Winter School Hückel Homework

Fernando Mata Rabadán

1 Theoretical background

The focus of this homework is to write a program that is able to obtain the eigenvalues and eigenstates of the Hückel Hamiltonian in carbon chains and rings. The Hückel Hamiltonian allows calculating the energies of conjugated π systems.

The Hückel Hamiltonian is built based on two parameters: α and β . They are defined as:

$$\alpha = -\frac{2}{2r^2} \qquad \beta = -\frac{1}{2r^2} \tag{1}$$

Where r is the distance of the electron with respect to the atom. The parameter α is the energy of an electron in a 2p orbital. The β parameter represents the energy of stabilization of an electron due to the adjacent atoms. This definition shows that the value of α depends on the atom type, as the size of the 2p orbital will vary with the atom. Also, the value of beta depends on the distance to the neighbouring atoms as it will affect the stabilization.

Both α and β are negative quantities, as they represent stabilization with respect to a free electron. The Hückel Hamiltonian matrix is built with α as the diagonal terms and β in the off-diagonal terms of interacting atoms. A simple example of pentane atoms would be:

$$\hat{H}_{\text{H\"{u}ckel}} = \begin{pmatrix} \alpha & \beta & 0 & 0 & 0 \\ \beta & \alpha & \beta & 0 & 0 \\ 0 & \beta & \alpha & \beta & 0 \\ 0 & 0 & \beta & \alpha & \beta \\ 0 & 0 & 0 & \beta & \alpha \end{pmatrix}$$
(2)

Where the first atom is connected to the second, the second to the third and so on. By diagonalizing this Hamiltonian, the energies and wave functions of the system are found:

$$\hat{H}_{\text{H\"{u}ckel}} |\Psi_i\rangle = E_i |\Psi_i\rangle \tag{3}$$

The Hückel Hamiltonian can also be applied to rings. In this case, this can be done by including the matrix elements that connect the first and last atoms. In the case of cyclopentane it would be:

$$\hat{H}_{\text{H\"{u}ckel}} = \begin{pmatrix} \alpha & \beta & 0 & 0 & \beta \\ \beta & \alpha & \beta & 0 & 0 \\ 0 & \beta & \alpha & \beta & 0 \\ 0 & 0 & \beta & \alpha & \beta \\ \beta & 0 & 0 & \beta & \alpha \end{pmatrix}$$

$$(4)$$

As the quantity of interest is the relative energy difference between states, it is possible to set $\alpha = 0$, as this does not affect the relative energy of eigenvalues. When there are different types of atoms, the value of α

will not be the same, and thus this must be reflected in the Hamiltonian matrix. An example of a chain of 5 alternating atoms would be:

$$\hat{H}_{\text{H\"{u}ckel}} = \begin{pmatrix} \alpha & \beta & 0 & 0 & 0 \\ \beta & \alpha' & \beta & 0 & 0 \\ 0 & \beta & \alpha & \beta & 0 \\ 0 & 0 & \beta & \alpha' & \beta \\ 0 & 0 & 0 & \beta & \alpha \end{pmatrix}$$
(5)

Where α corresponds to the first atom type and α' to the second. Similarly, the value of β depends on the interatomic distance.- Therefore, it is possible to express differences in bond lengths between atoms setting two different values of beta β^+ and β^- . In the case of pentadiene:

$$\hat{H}_{\text{Hückel}} = \begin{pmatrix} \alpha & \beta^{+} & 0 & 0 & 0 \\ \beta^{+} & \alpha & \beta^{-} & 0 & 0 \\ 0 & \beta^{-} & \alpha & \beta^{+} & 0 \\ 0 & 0 & \beta^{+} & \alpha & \beta^{-} \\ 0 & 0 & 0 & \beta^{-} & \alpha \end{pmatrix}$$
(6)

Where β^+ corresponds to a longer bond distance and β^- to a shorter one.

2 Program review

The initial intent was to write the program in FORTRAN, but due to problems with my compiler I could not. When calling a matrix diagonalization using llapack, the eigenvalues and eigenvectors were mismatched. Due to this, the program was rewritten in python, where this problem didn't occur.

2.1 Code analysis

The code is divided in two main parts: a class Chain that handles all the properties related to the Hückel Hamiltonian and a function run_calculation that handles the general flow of a calculation. The advantage of working with a class is that it is possible to create various instances of these chains or rings and compare results and plot them easily and directly.

2.1.1 The Chain class

The Chain class compacts both the calculation and plotting of the eigenvalues and eigenvectors of the system Hamiltonian. The Chain class has the following methods that allow the modification of internal variables:

These internal variables are accessed when building the Hamiltonian, modifying the different possible cases.

