Simulating photophysical properties of organic molecules and organometallic complexes using ORCA, DFT and Post-Hartree-Fock methods in a high-performance computing environment.

Domllermut C. Alamo | Dr. David Hrovat | Dr. Thomas R. Cundari | Department of Chemistry | University of North Texas

Simulating photophysical properties of inorganic and organic complexes have been made easier by the quantum chemistry package known as ORCA (especially in the latest versions from 5.0 and on). A compilation of techniques and experiences working on organic molecules and inorganic complexes in a High-Performance Computing (HPC) environment are presented in this poster as a guide for those who want to simulate photophysical properties such as uv-Vis absorption, infrared, fluorescence, phosphorescence, electro paramagnetic resonance (EPR) and Raman spectroscopy.

Density-functional theory (DFT) is the preferred computational quantum mechanical modelling method for many since its computational costs are relatively low when compared to traditional methods, but this lower cost is balanced out by a lower accuracy. With access to a powerful HPC, there is value on trying computational methods with higher computational cost and accuracy, like post-Hartree-Fock methods (STEOM, DLPNO-CCSD and others).

The information compiled here is a starting point to those interested in research aimed at photophysical modeling. Check the link to the github page for more details and examples. A summary of experiences are presented in this list:

- The first thing needed to run a job in ORCA is the initial geometry information. It could be a drawing from a commercial (Gaussview, Chemdraw) or open-source software (Avogadro). It could also be downloaded as the crystal information from a crystal database as Cambridge Crystallographic Data Centre (CCDC). Output files could be read using Chemcraft, notepad, or Avogadro. **<u>Important</u>: Mercury (could be found on the CCDC webpage) will also be needed to handle files from the crystal database, which might require to normalize the bond length to hydrogen atoms in the molecule (limitations of X-ray diffraction). Once the desired molecule is normalized and edited, then it could be used to build input files.**
- Several examples of those input files are presented here. For bigger molecules, post-Hartree-Fock methods could be so expensive that those calculations could take months to finish, so it is recommended to start small, with DFT, unless higher accuracy is a must. Also, higher order functional and basis set could increase computational cost. In these examples, for both DFT and CCSD, there was a massive increase in computational time to finish the job by simply using DEF2-TZVP instead of DEF2-SVP. A SVP job could take a couple of minutes while the equivalent on TZVP could take more than 10 hours, it doesn't sound so bad, but if the SVP job took 6 days to finish, the TZVP job could take several months. Choosing the right method, functional and basis set will impact dramatically the computational cost. **<u>WARNING!</u>: Word of advice, start as small as possible, then ramp up accuracy if needed. Doing the opposite could potentially take months of calculations, or the files could end up in an error because of the high demand on RAM or disk memory needed when very expensive methods are used.**
- Most jobs could be accelerated by increasing the available memory RAM per CPU core, or/and increase the number of cores on parallel calculations. But some methods in specific, like ESD, do not do parallel calculations so they are forced to run these jobs on a single CPU core.
- If ORCA is installed on a HPC, it will be set up to run under cluster management and job scheduling system (like SLURM). They require the building of a shell script to then run the input file. Per example, in SLURM, after build the .sh file, in order to submit the job, it will be required to run it as from the folder where it contains the files, on the HPC terminal: sbatch job_example.sh & (it will run your job_example.inp) (important: in the .sh, there are line that states which input and output will be considered, the most important one is the .out file extension, but others like .hess, .spectrum and others are needed too. They should be included.
- If ORCA was installed on a personal computer, no jobs scheduling system will be needed, just go to the folder where your inputs files are located, open terminal (Windows, Linux, or Macs), and run them as:
ORCA job_example.inp > job_example.out (for personal computers, be careful of the amount of RAM, disk space and numbers of cores needed for calculations. Also, flash drives lose their integrity after too many cycles of reading and writing. Magnetic disk drives are recommended since they don't lose integrity as much as their flash or SSD counterparts) .
- Lastly, on HPC, but more importantly on PC, the system could overwrite files that are in the folder. One way to avoid problem, is to make new folders for those calculations and use the main folder to store initial data.
- There is also access to a github page with manual, examples (input, output, bash files, etc), web addresses, how to guide, and other relevant information for installing and running jobs in ORCA on your HPC or PC. →

