Day 2 Hands on: DFT Bread and Butter

Tutors: Chiara Cignarella / Alberto Carta / Nelson Dzade / George Manyali / James Sifuna / Fatema Mohamed / Omamuyovwi Rita Jolayemi / Maram Ali Ahmed Musa

Brief description of the tutorials:
1) Single DFT calculations: benzene and graphene
2) Silicon: convergence test and lattice optimization
3) Optional: carbyne chain, convergence test for vacuum

Topics of the session

- 1. The basics: benzene and graphene
  - Single molecule calculations: benzene
  - Basics of post-processing: plotting wavefunctions
  - Periodic systems: graphene
- 2. DFT bread and butter: convergence tests and lattice optimization
  - Basic convergence tests on fcc Si: energy cutoff and k-grid
  - Optimizing the lattice constant: Extracting physical information: bulk modulus
- 3. Optional: the carbyne chain
  - Converge test and structure optimization

To get the latest version of the exercises, in the ASESMA 2025 folder execute: `git pull`

About Quantum ESPRESSO

- More info about Quantum ESPRESSO can be found in:
  - https://www.quantum-espresso.org/
- Quantum ESPRESSO (QE) documentation:
  - on-line (pw.x code) and (pp.x code) manuals and

  for input file description
- Useful resources:
  - Input generator from materials cloud
  - Pseudopotentials SSSP library and GBRV library

Exercise 1.1 : Benzene

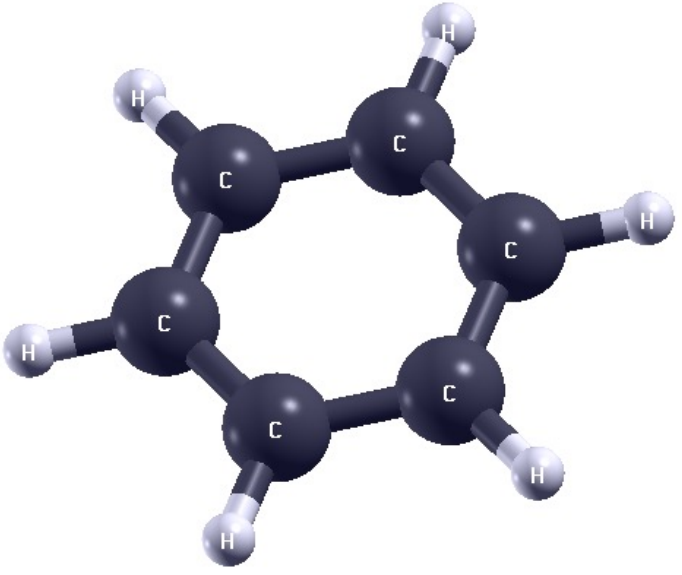
How to calculate and plot molecular orbitals of benzene ($(sign\ \psi(r)) \cdot |\psi(r)|^2$)

1. Exercise 1: Benzene

Step 0: View the Benzene Molecule

```
xcrysden --pwi pw.benzene.scf.in
```

Move the mouse around to take a look at the molecule.
The ".pwi" format stands for quantum espresso pw.x input file.
For a pw.x output file, that would be ".pwo”

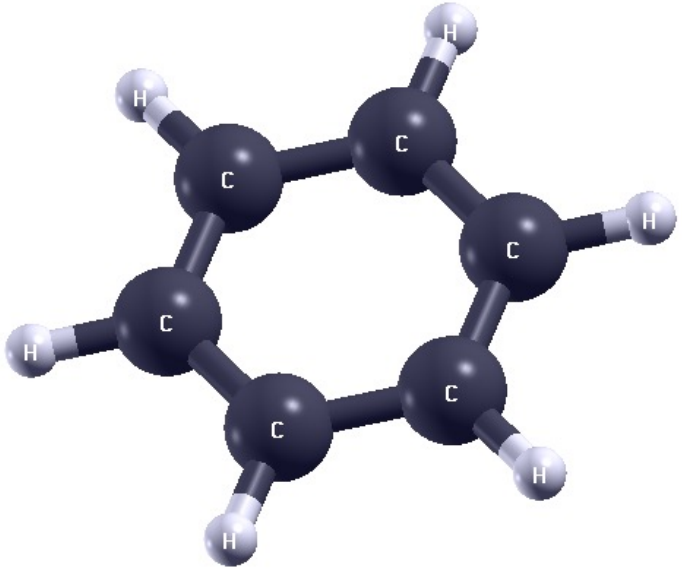


1. Exercise 1: Benzene

Step 0: View the Benzene Molecule

```
xcrysden --pwi pw.benzene.scf.in
```

Move the mouse around to take a look at the molecule.
The ".pwi" format stands for quantum espresso pw.x input file.
For a pw.x output file, that would be ".pwo"



Step 1: Perform the SCF Calculation

```
pw.x –in pw.benzene.scf.in > pw.benzene.scf.out
```

The pw.x code will now perform a DFT self - consistent field (SCF) calculation.