Chain.huckel_matrix: The Hamiltonian is stored as a property. The function considers different cases in the following way:

• Alternating bond distances: in order to simulate different bond distances, the value of the β value is modified in the Hamiltonian. In this case, the value of β^+ is set always to -1 while the value of β^- can be set to a different value with Chain.set_beta_minus. To take into account this alternation of betas, the Chain.huckel_matrix modifies the Hamiltonian matrix with:

```
for i in range(self.n_atoms - 1):
    if i % 2 == 0:
        huckel_matrix[i + 1, i] = huckel_matrix[i, i + 1] = -1
    else:
        huckel_matrix[i + 1, i] = huckel_matrix[i, i + 1] = (
            self._beta_minus
    )
```

It iterates over the matrix and changes the value of beta when the index is odd.

• Alternating atoms: Similarly to alternating distances, to take into account alternating atoms, it is possible to set a different value to the α value of the diagonal matrix elements with Chain.set_alpha_prime. As with the previous property, only the possibility of modifying one of the two α values was considered, and this is reflected in the Hamiltonian matrix with:

```
for i in range(self.n_atoms):
    if i % 2 == 0:
        huckel_matrix[i, i] = huckel_matrix[i, i] = 0
    else:
        huckel_matrix[i, i] = huckel_matrix[i, i] = self.
        _alpha_prime
```

• Rings: Rings are built by connecting the first and last atoms of a given chain. This is done explicitly as:

```
if self._ring:
    huckel_matrix[0, self.n_atoms - 1] = huckel_matrix[self.
    n_atoms - 1, 0] = -1
```

The condition of the object being a ring is checked and if so the last values of the first row and column are adjusted accordingly.

Chain.eigen: Once the Hamiltonian matrix is built, it can be diagonalized to obtain the eigenvalues and eigenvectors. The Chain.eigen stores as a property the eigenvalues and eigenvectors and calculates it via:

```
def eigen(self) -> Tuple[np.ndarray, np.ndarray]:
    self._eigenvalues, self._eigenvectors = np.linalg.eig(
        self.Huckel_matrix
)

# Sort eigenvalues and eigenvectors
    idx = np.argsort(self._eigenvalues) # returns the array with
        sorted indexes
    sorted_eigenvalues = self._eigenvalues[idx]
    sorted_eigenvectors = self._eigenvectors[:, idx]

self._eigenvalues = sorted_eigenvalues
    self._eigenvectors = sorted_eigenvectors
```

The Hückel matrix is diagonalized using the numpy linalg.eig function. Then the pairs of eigenvalues and eigenvectors are sorted to return them in ascending order.

Finally, the Chain class has two plotting functions:

```
Chain.plot_eigenvalues() # plots the eigenvalues of the Huckel
Hamiltonian
Chain.plot_eigenvectors() # plots the eigenvectors of the Huckel
Hamiltonian
```

2.1.2 The run_calculation Function and input handling

The general scheme of any calculation using the program would be:

- 1. Read the input.
- 2. Modify properties accordingly to the input.
- 3. Build and diagonalize the Hückel Hamiltonian.
- 4. Return the results in a log file and the eigenvalues and eigenvectors in different files for external plotting.

This workflow is managed by the run calculation function. The input file is read with:

```
def run_calculation(input_filename:str, output_filename:str='') ->
   None:
        content = load_input(input_filename)

if content is None:
        return

keyval_pairs = [
        tuple(line.replace('=', 'u').strip().split()[:2])
        for line in content
        if len(line.replace('=', 'u').strip().split()) >= 2
]

cont = dict(keyval_pairs)

if 'n_atoms' not in cont.keys():
        raise ValueError('Definition_of_number_of_atoms_is_mandatory')
```

The previous code reads all the content in the input file and generates a dictionary of property-value pairs. This way, it is possible to easily build the Chain object with the correct properties. Also, an error is raised if there is no information of the number of atoms of the system. Then the Chain object is created, and the properties modified with:

```
c = Chain(n_atoms=int(cont['n_atoms']))

if 'ring' in cont.keys():
    if cont['ring'].capitalize() == 'True':
        c.close_ring()

if 'alternate_alpha' in cont.keys():
    alt_alpha = float(cont['alternate_alpha'])
    c.set_alpha_prime(alt_alpha)

if 'alternate_beta' in cont.keys():
    alt_beta = float(cont['alternate_beta'])
    c.set_beta_minus(alt_beta)
```

Finally, the eigenvalues and eigenvectors are calculated with:

```
eigenval, eigenvec = c.eigen
```

And the results written in the corresponding files with:

```
with open(output_filename + '_eigenvalues', 'w') as f:
    for eigen in eigenval:
        f.write(f'{eigen:8.5f}')

with open(output_filename + '_eigenvectors', 'w') as f:
    for vec in eigenvec:
    f.write('u'.join(f'{val:>8.5f}' for val in vec))
```

In order to run the program from the terminal line, the sys module is imported, which allows reading the command line arguments. An error is raised if no input file is provided.

2.2 Usage and examples

The program can be run from the command line by:

```
python3 huckel.py input_file output_file
```

Where the specification of the output file is optional. A built-in help message is prompted in case of error. The input file should contain a list of keywords:

```
n_atoms = 100
```

And optional values such as cyclization, distance alternation and atom alternation can be set with:

```
ring = False
alternate_alpha = 0.0
alternate_beta = -1.0
```

The input is case-sensitive, and the program will not run if keywords are not exactly the same. Also, the values True and False must have the first capital letter.

