Tutorial: VPT2 Frequencies from PhysNet

Silvan Käser
silvan.kaeser@unibas.ch

University of Basel – Meuwly Research Group – November 3, 2023

1 Introduction

This tutorial describes the generation of a (local) PhysNet[1] based potential energy surface (PES) and how second-order vibrational perturbation (VPT2[2]) theory is used to obtain anharmonic frequencies. As example, the water dimer will be studied. To begin with, it is advisable to:

- Start *ab initio* optimizations and frequency calculations for water and the water dimer at the MP2/aug-cc-pVTZ level
- Start *ab initio* VPT2 frequency calculation for water and the water dimer at the MP2/aug-cc-pVTZ level (using the following Gaussian command *# opt=tight freq=anharmonic mp2/aug-cc-pvtz*)
- Search literature for experiments on the water dimer (infrared spectroscopy [fundamentals as well as overtones and combination bands])
- The following papers likely are useful for your understanding
  - PhysNet [1] (the neural network we use to represent the PES)
  - VPT2 + TL [3] (the paper in which we presented the route to VPT2 frequencies)
  - HyDRA [4]
  - General introduction to neural networks [5] (normal mode sampling is described)

2 Installing dependencies and first PhysNet calculations

Most Python scripts that are used to evaluate the PhysNet PESs make use of the atomic simulation environment (ASE)[6] and are written in Python. It is important to get used to ASE, which has very good tutorials online (`https://wiki.fysik.dtu.dk/ase/tutorials/tutorials.html#ase`), once all dependencies are installed.

The following steps will guide you through the installation of dependencies that are required for using PhysNet and the respective VPT2 codes (note that I use particular Python / Package versions). First, on your local computer, install the following.

- If not installed already, install Miniconda on your machine (see `https://docs.conda.io/projects/conda/en/latest/user-guide/install/linux.html`)
- Create a virtual environment with conda (this allows you to have specific versions of Python and respective packages).

```
conda create -n physnet-env python=3.6
conda activate physnet-env
```

- With activated environment, all dependencies can be installed.

```
pip install ase==3.19.1
```

```
pip install tensorflow==1.12
```

- Test the installation by running the examples for formic acid in *tests/formic_acid*

Note that all scripts need to be run with and all packages need to be installed with the activated conda environment. Once the environment is not required anymore, you can deactivate it by

```
conda deactivate
```

To practice how to use a PhysNet PES and to get used to ASE, you could write your own scripts that:

- predict energies, forces and dipole moments for a structure saved in .xyz file format
- performs an optimization starting from an .xyz file and saves the new, optimized structure to a new .xyz file
- performs a harmonic frequency calculation and saves the output into a text file. For comparison, run the same calculation using Molpro or Gaussian at the same level of theory.
- runs a *NVE / NVT* simulation

3 VPT2 Calculation using HCOOH Model

Once the dependencies have been installed on your local computer and you engaged with Python/-PhysNet/ASE, repeat the installation of the environment and all Python packages on the pc-studix cluster. Also, make sure to test the installations using the examples for formic acid in *tests/-formic_acid*. **Let me know if you have problems with this.**

Finally, with the activated environment install the following package:

```
pip install fortranformat==0.2.5
```

The VPT2 calculations are run on *pc-studix* (although in principle other clusters with Gaussian installations should work just as well). Thus, copy all required scripts/input files (available in *scripts/vpt2_hcooh*) to a suitable folder on pc-studix0. Change into the directory and, if not activated yet, activate your conda environment

```
conda activate physnet-env
```

Then, adapt the "NN part" in both *nn_energy_and_force.py* and *nn_freq.py* suitably. For the HCOOH test example this only requires you to adapt the path to the PhysNet models (i.e. the checkpoint path). After you looked at / understood all files in the folder, optimise the structure you want to predict using Gaussian (note that it is advisable to use a structure that has already been optimized using ASE/PhysNet) by going through the following workflow:

```
gview16 opt_hcooh.com
```

```
ctrl + g
```

```
submit
```

```
submit
```

```
yes
```

Note that this runs the calculation locally, but it should not be problematic to write a bash script that runs the calculation on a node. Once the optimisation has finished, copy the optimized structure from *opt_hcooh.log* (Cartesian coordinates are right after "Stationary point found" statement) and adapt suitably (i.e. remove the first three columns and replace them with capitalized letters for atom types). Then, start the VPT2 calculation as outlined above for the optimisation:

gview16 nnvpt2_hcooh.com
ctrl + g
submit
submit
yes

Finally, analyse the anharmonic frequencies and intensities given in the section "Anharmonic Infrared Spectroscopy" of the .log file. It is advisable to:

- go through literature and compare your results with experiment.