1. Exercise 1: Benzene

Step 2: Postprocess the wavefunction data

```
pp.x –in pp.benzene.psi2.in > pp.benzene.psi2.out
```

The resulting wavefunction amplitudes $sign\,\psi(r)) \cdot |\psi(r)|^2$ are written to files psi2.benzene_K001_B0*.xsf

1. Exercise 1: Benzene

Step 2: Postprocess the wavefunction data

```
pp.x –in pp.benzene.psi2.in > pp.benzene.psi2.out
```

The resulting wavefunction amplitudes $sign\,\psi(r)) \cdot |\psi(r)|^2$ are written to files psi2.benzene_K001_B0*.xsf

Step 3: Plot a single molecular orbital

```
xcrysden --xsf psi2.benzene_K001_B006.xsf
```

Make a fancy display of the molecular orbital (follow the instructions of the tutor). The README.md contains the info necessary to save the current state view. You can try and save that as MO-state.xcrysden and then use the same view with another orbital:

```
xcrysden --xsf psi2.benzene_K001_B006.xsf –script MO-state.xcrysden
```

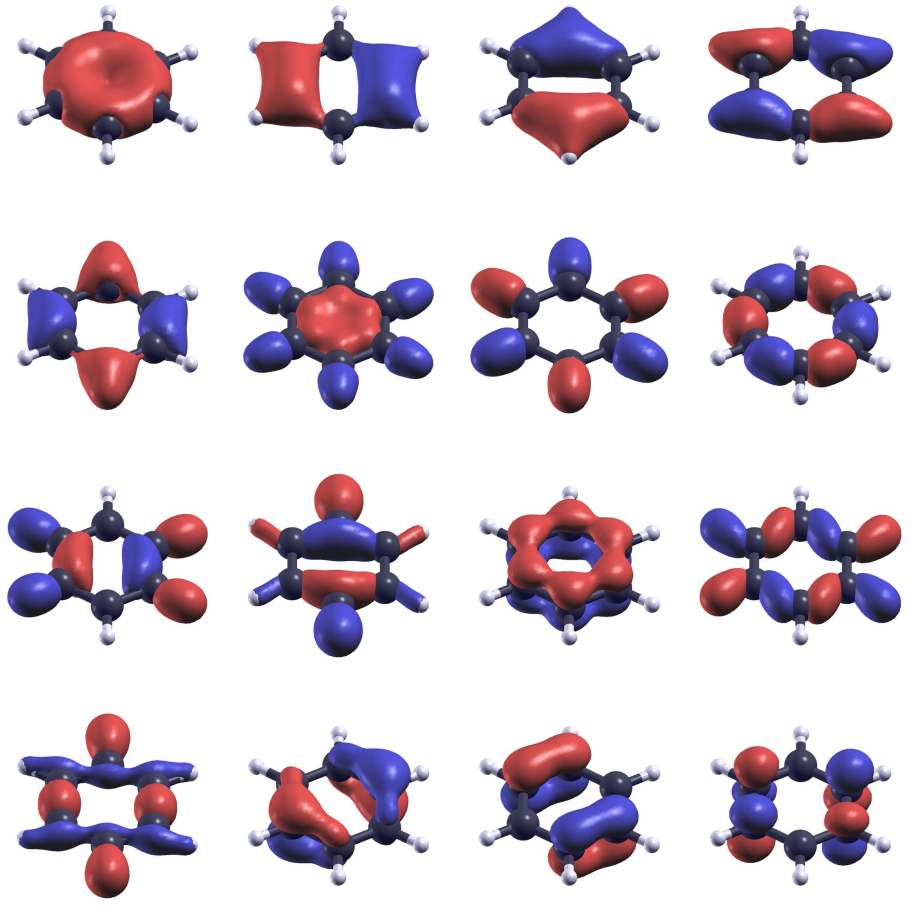
1. Exercise 1: Benzene

Step 4: Plot All Molecular Orbitals

Run in the terminal:

```
bash plot-psi2.sh
```

The shell script is trying to plot all wavefunctions you computed, be patient (this will take a while).



