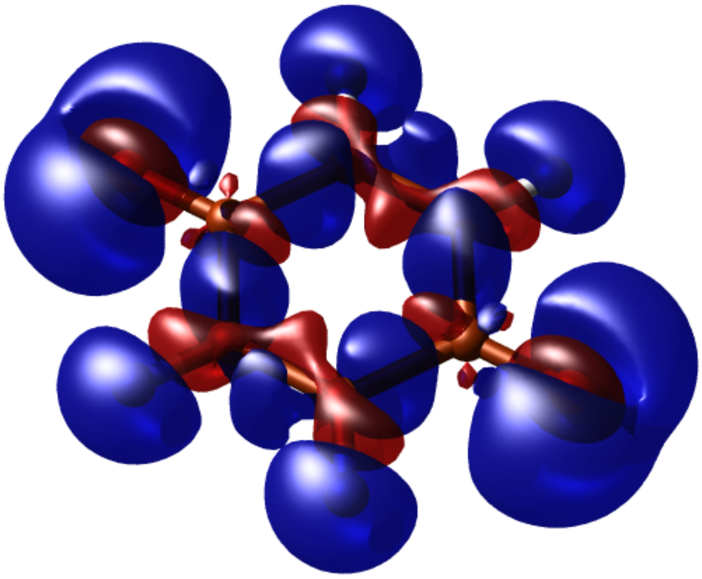
Density functional theory and beyond: Computational materials science for real materials

Trieste, August 06 – 15, 2013



Tutorial I: Basics of Electronic-Structure Theory
Manuscript for Exercise Problems

Prepared by Oliver T. Hofmann and Lydia Nemec
Fritz-Haber-Institut der Max-Planck-Gesellschaft
Trieste, August 8, 2013

A quick summary of the exercises

A guideline through the tutorial

This tutorial aims to give a basic introduction to electronic structure calculations for very simple systems. As every DFT code has its own philosophy, this tutorial should also familiarise you with fundamental aspects of using FHI-aims.
The goal of the first section is to explain the basic inputs of FHI-aims and to demonstrate that DFT calculations can have predictive power for many observable quantities. The second part introduces the geometry optimisation of a molecule and points out how to assess the reliability of the result. The third part is dedicated to efficiently obtaining and visualising electronic structure derived data. Some exercises are marked with a red exclamation mark (!). These exercises demonstrate pitfalls or limitations of the approach. Before we start working on the first problem, here is a short overview.

The practice session consists of three parts:

Part I: Basic electronic structure with FHI-aims

- Problem I: The hydrogen atom
- Problem II: Hydrofluoric acid (HF): bond length and dipole moment
- Problem III: Molecular oxygen - a critical look !

Part II: Local structure optimisation

- Problem IV: Planar hydronium cation H_3O^+
- Problem V: A vibrational analysis of planar H_3O^+
- Problem VI: Getting H_3O^+ right
- Problem VII: Pyramidal H_3O^+
- Problem VIII: Infra-red spectrum
- Problem IX: Limits of the harmonic approximation !

Part III: Electron density mixing and visualising electron densities and eigenstates

- Problem X: Making the SCF-cycle converge efficiently for Allicin
- Problem XI: Visualising density differences of *para*-benzoquinone

For every exercise, we also provide solutions, including sample input files. However, we strongly recommend to use the provided `control.in` and `geometry.in` files only in case of time shortage. You will maximise your learning progress by trying to generate the input files on your own. In case you get stuck with a particular problem, do not hesitate to ask one of the tutors. For the tutorials, an executable of FHI-aims will be provided on your workstation.

The very basics of FHI-aims

Each calculation should be done in a separate directory containing the two mandatory input files `control.in` and `geometry.in`. FHI-aims is then called in this directory.

In short:

- **Each calculation separate directory**
- **2 input files:**
  - `control.in`
  - `geometry.in`
- **Start calculation**

```
mpirun -np 4 aims.x | tee output
```

The above starts a parallel calculation on two processors, shows the main output on the screen and, at the same time, pipes it to the `output` file. The `output` file contains the basic information and results of the calculation such as the total energy, atomic forces, and so forth. Additional output files might be generated according to the specified settings.

geometry.in

The `geometry.in` file contains all information concerning the atomic structure of the system. This includes the nuclear coordinates, which are specified by the keyword `atom`, followed by cartesian coordinates (in units of Å) and the descriptor of the species. Fig. 1 shows an example `geometry.in` file for a hydrogen atom.

```
#The hydrogen atom
atom 0.0 0.0 0.0 H
```

Figure 1: An example `geometry.in` file for a hydrogen atom positioned at the origin.

For periodic calculations, lattice vectors can also be given in this file. This will be covered in the next tutorial. In the present tutorial, however, we will stick to non-periodic systems.

control.in

This file contains all physical and computational settings for the calculation. Fig. 2 shows a minimal example of a `control.in` file, which can be used as a template during the tutorial. These basic settings should be used as default for part I of this tutorial (unless specified otherwise).

```
#Sample input file for the calculation of a H atom
##################################################

xc  method
charge 0.0
spin collinear
default_initial_moment hund

#######SCF Accuracy###############################
sc_accuracy_eev 1E-2
sc_accuracy_etot 1E-5
sc_accuracy_rho 1E-4
sc_iter_limit 100

########Species###################################
```

Figure 2: Default physical and computational settings for `control.in`.

In this example, the following options are set:

- `xc`
  This keyword sets the method to be used. For example you can choose the option `hf`, which requests a Hartree-Fock calculation.

- `charge`
  Set the total charge of the system in units of $|e|$. For a neutral system, this is zero.

- `spin`
  This keyword governs the spin treatment. It can be set to `none`, which requests a spin-restricted (unpolarised) calculation, or to `collinear`, which requests a spin-unrestricted

(polarised) calculation. In a spin-restricted calculation, α and β spins are assumed to be equal. Only one spin-channel is treated and hence, the number of electrons is effectively halved. This usually causes a major speed-up for the calculations.

- `default_initial_moment hund`
  Sets the initial spin of the atoms. The value ”hund” requests an initial moment taken from Hund’s rule, but it is often more sensitive to provide a reasonable guess by hand. Only necessary for `spin collinear` calculations.

The other lines set the convergence criteria of the self-consistency cycle regarding the change of the density (`sc_accuracy_rho`), the sum of orbital eigenvalues (`sc_accuracy_eev`), and the total energy (`sc_accuracy_etot`) between two consecutive cycles. The remaining block of `control.in` defines all computational parameters associated with the elements (species) - most importantly the basis set. Each element listed in the `geometry.in` file must also be listed in `control.in`. The order of the species in the listing is irrelevant.

FHI-aims is shipped with pre-defined settings for all species which govern the key parameters regarding the numerical accuracy. They include, *inter alia*, the specification of all real-space integration grids, the accuracy of the Hartree potential and, of course, the basis set. For all elements, defaults are provided for three different levels of accuracy, light, tight, and really tight. They can be found in the directory

```
$HandsOnDFT/fhi-aims/species_defaults
```

and should be copied and pasted into `control.in`, e.g. via the command

```
cat $HandsOnDFT/fhi-aims/species_defaults/really_tight/01_H_defaults » control.in
```

which pastes the really_tight settings of the H-atom into the `control.in` file. Already the *tight* species_defaults are rather safe and *really tight* settings are for most purposes overconverged. In addition the number of basis functions can be varied, as well as the basis functions themselves. The basis functions associated with a given species are tabulated at the end of these default settings, as shown in Fig. 3.

The idea of keeping the species defaults out in the open is that, even if they are not modified, these are the critical accuracy parameters which one might look at to ensure numerical convergence. Each line denotes a specific basis function. They can be read as follows: The first keyword denotes the ”type" of the basis function. Hydro means that this is a hydrogen-like basis functions. Some basis functions are of the type ”ionic". They are described in more detail in the manual. The next two symbols correspond to the first two quantum numbers of the basis functions, and the final number corresponds to the ”effective nuclear charge" for which this basis function is created. `hydro 1 s 0.85` corresponds to the exact solution for the 1 s basis function of a hydrogen atom if it had a nuclear charge of only 0.85.