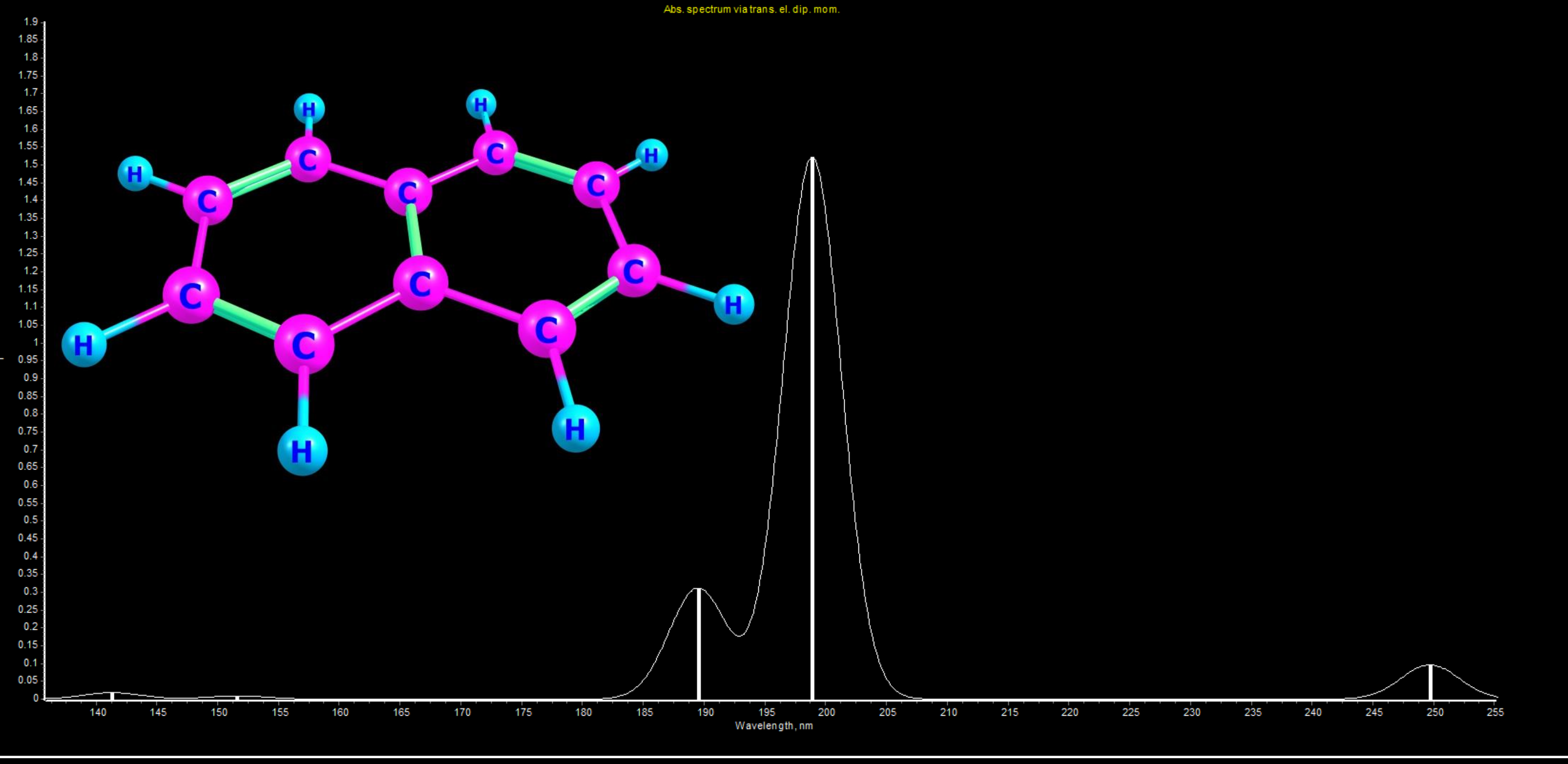
https://qrco.de/orca_output_examples

Uv-vis absorption

```
!STEOM-DLPNO-CCSD DEF2-TZVP DEF2-TZVP/C def2/J RIJCOSX TIGHTSCF CPCM(toluene) RI-SOMF(1X) ESD(ABS)

%maxcore  120000

%mdci
    NRoots 15 #no of roots (excited states)
    DOSOLV  TRUE   # (solvation)
    DONTO   TRUE   # (Natural Transition Orbitals)
end

%ESD
    GSHESSIAN "Abs_TZVP_S0_naphtalene.hess"
    DOHT TRUE
    HESSFLAG AHAS    # uv-Vis absorption method
END

* xyz 0 1
  C   -8.96121784812031    0.24268337776474    0.00033489819780
  C   -7.59059951968750    0.24371545417717    0.00015198745293
  C   -6.86693343298993    1.46030211281662    0.00023475747764
  C   -7.58799559947894    2.69205204792658    0.00050997612007
  C   -9.00310055342497    2.65617792309838    0.00069280404132
  C   -9.67467975506609    1.46137108557408    0.00060814266859
  H   -4.90710891896316    0.55933209328366   -0.00016767341217
  H   -9.50425962457938   -0.69409088109167    0.00026687164944
  H   -7.04027568972576   -0.68980140749936   -0.00005544925225
  C   -5.45183741157070    1.49616229462644    0.00005614405431
  C   -6.86437066397562    3.90865807672096    0.00059715121121
  H   -9.54784882567221    3.59299899012176    0.00091076313657
  H  -10.75737868171900    1.44626581403603    0.00074783188745
  C   -5.49375285629476    3.90967522182525    0.00042517705822
  C   -4.78028626336214    2.69100004727155    0.00015218718405
  H   -7.41470937199301    4.84216327382740    0.00079951152285
  H   -4.95071130646203    4.84645090655905    0.00050815101568
  H   -3.69758703691443    2.70613312896136    0.00001676798628
*
```

ESD method could be too expensive for uv-Vis absorption. A simpler version could be done without it.

Delete ESD(ABS) and the whole %ESD part of the input file if needed.



