User's Guide for PWneb (version 5.0)

Contents

1 Introduction

This guide covers the usage of PWneb, version 5.0: an open-source package for the calculation of energy barriers and reaction pathway using the Nudged Elastic Band (NEB) method.

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 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 this code 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

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`: used by utility `path_int.sh` that generates, starting from a path (a set of images), a new one with a different number of images. The initial and final points of the new path can differ from those in the original one.

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. To run the examples, you should follow this procedure:

1. Go to the `examples/` directory from the main Quantum ESPRESSO directory and edit the `environment_variables` file, 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. TMP_DIR must be a directory where you have read and write access to, with enough available space to host the temporary files produced by the example runs, and possibly offering high I/O performance (i.e., don't use an NFS-mounted directory). NOTA BENE: do not use a directory containing other data, the examples will clean it! 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 run by a command like this:

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

For example, if the command line is like this (as for an IBM SP):

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

you should 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 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

`PWneb` uses `PWscf` as energy and forces engine. It can take advantage from the two different parallelization paradigms currently implemented inside QUANTUM ESPRESSO. For a detailed information about QUANTUM ESPRESSO parallization, please refer to the QUANTUM ESPRESSO general documentation. All `PWscf`-specific parallelization options holds.

As `PWneb` makes several energy and forces evaluations ("images") for each NEB cycle it is possible to distribute them through Message-Passing (MPI).

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 "images" used by NEB.

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 images (4 processors each), for a path containing at least 4 points.

IBM SP machines, batch:

```
neb.x -nimages 4 -inp 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 - images - PWscf hierarchy
```

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

images: Processors can then be divided into different "images", corresponding to a point in configuration space (i.e. to a different set of atomic positions).

Communications: Images is loosely coupled and processors communicate between different images only once in a while, whereas processors within each image are tightly coupled and communications are significant.

Default value is: `-nimage 1` ;

5 Using PWneb

Reminder 1: NEB calculations are no longer performed by `pw.x`. In order to perform a NEB calculation, you should compile `NEB/src/neb.x` (command `make neb` from the main QUANTUM ESPRESSO directory, or commend `make` from the `NEB` directory).
Reminder 2: `neb.x` does not read from standard input. You cannot use input redirection, as in `neb.x < neb.in ...`.

A NEB calculation can be run in two different ways:

1. by reading a single input file, to be specified using command-line options `neb.x -inp` or `neb.x -input`;

2. by specifying the number N of images (i.e. points in the configuration space), using command-line option `neb.x -input_images N`, and providing input data in a `neb.dat` file and in files `pw_X.in`, $X = 1, ..., N$.

For case 1) the input file contains KEYWORDS (see below for format specifications). These KEYWORDS enable the code to distinguish which part of the file contains NEB-specific data and which parts contains input data for the energy and force computational engine (currently only PW). After the parsing, different files are generated: `neb.dat`, with NEB-specific input data, and a set of PWscf input files `pw_1.in`,..,`pw_N.in` PWscf input files, one for each set of atomic position. All options for a single SCF calculation apply.

The general structure of the file to be parsed is:

```
BEGIN
BEGIN_PATH_INPUT
~... neb specific namelists and cards
END_PATH_INPUT
BEGIN_ENGINE_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
```

For case 2), the `neb.dat` file and all `pw_X.in` should be already present.

A detailed description of all NEB-specific input variables is contained in files `Doc/INPUT_NEB.*`. See Example 17.

A NEB calculation will produce a number of files in the current directory (i.e. in the directory were the code is run) containing additional information on the minimum-energy path. The files are organized as following (where `prefix` is specified in the input file):

`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

"NEB calculation are a bit tricky in general and require extreme care to be setup correctly. NEB also takes easily hundreds of iteration to converge, of course depending on the number of atoms and of images. Here is some free advice:

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. Remember that QUANTUM ESPRESSO assumes continuity between the first and the last image at the initial condition. In other words, periodic images are NOT used; you may have to manually translate an atom by one or more unit cell base vectors in order to have a meaningful initial path. You can visualize NEB input files with XCrySDen as animations, take some time to check if any atoms overlap or get very close in the initial path (you will have to add intermediate images, in this case).

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.

(Courtesy of Lorenzo Paulatto)

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_int.sh` shell script.

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.