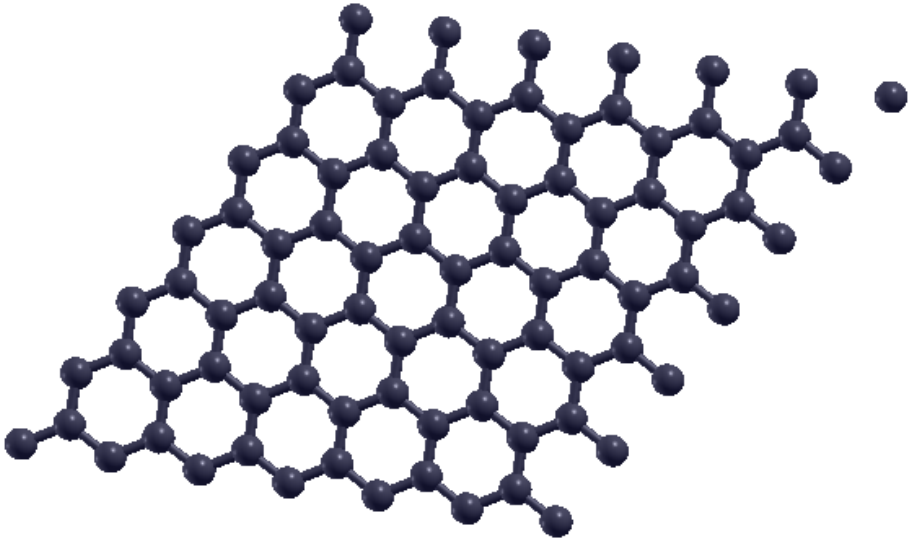
Exercise 1.2: Graphene

Introduction to periodic system

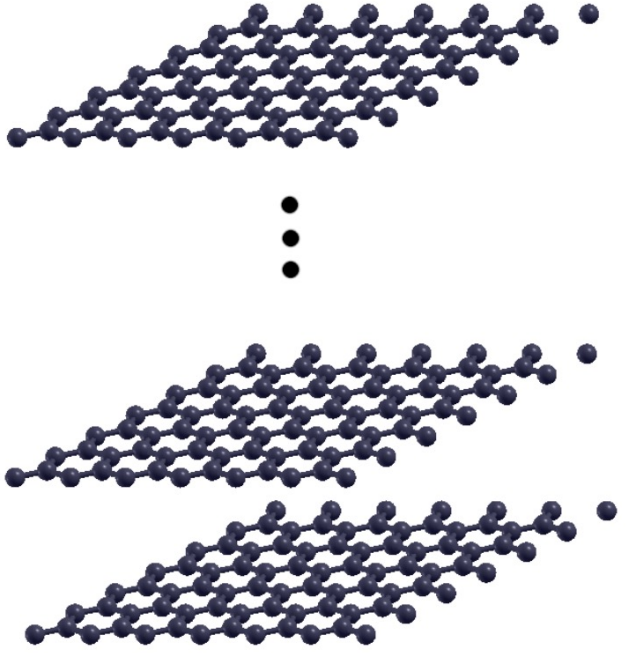
1. Exercise 2: Graphene

Graphene is a single sheet of carbon atoms

Periodic boundary conditions are applied in the plane



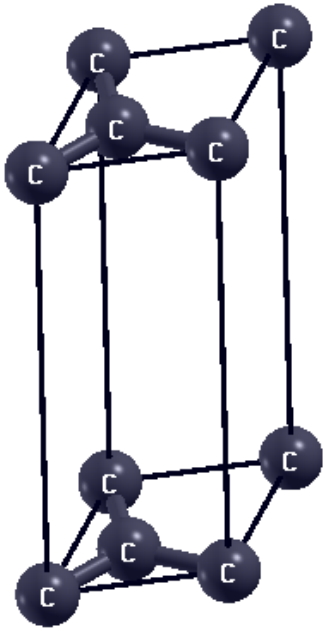
But in quantum espresso every direction is periodic

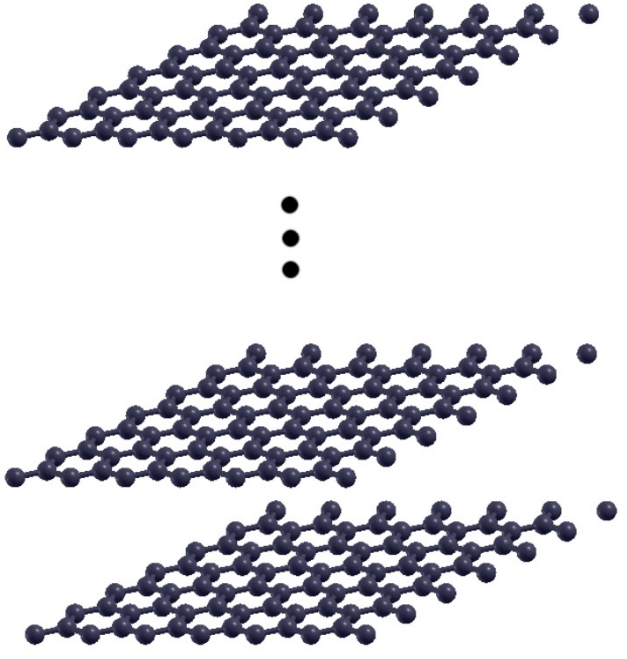


1. Exercise 2: Graphene

Graphene is a single sheet of carbon atoms

We can describe the graphene sheet with just 2 atoms in the unit cell





1. Exercise 2: DOS of Graphene

The scheme to compute the DOS is the following:

- **Perform the SCF Calculation:** Use pw.x to calculate the density (calculation = 'scf')

```
pw.x –in pw.graphene.scf.in > pw.graphene.scf.out
```

