Using Python at CHPC

Brett Milash

Center for High Performance Computing

10/28/2024

Overview

- Selecting a version of python
- Installing python
- Installing python packages
- Running python
  - scripts
  - Jupyter notebook
  - VSCode IDE

Which python?

```
$ python
bash: python: command not found...
$ /usr/bin/python3 --version
Python 3.6.8
$ /usr/bin/python2 --version
Python 2.7.18
```

- Python release cycle: https://python-release-cycle.glitch.me/
- Don’t use the system-provided pythons
  - They are really old!
  - You don’t control when it is updated
- Alternatives:
  - Use a CHPC-installed python module
  - Install your own

CHPC-installed python modules

```
$ module spider python
----------------------------------------------------------------------------
  python:
----------------------------------------------------------------------------
     Versions:
        python/2.7.18
        python/3.8.8
        python/3.9.15
        python/3.10.3
        python/3.11.3
        python/3.11.7
        python/3.12.4
$ module load python/3.10.3
$ python --version
Python 3.10.3
$ which python
/uufs/chpc.utah.edu/sys/installdir/r8/python/3.10.3/bin/python
```

Install your own python

- User-installed python at CHPC:
  - https://www.chpc.utah.edu/documentation/software/python-anaconda.php
- Miniforge: https://github.com/conda-forge/miniforge/
  - Requires 454 MB, ~ 1% of a 50 GB disk space quota

```
$ mkdir -p software/pkg/
$ wget https://github.com/conda-forge/...
$ bash ./Miniforge3-Linux-x86_64.sh -h
$ bash ./Miniforge3-Linux-x86_64.sh -s -b -p $HOME/software/pkg/miniforge3
$ cd $HOME/software/pkg/miniforge3/bin
$ export PATH=$(pwd):$PATH
```

- You can also create your own python module

```
$ mkdir -p $HOME/MyModules/miniforge3
$ cp /uufs/chpc.utah.edu/sys/installdir/python/modules/miniforge3/latest.lua
      $HOME/MyModules/miniforge3
```

Installing packages

- You can install packages yourself, using your own Python or CHPC's
- You can install precompiled packages, or install from source
- You can install packages where you want
- You can install using "pip" or from source code

Installing packages with pip

- https://docs.python.org/3/installing/index.html
- Installs packages from the Python Package: https://pypi.org/
- Pros:
  - “pip is the preferred installer program”
  - installs / upgrades prerequisites
  - can uninstall too
  - --dry-run option (added with python version 3.10)
  - --user (default if main library location is un-writable)
- Cons:
  - have been addressed in recent releases - older versions can be dangerous

Installing packages from source with setup.py

- Python packages include a script named “setup.py”

  `python setup.py install --help`

- Useful if package isn’t found on pypi.org
- Allows you to examine prerequisites in requirements.txt

Where do the packages go?

- By default:

  <main_python_location>/lib/python<version>/site-packages
- User directory:

  $HOME/.local/lib/python<version>/site-packages
- If using --prefix=

  <prefix_location>/lib/python<version>/site-packages
  - Need to add this location to PYTHONPATH

Methods to access resources at CHPC

Method	Attributes	Resource
ssh to interactive node	command line or GUI	interactive node
Fastx to interactive node	command line or GUI, persistence	interactive node
OnDemand cluster shell access	convenient, like ssh, command line only	interactive node
SLURM sbatch command	non-interactive (batch mode)	compute node(s)
SLURM salloc command	interactive command-line or GUI	compute node(s)
OnDemand applications (e.g. Jupyter Lab)	web-based access	compute node(s)

Remember the appropriate uses for interactive and compute nodes:

- Interactive nodes: writing code, installing code, small-scale testing, debugging, managing SLURM jobs
- Compute nodes: heavy-duty computing (simulations, stats, data visualization) whether interactive or not

Methods of using Python at CHPC

Command	Attributes
python, python scriptname.py …	Interactive, command line style
python scriptname.py …	Batch jobs, script
Jupyter Lab or Notebook	Interactive, document based, web browser GUI
vscode, pycharm	Integrated Development Environment, GUI
Workflows, e.g. snakemake, nextflow	Non-interactive workflow tools