Geometry optimization and frequency calculations

```
!UKS B3LYP DEF2-TZVP(-F) RIJCOSX D4 TIGHTSCF CPCM(CH2Cl2) OPT FREQ

%maxcore  62000
%PAL
   NPROCS 4
END

# Ni (II) (tppz)2 2+

* xyz 2 3
Ni    0.000000000000    0.000000000000    0.000000000000
N     0.000000000000    0.000000000000    2.022675000000
N     0.000000000000    0.000000000000    4.712268000000
C    -1.172167000000    0.227546000000    2.653816000000
C    -4.408622000000    0.000000000000   -0.095127000000
C    -3.683596000000    0.083969000000    2.213896000000
H    -3.911823000000    0.063590000000    3.270619000000
C    -4.718829000000   -0.005481000000    1.272335000000
C    -1.104881000000    0.425984000000    4.065166000000
N    -2.063651000000    0.150186000000    0.429508000000
C    -3.618275000000    2.912541000000    5.013229000000
N    -2.733523000000    2.195136000000    4.286369000000
C    -2.357844000000    0.773415000000    6.218507000000
H    -1.839208000000   -0.073458000000    6.653575000000
C    -2.356789000000    0.182621000000    1.765662000000
C    -3.927410000000    2.610596000000    6.351873000000
H    -4.648088000000    3.215222000000    6.892531000000
C    -3.061142000000    0.068196000000   -0.475206000000
C    -2.115184000000    1.138991000000    4.882086000000
C    -3.284531000000    1.521458000000    6.961635000000
H    -3.499186000000    1.259339000000    7.993005000000
H    -5.748992000000   -0.081760000000    1.604941000000
H    -4.087450000000    3.752754000000    4.509921000000
H    -5.182006000000   -0.062269000000   -0.852555000000
H    -2.775345000000    0.046475000000   -1.519919000000
C     1.172167000000   -0.227546000000    2.653816000000
C     4.408622000000    0.000000000000   -0.095127000000
C     3.683596000000   -0.083969000000    2.213896000000
H     3.911823000000   -0.063590000000    3.270619000000
C     4.718829000000    0.005481000000    1.272335000000
C     1.104881000000   -0.425984000000    4.065166000000
N     2.063651000000   -0.150186000000    0.429508000000
C     3.618275000000   -2.912541000000    5.013229000000
N     2.733523000000   -2.195136000000    4.286369000000
C     2.357844000000   -0.773415000000    6.218507000000
H     1.839208000000    0.073458000000    6.653575000000
C     2.356789000000   -0.182621000000    1.765662000000
C     3.927410000000   -2.610596000000    6.351873000000
H     4.648088000000   -3.215222000000    6.892531000000
C     3.061142000000   -0.068196000000   -0.475206000000
C     2.115184000000   -1.138991000000    4.882086000000
C     3.284531000000   -1.521458000000    6.961635000000
H     3.499186000000   -1.259339000000    7.993005000000
H     5.748992000000    0.081760000000    1.604941000000
H     4.087450000000   -3.752754000000    4.509921000000
H     5.182006000000    0.062269000000   -0.852555000000
H     2.775345000000   -0.046475000000   -1.519919000000
N     0.000000000000    0.000000000000   -2.022675000000
N     0.000000000000    0.000000000000   -4.712268000000
C     0.227546000000    1.172167000000   -2.653816000000
C     0.000000000000    4.408622000000    0.095127000000
C     0.083969000000    3.683596000000   -2.213896000000
H     0.063590000000    3.911823000000   -3.270619000000
C    -0.005481000000    4.718829000000   -1.272335000000
C     0.425984000000    1.104881000000   -4.065166000000
N     0.150186000000    2.063651000000   -0.429508000000
C     2.912541000000    3.618275000000   -5.013229000000
N     2.195136000000    2.733523000000   -4.286369000000
C     0.773415000000    2.357844000000   -6.218507000000
H    -0.073458000000    1.839208000000   -6.653575000000
C     0.182621000000    2.356789000000   -1.765662000000
C     2.610596000000    3.927410000000   -6.351873000000
H     3.215222000000    4.648088000000   -6.892531000000
C     0.068196000000    3.061142000000    0.475206000000
C     1.138991000000    2.115184000000   -4.882086000000
C     1.521458000000    3.284531000000   -6.961635000000
H     1.259339000000    3.499186000000   -7.993005000000
H    -0.081760000000    5.748992000000   -1.604941000000
H     3.752754000000    4.087450000000   -4.509921000000
H    -0.062269000000    5.182006000000    0.852555000000
H     0.046475000000    2.775345000000    1.519919000000
C    -0.227546000000   -1.172167000000   -2.653816000000
C     0.000000000000   -4.408622000000    0.095127000000
C    -0.083969000000   -3.683596000000   -2.213896000000
H    -0.063590000000   -3.911823000000   -3.270619000000
C     0.005481000000   -4.718829000000   -1.272335000000
C    -0.425984000000   -1.104881000000   -4.065166000000
N    -0.150186000000   -2.063651000000   -0.429508000000
C    -2.912541000000   -3.618275000000   -5.013229000000
N    -2.195136000000   -2.733523000000   -4.286369000000
C    -0.773415000000   -2.357844000000   -6.218507000000
H     0.073458000000   -1.839208000000   -6.653575000000
C    -0.182621000000   -2.356789000000   -1.765662000000
C    -2.610596000000   -3.927410000000   -6.351873000000
H    -3.215222000000   -4.648088000000   -6.892531000000
C    -0.068196000000   -3.061142000000    0.475206000000
C    -1.138991000000   -2.115184000000   -4.882086000000
C    -1.521458000000   -3.284531000000   -6.961635000000
H    -1.259339000000   -3.499186000000   -7.993005000000
H     0.081760000000   -5.748992000000   -1.604941000000
H    -3.752754000000   -4.087450000000   -4.509921000000
H     0.062269000000   -5.182006000000    0.852555000000
H    -0.046475000000   -2.775345000000    1.519919000000
*
```

Maxcore indicates the amount of memory per CPU core. 75 % of the maximum amount of memory ram is recommended for all calculations.

Nprocs, for the total number of requested CPU cores to run the job. (Note: open MPI, MS-MPI, or any other MPI needed to run cores in parallel)

* xyz 2 3

The first number denotes charge, the second number denotes multiplicity.