- **Perform the NSCF Calculation**: Use pw.x to calculate the electronic eigenvalues on more k-point grids (calculation = 'nscf')

```
pw.x –in pw.graphene.nscf.in> pw.graphene.nscf.out
```

- **Calculate DOS datafile**: Use dos.x to compute the total DOS and save it to *graphene.dos*.

```
dos.x –in dos.graphene.in > dos.graphene.out
```

- **Plot DOS with gnuplot**: Use gnuplot to plot the DOS saved in *graphene.dos*.

```
gnuplot dos.gp
```

1. Exercise 2: DOS of Graphene

Replot DOS shifted to the Fermi energy

- **Find the Fermi energy:**

```
grep “Fermi” pw.graphene.nscf.out
```

- Use the Fermi energy value from the output and edit dos.gp files accordingly.

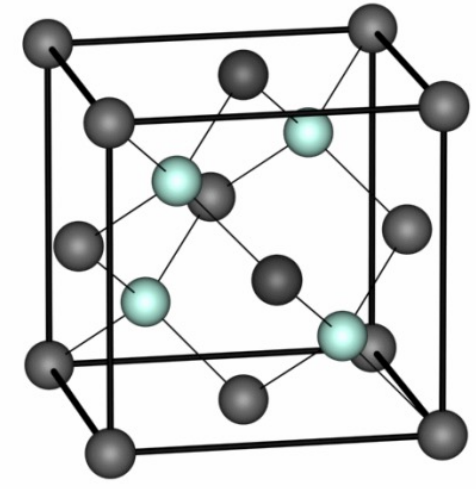
```
gnuplot dos.gp
```

Exercise 2 : Silicon

Bread and butter of real DFT calculations

2. Exercise 1: Silicon

bulk silicon is a face-centered cubic (FCC) lattice with 2 atoms in the unit cell at positions [0 0 0] and [1/4 1/4 1/4] (this is also called diamond or zinc-blend structure)



```
&CONTROL
calculation = 'scf',
prefix = 'silicon',
pseudo_dir = '../../pseudo/',
outdir = './tmp'
/
&SYSTEM
ibrav = 2,
celldm(1) = 10.2,
nat = 2,
ntyp = 1,
ecutwfc = 28,
/
&ELECTRONS
/
ATOMIC_SPECIES
Si 28.086  Si.pbe-rrkj.UPF
ATOMIC_POSITIONS
Si              0.00.   0.00.   0.00
Si              0.25.   0.25    0.25
K_POINTS automatic
4 4 4    1 1 1
```

2. Exercise 1: Silicon

- ibrav=2: meaning FCC lattice
- Just one: celldm(1)=10.2, lattice parameter *a* in Bohr
- nat=2: two atoms
- ntyp=1: one distinct atomic specie

- Where are the atoms located in the unit cell? See card ATOMIC POSITIONS: here, in Cartesian axes, in units of *a*

```
&CONTROL
calculation = 'scf',
prefix = 'silicon',
pseudo_dir = '../../pseudo/',
outdir = './tmp'
/
&SYSTEM
ibrav = 2,
celldm(1) = 10.2,
nat = 2,
ntyp = 1,
ecutwfc = 28,
/
&ELECTRONS
/
ATOMIC_SPECIES
Si 28.086  Si.pz-vbc.UPF
ATOMIC_POSITIONS
Si              0.00.   0.00.   0.00
Si              0.25.   0.25    0.25
K_POINTS automatic
4 4 4   1 1 1
```

2. Logic of the examples:

Convergence tests for Si bulk consist of the following steps:

1. Converge the basis-set
2. Converge the k-points
3. With converged basis-set and k-points, calculate the lattice parameter of FCC bulk Si
4. With converged basis-set, k-points, and lattice parameter, fit the bulk modulus of FCC Si

Description of folder structure:

- *ex1.ecutwfc/* Convergence tests for cutoff energy
- *ex2.kpoints/* Convergence tests for k-points
- *ex3.alat/* Search of lattice parameter of Si bulk (alat = a lattice parameter)

Kinetic Energy Cutoff (ecutwfc)

The kinetic energy cutoff **ecutwfc** (in Ry) determines the **size of the Plane-Wave (PW) basis set** used to expand wavefunctions (i.e. Kohn-Sham orbitals). The default value for the charge density is **ecutrho=4*ecutwfc**, which is suitable for norm-conserving pseudopotentials.

A manual test of convergence with respect to the kinetic energy cutoff involves the following tasks:
(Note: We will not perform this manually! Follow the instruction on the README.md file):

1. Change the value of ecutwfc in the **pw.si.scf.in** input file to different values such as 16, 20, 24, 28, 32 Ry.
2. For each value of ecutwfc, run pw.x and record the final total energy.
3. Store the data in a file, let's say **si.etot_vs_ecut** (each line should contain two values: ecutwfc and total energy).
4. Plot the energies collected in si.etot vs ecut using your preferred plotting program.

