PES2MP - Quick Guide

Apoorv Kushwaha, Quantum Dynamics Lab, IIT Ropar

2024.R1

1 About PES2MP program

PES2MP is a **potential energy surface (PES)** builder which can automate PES building and augmentation using Neural Networks (N-dimensional augmentation) or analytical fitting (restricted to radial augmentation only) while also automating PES fitting into **Radial Terms** using Legendre polynomials (2D) and Spherical Harmonics (4D) for studying rotational (de-)excitation of any rigid rotor.

The Python-based program automatically creates PES for various collisional pairs e.g. atom - atom (1D), rigid rotor - atom (2D) and rigid rotor - rigid rotor (4D). The program calculates and moves respective Rigid Rotor(s) (RR) to COM and generates PES using Psi4 (for rough estimation). The program also generates input files for external calculations using Molpro/Gaussian/Psi4 along with publication-ready plots with customization (title, axis, etc.) in pdf/eps formats. The users are expected to generate a PES using default coordinates which are sparse but can be changed to make coordinates denser. This sparse PES can then be augmented using ensemble of Neural Networks (NN) models based on the high-level library of TensorFlow i.e. **Keras** where users can choose multiple models to avoid dependency on a singly trained machine learning (ML) model. The final part of the program does multipole expansion of the PES to obtain radial terms which can be analytically fitted into various functions for MOLSCAT calculations.

The first part of the program will take atoms and bond lengths (optimised or experimental) for each collider along with the charge and multiplicity of the fragments. The users can choose from various input files for generating Molpro or Gaussian input files (CP corrected or CBS) and leave the rest of the parameters to default.

For the NN model, a cutoff for very high energy is required, as the program creates separate models for minima and high-energy region to preserve the features that are lost when both regions are trained on a single NN model. The program provides a residual plot for the NN-generated PES and also the PES generated after recreating the surface from radial coefficients (multipole (MP) expanded PES). Currently, the program uses Scipy and pyshtools to carry out MP expansion of 2D and 4D PES by calculating Legendre and spherical harmonics terms respectively.

The analytical fitting of these radial terms or 1D plot (optional) can be carried out using either $\sum_i \alpha_i R^{-\beta_i}$, $\sum_i \alpha_i e^{-\beta_i R}$ or mixed functions. The final output of these analytically fitted functions are provided in MOLSCAT readable format.

2 Instructions for Installation

There are two options for makefiles :

1. Quick install :: **Filename**: *install_pes2mp_quick.sh* || **Conda environment**: *pes2mp_q*)
2. Conda install :: **Filename**: *install_pes2mp.sh* || **Conda environment**: *pes2mp*)

The quick version uses **python -m pip** command (faster but does not check the compatibility among packages) while the normal version uses **conda** command to install required packages. While the conda installation is recommended, it can sometimes be painfully slow to solve environment conflicts among packages and therefore it is suggested that users install both makefiles in separate terminals. If the conda installation doesn't finish in 24 hours (i.e. conda is stuck in a loop and keeps solving the environment), just cancel the installation and use the quick version. If both install normally, users can use any of the two environments to run the program.

2.1 Prerequisite

Anaconda
Install anaconda by visiting anaconda webpage i.e. `https://www.anaconda.com/download`.

It can create environments with specific Python version and two environments are not affected by any change in the base (root) environment. For our work, we shall create **pes2mp** environment utilizing conda and **pes2mp_q** environment utilizing pip for installation of various packages (**makefile is provided**). The important packages are: Psi4, pyshtools, TensorFlow, SciPy, Matplotlib, tqdm, and many more.

2.2 Installing and Running PES2MP

1. Download zipped file from GitHub (link)

2. Goto "**makefile**" folder

3. Open **two** terminal windows (one for quick install and other for conda install) and type the following commands: Users can install either the quick version or the normal version. The instructions are provided for both versions (read above). The quick version installation does not interfere with conda installation. The quick version installs packages without rigorously checking for inter-compatibility and is therefore faster to install. However, conda install ensures that all packages are compatible and is therefore slow to install.

   Terminal 1 (Quick Install) || Terminal 2 (Conda Install):

   ```
   $ chmod +x install_pes2mp_quick.sh          ||    $ chmod +x install_pes2mp.sh
   $ ./install_pes2mp_quick.sh                 ||    $ ./install_pes2mp.sh
   ```

4. Adding pes2mp command to bashrc/bash_profile (Optional):

   After installation, the command to run the program can be shortened by adding a few lines in bash_profile (Mac OS) or bashrc (Ubuntu). This is not essential but makes the program more user friendly. Modify .bash_profile (MacOS) or .bashrc (Most Linux including Ubuntu) in terminal by typing:

   Mac OS || Ubuntu (Linux):

   ```
   $ open -t ~/.bash_profile                   ||    $ gedit ~/.bashrc
   ```

   Now the GUI text editor will open the bashrc/bash_profile. Paste the following lines at the bottom of the text file, save and exit.