4 Normal mode sampling for $(H_2O)_2$

Extensive *ab initio* data is required to train a machine learning-based model for a PES. This can be done based on various approaches (and of course also combinations thereof) including molecular dynamics (MD) simulations, Monte Carlo simulations, or normal mode sampling (NMS)[5, **?**]. Since VPT2 calculations require only a local representation of the PES around the minimum and its zeroth order is the harmonic approximation, NMS is the method of choice. If the PES is supposed to be used for additional evaluations (e.g. MD simulations) additional reference data is required! The following steps illustrate how structures are generated for H_2O and make use of a FORTRAN script available in *normal_mode_sampling*.

- If not already done, perform an optimization + frequency calculation for water. Then, create a new .xyz file on your local computer (e.g. *h2o_opt.xyz*) and from the Gaussian output file copy the optimized structure. It is crucial to use the structure in standard orientation (i.e. search for the first occurrence of "Stationary point found", scroll down a bit to find "Standard orientation"). See example files in *normal_mode_sampling*.
- Create a new .dat file on your local computer (e.g. *h2o_cart_disp.xyz*) and from the Gaussian output file copy the information on the displacement coordinates/force constants to the new file. See example files in *normal_mode_sampling*.
- Adapt the FORTRAN script to your needs.
- Compile the FORTRAN script:

gfortran nms.f90

- Run the program, which will ask for an index (at which the newly generated files' names will start at) and a temperature:

./a.out
0 50

- As defined in the FORTRAN script this will generate 1000 structures (i.e. p0.xyz to p999.xyz) sampled at 50K. Now redo at increasing temperatures:

```
./a.out
1000 100
./a.out
2000 300
./a.out
3000 500
./a.out
4000 1000
./a.out
5000 1500
./a.out
6000 2500
```

- This will leave you with a total of 7000 structures for H_2O, which you can combine into a single .xyz file by doing:

```
for i in {0..6999}; do cat p$i.xyz ; done > water_geoms.xyz
```

- Then, remove all the single files

```
rm p[0-9]*.xyz
```

- Finally, the above procedure can be repeated for the water dimer $(H_2O)_2$

To be able to train PhysNet we will need *ab initio* energies, forces and dipole moments which, for water and the water dimer will be calculated at the CCSD(T)-F12 level of theory. Example scripts to prepare Molpro input files are given in *ab_initio*. First,

- copy the newly generated structures, i.e. *water_geoms.xyz* to the *ab_initio* folder and change to that folder.

- convert the .xyz files to the .traj file format.

```
./xyz_to_traj.py -i water_geoms.xyz
```

- Then, prepare all molpro input files.

```
./SplitUpMOLPROCOC_ccf12.py -i water_geoms.traj
```

Maybe we can discuss when you reach this place.

5 Ab initio Molpro calculations

Ab initio calculations need to be run with Molpro. Copy the folder containing all .inp files you have created in the last chapter to pc-beethoven (or any other cluster that has Molpro installed) along side with the bash scripts that are in the *vpt2_tutorial/ab_initio* folder.

On pc-beethoven, change into the directory where all the .inp files are and make sure the molpro2022_par and the submit_jobs.sh script are there as well. Check the molpro2022_par (when run as molpro2022_par example_job.inp the file creates a .sh file and submits via sbatch command) and make sure it has specified the correct number of cores, memory, queue, etc. Then, create a textfile that lists all .inp files with the command to submit the Molpro jobs:

```
for i in *.inp; do echo ./molpro2022_par $i; done > jobs.queue
```

Then, using the submit_jobs.sh script (the script checks how many of your jobs are already running and queued. If you have less than 10 jobs in the queue it will submit new jobs, i.e. it is essentially a scheduler for your own jobs.) Submit the jobs via scheduler by doing

```
nohup sh submit_jobs.sh < jobs.queue &
```

If you have made a mistake and want stop the submissions (that are performed in the background) you can search for the PID using

```
ps aux —grep submit_jobs.sh
```

and kill the process using the appropriate PID.

```
kill -9 PID
```

Once all your jobs are terminated, make sure you have .out files for all of the jobs and that these jobs all terminated normally. I.e. check how many .inp and .out (note that there will be one "nohup.out" file that you can delete) files you have:

```
ls *.inp —wc -l
```

```
ls *.out —wc -l
```

