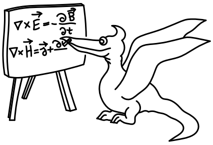
$\nabla\times\vec{E}=-\frac{\partial\vec{B}}{\partial t}$
$\nabla\times\vec{H}=\vec{J}+\frac{\partial\vec{D}}{\partial t}$

Serial: Quanta of molecules

In this part, we will finally look at how to use quantum physics for calculations involving molecules.

Nothing is that simple

As you may have already guessed, and the end of the second issue of this series may have told you, doing a calculation for a whole molecule is not that simple. During the process of finding the solution, we have to make a number of simplifications. We start with the decision of which physics of the problem we want to use, that is, what physical effects we take into account. Firstly, we usually do not consider the effects of special relativity, for example, that a moving electron is heavier. The second, more fundamental assumption, is that atomic nuclei are many times heavier than electrons, and thus will move on much slower timescales. As a result, the nuclei appear almost stationary to the electrons, and we can solve for their motion (their wave function) with the nuclei fixed, and possibly deal with the motion of the nuclei later. For the same reason we can also neglect the quantum nature of nuclei and consider them as point charges.

As a result, we solve the Schrödinger equation only for a set of electrons in electrostatic field of the fixed nuclei. We of course do not want to neglect the electrostatic interaction between electrons, so the Hamiltonian will contain the appropriate elements. However, if you recollect the end of the second part again, even this is not easy. We indicated there that we have no chance to find the exact solution of the Schrödinger equation when we have more than a few electrons. And let's remember that we are often interested in molecules that have tens of atoms and a few hundred electrons.

A solution to this problem is easy, just recollect the previous issue of the series. We introduced the basis functions. If we want to do calculations for molecules, we have many optimized sets available. Most of these are functions that resemble atomic orbitals as we know them, only they are defined in a way that makes them easier to compute numerically. During the calculation, we use one set of functions for each atom with parameters specific for that atom and centered around the position of that atom. Usually, we only need a few dozen of basis functions for each atom.

It could seem we are done – we have managed to reduce the problem to a few hundred or a few thousand basis functions, even though before introducing the basis we had more lattice points than there are particles in the universe! However, the opposite is true. We need to understand that we do not have a one-particle problem. That is, the main "building" blocks are not the orbitals themselves, but the states where we have electrons stacked in some way in a given set of orbitals. Each such state will then have its own unknown coefficient. How many such states are there? We can easily calculate that. We have N basis functions, so N different orbitals into which we can put electrons. In each orbital there can be an electron with spin up, down, both, or neither. That means, that we have $2N$ boxes into which we place electrons one by one. The number of possibilities is then, if the number of electrons in k, given by the binomial coefficient $\binom{2N}{k}$. This is still an astronomical number for most systems; for example, if we have 40 electrons and 100 orbitals, the number of possibilities is on the order of 10^{42}. And this still corresponds to a fairly small molecule, such as benzene. Fortunately, there is an approximation that can get us out of this mess.

The Hartree-Fock approximation

How solve this? The problem is that we are trying to solve the problem for all electrons at once. We need to try to divide the problem so that we have an independent equation for each electron. But at the same time we do not want to neglect the repulsion between the electrons as we have been doing until now. The first step is to abandon the idea of having different configurations in the wave function, that is, that all the different ways in which the orbitals can be occupied contribute to the wave function. Instead, we use the fact that the wave function of most molecules can be described by a single configuration, such that we sequentially fill the orbitals with the lowest energy from the bottom up with two electrons. The whole problem then simplifies to finding the correct shape of these occupied orbitals. But how do we find these orbitals? Let's imagine someone told us what all occupied orbitals look like except for one. Then we could take the wave function of all these orbitals and produce from them the probability density which we introduced in the second part of the series. However, if we multiply this probability density by the charge of an electron, we get the charge density!

Thus, we know both the electrostatic potential of the nuclei and the charge distribution of the other electrons. Now nothing can stop us from formulating the Schrödinger equation for the last orbital using the potential given by this charge distribution. We can solve it easily, as we only have a few thousand unknowns at a time, which is a piece of cake for a computer. (We are looking for only one unknown coefficient for each basis function.) The result is the wave function of the last orbital and the energy of the electron in that orbital.

And how do we deal with the fact that we do not know the wave functions of the individual occupied orbitals? In the beginning, we make some sort of guess. We then use these orbitals to construct the hamiltonians for the individual electrons. We solve these equations and obtain a new set of orbitals that are more accurate than our original estimate. Then we repeat the whole process with the new orbitals, continuing until we get the same orbitals we used to make the hamiltonian, i.e. the orbitals converge.

