PWneb User's Guide (v. 6.2)

(only partially updated)

Contents

1 Introduction

This guide covers the usage of PWneb, version 6.2: an open-source package for the calculation of energy barriers and reaction pathway using the Nudged Elastic Band (NEB) method.

Important notice: due to the lack of time and of manpower, this manual is only partially updated and may contain outdated information.

This guide assumes that you know the physics that PWneb describes and the methods it implements. It also assumes that you have already installed, or know how to install, QUANTUM ESPRESSO. If not, please read the general User's Guide for QUANTUM ESPRESSO, found in directory Doc/ two levels above the one containing this guide; or consult the web site: http://www.quantum-espresso.org.

PWneb is part of the QUANTUM ESPRESSO distribution and uses the PWscf package as electronic-structure computing tools ("engine"). It is however written in a modular way and could be adapted to use other codes as "engine". Note that since v.4.3 PWscf no longer

performs NEB calculations. Also note that NEB with Car-Parrinello molecular dynamics is not implemented anymore since v.4.3.

2 People and terms of use

The current maintainers of PWneb are Layla Martin-Samos, Paolo Giannozzi, Stefano de Gironcoli. The original QUANTUM ESPRESSO implementation of NEB was written by Carlo Sbraccia.

PWneb is free software, released under the GNU General Public License.
See http://www.gnu.org/licenses/old-licenses/gpl-2.0.txt, or the file License in the distribution).

We shall greatly appreciate if scientific work done using QUANTUM ESPRESSO distribution will contain an explicit acknowledgment and the following reference:

> P. Giannozzi, S. Baroni, N. Bonini, M. Calandra, R. Car, C. Cavazzoni, D. Ceresoli, G. L. Chiarotti, M. Cococcioni, I. Dabo, A. Dal Corso, S. Fabris, G. Fratesi, S. de Gironcoli, R. Gebauer, U. Gerstmann, C. Gougoussis, A. Kokalj, M. Lazzeri, L. Martin-Samos, N. Marzari, F. Mauri, R. Mazzarello, S. Paolini, A. Pasquarello, L. Paulatto, C. Sbraccia, S. Scandolo, G. Sclauzero, A. P. Seitsonen, A. Smogunov, P. Umari, R. M. Wentzcovitch, J.Phys.:Condens.Matter 21, 395502 (2009), http://arxiv.org/abs/0906.2569

Note the form QUANTUM ESPRESSO for textual citations of the code. Please also see package-specific documentation for further recommended citations. Pseudopotentials should be cited as (for instance)

> [ ] We used the pseudopotentials C.pbe-rrjkus.UPF and O.pbe-vbc.UPF from http://www.quantum-espresso.org.

3 Compilation

PWneb is a package tightly bound to QUANTUM ESPRESSO. For instruction on how to download and compile QUANTUM ESPRESSO, please refer to the general Users' Guide, available in file Doc/user_guide.pdf under the main QUANTUM ESPRESSO directory, or in web site http://www.quantum-espresso.org.

Once QUANTUM ESPRESSO is correctly configured, PWneb can be automatically downloaded, unpacked and compiled by just typing make neb, from the main QUANTUM ESPRESSO directory. make neb will produce the following codes in NEB/src:

- neb.x: calculates reaction barriers and pathways using NEB.
- path_int.x: generates a reaction path (a set of points in the configuration space of the atomic system, called "images"), by interpolating the supplied path. The new path can have a different number of images than the old one and the initial and final images of the new path can differ from the original ones. The utility path_interpolation.sh in the tools/ directory shows how to use the code.

Symlinks to executable programs will be placed in the bin/ subdirectory of the main QUANTUM ESPRESSO directory.

3.1 Running examples

As a final check that compilation was successful, you may want to run some or all of the examples (presently only one). To run the examples, you should follow this procedure:

1. Edit the `environment_variables` file in the main QUANTUM ESPRESSO directory, setting the following variables as needed:

   BIN_DIR: directory where executables reside
   PSEUDO_DIR: directory where pseudopotential files reside
   TMP_DIR: directory to be used as temporary storage area

The default values of BIN_DIR and PSEUDO_DIR should be fine, unless you have installed things in nonstandard places. The TMP_DIR variable must point to a directory where you have read and write permissions, with enough available space to host the temporary files produced by the example runs, and possibly offering good I/O performance (i.e., don't use an NFS-mounted directory). **N.B.** Use a dedicated directory, because the example script will automatically delete all data inside TMP_DIR. If you have compiled the parallel version of QUANTUM ESPRESSO (this is the default if parallel libraries are detected), you will usually have to specify a driver program (such as `mpirun` or `mpiexec`) and the number of processors: see Sec.4.1 for details. In order to do that, edit again the `environment_variables` file and set the PARA_PREFIX and PARA_POSTFIX variables as needed. Parallel executables will be started with a command line like this:

```
$PARA_PREFIX neb.x $PARA_POSTFIX -inp file.in > file.out
```

For example, the command for IBM's POE looks like this:

```
poe neb.x -procs 4 -inp file.in > file.out
```

therefore you will need to set PARA_PREFIX="poe", PARA_POSTFIX="-procs 4". Furthermore, if your machine does not support interactive use, you must run the commands specified below through the batch queuing system installed on that machine. Ask your system administrator for instructions.