You can check whether all calculations terminated normally by doing (which should give you the same number as .out files.

```
grep "Molpro calculation terminated" *.out —wc -l
```

If this is not the case you can check which files did not terminated normally by doing

```
grep -L "Molpro calculation terminated" *.out —wc -l
```

This allows you to check for potential error messages. You can run "remove_faulty_out.sh" to list all the files that did not terminate normally (and if you checked that it only lists the files you want you can delete the comment ("#") to let the file actually remove faulty calculations. Then, finally, create a new joblist with the remaining jobs by running

```
./create_joblist.sh > jobs.queue
```

and then submit once more using submit_jobs.sh

6 PhysNet training and evaluation

Before training a NN potential, the *ab initio* data (including energies, forces and dipole moments) needs to be prepared. This can be done with the *extract_ccsdtf12.py* script, for the present case of the water dimer. This script needs to be used with care and adapted suitably if one uses a different level of theory. **[Sena: let me know if something does not work.]** Also note that atomic energies are subtracted from the total energy one obtains from quantum chemistry. Thus, these need to be recalculated at a new level of theory and the script currently only contains the atomic energies for H-, C- and O-atoms!

For the water dimer you only need to adapt lines 12 (in which you give the path to your *ab initio* data) and 123 where you give the data set a suitable name. Then, create the .npz file

```
./extract_ccsdtf12.py
```

and analyze how your data looks by running

```
./analyze_energy.py dataset.npz
```

Now, copy the dataset and the folder *physnet_training* to pc-studix0, where you need to create an additional conda environment following the steps outlined in Chapter 2. This time, however, call it e.g. physnet-env-gpu and install the tensorflow for GPUs (the rest of the steps are the same) using

```
pip install tensorflow-gpu==1.12
```

[Once you are at this step, we can sit together and discuss. I will show you how

to submit a training job to a GPU.]

After installing the tensorflow for GPUs, remove the previous file you did copy and copy your .npz file into *datasets*. Here, open *run_waterdimer.inp* change the file name and adapt the parameters if you need. You can prepare two different simulations by changing *seed* basically.
Now, edit your *run_script.sh* file according to your file, directory, and environment (*physnet-env-gpu*) and save it and run .sh file:

```
sbatch run_script.sh
```

To Do:

- Extend the manual with the steps on how to submit a training to GPUs
- Extend the manual with the steps on how to visualize the training progress with tensorboard (and how to copy the final models)
- Run harmonic frequency calculations using the NN models for the water monomer and dimer
- Run VPT2/anharmonic frequency calculations using the NN models for the water monomer and dimer
- Compare the harmonic frequencies with *ab initio* reference
- Compare the VPT2 frequencies with experiment

References

[1] Oliver Thorsten Unke and Markus Meuwly. Physnet: a neural network for predicting energies, forces, dipole moments and partial charges. *J. Chem. Theory Comput.*, 2019.

[2] Vincenzo Barone. Anharmonic vibrational properties by a fully automated second-order perturbative approach. *J. Chem. Phys.*, 122(1):014108, 2005.

[3] Silvan Käser, Eric D Boittier, Meenu Upadhyay, and Markus Meuwly. Transfer learning to ccsd (t): Accurate anharmonic frequencies from machine learning models. *J. Chem. Theory Comput.*, 17(6):3687–3699, 2021.

[4] Taija L Fischer, Margarethe Bödecker, Sophie M Schweer, Jennifer Dupont, Valéria Lepère, Anne Zehnacker-Rentien, Martin A Suhm, Benjamin Schröder, Tobias Henkes, Diego M Andrada, et al. The first hydra challenge for computational vibrational spectroscopy. *Phys. Chem. Chem. Phys.*, 25(33):22089–22102, 2023.

[5] Silvan Käser, Luis Itza Vazquez-Salazar, Markus Meuwly, and Kai Töpfer. Neural network potentials for chemistry: concepts, applications and prospects. *Digital Discovery*, 2(1):28–58, 2023.

[6] Ask Hjorth Larsen, Jens Jørgen Mortensen, Jakob Blomqvist, Ivano E Castelli, Rune Christensen, Marcin Dułak, Jesper Friis, Michael N Groves, Bjørk Hammer, Cory Hargus, et al. The atomic simulation environment: a python library for working with atoms. *J. Phys. Condens. Matter*, 29(27):273002, 2017.