Installing EON Reproducibly

– By Rohit and Amrita Goswami

for the 2019 RARE Summer Workshop at IISc Bangalore

Python Dependency Management

Problem Statement

The problem is that most Linux distributions include a lot of pre-packaged `python` packages and do not play nice with a lot of older programs. In any case, it is considered to be a bad idea to use the system package manager for these things. Indeed it seems like the necessary `python` version for EON is 2.x which needs special consideration for some newer distros.

Solution

Full details and some more stuff is in the poetry section of this file. The code is simply:

```
# Assumes you have direnv, pyenv and virtualenvwrapper
export verPy="2.7.16"
export cuteName="rareEventsBang"
pyenv install $verPy
source /usr/bin/virtualenvwrapper_lazy.sh
mkvirtualenv -p $HOME/.pyenv/versions/$verPy/python $cuteName
echo "layout virtualenvwrapper $cuteName" >> .envrc
direnv allow
```

Essentially:

- GOTO `$HOME/Git/Gitlab/rareEventsBang`
- A local `python` virtualenv is activated

Further Installation

For anything under the folder, simply use:

```
# After poetry init
poetry add $pkgname
```

This grabs dependencies as well.

Git Stuff

It is actually a good idea to commit the `poetry` files and the `envrc`, however in that case you must remember the full set of instructions if you need them. Do not initialize a repository in the main folder though, since there are some other project folders there. It's really a sort of meta holding area to define `python` in it and all lower folders (can be over-ridden by another `poetry` + `direnv` combo)

EON

This needs `svn`, which is sort of like `git`. In any case, since they simply want us to add on to the `$PATH`, we can use our `direnv` setup.

```
# Only to be run once
cd $HOME/Git/Gitlab/rareEventsBang
export eonPath="$HOME/Git/Gitlab/rareEventsBang/eon"
echo 'export PYTHONPATH="$HOME/Git/Gitlab/rareEventsBang/eon/eon:$PYTHONPATH"' >> .envrc
echo "# Use a helper function\nPATH_add eon/bin" >> .envrc
direnv allow
```

In any case, having now set our paths without bothering our global setup, we can now move on to the installation, which needs `fortran` as well.

```
cd $HOME/Git/Gitlab/rareEventsBang
export eonPath="$HOME/Git/Gitlab/rareEventsBang/eon"
cd $eonPath
python setup.py install
cd eonclient
make
cp eonclient ../bin
```

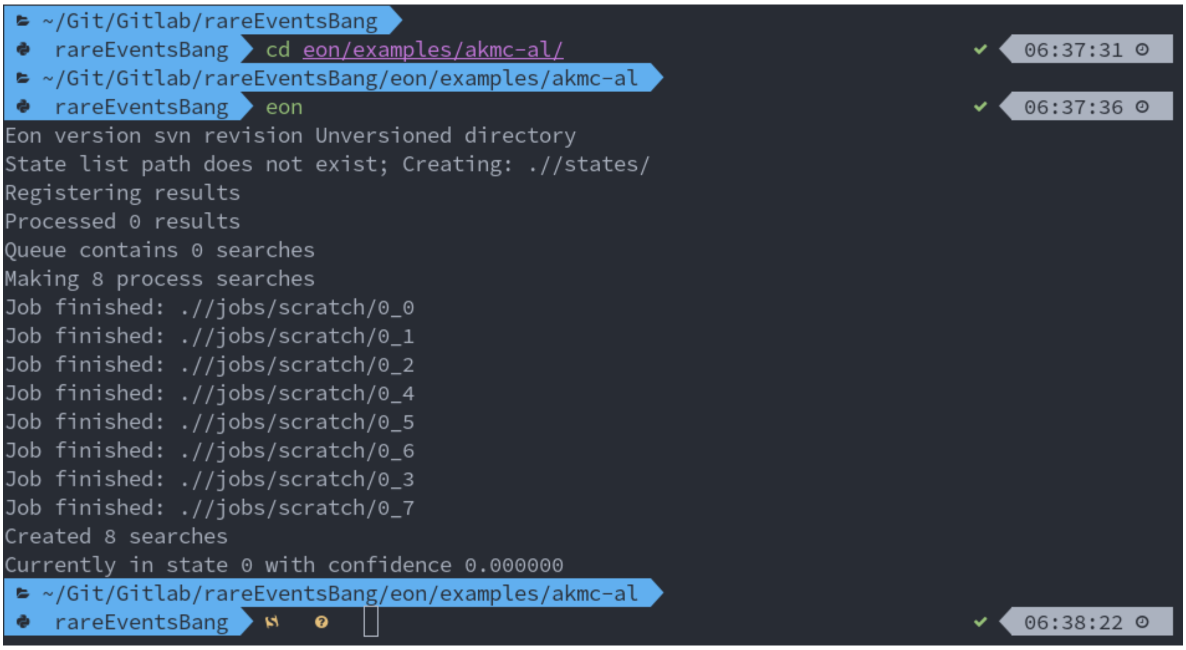
Now we can start to work on more interesting problems, after we run the tests of course.

```
export eonPath="$HOME/Git/Gitlab/rareEventsBang/eon"
cd $eonPath
cd examples/akmc-al
eon
```