The basis functions are classified in ”tiers”. Not all basis functions are enabled by default. Rather, some are commented out using the ” # " symbol. They can be included in the calculation by uncommenting the corresponding lines. Systematically improved calculations can be performed by enabling additional tiers one after another.

```
##########################################################
#
#  Definition of "minimal" basis
#
##########################################################
#     valence basis states
    valence      1  s   1.
#     ion occupancy
    ion_occ      1  s   0.5
##########################################################
#
#  Suggested additional basis functions. For production
#  calculations, uncomment them one after another (the
#  most important basis functions are listed first).
#
#  Basis constructed for dimers:
#  0.5 A, 0.7 A, 1.0 A, 1.5 A, 2.5 A
#
##########################################################
#  "First tier" - improvements: -1014.90 meV to -62.69 meV
     hydro 2 s 2.1
     hydro 2 p 3.5
#  "Second tier" - improvements: -12.89 meV to -1.83 meV
#     hydro 1 s 0.85
#     hydro 2 p 3.7
#     hydro 2 s 1.2
#     hydro 3 d 7
#  "Third tier" - improvements: -0.25 meV to -0.12 meV
#     hydro 4 f 11.2
#     hydro 3 p 4.8
#     hydro 4 d 9
#     hydro 3 s 3.2
```

Figure 3: Tabulated basis functions for hydrogen. The basis functions are classified in ”`tiers`”. In this example only the `tier 1` and minimal basis functions are enabled.

Additional tools and programs

Scripts: For some exercises, scripts are required for dedicated tasks, such as for the manipulation of cube files. All scripts you will need for this tutorial can be found in

```
$HandsOnDFT/tutorials/tutorial1/utilities
```

Visualisation tools: To visualise structures, vibrational modes, charge density plots, etc., several programs including molden, vmd, jmol, gdis, and XCrysden are installed on your work stations. In this tutorial we will use `molden` as visualization tool, although most things work fine with jmol or other programs, too. A brief introduction on how to use molden can be found in Appendix II of this handout.

Plotting and Editing: To plot data, gnuplot, xmgrace and qtiplot are provided. Also several editors are available, including gedit, emacs, vi, and kwrite. The simplest one to handle is probably gedit.

Part I: Basic electronic structure with FHI-aims

Problem I: The hydrogen atom

In this exercise, we aim to convey the basics of FHI-aims using the hydrogen atom. The hydrogen atom is the simplest system possible and the only one for which the exact analytic solution is known. By the end of the first exercise, we see how various computational methods compare to each other and to the exact solution. From a technical perspective, we learn how to generate input files, read the standard FHI-aims output, and perform basis set convergence tests.

Getting started - the hydrogen atom

- Generate a simple `geometry.in` and `control.in` file and run FHI-aims.
- Test the convergence of the total energy with basis size.
- Compare the total energy of the hydrogen atom computed with different methods implemented in FHI-aims. Do all methods converge to the same result?

1. Generate a simple `geometry.in` file by hand, which contains only a single hydrogen atom. Please use the example shown in Fig. 1. This corresponds to a single hydrogen atom in the gas phase. It is located at the origin of the coordinate system, although its position, of course, does not matter here.

2. Generate a simple `control.in` file by hand. Please use the example `control.in` file given in Fig. 2. Systems with only a single electron can be solved exactly (within the Born-Oppenheimer approximation) using Hartree-Fock. Therefore, replace the word ”method" in Fig. 2 by `hf`. This will calculate the Hartree-Fock total energy for an uncharged, spin-polarised hydrogen atom. Finally, paste the ”*really_tight*" species data of H into the `control.in` file, e.g. via the command

   ```
   cat $HandsOnDFT/fhi-aims/species_defaultsreally_tight/01_H_default
   >> control.in
   ```

3. Now, run FHI-aims:

   ```
   aims.x | tee output
   ```

   When done, open the `output` file. If you find the line ”`Self-consistency cycle converged.`" near the end, then your calculation is converged. We are now interested in the total energy. Search for the block

   ```
    | Total energy uncorrected   :  -0.136054662652353E+02  eV
    | Total energy corrected     :  -0.136054662652353E+02  eV
    | Electronic free energy     :  -0.136054662652353E+02  eV
   ```

   Although in this special case, all energies are equal, this is is not the case if, for some reasons fractionally orbitals were found! For non-metallic systems, as in this tutorial, always use the **Total energy uncorrected** value. Compare it with the exact result for the hydrogen atom (0.5 Hartree ≈ 13.6057eV).

   TIP:
   In later exercises, to find this value fast and efficiently, use the command

   ```
   grep 'Total energy uncorrected' output
   ```

4. Redo the calculation with different basis sets (`minimal`[1], `tier1`, `tier2`, `tier3`) by uncommenting the basis functions at the end of the control.in file. Plot the total energy as function of the basis set size. Do the results change? From which tier on are the numbers converged?

   > **TIP**:
   > To plot the results, simply create a text file (e.g., *file.dat*) with two columns, the number of basis functions and the obtained total energy. This file can be plotted directly using the command
   > `xmgrace file.dat`

5. Finally, compare different methods implemented in FHI-aims by replacing `hf` in control.in with

   - `pw-lda`
   - `pbe`
   - `pbe0`

   and test the basis set convergence again. Do all methods converge with basis set size? Do all converge to the same result?

Optional: An optimal basis set

Optional part: If you browse through the hydrogen basis set you will note that no hydrogen 1s radial function is included. Change that by adding the line

```
hydro 1 s 1
```

at the end of the `control.in` file. Comment out all other basis functions and run the Hartree-Fock calculation again. How close does it get to the exact result?

Problem II: Hydrofluoric acid (HF): bond length and dipole moment

Hydrofluoric acid (HF)

One of the most influential papers in chemistry which systematically investigated the performance of DFT was Johnson et al.[1], in which for a large number of diatomic systems, several properties were consistently computed and compared to experimental values. In the style of this work, we will calculate the binding curve, atomization energy (ΔH_{at}), and the dipole moment for hydrogen fluoride (HF) with two methods. From a technical perspective, this exercise teaches how simple shell scripting can be used to make your (computational) life easier.

- Find the equilibrium bond distance of HF with the help of a simple shell script. Which bond length corresponds to the lowest energy?
- Compare the HF bond length for different methods. How does the optimal bond length change? How much does the total energy change?
- Calculate the atomization energy (ΔH_{at}).
- Compute the dipole moment for different methods and bond lengths. Do all methods converge to the same result?
- [optional] Use the Mulliken partitioning schemes to calculate the dipole moment within the point dipole approximation.

[1] If you are using parallelisation, please note, that FHI-aims requires at least as many basis functions as processors to be present. For a minimal basis set calculation of hydrogen, only one CPU can be used;

1. The first task of this exercise will be to find the equilibrium bond distance of HF from a series of single point calculations. Start by creating a `geometry.in` file which contains a H and a F atom, as shown in fig. 4

```
#HF at variable bond distances
atom 0.0 0.0 0.0 H
atom 0.0 0.0 Dist F
```

Figure 4: The geometry data for a HF molecule. H is put in the origin and F is located `Dist` Å away from the origin along the z-axis.

   In this example, H is put in the origin and F is located `Dist` Å away from the origin along the z-axis. Hereby, `Dist` is a placeholder which will be replaced by the actual distance later via a script.

2. Create a `control.in` file, and specify a `hf` calculation for a neutral system. Feel free to copy the file from the previous exercise, but remember to paste in the ”tight" species defaults for H **and** F. We also want to compute the electronic dipole moment, which is requested using the keyword `output dipole`. Note that although it doesn't matter where you put the keyword, it is recommended to put all manually set keywords *before* the species tags.