An advantage of working with classes is that the simplicity to calculate and plot different chains and cases, for example to compare the open chain and ring cases it would be:

```
from huckel import Chain
a = Chain(10)
b = Chain(10)

b.close_ring()

a.plot_eigenvalues()
b.plot_eigenvalues()
```

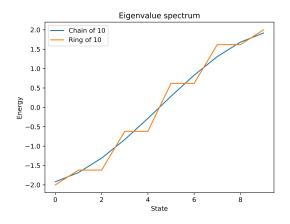
And the same with the set_alpha_prime() and set_beta_minus() methods. All the scripts to obtain the figures in the next section can be found in the plots/ directory.

3 Results

Having defined the whole framework in the previous class, it is possible to calculate the eigenvalues and eigenvectors of different systems.

3.1 Chain vs Ring

First, the eigenvalues and eigenvectors of both a chain and a ring of 10 atoms were calculated. The results are shown in Figure 1. It is possible to appreciate a clear difference between the two cases, with the ring having a twofold degeneracy in the eigenvalues. Eigenvectors of the first states are shown in Figures 3 and 4.



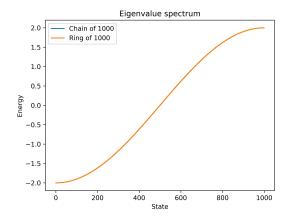
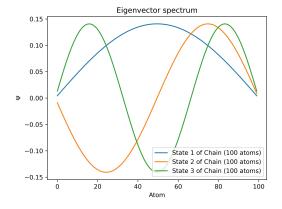


Figure 1: Comparison of eigenvalues for a chain and a ring of 10 atoms.

Figure 2: Comparison of eigenvalues for a chain and a ring of 100 atoms.

There is a big difference in the eigenvector spectrum. While the wave function of the ground state is localized in the central atoms in the chain as in the particle in a box, the wave function of the ring is delocalized over all the atoms. This is a consequence of the periodic boundary conditions of the ring introduced in the Hückel Hamiltonian. Due to the first and last atoms being connected, the ground state wave function will allow free movement of electrons in the wave function, resulting in a delocalized state.



Eigenvector spectrum

0.15

0.10

0.05

-0.05

-0.05

State 1 of Ring (100 atoms)
State 2 of Ring (100 atoms)
State 3 of Ring (100 atoms)
Atom

Atom

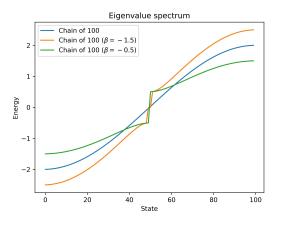
Figure 3: Eigenvectors for a chain of 10 atoms.

Figure 4: Eigenvectors for a ring of 10 atoms.

Finally, the eigenvalues of the infinite chain/ring limit (considering that a chain/ring of 1000 atoms converges to said limit) is shown in Figure 2. It is possible to see that, as the ring is so large, the eigenvalues of the ring converge to the eigenvalues of the chain. This can be explained considering that an infinitely big ring will behave in the same way as an infinite chain, due to the periodic boundary conditions only being applied at infinity.

3.2 Alternating atom distances

It is possible to simulate the effect of different alternating atoms distances. We refer to the definition of β in equation 1. A higher distance between atoms will result in a smaller value of β , implying that the stabilization due to the neighbouring atoms will be lower. On the other hand a larger value of beta implies proximity and thus higher stabilization. To test the effect of this, figures 5 and 6 show the eigenvalues of a chain and a ring with larger and smaller distances between atoms. Figures 7 and 8 show the eigenvectors of the first states.



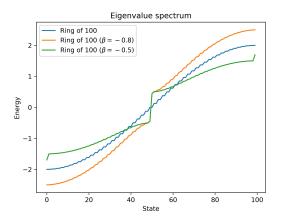


Figure 5: Eigenvalues for a chain with alternating bond distances.

Figure 6: Eigenvalues for a ring with alternating bond distances.

The first thing that can be noticed is the difference in the eigenvalues. Both the chain and ring with alternating distances present a new band structure, with an energy gap between them. The energy cap can be seen to be symmetrical with respect to 0, and is the same independently of the different beta value. On the other hand, the energies of the bands up to the gap differ, with the systems with a smaller beta presenting lower ground state energies.

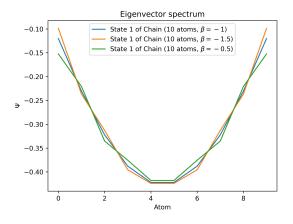


Figure 7: Eigenvectors for a chain with alternating bond distances.

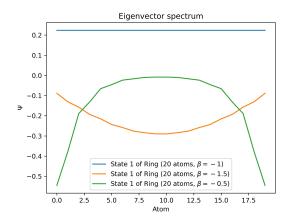


Figure 8: Eigenvectors for a ring with alternating bond distances.