Why is the result of this approximation not exact? The catch is that the electrostatic potential from other electrons in the Schrödinger equation for the orbital is independent of the position of this electron. In the real world, we can certainly imagine that depending on where this electron is located, it will deform other orbitals due to electrostatic repulsion. Unfortunately, we cannot describe this effect by solving an equation for each single electron separately.

The total error of this approximation is around 1 % in the resulting energy. Unfortunately, for describing chemical processes we usually need significantly higher accuracy. As a result a large part of the development of methods for quantum chemical calculations focuses on how to overcome this limitation without significantly increasing the computational complexity. But why is one percent such a large error?

We have to realize that the total energy of a molecule which we calculate is the energy that we would need to "dissassemble" the molecule into individual electrons and nuclei and to move them to infinite distance form each other. This energy is enormous compared to the energies of chemical reactions. In chemical reactions, only the orbitals in the valence shell rearrange. Therefore, when we calculate the energy of a reaction, we have to calculate the total energy of the products and subtract the total energy of the reactants. If you have experience with how deviations change under mathematical operations, you know that the error in the interaction energy will be enormous. It can be even higher than the energy of the reaction itself. That is why we try to compensate for the error in the energy of reactants and products as well as possible – we have to use the same method and basis set for both calculations.

And what about the geometry of the molecule?

As we already mentioned in the beginning, the positions of the nuclei are fixed during the entire calculation and we need to input their coordinates at the beginning of the program. But where do we get these coordinates? While we can sometimes obtain them from some experiment, such as X-ray diffraction, in most cases we do not know them in advance. What do we do then? We only need to know what we want the molecule to look like, and from that we can determine its approximate geometry. For that we calculate the energy. Then, we slightly move the nuclei in a certain direction and see if the energy decreases or not. Gradually, we can arrive at geometry with the lowest energy. An advantage is that we do not have to "blindly" look for these movements of nuclei, but we can directly calculate the energy gradient from the wave function, i.e. we can see the direction in which the energy decreases the fastest.

Thanks to this, we can even find structures of molecules and complexes that are almost impossible to observe experimentally, such as short-lived intermediates of chemical reactions.

And what does it look like in practice?

Now, we will look at how the calculations look like in practice. We will use the program Psi4, which is a free software. (`https://psicode.org/`) Let's try to illustrate the whole process with a molecule of water. We need to prepare an input file where we tell the program what we want it to do.

First, we need to describe the molecule. In the input file we enter the geometry of a molecule of water, this is done by

```
molecule{
0 1
 O -1.818556 -0.320567 1.225399
 H -0.898992 -0.062457 1.771117
 H -2.278145 -1.054597 1.903623
}
```

Primarily, we should be interested in the first line `0 1`. The first number indicates the charge of the molecule. Therefore, the number `0` says, that we have a neutral molecule. In the case of a cation, we would have a positive number, for an anion we would have a negative number. The second number indicates a spin multiplicity. This is a somewhat complicated concept, which describes the total number of unpaired electrons, i.e. how many more electrons have the spin up than down. For historical reasons, it is designated by a number one higher than the number of unpaired electrons. If we have all electrons paired, it is a singlet state, and we denote it by `1`. The state with two unpaired electrons is a triplet (`3`). Radicals can also have one unpaired electron, then we have a doublet. Anyway, the vast majority of molecules are singlet in the ground state, so the first line is rarely something other than `0 1`. A notorious exception is the oxygen molecule O_2 which is a triplet. The coordinates of the nuclei given in angstroms follow.

We set the basis with the line `set basis cc-pvdz`. The abbreviation `cc-pvdz` denotes one of the commonly used basis. Other possible options are, increasing in size (i.e. with increased precision and also computational complexity), `sto-3g, cc-pvdz, cc-pvtz, cc-pvqz`.

Now that we have this defined, we can add the following line at the end `optimize("HF")`, which starts the geometric optimization. `"HF"` says, that we want to optimize by the Hartree-Fock method. If we already had the correct coordinates of nuclei and wanted only to calculate the energy, we would need only to use `energy("HF")`.

If we put all these lines together, we get the whole input for the program:

```
set basis cc-pvdz
molecule {
0 1
 O -1.818556 -0.320567 1.225399
 H -0.898992 -0.062457 1.771117
 H -2.278145 -1.054597 1.903623
}
optimize("HF")
```

which we can save as a file called, for example, `input.inp`. The calculation can be started in the command line using `psi4 input.inp`.