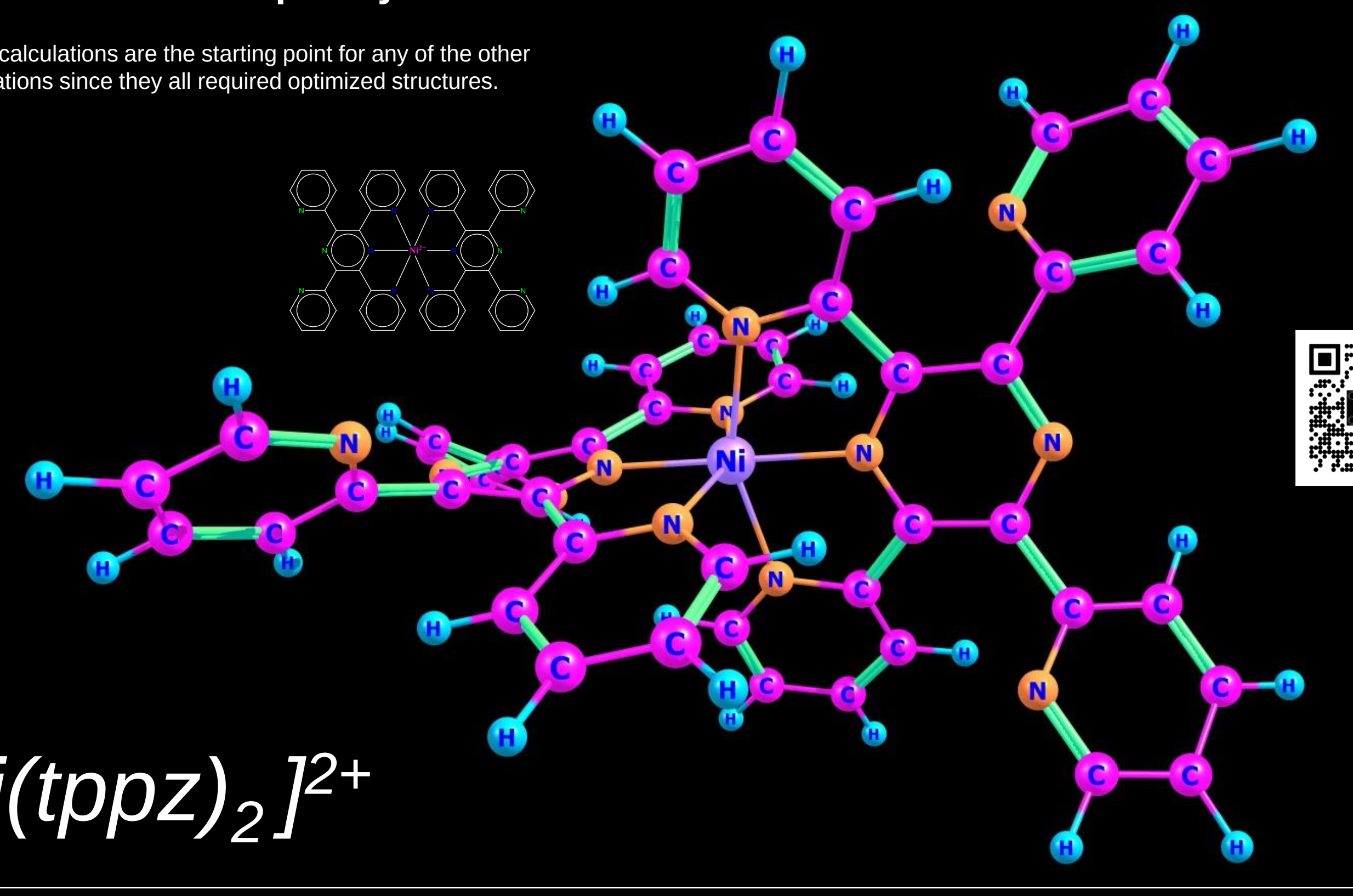
In this case $[Ni(tppz)_2]^{2+}$ is a molecule with charge 2+ and in a triplet ground state.

- ! Denotes the beginning of the main instruction.
- UKS, Kohn-Sham wavefunction type, there are different ones for open and closed shells systems, UKS will work for both.
- B3LYP DEF2-TZVP(-F) , this section refers to the functional / basis set. For this example, B3LYP for this type of DFT functional, and DEF2-TZVP(-F) for basis set.
- RIJCOX and other ones that looks like that are auxiliary instructions to accelerate or include information not present in the basis set.
- D4 is for grid, most of the time it is not needed to be added.
- CPCM, for the solvation model, solvents!
- OPT FREQ, optimization and frequency calculation. This part could change in many ways since not all calculations are OPT FREQ. There is a plethora of different types of calculations requested to the system.
- The same ideas could be applied to the rest of the calculations presented here.

Geometric information:

Denoted as (…) to save space.

Opt Freq calculations are the starting point for any of the other calculations since they all required optimized structures.