3. Next, create a bash-script which runs FHI-aims for a series of bond distances (ideally between 0.7Å and 1.3Å with 0.1Å steps, and a denser stepwidth of 0.02Å between 0.9Å and 1.0Å).

   For each distance, it should

   - create a unique directory
   - copy the `control.in` and `geometry.in` file
   - replace the bond distance place holder `Dist` with the bond distance and
   - start FHI-aims.

   Appendix I holds a short sample bash script (fig. 10) for this task. Alternatively, you may also use the script we provide in

   `$HandsOnDFT/tutorials/tutorial1/reference/exercise_02_HF/run_HF.sh`

   Run the script and plot the total energies vs. the bond length. This can be scripted as well. An example script is also provided in

   `$HandsOnDFT/tutorials/tutorial1/reference/exercise_02_HF/scripts/pp.sh`.

   Which bond length corresponds to the lowest energy?

4. Repeat the bond length determination using `pbe0`. How does the optimal bond length change? How much does the total energy change? How does the bond length compare to the experimental bond length of 0.917Å?

5. For comparison with experimental values, we compute the atomization energy (ΔH_{at}). In order to calculate the ΔH_{at}, we need the total energy of the isolated H and F atoms as well. Compute the single atoms for `hf` and `pbe0`. Check the result carefully - is Hund's rule fulfilled? Does the electronic configuration correspond to what you expect from the aufbau principle?

Note:
Atoms are highly symmetric systems, often with multiple degenerate solutions. In the case of fluorine, for example, the unpaired p-electron might sit in the px, py, or pz orbital. All three solution are equivalent. If the calculation is started unbiased, it might converge to a superposition of these three cases, which is a saddle point on the potential energy surface. It results in partial electron occupations. Although in DFT non-integer occupations are in principle possible, one should be very suspicious when obtaining such a solution for non-metallic systems. Typically, an energetically lower lying solution exists. They can be found by breaking the inherent symmetry of the problem, for example by applying a small external field at the beginning of the SCF-cycle.

To break the inherent symmetry of an atom and ensure integer occupation set the keyword `switch_external_pert 10 safe`; for 10 iterations of the SCF-cycle a small external field in z-direction is applied and then switched off. Usually, this is sufficient to perturb the SCF out of the symmetric solution and towards the correct electronic structure.

Calculate the atomization energy (ΔH_{at}) of HF by subtracting the free-atom energies from the total energy in the minimum.

$$\Delta H_{at} = E_{tot}^{HF} - E_{atom}^{H} - E_{atom}^{F} \tag{1}$$

How does this compare to the experimental value of $\Delta H_{at} = 135.2\text{kcal mol}^{-1}$ (5.86 eV)? How much does the atomization energy differ between methods?

6. Now, let us look at the dipole moment. Search for the corresponding line in the output file. Does the dipole differ much between the two functionals used? How does the dipole at the equilibrium distance compare with the experimental value of 1.82 Debye? Plot the dipole moment vs. the bond distance. You will find a (mostly) linear correspondence. Do you expect this to go on infinitely? Why?

Optional: Charge partition schemes.

Chemical reactivity and many physical properties are often explained in terms of atomic charges. However, atomic charges are not physical observables, since no unique operator exists to determine this quantity. They rather depend on the chosen charge partition scheme. Perhaps the most common one is Mulliken [2]. In FHI-aims, you can request it by specifying `output mulliken` in `control.in`.

For the equilibrium structure of HF, compute the atomic charges with this scheme using `PBE0`. Use them to calculate the dipole moment p in the point dipole approximation Within this approximation for a two atomic system the dipole moment is defined as:

$$p = q \cdot |\vec{r_H} - \vec{r_F}| \tag{2}$$

where q is the atomic partial charge, $\vec{r_H}$ and $\vec{r_F}$ are the atomic positions of the atoms. The absolute value of the difference $|\vec{r_H} - \vec{r_F}|$ is the displacement. Compare the dipole moment to the one computed by FHI-aims. How do they compare?

Problem III: Molecular oxygen - a critical look !

Bonus: An important part of every calculation is to always look critically at the output and ensure that the result is reasonable. For some systems, defaults may not be adequate, or assumptions which commonly work well may prove to be wrong. A prime example is the spin treatment in systems with degenerate orbitals, such as O_2.

[**Bonus: O_2 !**]

- Find the equilibrium bond distance of O_2. Which spin treatment is used if it is not specified `control.in`?
- `Repeat the calculation with spin-polarised settings. How does the occupation of the Kohn-Sham orbitals changes?`
- `Calculate the atomization energy (`ΔH_{at}`).`

1. Set up a calculation similar to the previous exercise for O_2. Leave out the `spin` keyword in `control.in`
2. Using PBE and PBE0, calculate the binding curve in the interval $[0.8, 1.6]$ Å with a stepwidth of 0.1 Å and the atomization energy (ΔH_{at}). Look up the occupation of the Kohn-Sham orbitals. Which spin treatment is chosen by default? Does this make sense here? Is Hund's rule fulfilled?
3. Repeat the calculation with spin-polarised settings - is Hund's rule now fulfilled?

Compare the results: Do both spin settings yield the same equilibrium bond length? Calculate the difference in the total energy at the equilibrium bond length. Which one is lower? How does it compare to the experimental value of 1.0 Å? How does the atomization energy (ΔH_{at}) compare to the experimental value of 5.18 eV?

Part II: Local structure optimisation

Problem IV: Planar H_3O^+

This exercise covers how to perform geometry optimisations. Specifically, we relax the H_3O molecule starting from a planar initial guess for the geometry.

- Generate a `geometry.in` and a `control.in` file for a structure relaxation run.
- Visualise the relaxation with `molden`.
- What does the fully relaxed structure look like?

1. Fig. 5 shows a `geometry.in` file. Please use this as the starting point for your structure relaxation.

```
# Initial guess for planar H3O+
atom   0.00   0.00   0.00   O
atom   0.92  -0.53   0.00   H
atom  -0.92  -0.53   0.00   H
atom   0.00   1.06   0.00   H
```

Figure 5: This `geometry.in` file gives a starting point for the planar H_3O^+ cation.

2. Create a `control.in` file. Start using the `control.in` file from the previous exercises as a template. As `xc` functional specify `pw-lda`. This time use `spin none` and, as we are interested in a cation, `charge 1.0`. Finally, add the keyword for structure relaxation `relax_geometry trm 1E-3`. Your `control.in` should look similar to the example shown in fig. 6.

```
# Sample input file for the relaxation
# of the H3O+ molecule
#######################################
  xc  pw-lda
  charge 1.0
  spin none

#######SCF Accuracy####################
  sc_accuracy_eev 1E-2
  sc_accuracy_etot 1E-5
  sc_accuracy_rho 1E-4
  sc_accuracy_forces 5E-4
  sc_iter_limit 100

#######Relaxation######################
  relax_geometry trm 1E-3
```

Figure 6: For comparison, the `control.in` file used for the structure relaxation of the cation H_3O^+.

As in the exercises before the basis set has to be included in the `control.in` file as well. Use the "*light*" species defaults for O and H atoms.

```
cat $HandsOnDFT/fhi-aims/species_defaultslight/01_H_default
>> control.in
cat $HandsOnDFT/fhi-aims/species_defaultslight/08_O_default
>> control.in
```

3. Run FHI-aims.

```
mpirun -np 4 aims.x | tee aims.H3O+_planar_relaxation.out
```

4. To visualise the results, copy the tool `create_relax_movie.pl` from $HandsOnDFT/tutorials/tutorial1/utilities into the working directory. Apply it to the output and pipe the result to a new file using the command
   ```
   ./create_relax_movie.pl H3O+_planar_relaxation.out > H3O+.molden
   ```
   Open the file `H3O+.molden` using molden.
   ```
   molden H3O+.molden
   ```

What does the fully relaxed structure look like. Do you believe this is the actual total energy minimum?