Go to `NEB/examples/examplex01` and execute:

```
./run_example
```

This will create a subdirectory `results/` containing the input and output files generated by the calculation.

The `reference/` subdirectory contains verified output files, that you can check your results against. They were generated on a Linux PC using the Intel compiler. On different architectures the precise numbers could be slightly different, in particular if different FFT dimensions are automatically selected. For this reason, a plain diff of your results against the reference data doesn't work, or at least, it requires human inspection of the results.

4 Parallelism

The `PWneb` code is interfaced to `PWscf`, which is used as computational engine for total energies and forces. It can therefore take advantage from the two parallelization paradigms currently implemented in QUANTUM ESPRESSO, namely Message Passing Interface (MPI) and OpenMP

threads, and exploit all PWscf-specific parallelization options. For a detailed information about parallelization in QUANTUM ESPRESSO, please refer to the general documentation.

As PWneb makes several independent evaluations of energy and forces at each step of the path optimization (one for each point in the configurational space, called "image", consituting the path) it is possible to distribute them among processors using an additional level of parallelization (see later).

4.1 Running on parallel machines

Parallel execution is strongly system- and installation-dependent. Typically one has to specify:

1. a launcher program (not always needed), such as poe, mpirun, mpiexec, with the appropriate options (if any);
2. the number of processors, typically as an option to the launcher program, but in some cases to be specified after the name of the program to be executed;
3. the program to be executed, with the proper path if needed: for instance, ./neb.x, or $HOME/bin/neb.x, or whatever applies;
4. other PWscf-specific parallelization options, to be read and interpreted by the running code;
5. the number of image groups used by NEB (see next subsection).

Items 1) and 2) are machine- and installation-dependent, and may be different for interactive and batch execution. Note that large parallel machines are often configured so as to disallow interactive execution: if in doubt, ask your system administrator. Item 3) also depend on your specific configuration (shell, execution path, etc). Item 4) is optional but may be important: see the following section for the meaning of the various options.

For illustration, here is how to run neb.x on 16 processors partitioned into 4 image groups (4 processors each), for a path containing at least 4 images with POE:

```
poe neb.x -procs 16 -ni 4 -i input
```

4.2 Parallelization levels

Data structures are distributed across processors. Processors are organized in a hierarchy of groups, which are identified by different MPI communicators level. The groups hierarchy is as follow:

```
world - image_group - PWscf hierarchy
```

world: is the group of all processors (MPI_COMM_WORLD).

image_group: Processors can then be divided into different image groups, each of them taking care of one or more NEB images.

Image parallelization is of loosely coupled type, so that processors belonging to different image groups communicate only once in a while, whereas processors within the same image group are tightly coupled and communications are more significant (please refer to the user guide of PWscf).

The default number of image groups is one, corresponding to the option -ni 1 (or, equivalently, -nimage 1).

5 Using PWneb

NEB calculations with neb.x can be started in two different ways:

1. by reading a single input file, specified with the command line option -i (or -in, or -inp);
2. by specifying the number N of images with the command line option -input_images N, and providing the input data for PWneb in a file named neb.dat and for the PWscf engine in the files pw_X.in ($X = 1, ..., N$, see also below).

In the first case, the input file contains keywords (described here below) that enable the code to distinguish between parts of the input containing NEB-specific parameters and parts containing instructions for the computational engine (only PW is currently supported).
N.B.: the neb.x code does not read from standard input, so that input redirection (e.g., neb.x < neb.in ...) cannot be used.