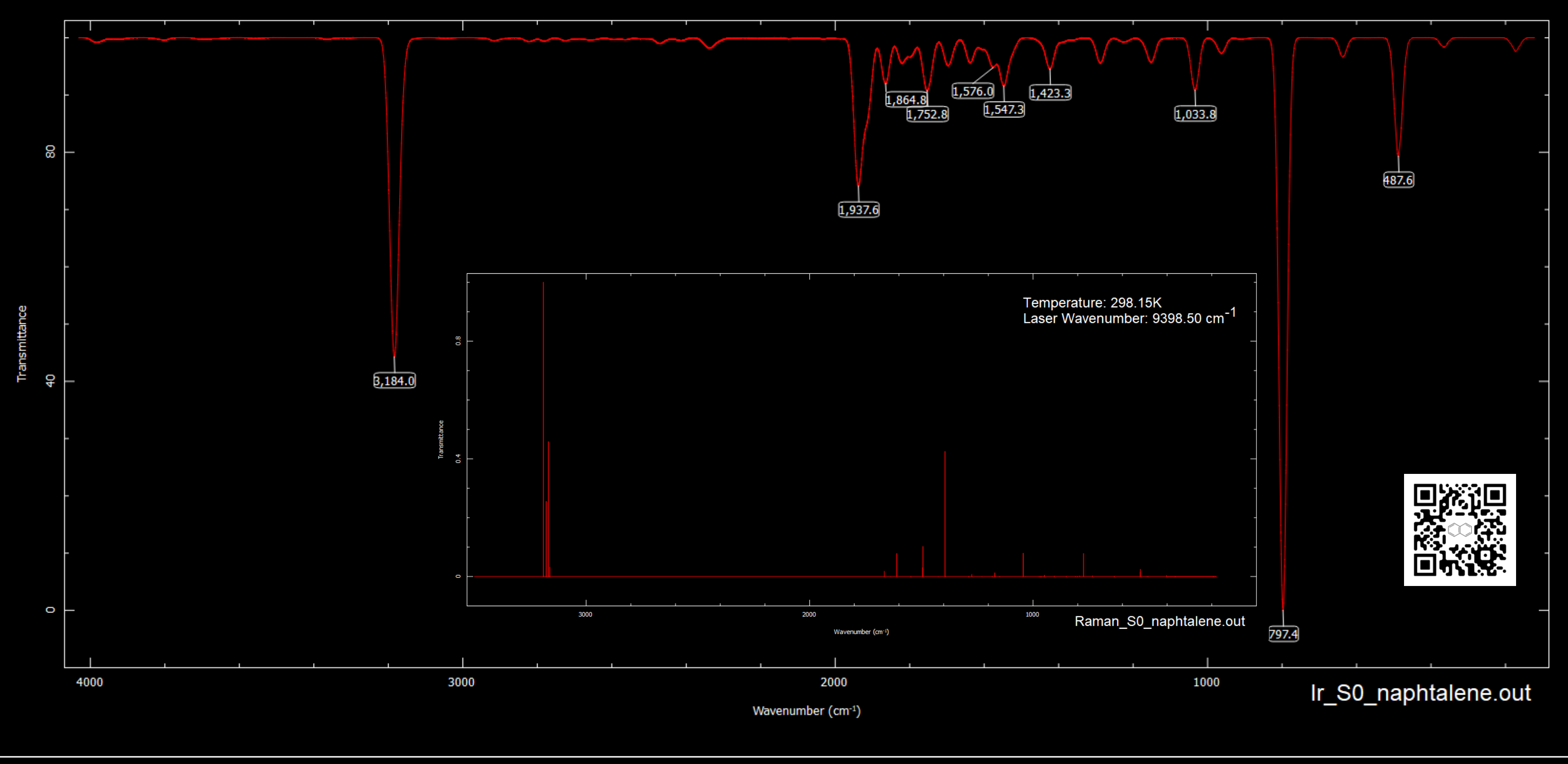
$[Ni(tppz)_2]^{2+}$

Infrared and raman spectroscopy

```
!UKS B3LYP DEF2-TZVP(-F) RIJCOSX TIGHTSCF CPCM(toluene) TIGHTOPT FREQ NEARIR

%maxcore  20000
%PAL
   NPROCS 10
END

%FREQ
    XTBVPT2 FALSE  # False, this is an optional addon that could be installed.
END

# Naphtalene

* xyz 0 1
    …
*
```

Near infrared data can also be obtained this way!

```
 !UKS B3LYP DEF2-TZVP(-F) RIJCOSX TIGHTSCF CPCM(toluene) TIGHTOPT NUMFREQ

 %maxcore  20000
 %PAL
    NPROCS 10
 END

 %ELPROP           # there is not a specific raman module
    POLAR 1    # but numfreq + %elprop Polar 1 prints the raman spectrum
 END

 # Naphtalene

 * xyz 0 1
     …
 *
```



Single point energy calculation

Sometimes, a more accurate energy calculation is needed, CCSD methods are recommended, but since they are so expansive, a nice shortcut is to use DFT to get the geometry optimization of the molecule, and then, with the optimized geometry, CCSD methods are used to get a more accurate energy calculation using a minimal number of computational resources. This way, computational resources are not spent on multiple CCSD steps for geometry calculations, but they are focused instead on finding the energy of what is described here as a single point.

```
!DLPNO-CCSD(T) DEF2-TZVP DEF2-TZVP/C def2/J RIJCOSX TIGHTSCF CPCM(CH2Cl2) RI-SOMF(1X)

%maxcore  120000

%mdci
    DOSOLV  TRUE
    Maxiter 1000 # maximum amount of iterations
end

# pyrazine_Nickel

* XYZ 0 3
    …
*
```

Orca outputs can provide the electronic energy, enthalpy, free energy of the system, and many other type of data, that information could be found at the bottom of the output files.

```
-------------------------   --------------------
FINAL SINGLE POINT ENERGY     -4008.246306545236
-------------------------   --------------------
```

