

Manual of running Climbing Nudged Elastic Band (CNEB) method with Gaussian on Linux

- Purpose of CNEB method: find a "guessed" transition state geometry
- All scripts needed can be found at: /lustre/lwork/yclee/python_practice/cneb
- Example of CNEB: /lustre/lwork/yclee/python_practice/cneb/example
- For detailed explanation of CNEB method, please refer to the papers below:

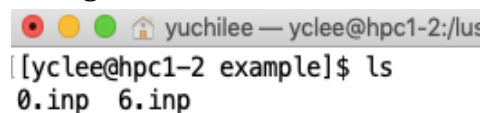
<https://aip.scitation.org/doi/10.1063/1.1329672>

<https://aip.scitation.org/doi/abs/10.1063/1.1323224>

Steps of running CNEB

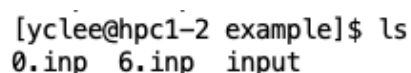
Step1. Prepare the initial state structure and the final state structure in your working directory. The orientation of initial state structure and the final state structure must be the same. Please rename the initial state structure as 0.inp and the final state structure as "number of images+1.inp" (ex: if you wish to have 5 images, rename the final state structure as 6.inp)

* images: the intermediate structures between initial state and final state

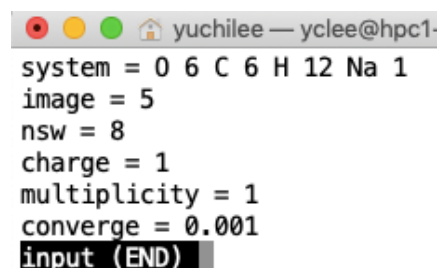


```
yuchilee — yclee@hpc1-2:/lus
[yuchilee@hpc1-2 example]$ ls
0.inp 6.inp
```

Step2. Create an "input" file in your working directory. Please refer to the example below:



```
[yuchilee@hpc1-2 example]$ ls
0.inp 6.inp input
```



```
yuchilee — yclee@hpc1-
system = 0 6 C 6 H 12 Na 1
image = 5
nsw = 8
charge = 1
multiplicity = 1
converge = 0.001
input (END)
```

* system: the atoms and their respective numbers in your structure. Please note that the order of atoms written here should be corresponding to the order in your Gaussian inp files.

* image: the number of images you want

- * nsw: the maximum iterations of CNEB. When reaching this number, CNEB will stop automatically.
- * charge: the charge of your structure (same as the inp files)
- * multiplicity: the multiplicity of your structure (same as the inp files)
- * converge: the convergence criteria of CNEB (unit: Hartree/Bohr) When the maximum perpendicular force acting on atoms becomes smaller than this value, CNEB will stop automatically.

Step3. Use the script "get_image.py" to generate images (intermediate structures).

```
[ycllee@hpc1-2 example]$ python /lustre/lwork/ycllee/python_practice/cneb/get_image.py
[ycllee@hpc1-2 example]$ ls
0.inp 1.inp 2.inp 3.inp 4.inp 5.inp 6.inp input
```

Step4. Use the script "inp_makemovie.py" to create a movie. Please check this movie to ensure that the reaction path and structures make sense.

```
[ycllee@hpc1-2 example]$ python /lustre/lwork/ycllee/python_practice/cneb/inp_makemovie.py
[ycllee@hpc1-2 example]$ ls
0.inp 1.inp 2.inp 3.inp 4.inp 5.inp 6.inp input movie.xyz
```

Step5. Copy rung09 to your working directory. Run CNEB by submitting the job "qsub rung09". The job will automatically stop when the iteration number exceeds nsw.

```
[ycllee@hpc1-2 example]$ ls
0.inp 1.inp 2.inp 3.inp 4.inp 5.inp 6.inp input rung09
[ycllee@hpc1-2 example]$ qsub rung09
50628.master
```

Step6. After running CNEB, use the script "get_scf.py" to learn which image is the "guessed transition state structure". For the example below, image 1 has the highest energy, and thus 1.inp should be the guessed transition state structure.

```
[ycllee@hpc1-2 example]$ ls
0.chk 1.chk 2.chk 3.chk 4.chk 5.chk 6.chk 7.chk 8.chk g09.o50624      rung09
0.inp 1.inp 2.inp 3.inp 4.inp 5.inp 6.inp ave_normal_force max_normal_force
0.log 1.log 2.log 3.log 4.log 5.log 6.log decision      movie.xyz
[ycllee@hpc1-2 example]$ python /lustre/lwork/ycllee/python_practice/cneb/get_scf.py
1
[-849.545286889, -849.427892672, -849.432653914, -849.453021619, -849.483551388, -849.522283025, -849.546927191]
```

Functions of the scripts not mentioned above:

first_ok.py: create a "decision" file which records the number of iteration

find_first.py: locate the initial state inp file

find_last.py: locate the final state inp file

rm_small_dis_new.py: if the atoms are too close, which may cause Gaussian failure, this script will enlarge the distance between atoms

new_decision_ok.py: after each CNEB iteration, copy log files to the directory named by the number of iteration. (ex: after the first Gaussian calculations are done, a new directory "0" will be created, and the log files will be copy to this directory) This script also update the number of iteration in the "decision" file

print_max_normal_force.py: record which atom has the largest perpendicular (perpendicular to the reaction path) force and the magnitude of this force after each iteration.

print_ave_normal_force.py: record the magnitude of average perpendicular force acting on atoms after each iteration.

new_total_ok_update_allsp_allc.py: main script of the CNEB method. This script adjusts the coordinates of each image according to the perpendicular force and distance between images. New (updated) inp files will be created and replace the old ones.

step_limit.py: if the number of iteration reaches nsw, this script will terminate CNEB

get_dis.py: get the distances between neighboring images

get_input.py: read the data in the "input" file

makemovie.py: make a movie of the log files (inp_makemovie.py make a movie of the inp files)