Problem V: A vibrational analysis of planar H_3O^+ !

Now we will check whether our obtained geometry is a local minimum or saddle point by performing a vibrational analysis.

- Perform a vibrational analysis of the planar H_3O^+ molecule.
- Visualise the vibration with `molden`.
- What does the mode with the imaginary frequency look like?

We use the wrapper `aims.vibrations.hands-on-2013.mpi.pl` in a folder that contains a control.in and geometry.in file.

1. Use the same control.in as in Problem IV, but comment out the `relaxation`-flag. Next copy the optimised geometry (from the output of Problem IV grep "Final atomic structure").

2. Run the vibrational script.
   ```
   > aims.vibrations.hands-on-2013.mpi.pl  H3O+_planar
   ```
   (The suffix `H3O+_planar` labels your calculation output)

3. Inspect the `H3O+_planar.vib.out` file. The eigenmodes of this molecule can be visualised via the `troublemaker.pl` tool, which can be found in the folder `$HandsOnDFT/tutorials/tutorial1/reference/utilities`.

   In order to generate the molden-readable file `H3O+_planar.molden`, apply it to the `*xyz` file of the output using the syntax
   ```
   > ./troublemaker.pl -xyz2molden H3O+_planar.xyz > H3O+_planar.molden
   ```

   Visualise your result using molden
   ```
   > molden H3O+_planar.molden
   ```

The script `aims.vibrations.hands-on-2013.mpi.pl` produces several output files - most importantly the `H3O+_planar.vib.out` and the `H3O+_planar.xyz` file. The `H3O+_planar.vib.out` file contains basic information regarding vibrational frequencies, zero point energies and infrared (IR) intensities. In addition, the `H3O+_planar.xyz` file contains geometrical information about the vibrational modes and can be read by visualization programs such as molden.

If the Hessian is positive definite (i.e., there are only positive eigenvalues), the structure is in a local minimum. If not we found a saddle point or a maximum. Formally, this corresponds to imaginary vibrational frequencies. However it is also common to output them as negative frequencies. There are 3N-6 non-zero modes (3N-5 for linear molecules), where N is the number of atoms. Due to numerical errors you will typically find the 'zero' frequencies to be in the range of $\sim \pm 15$ cm^{-1}.

What does the mode with the imaginary frequency look like?

Problem VI: Getting H_3O^+ right

An imaginary frequency indicates that the curvature of the potential energy surface (PES) along that mode is negative. In other words, following that mode leads to a structure with lower energy. Therefore, a reasonable guess would be to distort the geometry along this direction and optimise the structure again. This is what we are going to do next!

- Distort the planar geometry along the imaginary mode.
- Optimise the structure.

  What does the energy compare to the planar optimised structure of exercise Problem IV?
- Visualise the relaxation with `molden`.

  What does the optimised structure look like?

1. Taking the `H3O+_planar.xyz` file from exercise Problem V and distort the planar geometry along the imaginary mode using the script

   ```
   > ./troublemaker.pl H3O+_planar.xyz
   ```

2. For the distorted geometry optimise the structure again (as described in Problem IV).

   ```
   > mpirun -np 4 aims.x | tee aims.H3O+_distorted_relaxation.out
   ```

How does the optimised structure look like?
How does the energy compare to the planar optimised structure of exercise Problem IV?

Problem VII: Pyramidal H_3O^+

Check whether the 3D geometry of H_3O^+ is stable by performing a vibrational analysis as described in the previous exercise Problem V.

- Perform a vibrational analysis of the pyramidal H_3O^+ cation.

  Is it stable?
- Visualise the vibrations with `molden`.

  How does the lowest mode in energy (which is not a translation or rotation) look like?

Problem VIII: IR spectrum

Besides information about the structural stability, a vibrational calculation also gives the IR spectrum. In this exercise we will compare the IR spectrum for the planar and pyramidal geometry for different numerical settings and different functionals.

- Visualise the IR-spectrum of the H_3O^+ cation.
- Repeat it for the planar structure. Compare these two IR spectra. What can you see?
- Compare the IR-spectra for different numerical settings:
  - Size of the basis set
  - xc-functional – PBE vs. LDA
- Compare your computed spectra with experiment.

To get started we can use the data calculated in Problem V and Problem VII. Compare the vibrational frequencies obtained in both problems.

Note:
As you see, the IR-spectrum is very sensitive to the structure. Comparing a calculated IR-spectrum to an experimental on can shed light on the structure that is predominantly present in experiment.

- Perform convergence tests with respect to light, tight and really tight settings for the infrared spectrum and for the 3D geometry (pw-lda functional).

  Important:
  Since a harmonic IR-spectrum only makes sense for an optimised geometry, make sure that you request a geometry optimisation in your control.in file, i.e. set `relax_geometry trm 1E-3`.
  If set, the vibrational script optimises the geometry before doing the actual calculation.

- For light settings, compute the PBE IR spectrum and compare it with the pw-lda spectrum.

- Compare your obtained spectrum with the experimentally obtained vibration frequencies. You can look them up online at the NIST webbook: http://webbook.nist.gov/chemistry/form-ser.html.

Problem IX: Limits of the harmonic approximation !

In order to demonstrate the range of the validity of the harmonic approximation which is the basic assumption for the vibrational analysis, we first restrict the potential energy surface (PES) to one dimension.
We consider a cut through the PES where the H atoms remain in one plane and the O moves along the normal of this plane. Denote the distance between the plane of H atoms and the O atom as Q.

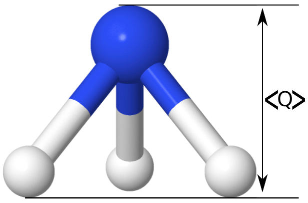


Figure 7: The hydronium cation: The H atoms form a plane in the xy-plane. The oxygen atom is $\langle Q \rangle$ Å above the plane.

The PES can be calculated by performing a structure relaxation under the constraint that the H-atoms are fixed within a plane for a given distance Q between O and that plane. The information about the constraints goes into the geometry.in file, where the keyword has to follow immediately after the corresponding atom: `constrain_relaxation .true.` indicates that the position of the atom is fixed during the optimisation process in all three spatial coordinates, and `constrain_relaxation z` fixes only the z-coordinate of the atom.

- Perform a set of constrained relaxations
- Calculate an approximate PES from distortions along the imaginary mode.

```
# Initial guess for planar H3O+
atom   0.00    0.00     <Q>    O
constrain_relaxation .true.
atom   0.92   -0.53    0.00    H
constrain_relaxation z
atom  -0.92   -0.53    0.00    H
constrain_relaxation z
atom   0.00    1.06    0.00    H
constrain_relaxation z
```

Figure 8: This is a sample `geometry.in` file constraining the O atom at its position. For the H atoms only the z coordinate is constrained.

- Generate a `geometry.in` template file including constraints on the atomic position during structure optimisation.
- Calculate the PES by performing constrained structure optimisations for Q between 0Å and 1.2Å. Choose a reasonable stepwidth.
- Inspect the output file and look for the final geometry. Are the hydrogen atoms still located at the same z-coordinate?
- Collect the total energy for each calculation.
- Approximate the PES by distorting along the out-of-plane mode relative to the saddle point. The structure for $Q = 0$ in the above construction corresponds to the saddle point as we have found it in Problem IV . Therefore, within the vicinity of $Q = 0$ the PES should be well described by the out-of-plane normal mode of the planar geometry, i.e., its imaginary mode. The script `troublemaker.pl -mode` M `-norm` N `*xyz` distorts the equilibrium geometry along a chosen mode M with a given extension N. Here, N is simply a scaling factor which is applied to all atomic displacements in the given mode.