Fluorescence and phosphorescence spectroscopy

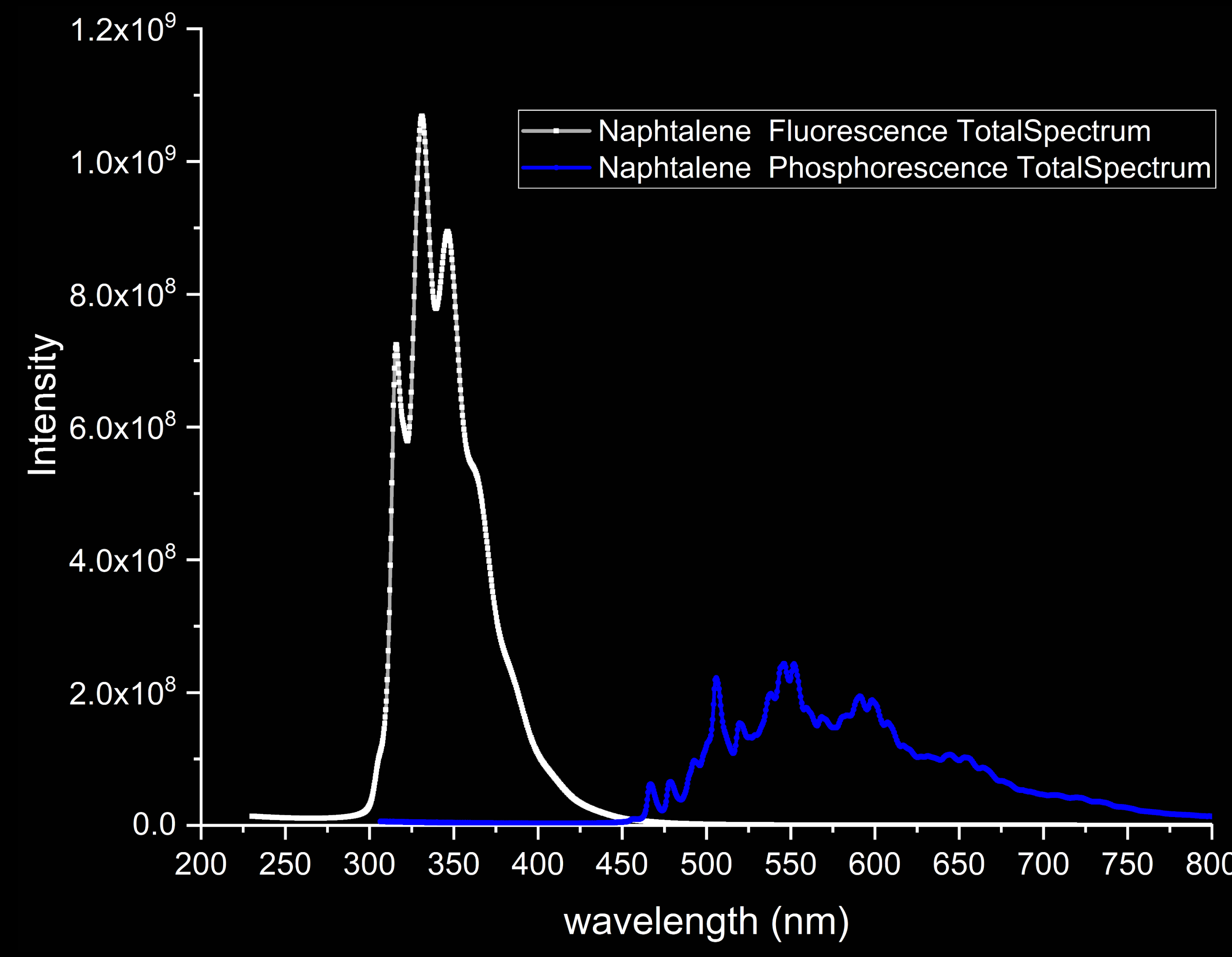
```
!STEOM-DLPNO-CCSD DEF2-SVP DEF2-SVP/C def2/J RIJCOSX TIGHTSCF CPCM(toluene) RI-SOMF(1X) ESD(FlUOR)

%maxcore  120000

%mdci
    NRoots 5 #no of roots
    IROOT 1
    DOSOLV  TRUE
end

%ESD
    GSHESSIAN "Fluor_naphtalene.GS.hess"
    ESHESSIAN "Fluor_naphtalene.ES.hess"
    DOHT TRUE
    LINES VOIGT   # these next 3 lines are about
    LINEW 75      # formating of the spectra
    INLINEW 200
END

* xyz 0 1
    …

*
```

Iroot 1, fluorescence and phosphorescence jobs could study the vertical transition of different states to the ground state. In this case, roots 1, 2 and 3 are being studied for phosphorescence.

The adiabatic hessian method is the one used to get the absorption, fluorescence and phosphorescence spectra from the ESD module. They require the information stored on the Hessian files. They are obtained when submitting the initial opt freq jobs. Simple DFT jobs can do that.