Gnuplot:

```
plot "si.etot_vs_ecut" with linespoint
```

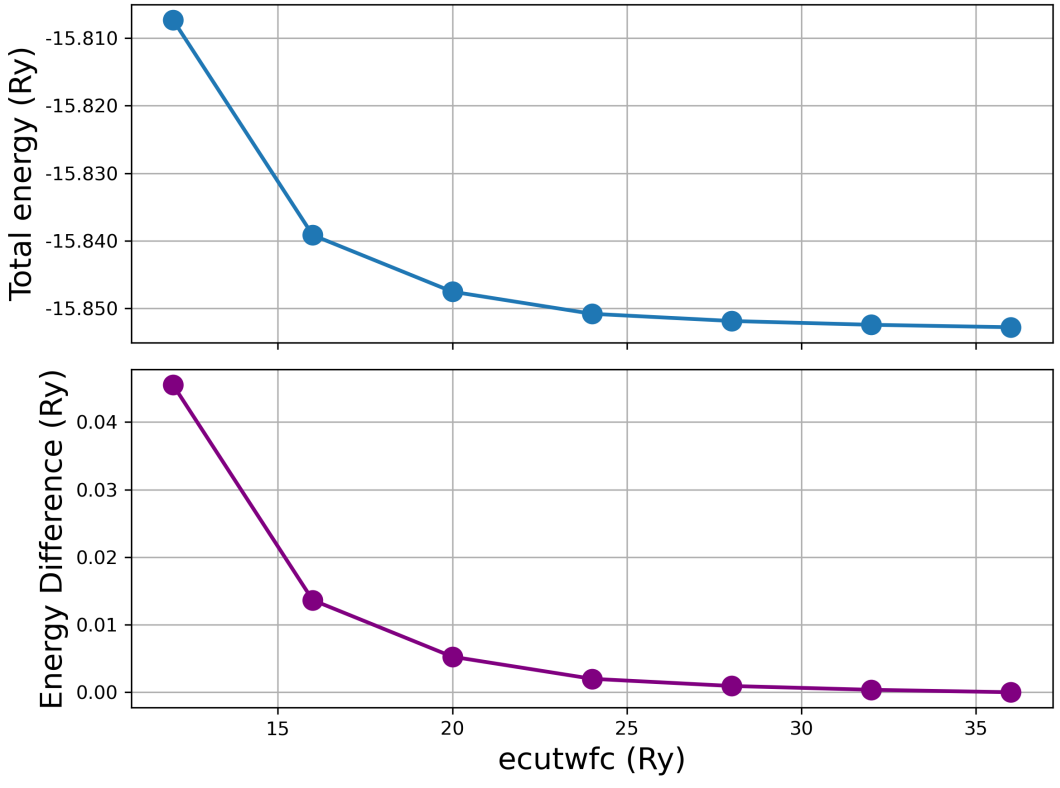
Python:

```
python plot_etot_ecut.py
```

Kinetic Energy Cutoff (ecutwfc)

In the end, we want something like this:

```
12.0000 -15.80731200
16.0000 -15.83916740
20.0000 -15.84754590
24.0000 -15.85081789
28.0000 -15.85188267
32.0000 -15.85244512
36.0000 -15.85280759
```



We are looking for differences of < mRy

Note: Absolute values of total energy do not have any physical meaning: only energy differences are meaningful!

Convergence with Respect to K-Points

A sufficiently **dense grid of k-points** is required to accurately represent the periodicity of the system. To test the convergence with respect to k-points, you can modify the K_POINTS card in your input file. Request **automatic** Monkhorst-Pack grids using the following format:

```
K_POINTS automatic
nk1 nk2 nk3      k1 k2 k3
```

Gradually increase the values of nk1, nk2, and nk3 while keeping k1, k2, and k3 equal to 1. For example, you can try increasing nk1 = nk2 = nk3 to 2, 4, 6, 8, and so on. Run the pw.x calculation for each set of k-point values.

- The first three nk1 nk2 nk3 numbers indicate the number of grid points along crystal axes 1, 2, 3.
- The second three k1 k2 k3 numbers, either 0 or 1, indicate whether the grid starts from 0 or is displaced by half a step along crystal axes 1, 2, 3.

Note:

- Convergence is not necessarily monotonic, as there is no variational principle with respect to the number of kpoints. Why do you think this is the case? Try to repeat the example with odd values of nk.

Lattice parameter determination

In silicon (Si), the equilibrium state is determined solely by the minimum-energy lattice parameter. Due to symmetry, there are no forces on the atoms.
(You can verify this by setting tprnfor=.true. in the namelist &CONTROL and checking for the forces reprinted at the end of the calculation.)

Lattice parameter determination

In silicon (Si), the equilibrium state is determined solely by the minimum-energy lattice parameter. Due to symmetry, there are no forces on the atoms.
(You can verify this by setting tprnfor=.true. in the namelist &CONTROL and checking for the forces reprinted at the end of the calculation.)