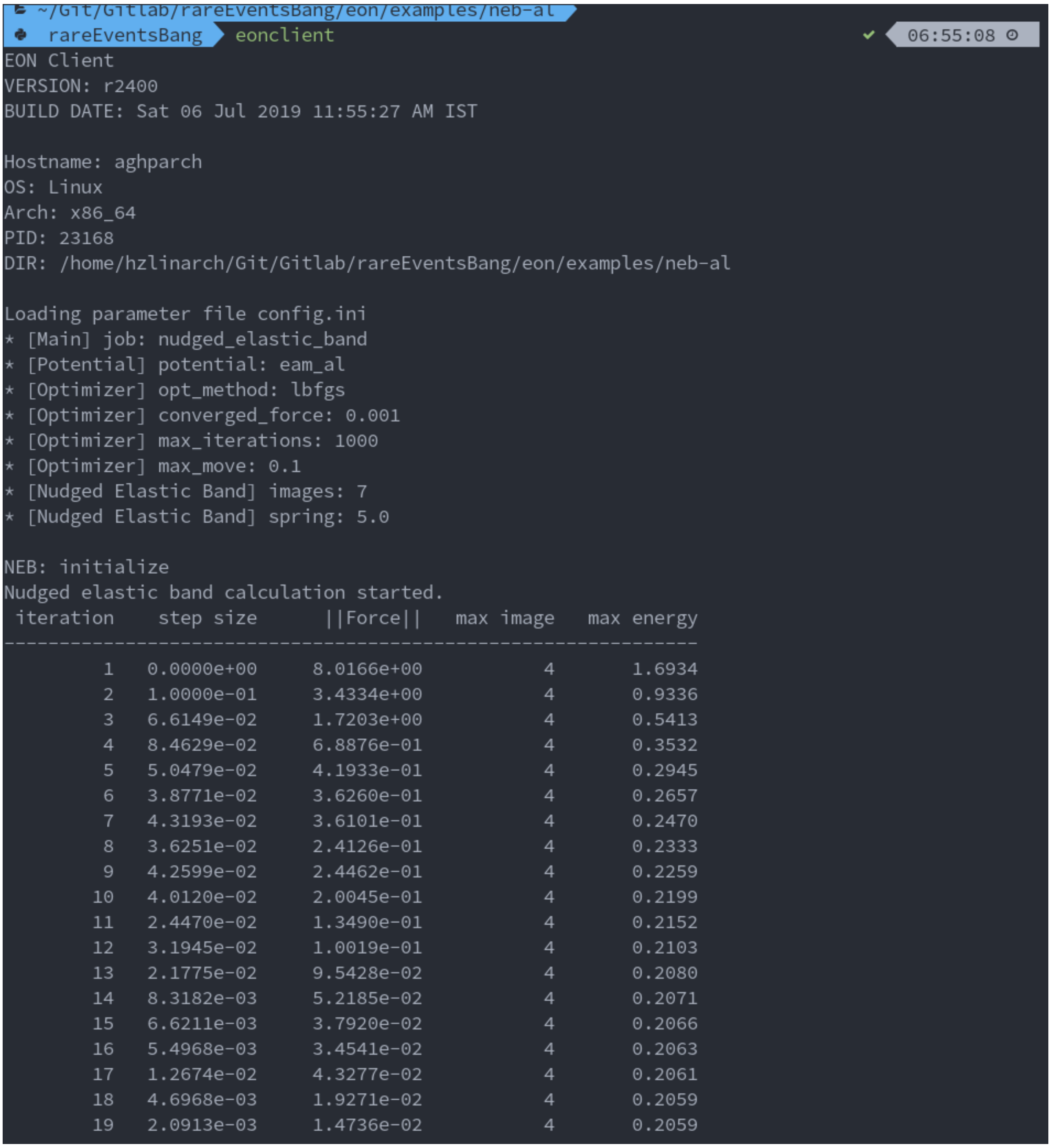
This throws an error because `yaml` support is missing, so we can get that rather easily.

```
export rareBangPath="$HOME/Git/Gitlab/rareEventsBang"
poetry add pyyaml
```

