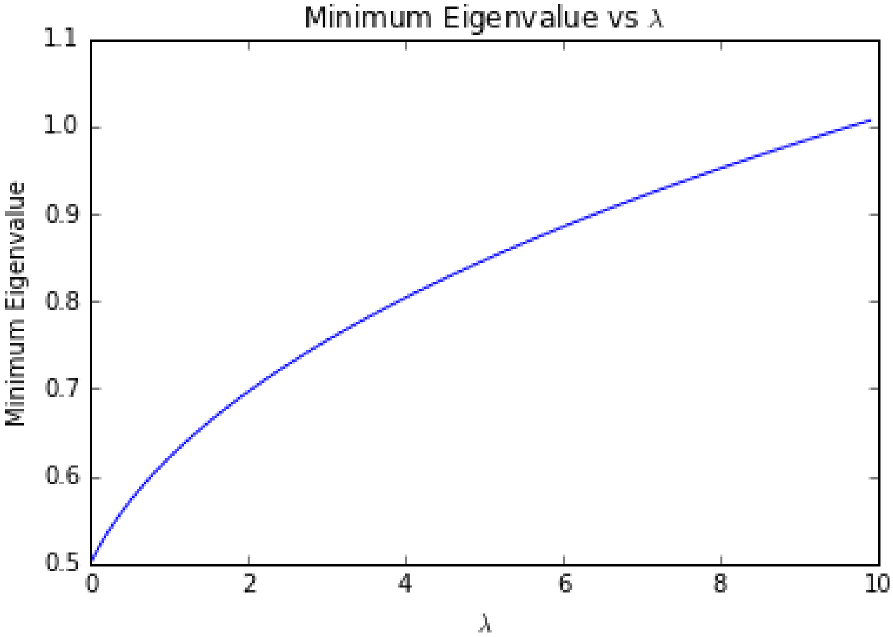


Figure 4: Lowest eigenenergy against *lambda*

2.3 Good eigenvalue for $\lambda = 100$

In order to find an eigenvalue for a given λ, we must check that the value produced does not change appreciably for a larger matrix size.
Figure 5 below shows the minimum eigenvalue for $\lambda = 100$ for different matrix sizes.

Minimum eigenvalue	Matrix size
19.25	1
2.52894319219	10
2.00199641611	100
2.00199638415	1000

Figure 5: Table of minimum eigenvalues for $\lambda = 100$ for different matrix sizes

Thus clearly we can ascertain that 2.00199 is an eigenvalue to a good approximation.