GS denote ground state, ES excited state, and TS triplet state.

```
!STEOM-DLPNO-CCSD DEF2-SVP DEF2-SVP/C def2/J RIJCOSX TIGHTSCF CPCM(toluene) RI-SOMF(1X) ESD(PHOSP)

%maxcore  120000

%mdci
     NRoots 6 #no of roots
     IROOT 1
     DOSOC TRUE   # for spin orbit coupling. More important for heavier atoms.
     DOSOLV  TRUE
end

%ESD
     GSHESSIAN "Phos_naphtalene.GS.hess"
     TSHESSIAN "Phos_naphtalene.TS.hess"
     DOHT TRUE
     DELE 22650.19 # Energy difference from the single point files between GS and TS
END

* xyz 0 1
     …
*

$NEW_JOB

!STEOM-DLPNO-CCSD DEF2-SVP DEF2-SVP/C def2/J RIJCOSX TIGHTSCF CPCM(toluene) RI-SOMF(1X) ESD(PHOSP)

%maxcore  120000

%mdci
     NRoots 6 #no of roots
     IROOT 2
     DOSOC TRUE
     DOSOLV  TRUE
end

%ESD
     GSHESSIAN "Phos_naphtalene.GS.hess"
     TSHESSIAN "Phos_naphtalene.TS.hess"
     DOHT TRUE
     DELE 22650.19
END

* xyz 0 1
     …
*

$NEW_JOB

!STEOM-DLPNO-CCSD DEF2-SVP DEF2-SVP/C def2/J RIJCOSX TIGHTSCF CPCM(toluene) RI-SOMF(1X) ESD(PHOSP)

%maxcore  120000

%mdci
     NRoots 6 #no of roots
     IROOT 3
     DOSOC TRUE
     DOSOLV  TRUE
end

%ESD
     GSHESSIAN "Phos_naphtalene.GS.hess"
     TSHESSIAN "Phos_naphtalene.TS.hess"
     DOHT TRUE
     DELE 22650.19
END

•  xyz 0 1
      …
*
```



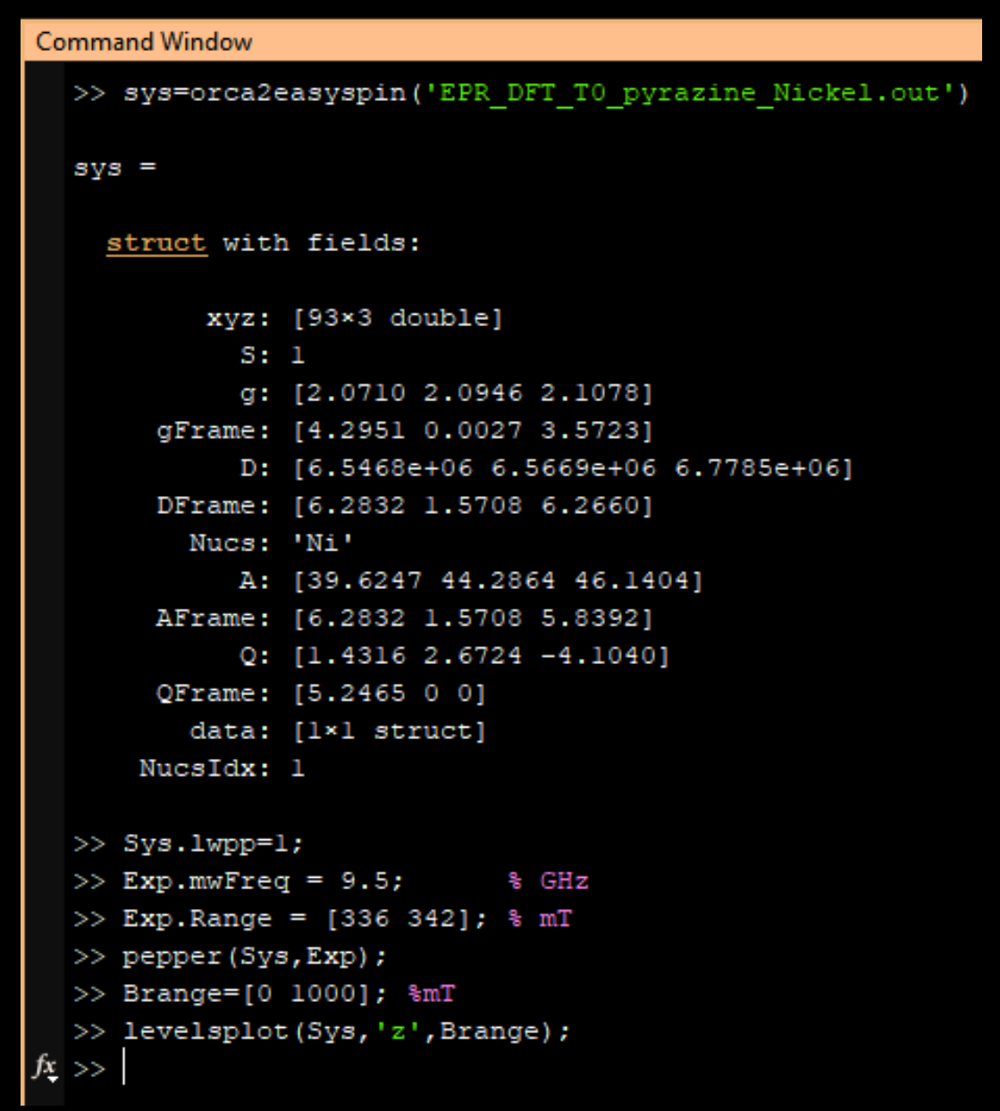
Simulated electro paramagnetic resonance (EPR) spectra

```
!B3LYP ZORA ZORA-DEF2-TZVP SARC/J DEF2-TZVP/C def2/J RIJCOSX TIGHTSCF CPCM(CH2Cl2) RI-SOMF(1X) UNO
%maxcore  30000
%PAL
   NPROCS 15
END
#%BASIS
    #NEWGTO Ni   "SARC-ZORA-TZVP" END
#END

# pyrazine_Nickel

* XYZ 0 3
   …
*

%eprnmr
        gtensor true printlevel 3
        nuclei = all Ni {aiso, adip, aorb, fgrad, rho}
        DTENSOR SSANDSO
        DSS     UNO
        DSOC    CP
end
```