Now we can go back and run the tests.



```
~/Git/Gitlab/rareEventsBang
 rareEventsBang  cd eon/examples/akmc-al/                        ✓  06:37:31
~/Git/Gitlab/rareEventsBang/eon/examples/akmc-al
 rareEventsBang  eon                                             ✓  06:37:36
Eon version svn revision Unversioned directory
State list path does not exist; Creating: .//states/
Registering results
Processed 0 results
Queue contains 0 searches
Making 8 process searches
Job finished: .//jobs/scratch/0_0
Job finished: .//jobs/scratch/0_1
Job finished: .//jobs/scratch/0_2
Job finished: .//jobs/scratch/0_4
Job finished: .//jobs/scratch/0_5
Job finished: .//jobs/scratch/0_6
Job finished: .//jobs/scratch/0_3
Job finished: .//jobs/scratch/0_7
Created 8 searches
Currently in state 0 with confidence 0.000000
~/Git/Gitlab/rareEventsBang/eon/examples/akmc-al
 rareEventsBang                                                  ✓  06:38:22
```



```
~/Git/Gitlab/rareEventsBang/eon/examples/neb-al
 rareEventsBang  eonclient                                       ✓  06:55:08
EON Client
VERSION: r2400
BUILD DATE: Sat 06 Jul 2019 11:55:27 AM IST

Hostname: aghparch
OS: Linux
Arch: x86_64
PID: 23168
DIR: /home/hzlinarch/Git/Gitlab/rareEventsBang/eon/examples/neb-al

Loading parameter file config.ini
* [Main] job: nudged_elastic_band
* [Potential] potential: eam_al
* [Optimizer] opt_method: lbfgs
* [Optimizer] converged_force: 0.001
* [Optimizer] max_iterations: 1000
* [Optimizer] max_move: 0.1
* [Nudged Elastic Band] images: 7
* [Nudged Elastic Band] spring: 5.0

NEB: initialize
Nudged elastic band calculation started.
 iteration    step size      ||Force||    max image   max energy
----------------------------------------------------------------
         1  0.0000e+00     8.0166e+00            4       1.6934
         2  1.0000e-01     3.4334e+00            4       0.9336
         3  6.6149e-02     1.7203e+00            4       0.5413
         4  8.4629e-02     6.8876e-01            4       0.3532
         5  5.0479e-02     4.1933e-01            4       0.2945
         6  3.8771e-02     3.6260e-01            4       0.2657
         7  4.3193e-02     3.6101e-01            4       0.2470
         8  3.6251e-02     2.4126e-01            4       0.2333
         9  4.2599e-02     2.4462e-01            4       0.2259
        10  4.0120e-02     2.0045e-01            4       0.2199
        11  2.4470e-02     1.3490e-01            4       0.2152
        12  3.1945e-02     1.0019e-01            4       0.2103
        13  2.1775e-02     9.5428e-02            4       0.2080
        14  8.3182e-03     5.2185e-02            4       0.2071
        15  6.6211e-03     3.7920e-02            4       0.2066
        16  5.4968e-03     3.4541e-02            4       0.2063
        17  1.2674e-02     4.3277e-02            4       0.2061
        18  4.6968e-03     1.9271e-02            4       0.2059
        19  2.0913e-03     1.4736e-02            4       0.2059
```

Visualization

There exists an old `atomview` visualization script, but that depends on `pygtk` which has been depreciated for a long time now. The file types used are best visualized by ASE (Atomic Simulation Environment) which is a dependency. However, the GUI requires some slight additional packages.

- If `tk` has been installed prior to building `python` with `pyenv`, then there is no further work required.
- If `tk` has been installed *after* the `python` environment has been initialized, simply reinstall it with `pyenv`

Dependencies

Poetry

For posterity, the `pyproject.toml` file is reproduced below for use with `poetry`:

```toml
[tool.poetry]
name = "rarebang"
version = "0.0.1"
description = "A meta package with dependencies for the Rare Events Workshop at Bangaloree"
authors = ["HaoZeke"]
license = "MIT"

[tool.poetry.dependencies]
python = "^2.7"
numpy = "^1.16"
ase = "^3.17"
pyyaml = "^5.1"
PyGObject = "^3.32"
pmw = "^2.0"

[tool.poetry.dev-dependencies]

[build-system]
requires = ["poetry>=0.12"]
build-backend = "poetry.masonry.api"

```

Direnv

Similarly, a sample `.envrc` file is:

```bash
layout virtualenvwrapper rareEventsBang
export PYTHONPATH="$HOME/Git/Gitlab/rareEventsBang/eon/eon:$PYTHONPATH"
PATH_add eon/bin
```