To determine the lattice parameter of bulk Si, you can follow these steps:
(as before, you can follow the instruction on the README.md file)

- Choose suitable values for ecutwfc (e.g., 36 Ry) and the k-point grid (e.g., 6 6 6 1 1 1);
- Run pw.x for values of **celldm(1)** ranging from 9.7 to 10.8 Bohr in steps of 0.1 Bohr;
- Store the final energy for each calculation in a file;
- Plot the results:

Gnuplot:

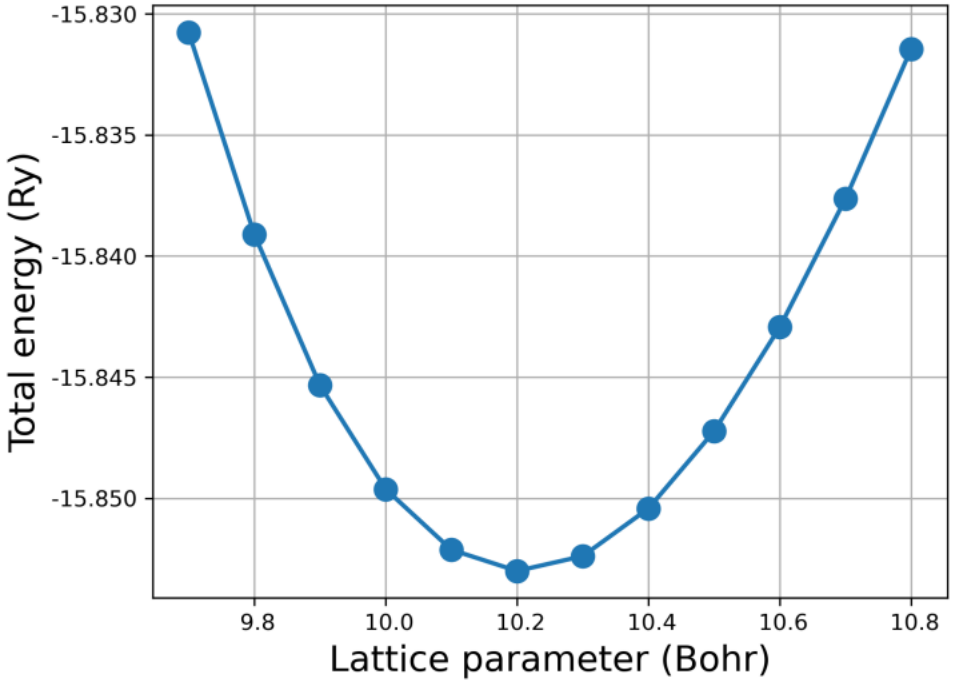
```
plot 'Etot-vs-alat.dat' with linespoint
```

Python:

```
python plot_alat.py
```

Lattice parameter determination

In silicon (Si), the equilibrium state is determined solely by the minimum-energy lattice parameter. Due to symmetry, there are no forces on the atoms.



Extracting physical parameters: the Bulk modulus

The bulk modulus is defined as:

$$B = -V\left(\frac{dP}{dV}\right)$$

We can use known relationship between volume and pressure (**equation of state**) to obtain B from the **E-vs-alat** data we calculate before.

The *Murnaghan equation of state* is expressed: $P(V) = \frac{B_0}{B_0{}'}\left[\left(\frac{V}{V_0}\right)^{-B_0'} - 1\right]$

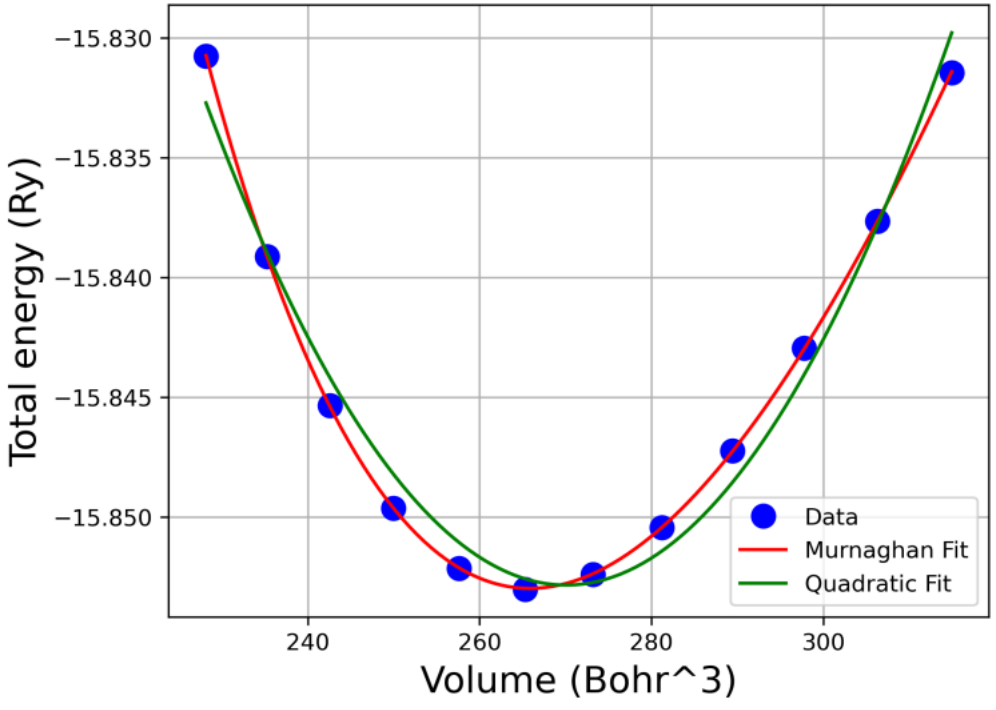
and in term of energy:

$$E(V) = E_0 + B_0 V_0\left[\frac{1}{B_0'(B'_0 - 1)}\left(\frac{V}{V_0}\right)^{1-B_0'} + \frac{1}{B_0'}\frac{V}{V_0} - \frac{1}{B_0' - 1}\right]$$

Extracting physical parameters: the Bulk modulus

We can fit our data with $E(V)$, and compare with a quadratic fit.

- Run `python fit_volume.py`
- Alternatively, you can you the built-in module of QuantumEspresso `ev.x`



The bulk modulus B_0 and the equilibrium volume V_0 are obtained as parameter of the fit.

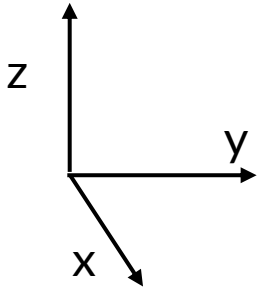
Exercise 3 : Carbon chain

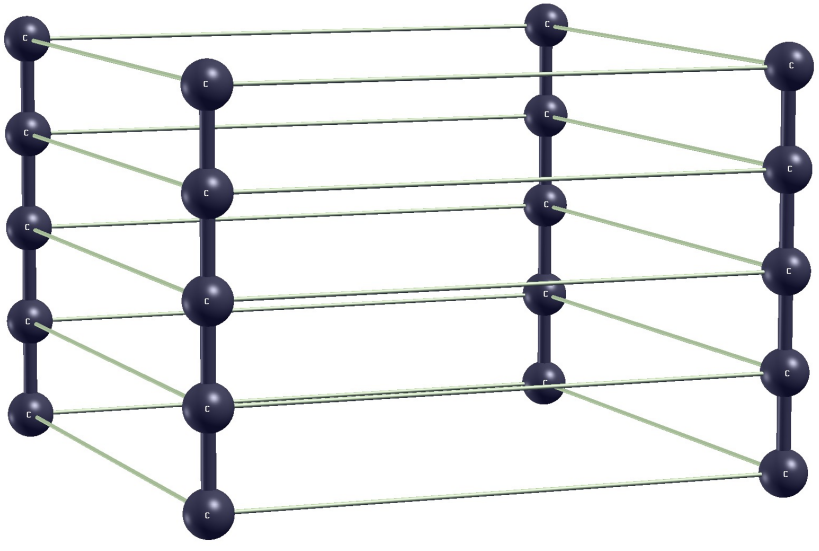
Introduction to low-dimensionality

3. Carbyne, the carbyne chain

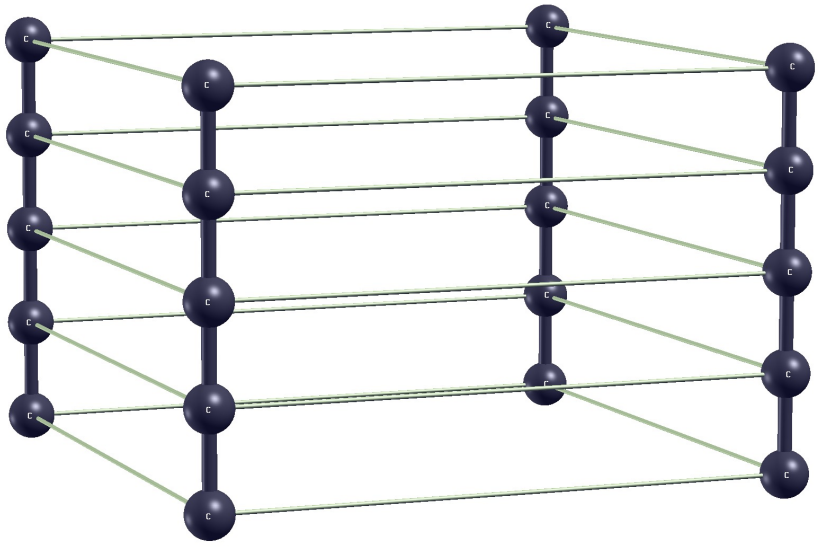
Carbyne is a chain composed of carbon

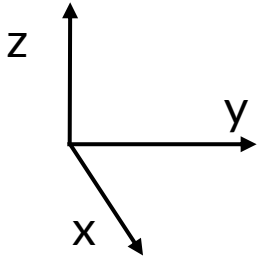
Periodic boundary conditions are applied in both directions. As seen in graphene, quantum espresso use periodic boundary condition.