Python use methods vs. CHPC access methods

	python, interactive	python, script	Jupyter	VSCode, Pycharm	Workflows
ssh (to interactive node)					
Fastx (to interactive node)					
OnDemand cluster shell access (runs on interactive node)					
SLURM sbatch (compute node)					
SLURM salloc (compute node)					
OnDemand system installed applications (runs on compute nodes)					

ssh, Fastx, and OnDemand shell access demo

- These methods provide access to interactive nodes
- Graphics (whether GUI or graphical output) requires X-forwarding
  - On Mac use “ssh -Y *username@hostname”*
  - On Windows use Xming (https://xming.en.softonic.com/)
- X-forwarding can be slow without some help

Python use methods vs. CHPC access methods

	python, interactive	python, script	Jupyter	VSCode, Pycharm	Workflows
ssh (to interactive node)	✅ good for testing	✅ good for testing	Inefficient - not recommended	VSCode ❌ PyCharm ✅	✅ good for testing
Fastx (to interactive node)	✅ good for testing	✅ good for testing	Inefficient - not recommended	VSCode ❌ PyCharm ✅	✅ good for testing
OnDemand cluster shell access (on interactive node)	✅ good for testing, but no graphics	✅ good for testing, but no graphics	❌ - requires X windows	❌ - requires X windows	✅ good for testing, but no graphics
SLURM sbatch (compute node)					
SLURM salloc (compute node)					
OnDemand system installed applications (on compute nodes)					

"✅ good for testing" means software works well within computing limits of interactive node

SLURM sbatch and salloc demo

- Both methods provide access to compute nodes
- sbatch is batch oriented - therefore non-interactive
- salloc starts an interactive shell session on a compute node
- https://chpc.utah.edu/documentation/software/slurm.php

Python use methods vs. CHPC access methods

	python, interactive	python, script	Jupyter	VSCode, Pycharm	Workflows
ssh (to interactive node)	✅ good for testing	✅ good for testing	Inefficient - not recommended	VSCode ❌ PyCharm ✅	✅ good for testing
Fastx (to interactive node)	✅ good for testing	✅ good for testing	Inefficient - not recommended	VSCode ❌ PyCharm ✅	✅ good for testing
OnDemand cluster shell access (on interactive node)	✅ good for testing, but no graphics	✅ good for testing, but no graphics	❌ - requires X windows	❌ - requires X windows	✅ good for testing, but no graphics
SLURM sbatch (compute node)	❌	⭐⭐⭐⭐⭐	❌	❌	⭐⭐⭐⭐⭐
SLURM salloc (compute node)	✅	✅	Inefficient - not recommended	VSCode ✅ PyCharm ✅	✅
OnDemand system installed applications (on compute nodes)					

"✅ good for testing" means software works well within computing limits of interactive node

OnDemand demo

- Web portal
- Access to compute nodes
- Very good for web and GUI applications

Python use methods vs. CHPC access methods

	python, interactive	python, script	Jupyter	VSCode, Pycharm	Workflows
ssh (to interactive node)	✅ good for testing	✅ good for testing	Inefficient - not recommended	VSCode ❌ PyCharm ✅	✅ good for testing
Fastx (to interactive node)	✅ good for testing	✅ good for testing	Inefficient - not recommended	VSCode ❌ PyCharm ✅	✅ good for testing
OnDemand cluster shell access (on interactive node)	✅ good for testing, but no graphics	✅ good for testing, but no graphics	❌ - requires X windows	❌ - requires X windows	✅ good for testing, but no graphics
SLURM sbatch (compute node)	❌	⭐⭐⭐⭐⭐	❌	❌	⭐⭐⭐⭐⭐
SLURM salloc (compute node)	✅	✅	Inefficient - not recommended	VSCode ✅ PyCharm ✅	✅
OnDemand system installed applications (on compute nodes)	❌	✅	⭐⭐⭐⭐⭐	VSCode⭐⭐⭐ PyCharm ✅	❌

"✅ good for testing" means software works well within computing limits of interactive node