   ```
   pes2mp ()
   {
           python3 pes2mp.py $1
   }
   ```

5. To run the program, simply use the command (e.g. Input file = pesgen1D.py):

   Step 4 completed || Step 4 incomplete:

   ```
   $ pes2mp pesgen1D                    ||     $ python3 pes2mp.py pesgen1D
   ```

   The first command works only if the bashrc/bash_profile is edited. If somehow, the bashrc could not be found/edited, the users can still run the program using the longer script:

   Important: The input file must have the "**.py**" extension e.g. plot2D.py. However, while executing (running) the program, simply type **pes2mp plot2D** without the "**.py**" extension.

2.3 Test Calculations

Run test calculation in folder "**test_files**" by typing:

```
$ chmod +x run_test.sh
$ ./run_test.sh
```

2.4 Sample Input Files & Auxiliary Codes

To run the PES2MP program, keep the 3 python files i.e.

```
(1) pes2mp.py, (2) pes2mp_driver.py and (3) $inputfile.py
```

in the same folder. For calculating the PES using external programs like Molpro, Psi4 and Gaussian, the batch files and provided. Similarly, once the calculations are finished, a python script to collect the results (in required format) is provided as well.
To run auxiliary scripts use following commands:

```
Python scripts : $ python3 script.py

Bash scripts   : $ chmod +x script.sh
                 $ ./script.sh
```

3 Capabilities

The input files for each are provided with the program.

1. **PES Calculation :: 1D/2D/4D rigid rotor PES.**
   Simple templates (input_files.py) are provided for Psi4, Gaussian and Molpro which automatically creates required input files (for each individual coordinate in XYZ format) for respective *ab initio* software packages. Sample input files are provided for Single point, BSSE-corrected and CBS-extrapolation schemes for Molpro. Gaussian template defaults to BSSE corrected PES, while Psi4 can be used both internally (using pes2mp) or externally (recommended for large calculations).

2. **Plots :: Plot PES, residual and curve fits.**
   PES plots are available as 1D (R vs E) and polar (R,θ vs E). For 2D/4D PES, certain angle combinations can be chosen for plots to prevent cluttering. Residual (error) plots can be automatically generated for NN/radial-terms generated PES. The curve fit with various functions can also be visualized along with any extrapolation function (if used). Apart from these, many other relevant plots are also provided for NN module.

3. **NN augmentation :: Create NN model to augment and/or get missing data points.**
   The program can be utilized to separately fit minima and high-energy regions for a good fit. The underlying package for the NN model is TensorFlow, which can learn multiple outputs simultaneously and is not restricted by dimensionality (i.e. data can have N inputs and M output columns). The program also keeps boundary elements (of input features) in the training dataset to prevent boundary errors. There are two models: PES-specific (default) and Generic. The PES-specific model uses Gaussian activation functions (custom-made) and a modified Slater function for high-energy and long-range extrapolation. In the generic model, GELU is the default activation function. For finding the most suitable model (layers/nodes/activation function), keras-tuner's (package) Bayesian Optimization tuning is used which utilises the Gaussian process to find optimum parameters within a reasonably defined search space.

4. **Multipole expansion :: Calculate radial terms (v_Λ).**
   The radial terms (v_Λ) are obtained by expanding the interaction potential (i.e. 2D/4D PES) in a set of orthogonal functions (Legendre [2D] and two Spherical harmonics [4D]) of the internal coordinates ([R, θ] and [R, ϕ, θ_2, θ_1]). The current implementation takes PES (must not have any coordinate missing) and multiples it with the pseudo-inverse matrix containing Legendre/Spherical harmonics coefficients to get radial terms. Warning: The number of angles must not be less than λ as it will result in poor fit which can be visualized by recreating the PES.

5. **Curve fit :: Fit 1D PES and/or radial terms into analytical expression.**
   The radial terms (v_Λ) can be fitted into a series of power/exponential functions provided with the pes2mp package or can be defined in the sample template for a more complex fit. Extrapolation schemes (R^{-6}/R^{-4}/R^{-3}/R^{-x} etc.) are also available with plots (optional). After fitting the PES into MOLSCAT readable functions, the PES is recreated (with a residual plot) to verify whether a good fit is obtained or not.

6. **MOLSCAT readable output :: Available for general-purpose version of POTENL subroutine**
   The curve fit produces an output which is MOLSCAT readable for direct utilization in **&potl** block. The feature is available in a general-purpose version of the subroutine POTENL of MOLSCAT 2020. The same can be utilized to calculate cross-sections for rotational (de-)excitation of two colliding species at cold and ultracold temperatures.