  Create a set of different geometries from the *xyz-file of exercise V by using different scaling factors. Inspect the obtained geometries to obtain a relation between N and Q.
  `> ./troublemaker.pl -mode 1 -norm` N `H3O+_planar.xyz`
  Use the output to create a geometry.in file. For every geometry obtained in this way, calculate the total energy.
  `> mpirun -np 4 aims.x | tee H3O+`Q`.out`

Important: Do not perform a structure optimisation!

Plot the PES obtained from constrained structure relaxations from the distortion along the imaginary mode. Take as common reference the total energy of the fully optimised planar structure. Is there a range where the two lines coincide?

Part III: Electron density mixing and visualising electron densities and eigenstates

Italian chef Antonello Colonna, said in an interview with an American daily newspaper: "Garlic is the king of the kitchen, to eliminate it is like eliminating violins from an orchestra."
The garlic bulb releases a number of low-molecular-weight organosulfur compounds rarely encountered in nature.[3]. One of them is the molecule Allicin (see Fig. 9). Allicin is responsible for the characteristic smell of freshly crunched garlic.
In this part of the exercise, we will use this molecule to compare the influence of different electron density mixing settings.

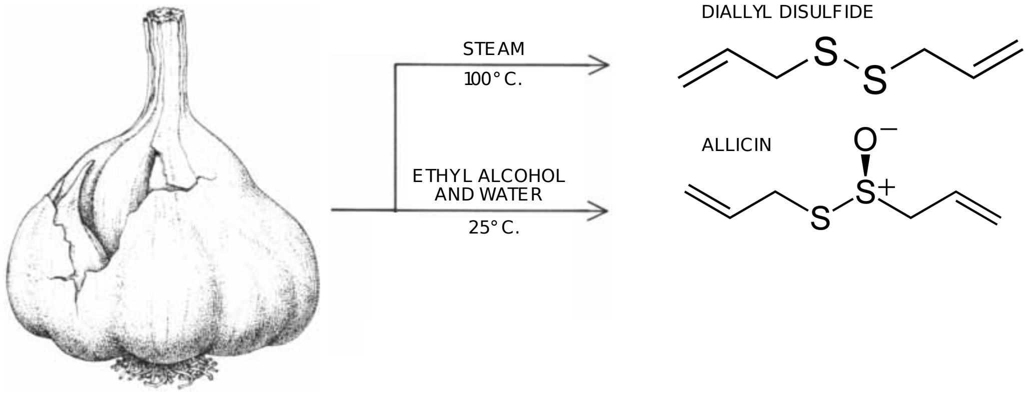


Figure 9: The allicin molecule features a thiosulfinate functional group (R-S(O)-S-R), which gives freshly crunched garlic its unique smell. (Figure adapted from Block [3])

Problem X: Converging efficiently

During the self-consistent solution of the calculation, it is beneficial to use a history of densities from previous SCF-cycles. FHI-aims provides two mainstream density mixing algorithms, linear and pulay (also known as Direct Inversion in Iterative Subspace, DIIS).

- Compare different electron density mixing schemes:
  `linear` and `pulay`.
- Start with the linear mixer.
- Test different settings for the `pulay` mixer.
- List the number of SCF-cycles and the total CPU time needed for each setting.
  What is the most efficient setting? How much CPU time do you save compared to the linear mixer and the default pulay settings?

The geometry of the molecule for this exercise, as well as a sample `control.in` file, can be found in the `$HandsOnDFT/tutorials/tutorial1/reference/Templates` directory.

1. First create a directory for a calculation using the `linear` mixer. In the template `control.in` file you find three new keywords:

```
mixer TYPE
charge_mix_param PARA
n_max_pulay NUM
```

For the first part ignore the keywords `charge_mix_param` and `n_max_pulay` – add a hash (#) in front of these two lines with `#` to deactivate them. The `mixer` keyword specifies the electron density mixing algorithm used. Here we use the *t*ype `linear`, so replace the place holder `TYPE` by `linear`.

2. Run the calculation and open the FHI-aims output file. How many SCF-cycles were needed to reach self-consistency? Do you think it has to take that long to converge a rather small molecule?

3. Next we will use a more advanced mixing scheme, the `Pulay` mixer.
This mixer also takes into account the history of the self-consistent problem. For this algorithm, `n_max_pulay` *n* and `charge_mix_param` *v* have to be specified, n determines the number of past iterations $\mu - k(k = 1, \cdots, n)$ to be mixed with the Kohn-Sham output density of iteration μ. The value v for the keyword `charge_mix_param` determines an additional (system-dependent) linear factor that is multiplied with the output density change of the Pulay mixer.
We will try three different settings:
   - `mixer pulay`
     `n_max_pulay 3`
     `charge_mix_param 0.1`
   - Use the FHI-aims default settings:
     `mixer pulay`
     `n_max_pulay 8`
     `charge_mix_param 0.2`
   - Finally use the settings specifically optimised for Allicin.
     `mixer pulay`
     `n_max_pulay 5`
     `charge_mix_param 0.3`

4. Compare the number of SCF-cycles and computational time with the linear mixer. Did the choice of mixer speed up your calculation? How do the optimised settings compare to the previous calculations?

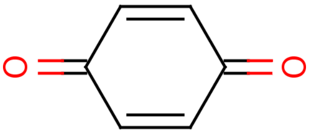
> This exercise can be nicely scripted by automatically replacing the place holder `TYPE, PARA, NUM`. Have a look at the example script in the Appendix I: Sample bash scripts.

Problem XI: Visualisation of Kohn-Sham orbitals and electron densities

Chemical reactions and many important physical effects (such as the formation of interface dipoles) are triggered by electron rearrangement. These are often associated with filling or emptying of molecular frontier orbitals. In this exercise, we will demonstrate how charge distributions are visualised and investigate how well the "actual" filling of a molecular orbital corresponds to the associated eigenstate of the neutral molecule.

- Set up the molecular geometry
  `linear` and `pulay`.
- Calculate and visualise the Kohn-Sham orbitals of the neutral molecule
- Calculate the negatively charged anion and subtract its energy from the energy of the neutral moiety. How well does the electron affinity compare to the experimental result?
- Calculate the change in electron density upon charging. What are the differences to the neutral LUMO?

1. The molecule for the present example is para-benzoquinone, a strong yet small electron acceptor which is readily available. You can either create the geometry input for this molecule yourself (see the appendix on how this can be done with `molden`) or use the provided input file from the Templates folder.



2. Create a control.in input file for a spin-unpolarised, uncharged PBE calculation of the molecule. We want to visualise the highest occupied molecule orbital (HOMO) and the lowest unoccupied molecular orbital (LUMO) in form of a standard cube file. Request the output of the wave functions using the keywords

```
output cube eigenstate homo
cube filename HOMO.cube
output cube eigenstate lumo
cube filename LUMO.cube
```

We also want to look at charge density differences, so we need the total density of the molecule. Add

```
output cube total_density
cube filename total_density_uncharged.cube
```

to the `control.in` file. By default, FHI-aims will figure out the region to plot into the cube files by itself.

However, you are also free to choose the cube region yourself by adding the following lines after the `output cube` keyword:
The line
`cube origin` *x y z*
allows to set the center of the space to be plotted, with x y z being its cartesian coordinates in Å. The edges of the cube file can be configured using the syntax
`cube edge` *n dx dy dz*,
where n indicates the number of steps of a particular edge (voxel), and dx, dy, dz indicate the length of each individual step in x, y, and z direction, respectively. Note that each cube file, being a 3D-object, requires 3 edges to be completely defined. E.g., a complete definition of a cube file could look like

```
output cube total_density
```

```
cube filename total_density_uncharged.cube
cube origin 0 0 0
cube edge 100 0.1 0.0 0.0
cube edge 150 0.0 0.1 0.0
cube edge 50 0.0 0.0 0.1
```

3. Run the calculation and look at the output. In the final occupation numbers, search for the energies of the HOMO and the LUMO. How do these energies compare to the experimental values of 10.7 eV for the ionisation potential and 1.9 eV for the electron affinity? Next, use your favourite tool to plot the cube files. (See appendix on how this is done with molden.) Do HOMO and LUMO both have π character?