The general structure of the file to be parsed should be as follows:

```
BEGIN
BEGIN_PATH_INPUT
~... neb specific namelists and cards
END_PATH_INPUT
BEGIN_ENGINE_INPUT
~...pw specific namelists and cards
BEGIN_POSITIONS
FIRST_IMAGE
~...pw ATOMIC_POSITIONS card
INTERMEDIATE_IMAGE
~...pw ATOMIC_POSITIONS card
LAST_IMAGE
~...pw ATOMIC_POSITIONS card
END_POSITIONS
~... other pw specific cards
END_ENGINE_INPUT
END
```

After the parsing is completed, several files are generated by PWneb, more specifically: neb.dat, with NEB-related input data, and a set of input files in the PWscf format, pw_1.in, ..., pw_N.in, one for each set of atomic position (image) found in the original input file. For the second case, the neb.dat file and all pw_X.in should be already present in the directory where the code is started. A detailed description of all NEB-specific input variables is contained in the input description files Doc/INPUT_NEB.*, while for the PWscf engine all the options of a scf calculation apply (see PW/Doc/INPUT_PW.* and example01 in the NEB/examples directory).

A NEB calculation will produce a number of output files containing additional information on the minimum-energy path. The following files are created in the directory were the code is started:

prefix.dat is a three-column file containing the position of each image on the reaction coordinate (arb. units), its energy in eV relative to the energy of the first image and the residual error for the image in eV/a_0.

`prefix.int` contains an interpolation of the path energy profile that pass exactly through each image; it is computed using both the image energies and their derivatives

`prefix.path` information used by QUANTUM ESPRESSO to restart a path calculation, its format depends on the input details and is undocumented

`prefix.axsf` atomic positions of all path images in the XCrySDen animation format: to visualize it, use `xcrysden --axsf prefix.axsf`

`prefix.xyz` atomic positions of all path images in the generic xyz format, used by many quantum-chemistry softwares

`prefix.crd` path information in the input format used by `pw.x`, suitable for a manual restart of the calculation

where `prefix` is the `PWscf` variable specified in the input. The more verbose output from the `PWscf` engine is not printed on the standard output, but is redirected into a file stored in the image-specific temporary directories (e.g. `outdir/prefix_1/PW.out` for the first image, etc.).

NEB calculations are a bit tricky in general and require extreme care to be setup correctly. Sometimes it can easily take hundreds of iterations for them to converge, depending on the number of atoms and of images. Here you can find some advice (courtesy of Lorenzo Paulatto):

1. Don't use Climbing Image (CI) from the beginning. It makes convergence slower, especially if the special image changes during the convergence process (this may happen if `CI_scheme='auto'` and if it does it may mess up everything). Converge your calculation, then restart from the last configuration with CI option enabled (note that this will *increase* the barrier).

2. Carefully choose the initial path. If you ask the code to use more images than those you have supplied on input, the code will make a linear interpolation of the atomic positions between consecutive input images. You can visualize the `.axsf` file with XCrySDen as an animation: take some time to check if any atoms overlap or get very close in some of the new images (in that case you will have to supply intermediate images). Remember that QUANTUM ESPRESSO assumes continuity between two consecutive input images to initialize the path. In other words, periodic images are not used by default, so that an unwanted path could result if some atom crosses the border of the unit cell and it is refolded in the unit cell in the input image. The problem can be solved by activating the mininum_image option, which choses an appropriate periodic replica of any atom that moves by more than half the size of the unit cell between two consecutive input images. If this does not work either, you may have to manually translate an atom by one or more unit cell base vectors in order to have a meaningful initial path.

3. Try to start the NEB process with most atomic positions fixed, in order to converge the more "problematic" ones, before leaving all atoms move.

4. Especially for larger systems, you can start NEB with lower accuracy (less k-points, lower cutoff) and then increase it when it has converged to refine your calculation.

5. Use the Broyden algorithm instead of the default one: it is a bit more fragile, but it removes the problem of "oscillations" in the calculated activation energies. If these oscillations persist, and you cannot afford more images, focus to a smaller problem, decompose it into pieces.

6. A gross estimate of the required number of iterations is (number of images) * (number of atoms) * 3. Atoms that do not move should not be counted. It may take half that many iterations, or twice as many, but more or less that's the order of magnitude, unless one starts from a very good or very bad initial guess.

The code `path_int.x` is is a tool to generate a new path (what is actually generated is the restart file) starting from an old one through interpolation (cubic splines). The new path can be discretized with a different number of images (this is its main purpose), images are equispaced and the interpolation can be also performed on a subsection of the old path. The input file needed by `path_int.x` can be easily set up with the help of the self-explanatory `path_interpolation.sh` shell script in the `NEB/tools` folder.

6 Performances

`PWneb` requires roughly the time and memory needed for a single SCF calculation, times `num_of_images`, times the number of NEB iterations needed to reach convergence. We refer the reader to the PW user_guide for more information.

7 Troubleshooting

Almost all problems in `PWneb` arise from incorrect input data and result in error stops. Error messages should be self-explanatory, but unfortunately this is not always true. If the code issues a warning messages and continues, pay attention to it but do not assume that something is necessarily wrong in your calculation: most warning messages signal harmless problems.