3. Carbyne, the carbyne chain






```
&CONTROL
   calculation='?',
   prefix='carbyne',
   pseudo_dir='../../pseudo',
   outdir='./tmp'
   tprnfor = .true. ,     tstress=.true.
  /
  &SYSTEM
    ibrav =  0,       nat =  ?,       ntyp = ?,
    ecutwfc = 40,
  /
  &ELECTRONS
  /
  ATOMIC_SPECIES
   C    C.pbe-n-kjpaw_psl.1.0.0.UPF
  ATOMIC_POSITIONS crystal
   C 0.00 0.00 0.00
  K_POINTS automatic
   ? ? 30 0 0 0
  CELL_PARAMETERS bohr
  10              0.00000         0.0000
  0.00000         10               0.0000
  0.00000         0.00000         3.36
```

3. Carbyne, the carbyne chain

1. Visualize the structure using xcrysden.
2. Open the script scan_vacuum.sh and look at the input.
3. Fill the part left empty:

- how many atom are in the unit cell, according to what you see with xcrysden?
- How many atomic type?
- What you expect to be a good value for the k-points along *x* and *y* direction?

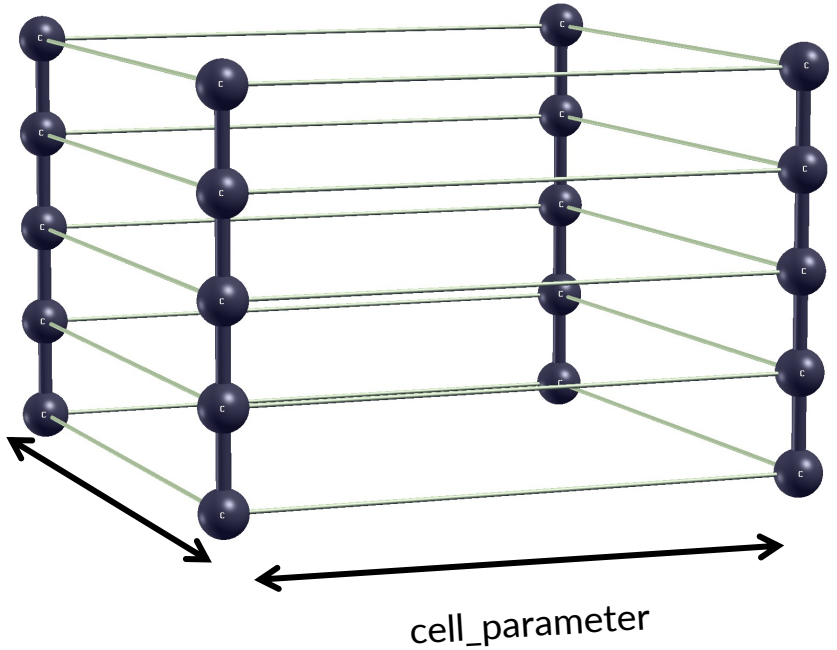
```
&CONTROL
   calculation='?',
   prefix='carbyne',
   pseudo_dir='../../pseudo',
   outdir='./tmp'
   tprnfor = .true. ,     tstress=.true.
  /
  &SYSTEM
    ibrav =  0,       nat =  ?,       ntyp = ?,
    ecutwfc = 40,
  /
  &ELECTRONS
  /
  ATOMIC_SPECIES
   C    C.pbe-n-kjpaw_psl.1.0.0.UPF
  ATOMIC_POSITIONS crystal
   C 0.00 0.00 0.00
  K_POINTS automatic
   ? ? 30 0 0 0
  CELL_PARAMETERS bohr
  10              0.00000         0.0000
  0.00000         10               0.0000
  0.00000         0.00000         3.36
```

3. Vacuum space convergence

The space between each repetition of the chain need to be converged.

4. Converge the vacuum using the script `scan_vacuum.sh` , trying different values of the cell parameter along the vacuum direction
5. Store the result in a file, running `gather_data.sh`

(Note: some parts need to be filled!)



3. Vacuum space convergence

6. Plot the result, with python plot_etot_vacuum.py or gnuplot plot.gp.
7. Open the script: plot_etot_diff_vacuum.py, try to understand it and fill the part left empty to plot the energy differences.

- Try smaller values of cell parameter (e.g., 5 bohr) and plot the energy again. What do you see? Visualize the input file with xcrysden

- What do you expect the total stress would be when everything is well-converged?

Thank you!

Questions?