4. To see how well the eigenvalues and eigenstates correspond to the physical situation of charging a molecule, prepare and run two more inputs for the molecule with one negative charge. Remember that the total number of electrons is odd, and therefore, `spin collinear` should be used.

   Obtain the electron addition and removal energies by subtracting the total energy of the charged system from the total energy of the uncharged system. This is also known as the ΔSCF approach. Compare the values with the HOMO and LUMO eigenvalue of the neutral molecule. How much do they differ?

5. The change in the electron density upon charging can be obtained by subtracting the total density of the charged system from the total density of the uncharged system. To that end, a python-script called 'subtract_cubes.py' is provided in the utilities folder. Copy this script and the total_density.cube files into a separate folder and execute

   ```
   > python subtract_cubes.py total_density_uncharged.cube
   total_density_charged.cube chargediff.cube
   ```

   to subtract the file given as second argument from the file given as first argument. The result will be written to the file provided as final argument. Note that the cube files need to be generated with the same cube file settings in `control.in`. Visualise the density differences. Do you see qualitative differences to the eigenstates? Can you explain (or at least speculate) on the reasons for them?

We like to thank you for participating in the first Tutorial of this workshop. We wish you an interesting and inspiring time here at the 'Density functional theory and beyond' workshop.

Appendix

Appendix I: Sample bash scripts

Sample script for Exercise 2

Below, you find a sample script to calculate a molecule with pre-defined bond distances.

```
#!/bin/bash -l
# Comment: This script calculates the bond distance for HF


# Please enter your own aims executable here
AIMS_BIN=aims.x

#Directory containing all the templates
TemplateDir=/afs/ictp.it/public/shared/smr2475/tutorials/tutorial1/reference/templates/ex


Methods='hf pbe0'


#Variable definition
BondDistance='0.7 0.8 0.9 0.92 0.94 0.96 0.98 1.0 1.1 1.2 1.3'

for Method in $Methods; do # Cycles through all methods
   echo "Start␣calculations␣with␣xc-functional␣" $Method
   mkdir $Method
   cd $Method
   for Dist in $BondDistance; do #Cycles through all distances
      mkdir $Dist #Create directory
      #################Prepare input #############################
      #copy geometry.in
      cp ../geometry.in.template $Dist/geometry.in
      #copy control.in
      cp ../control.in.template $Dist/control.in
      #Add species
      cat ../Basis_tight_tier2 >> $Dist/control.in
      #search/replace distance
      sed -i "s/Dist/$Dist/g" $Dist/geometry.in
      #search/replace method
      sed -i "s/method/$Method/g" $Dist/control.in
      ###################Input done##############################
     cd $Dist #Change directory and run
     echo "␣␣␣␣␣Run␣calculation␣for␣bond␣length␣" $Dist
     $AIMS_BIN > output

      cd .. # Leave directory $Dist
   done # Loop over distances done

   # Before we switch to the next method,
   # we also calculate the free atoms
   for  Atom in H F; do
     echo "Calculating␣$Atom"
     #################Prepare calculation ##########################
     mkdir $Atom
     echo "␣atom␣0.0␣0.0␣0.0␣$Atom" > $Atom/geometry.in
     cp  $TemplateDir/control.in.template $Atom/control.in
     #search/replace method
     sed -i "s/method/$Method/g" $Atom/control.in
     #Add species
     cat ../Basis_tight_tier2 >> $Atom/control.in
     ################ Preparation done ##############################
     cd $Atom #Hop into directory
     $AIMS_BIN  > output
     cd .. #Hop out again
   done;
   cd .. # Leave directory $method
done # Loop over Methods
```

Figure 10: This is a sample bash script for calculating a molecule with different pre-defined bond distances.

Sample script for Exercise 9 (first part)

Below, you find a sample bash script that creates geometry.in files from a template "geometry_planar_temp", executes a calculation and writes the total energy with respect to the zero-point energy into the file "PES_energy.dat".

```
#!/bin/bash -l
# -N H3O_pes
pushd ../.. >/dev/null; source tut1.env; popd >/dev/null

echo "****************************************************"
echo "EXERCISE␣9:␣constrained␣relaxation"
echo "****************************************************"

python_calc () {
    python -c "print␣($@)"
}

# Get template of control.in
cp $REF_DIR_T1/control_template.in  ./control.in
# Feed it with the species
cat $LIGHT_SPECIES/01_H_default >> control.in
cat $LIGHT_SPECIES/08_O_default >> control.in

for Q in {0..30}
do
   Q=$(python_calc "0.0+$Q*0.04")
   # Get geometry file
   cat geometry_planar_temp | sed "s/<Q>/$Q/g" >  geometry.in
   # Run aims
   echo "----------------------------------------"
   echo "Running␣distance␣Q="$Q  "␣␣Angstroms"
   mpirun -np $NP  $AIMS_BIN > aims.H3O+_PES_Q_${Q}.out
   # reference-point
   if [ $Q == 0.0 ] ; then
      ZP=$(cat aims.H3O+_PES_Q_${Q}.out |\
      grep ’Total energy uncorrected’| tail -1 |\
      awk ’{printf "%12.9f", $6}’)
   fi
   # Read and write energy
   cat aims.H3O+_PES_Q_${Q}.out | grep ’Total energy uncorrected’ |\
   tail -1 | awk ’{printf "%2.2f\t%12.9f\n", ’$Q’, ($6 - ’$ZP’)}’ >> PES_energy.dat
done
```

Figure 11: This is a sample bash script calculates the energy for different distance $\langle Q \rangle$ and writes the energies in a file "PES_energy.dat".

Sample script for Exercise 10

Below, you find a sample bash script that adapts the `control.in` file from the `Templates` directory, to test different settings for the density mixing scheme (`linear` and `pulay`).

```
#!/bin/bash -l

pushd .. >/dev/null; source tut1.env; popd >/dev/null

#Variable definitions are found in LocalVar.env
##############################################################
source LocalVar.env

MIXER=’linear pulay’
N_MAX=’3 5 8’
PARAM=(0.1 0.3 0.2)

echo "*****************************************************"
echo "EXERCISE␣X:"
echo "*****************************************************"

for type in $MIXER; do

  mkdir $type
  cd $type

  if [ $type != ’linear’ ]; then
     for num in $N_MAX; do
        if [ $num == ’3’ ];then
           para=${PARAM[0]}
        elif [ $num == ’5’ ];then
           para=${PARAM[1]}
        elif [ $num == ’8’ ];then
           para=${PARAM[2]}
        fi
        mkdir $num-$para
        cd $num-$para
        cp ../../Templates/geometry_template.in geometry.in
        cp ../../Templates/control_template.in control.in
        cat ../../Templates/Basis_light >> control.in
        sed -i "s/TYPE/$type/g" control.in
        sed -i "s/PARA/$para/g" control.in
        sed -i "s/NUM/$num/g" control.in
        mpirun -np $NP $AIMS_BIN > aims.out
        cd ..
     done
  else
     cp ../Templates/geometry_template.in geometry.in
     cp ../Templates/control_template.in control.in
     cat ../Templates/Basis_light >> control.in
     sed -i "s/TYPE/$type/g" control.in
     sed -i "s/charge_mix_param/#charge_mix_param/g" control.in
     sed -i "s/n_max_pulay/#n_max_pulay/g" control.in
     mpirun -np $NP $AIMS_BIN > aims.out
  fi
  cd ..
done
```