Sometimes, the basis set does not have information on a particular atom, so an auxiliary instruction is added. In this case is deactivated by the # symbol, since it is not needed for nickel, but, per example, if the atom of Platinum was used here, Pt will be placed where Ni is at, and the # should be removed

Just like in Python, # could be used to deactivate the commands after it and use them as way to write comments. Like descriptions or titles in this case

The EPR block comes after the optimized structure in the input file. Uncommon

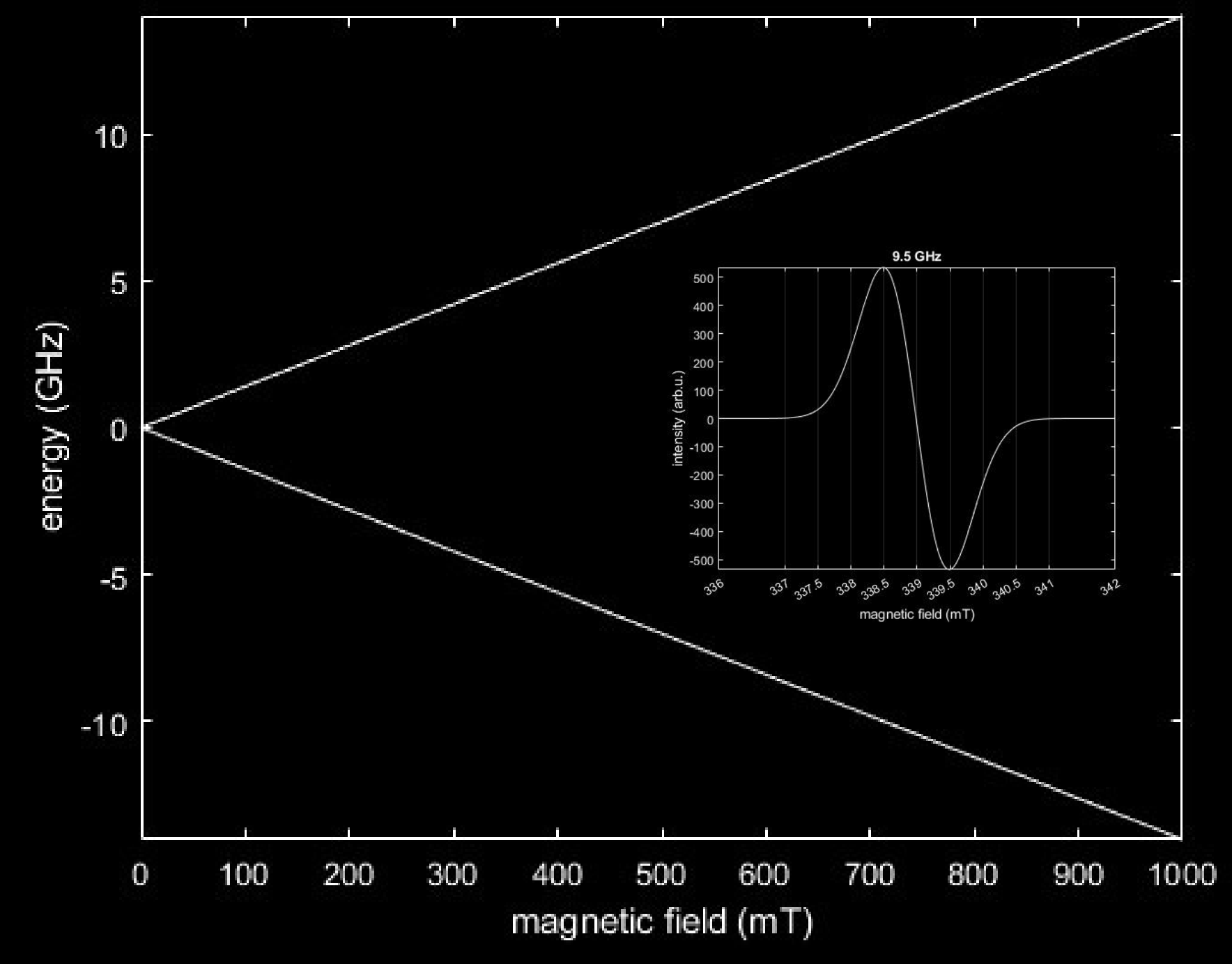


Command Window

```
>> sys=orca2easyspin('EPR_DFT_T0_pyrazine_Nickel.out')

sys =

  struct with fields:

       xyz: [93×3 double]
         S: 1
         g: [2.0710 2.0946 2.1078]
    gFrame: [4.2951 0.0027 3.5723]
         D: [6.5468e+06 6.5669e+06 6.7785e+06]
    DFrame: [6.2832 1.5708 6.2660]
      Nucs: 'Ni'
         A: [39.6247 44.2864 46.1404]
    AFrame: [6.2832 1.5708 5.8392]
         Q: [1.4316 2.6724 -4.1040]
    QFrame: [5.2465 0 0]
      data: [1×1 struct]
   NucsIdx: 1

>> Sys.lwpp=1;
>> Exp.mwFreq = 9.5;      % GHz
>> Exp.Range = [336 342]; % mT
>> pepper(Sys,Exp);
>> Brange=[0 1000]; %mT
>> levelsplot(Sys,'z',Brange);
>> |
```

Easyspin (in Matlab) can read the ORCA output files and simulate the EPR spectra.

It is also possible to simulate NMR spectra.



Acknowledgement:

Calculations were performed at UNT, Cruntch3, NSF grant CHE-1531468 and at Texas Advanced Computing Center (TACC).