Linear Variational Method

January 25, 2019

1 Computational Exercise 1: Linear Variational Principle

We will apply the Linear Variational Method to the particle in a box of length 10 atomic unitw with a delta function potential centered at $x = 5$ atomic units. In particular, we will optimize the trial wavefunction given by

$$\Phi(x) = \sum_{i=1}^{N} c_i \psi_i(x) \tag{1}$$

where the coefficients c_i are real numbers and $\psi_i(x)$ are the energy eigenfunctions of the particle in a box with no potential:

$$\psi_n(x) = \sqrt{\frac{2}{10}} \sin\left(\frac{n\pi x}{10}\right). \tag{2}$$

We will seek to minimize the energy functional through the expansion coefficients, where the energy functional can be written as

$$E[\Phi(x)] = \frac{\int_0^\infty \Phi^*(x)\, \hat{H}\, \Phi(x) dx}{\int_0^\infty \Phi^*(x)\, \Phi(x) dx}. \tag{3}$$

The Hamiltonian operator in the box is given by

$$\hat{H} = -\frac{\hbar^2}{2m}\frac{d^2}{dx^2} + \delta(x-5); \tag{4}$$

in natural units, $\hbar$ and m are equal to 1.
As we saw in class, $E[\Phi(x)]$ can be expanded as

$$E[\Phi(x)] \sum_{i=1}^{N}\sum_{j=1}^{N} c_i c_j S_{ij} = \sum_{i=1}^{N}\sum_{j=1}^{N} c_i c_j H_{ij} \tag{5}$$

where

$$S_{ij} = \int_0^L \psi_i(x)\psi_j(x) dx = \delta_{ij} \tag{6}$$

and

$$H_{ij} = \int_0^L \psi_i(x)\hat{H}\psi_j(x) dx. \tag{7}$$

1.0.1 Questions Part 1:

1. Work out a general expression for the integrals H_{ij}
2. Write a python function that takes the indices i and j and returns the value of the integral H_{ij}. Skeleton code for this function follows.
3. Show that differentiating the energy functional with respect to all coefficients and setting the derivative to zero results in the following set of equations:

$$\sum_{i}^{N} H_{ik} c_i = E[\Phi(x)] c_k \tag{8}$$

This can be written as an eigenvalue equation

$$\mathbf{Hc} = E\mathbf{c}, \tag{9}$$

where $\mathbf{H}$ is the matrix whose elements are given by H_{ij} and $\mathbf{c}$ is the vector of coefficients.

```
In [ ]: ### Function to return integrals involving Hamiltonian and basis functions
        def H_ij(i, j):
            ### if i==j, you need to worry about kinetic and potential
            ### if i!=j, you only need to worry about potential...
            ### so check if i==j and handle the two cases accordingly
            ### store the result in the variable called ham_int

            if i==j:
                ### code to evaluate integral H_ii here!
            else:
                ### code to evaluate H_ij here!

            return ham_int
```

Create an array called H_mat that can be used to store the Hamiltonian matrix elements. We can start by considering a trial wavefunction that is an expansion of the first 3 PIB energy eigenfunctions, so our Hamiltonian in this case should be a 3x3 matrix.

```
In [ ]: import numpy as np
        H_mat = np.zeros((3,3))
```

You can use two nested *for* loops along with your H_{ij} function to fill out the values of this matrix.

```
In [ ]: ### loop over indices of the basis you are expanding in
        ### and compute and store the corresponding Hamiltonian matrix elements
        for i in range(0,3):
            for j in range(0,3):
                H_mat[i][j] = H_ij(i, j)

        ### Print the resulting Hamiltonian matrix
        print(H_mat)
```

Before systematically identifying the optimal coefficients, it is instructive to try a few "trial" wavefunctions "by hand". A few suggestions include:

$$\mathbf{c} = (1,0,0)\ \ \mathbf{c} = (0,1,0)\ \ \mathbf{c} = (0,0,1) \tag{10}$$

$$\mathbf{c} = \left(\sqrt{1/2},\sqrt{1/2},0\right)\ \ \mathbf{c} = \left(0,\sqrt{1/2},\sqrt{1/2}\right)\ \ \mathbf{c} = \left(\sqrt{1/2},0,\sqrt{1/2}\right) \tag{11}$$

In Matrix form, the energy functional can be computed as follows:

$$E = \mathbf{c}^t\mathbf{H}\mathbf{c} \tag{12}$$

where $\mathbf{c}^t$ is just the transpose of $\mathbf{c}$. Using numpy, this can be done with the Hamiltonian matrix defined abouve and a vector $\mathbf{c} = (1,0,0)$ as follows:

```
In [ ]: ### create an empty numpy array for the c vector
        c = np.zeros(3)
        ### assign c vector to be (1, 0, 0)
        c[0] = 1

        ### compute H_mat * c and store it to a new array called Hc
        Hc = np.dot(H_mat,c)

        ### compute c^t * Hc and store it to a variable E
        E = np.dot(np.transpose(c),Hc)

        ### print the result
        print(E)
```

Continue evaluating the energy of different trial wavefunctions by changing the values of the **c** vector and repeating the calculation above. Does increasing the contribution of excited states impact the energy as you expect? Explain.

Finally, to get the optimal values of the **c** vector, we can find the set of vectors (there will be 3) that satisfy the eigenvalue equation we wrote before. We can use the *eig* function of numpy to do this in one line:

```
In [ ]: ### compute eigenvalues and eigenvectors of H_mat
        ### store eigenvalues to E_opt and eigenvectors to c_opt
        E_opt, c_opt = np.linalg.eig(H_mat)

        ### print lowest eigenvalues corresponding to the
        ### variational estimate of the ground state energy
        print(E_opt[0])

        ### print coefficients that define the trial wavefunction with the lowest eigenvalue
        ### which corresponds to the variational estimate of the ground state wavefunction
        print(c_opt[0])
```

1.0.2 Questions Part 2:

1. Is the energy you calculated above higher or lower than the ground state energy of the ordinary particle in a box system (without the delta function potential)?
2. Why do you think mixing in functions that correspond to excited states in the ordinary particle in a box system actually helped to improve (i.e. lower) your energy in the system with the delta function potential?
3. Increase the number of basis functions to 6 (so that **H** is a 6x6 matrix and **c** is a vector with 6 entries) and repeat your calculation of the variational estimate of the ground state energy. Does the energy improve (lower) compared to what it was when 3 basis functions were used?