After running the program, we will find in the folder the file `input.inp.dat`, possibly `input.out` or similar, which contains all information from the program run. If all went well, we will find the optimized coordinates of individual atoms and the total energy of the whole molecule at the end of the file. In our case, we get the following at the end of the file:

```
...
 ==> Convergence Check <==

 Measures of convergence in internal coordinates in au.

 Criteria marked as inactive (o), active & met (*), and active & unmet ( ).

 ---------------------------------------------------------------------------
 Step Total Energy Delta E Max Force RMS Force Max Disp RMS Disp
 ---------------------------------------------------------------------------
 Convergence Criteria 1.00e-06 * 3.00e-04 * o 1.20e-03 * o
 ---------------------------------------------------------------------------
 7 -76.02703278 -6.09e-06 3.51e-05 * 2.65e-05 o 1.79e-04 * 1.05e-04 o
 ---------------------------------------------------------------------------

Next Geometry in Ang
 Fragment 1 (Ang)

 O 0.0057046143 -0.0645290211 0.0000000000
 H -0.7911519198 0.4458305979 0.0000000000
 H 0.7006716423 0.5777071999 0.0000000000


 Final optimized geometry and variables:
 Molecular point group: cs
 Full point group: Cs

 Geometry (in Angstrom), charge = 0, multiplicity = 1:

 O 0.005700842385 -0.064489709337 0.000000000000
 H -0.791211319490 0.445805409758 0.000000000000
 H 0.700734813858 0.577693076293 0.000000000000


 Psi4 stopped on: Wednesday, 04 January 2023 05:36PM
 Psi4 wall time for execution: 0:00:01.40

*** Psi4 exiting successfully. Buy a developer a beer!
```

From the column marked `Total energy` in the table, where we find the number `-76.02703278`, we find that the total energy of the molecule is $-76.027\,\mathrm{Ha}$. The units Ha denote hartree, and one hartree corresponds to $27.21\,\mathrm{eV}$, so about $2625.5\,\mathrm{kJ{\cdot}mol^{-1}}$ or $4.36\cdot10^{-18}\,\mathrm{J}$. If we had an optimized geometry and calculated only the energy, we would only need to look for the line `Total Energy = -76.02703278`. Just before the end of the file is the resulting optimized geometry of the molecule.

A few remarks at the end

Both the search for the right orbitals and the geometric optimization are iterative procedures. Thus it can happen, and quite often does, that the iterations do not converge, so even if we'd let the program run infinitely long, we would not get the correct result. For this reason, the program is set to throw an error and end if it does not converge after a certain number of iterations. In such case, you have to try to run it again with a slightly different input geometry, or even different program settings, but that already requires an experienced quantum chemist.

At the same time, the Hartree-Fock method, in its original formulation, assumes a molecule that has all the orbitals of the molecule either occupied by two electrons or empty. This automatically implies that the molecule is in a singlet state. If we want to calculate a molecule with a different spin (for example, the triplet of the aforementioned oxygen), we need to lift this requirement with the line `set reference rohf`.

When preparing the geometry of more complicated molecules, it is often useful to use graphical programs, where we can "draw" the molecule. A good program in this respect is for example Avogadro.

And how to install and use Psi4 on Windows?

You can download the installation program for Windows directly from the PsiCode website[1]. During the installation, you can check the option "Add Psi4conda to my PATH environment variable", then it is possible to use the Windows command line directly to run it. In either case, after the installation, an "Anaconda prompt" program will appear in the Start menu, which you can use in exactly the same way. If we open the Anaconda prompt, we can type "psi4" to verify that everything works. The program should start and scold us for not having an input file.

The next step is, therefore, to take the input file (you can use, for example, the one from this text), and pass it to the program as a parameter. The only thing to watch out for is what folder we are in at the command line and where the input file is stored. It is either necessary to pass the full path to the file (e.g.: `psi4 Documents\psi4\input.inp`, if we create a folder for inputs named psi4 in Documents). The second option is to use the command cd, which allows us to move between folders. (So we would use the command `cd Documents`, then `cd psi4`, and

[1] `http://vergil.chemistry.gatech.edu/psicode-download/Psi4conda-1.7-py38-Windows-x86_64.exe`

finally run the program using `psi4 input.inp`) If we wanted to move to the parent folder, there is the command `cd ..` for that. And the command `dir` lists the contents of the current folder. The result of running the program is in this case the file `input.inp.out`, which we can open using Notepad.

This concludes this part of the series, and next time we will follow up by showing what happens when we allow the nuclei to move after all.

FYKOS is organized by students of Faculty of Mathematics and Physics of Charles University.
It's part of Media Communications and PR Office and is supported by Institute of Theoretical Physics of MFF UK, his employees and The Union of Czech Mathematicians and Physicists.
The realization of this project was supported by Ministry of Education, Youth and Sports.