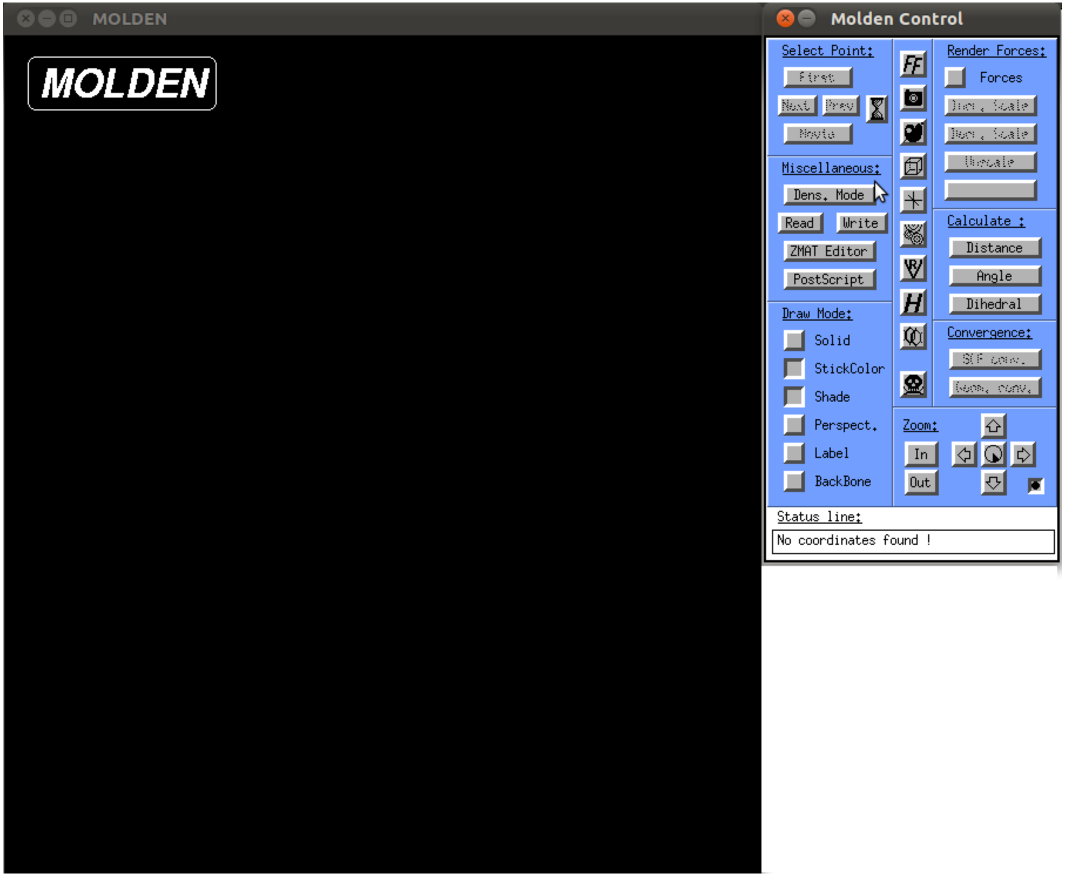
Figure 12: This is a sample bash script that tests different settings for the density mixing schemes.

Appendix II: Molden

Molden is a small, free tool which allows to visualize non-periodic structures, geometry optimizations, vibrations, and cube files. It can be used for all visualisations during tutorial 1. In the utilities folder, we provide several scripts which convert the output from FHI-AIMS to the molden format.

To open molden, simply type
`molden` or `molden` *`filename`*
in the command line.

Two windows will open, one named "MOLDEN", which contains the structure (and is empty if you didn't provide a file), and one named "MOLDEN Control" which contains several buttons. To open a new file, use the "Read" button.



No coordinates found !

Below, we will describe how to visualize the output from the individual task. Of course, molden can do much more than what is shown here - feel free to play around. If you want to close molden, please use the skull icon in the middle of the control window.

Geometry single points

1. Create a molden file by appyling the script `create_relax_movie.pl` to the FHI-aims output of a single point calculation

2. Open the file typing `molden` *`filename`*. On the top left corner of the Molden Control Window. The geometry of the system should appear in the Molden Window.

3. You can change the display options by playing around with the Draw Options in the bottom left corner of the Molden Control Window. Good results are often achieved by disabling shade and setting "Solid" to "Ball and Stick". Feel free to experiment!

Geometry relaxations

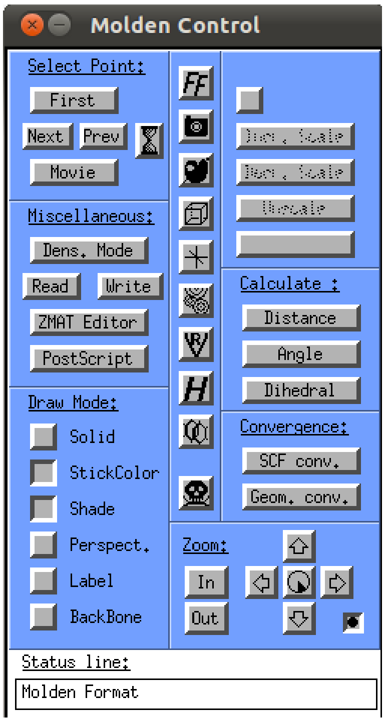
1. Create a molden file by applying the script `create_relax_movie.pl` to the FHI-aims output of a geometry relaxation.

2. Open the file typing `molden` *`filename`*. On the top left corner of the Molden Control Window, the "Select Point" field should be active now. (See picture)

3. Watch a movie of the relaxation by clicking on the "Movie" button. If the movie is too fast, use the "sand-clock" button to set a delay between the individual frame.

4. You can also walk though each point of the geometry optmisation using the "Next" and "Prev' buttons.

5. Clicking on the "Geom. Conv" button in the bottom right corner of the Molden Control window will open a new window with information about the change of energy, maximum force, and maximum step size for each step.

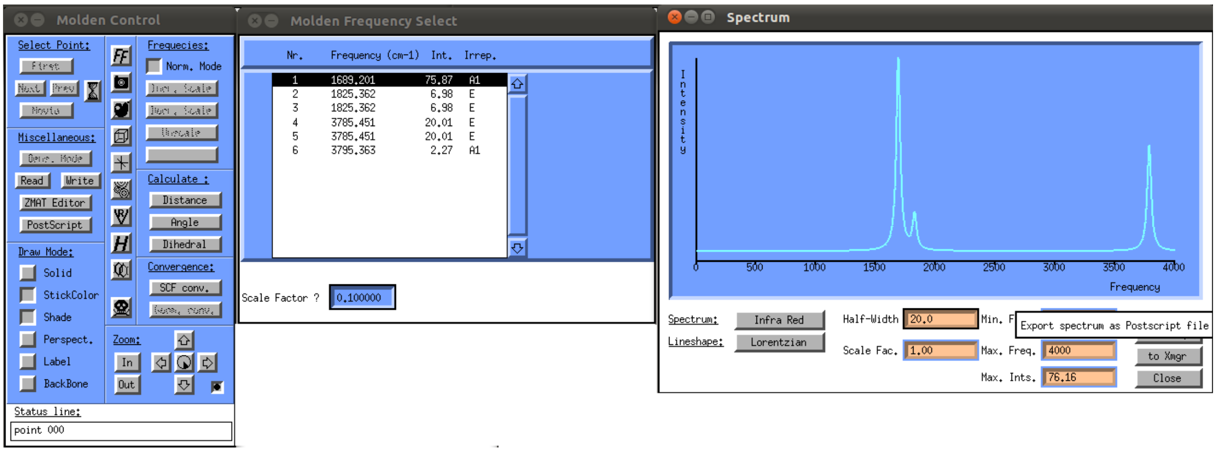


Vibrations

1. Create a molden file by applying the script `> ./troublemaker.pl -xyz2molden H3O+.xyz > H3O+.molden` to the FHI-aims output `H3O+.xyz` of a vibration (`troublemaker.pl` can be found in the `$HandsOnDFT/tutorials/tutorial1utilities` folder). calculations.

2. Open the file typing `molden` *`filename`*. On the top right corener fo the Molden Control Window, the "Norm. Mode" button should be active now, as highlighted in the picture.

3. Click on the button. Two more windows will open. The window called "spectrum" will provide you with an simulated infra-red spectrum based on the eigenvalues and their intensity. The bottom of this window provides several possibilities to alter the appearance of the spectrum or print it. The second window is called "Molden Frequency Select" and contains a list of all eigenmodes along with their frequency and IR intensity. By choosing one of these, the geometry in the molden window becomes animated and displays the corresponding eigenmode. The box scale factor at the bottom of the window can be used to amplify or damp the movement of the atoms.



Status line:
point 000

Nr.	Frequency (cm-1)	Int.	Irrep.
1	1689.201	75.87	A1
2	1825.362	6.98	E
3	1825.362	6.98	E
4	3785.451	20.01	E
5	3785.451	20.01	E
6	3795.363	2.27	A1

Scale Factor ? 0.100000

Half-Width 20.0
Min. F
Scale Fac. 1.00
Max. Freq. 4000
Max. Ints. 76.16

Cube files

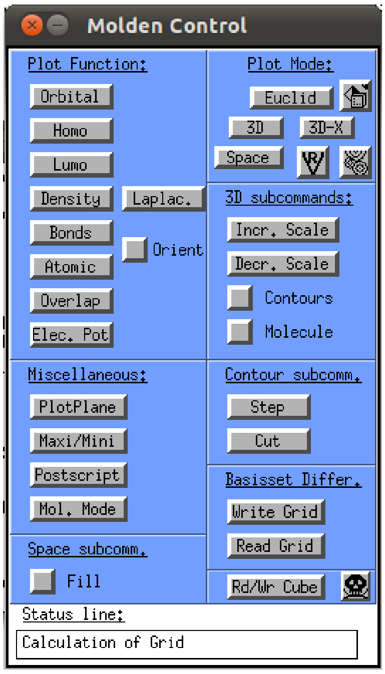
1. Cube files are generated directly by FHI-AIMS using the `output cube` syntax, see manual section 3.19

2. open molden without any filename by typing `molden`

3. Click on the "Dens Mode" button in the left central panel. The Molden control window will change and show different buttons, see picture.

4. Click on the "Rd/Wr Cube" button in the bottom right corner. Choose read, Gaussian and give input the file name of your cube file when asked.

5. When the file opens, use the "Space" button in the top right corner to choose an isovalue for the display of the cube file.



Status line:
Calculation of Grid

Creating geometries

Besides pure visualization, molden also allows to efficiently generate and save geometries for molecular structures. Molden uses a z-matrix editor. A detailed explanation of z-matrices is beyond the scope of this work, but can be found online, e.g. under http://www.cmbi.ru.nl/molden/zmat/zmat.html. Even for people not familiar with z-matrices, generating input structures is usually easy and intuitive. This function is briefly explained here for the example of para-benzoquione, the molecule used in the last part of Tutorial 1, and shown in the picture below.

1. Open molden and go to the z-matrix editor. A new window will appear which contains an empty field and several buttons, as shown in the picture below.

2. Begin by clicking on the "Add Line' button and select a carbon atom. A green circle should appear on your screen now.

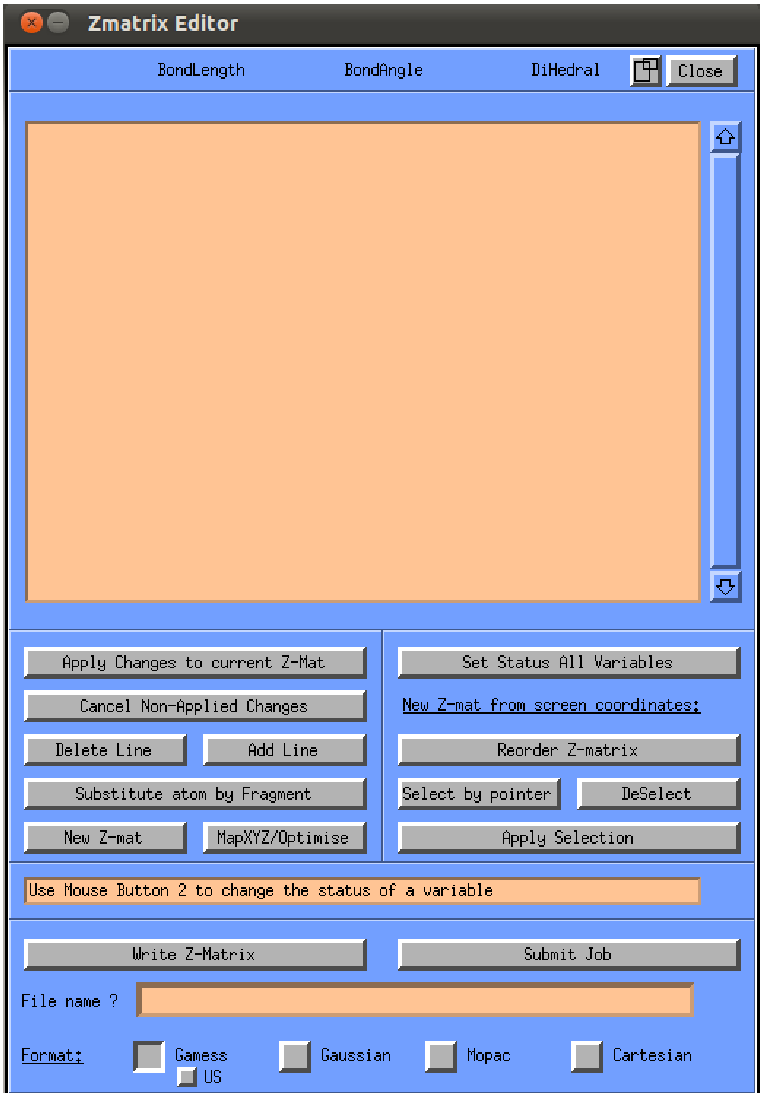
3. The fastest way to proceed is to mark the carbon atom, choose 'Substitute atom by fragment' from the Zmatrix Editor, and select -Phenyl. This will turn the atom into a completed ring, saturated with hydrogen atoms on all carbon atoms but one. The formally empty box will now contain a list of all atoms. The first row is for bond distance, the second row is bond angles and the forth row is dihedral angle.

4. To add additional atoms, select the "Add Line" button again. Choose "double" as bond length, and an oxygen atom from the periodic table. Molden will now ask you to select 3 atoms to define the connectivity. These are the atoms to which bond length, angle and dihedral angle are defined. Click on the only carbon atom which has no hydrogen on it, and sub-sequentially on 2 neighbouring carbon atoms. A new line should appear.

5. If you made a mistake and want to delete the atom, click on the "delete line" button. If you were successful, delete the hydrogen atom opposite to the oxygen atom. Then, add another oxygen atom in its place, the same way you added the first one.

6. Once you are done, save the geometry by clicking on "Cartesian" in the lower right corner of the Zmatrix Editor, choose "XYZ" and provide a file name. Save by pressing return, and quit molden via the Skull icon.

7. Use an editor or an script to adjust the xyz-format to the FHI-aims `geometry.in` format.



Use Mouse Button 2 to change the status of a variable

Acknowledgments

We like to thank testers of this tutorials for their invaluable feedback.

References

[1] B. G. Johnson, P. M. W. Gill and J. A. Pople, *The performance of a family of density functional methods*, The Journal of Chemical Physics **98**, 5612 (1993).

[2] R. S. Mulliken, *Electronic Population Analysis on LCAO-MO Molecular Wave Functions. II. Overlap Populations, Bond Orders and Covalent Bond Energies*, The Journal of Chemical Physics **23**, 1841 (1955).

[3] E. Block, *The Chemistry of Garlic and Onions*, Scientific American **252**, 114 (1985).