Python package and environment management on HPC

Jielin Yu
HPC User Services
LSU HPC  LONI
sys-help@loni.org

Louisiana State University
Baton Rouge
March 30, 2022
Outline

- Things to be covered in the training
  - Python package management tools
  - Python versions on HPC
  - How to use pip on HPC
  - How to use conda on HPC

- Please help us finish the survey before you leave: Survey
Python package management tools

- **Python package management tools**
  - Allow us to easily manage the dependencies for our project that are not part of the Python standard library.
  - A dependency is code that is required for your program to function properly. These often come in the form of packages.

- **Why we need Python package management tools**
  - Packages can also have their own dependencies. Managing all these dependencies can be hard because packages may require specific versions of their dependencies.
  - It’s easy to break something by modifying dependencies manually.
Python package management tools

- List of Python package management tools
  - pip
  - conda
  - pdm
  - pyenv
  - setuptools
  - venv
  - virtualenv
  - ...

Python package and environment management on HPC Spring 2022
Python versions on HPC

- Python versions on HPC
  - Python 2 & 3 are available on all of our clusters
  - Use “module av python” command
  - Python 3 will be used for this session

- Conda and pip are installed with most of the Python versions on HPC, so you will need to load one Python module before use conda and pip
- Commands might work differently on your local computer
Some useful commands for pip

- pip is the package installer for Python. You can use it to install packages from the Python Package Index and other indexes.

- Run “unset PYTHONPATH” and “unset PYTHONHOME” commands

- Check packages installed
  - pip list
  - pip show package_name

- Install a single package
  - pip install --user package_name==version
    - Packages will be installed in:
      /home/username/.local/lib/python_version/site-packages/

  - Add executables in bin to your PATH environment variable:
    echo 'export PATH=/home/username/.local/bin/:$PATH' >> ~/.bashrc
    source ~/.bashrc
Some useful commands for pip

- **Install multiple packages**
  
  ```
  pip install --user package_name1 package_name2 ...
  ```

- **Upgrade packages**
  
  ```
  pip install --upgrade --user package_name1 package_name2 ...
  ```

- **Uninstall packages**
  
  ```
  pip uninstall package_name1 package_name2 ...
  ```
Some useful commands for pip

- **Install a package to a specific location**
  
  ```bash
  pip install --prefix=/path/to/folder package_name==version
  ```

- Add packages and executables in bin to your PATH environment variable:
  
  ```bash
  echo 'export PATH=/path/to/folder/bin/:$PATH' >> ~/.bashrc
  echo 'export PYTHONPATH=/path/to/folder/lib/python_version/site-packages/:$PYTHONPATH' >> ~/.bashrc
  source ~/.bashrc
  ```
Some useful links for pip

- pip documentation
- Python Package Index
  https://pypi.org/
Creating a conda environment

- Conda is an open-source package management and environment management system for multiple programming languages.

- With conda, you can set up a totally separate environment to run different versions of Python, while continuing to run your usual version of Python in your normal environment.

- Instruction on creating conda environment on HPC website: http://www.hpc.lsu.edu/docs/faq/installation-details.php#TensorFlow
Creating a conda environment

- **Before creating a conda environment on HPC:**
  - Run “unset PYTHONPATH” and “unset PYTHONHOME” commands
  - Default conda envs and pkgs directory:
    - `/home/your_username/.conda/pkgs`
    - `/home/your_username/.conda/envs`
  - Add/change lines below in your ~/.condarc file:
    ```
    envs_dirs:
    - /work/your_username/test-env/envs
    pkgs_dirs:
    - /work/your_username/test-env/pkgs
    ```
  - If you do not have a ~/.condarc file, use the command below to create one:
    ```
    touch ~/.condarc
    ```
  - Check information about current conda install:
    ```
    conda info
    ```
Creating a conda environment

- **Create a conda environment**
  
  ```
  conda create -n env_name python=version
  ```

- **Activating/deactivating a conda environment**
  
  ```
  source activate env_name
  source deactivate
  ```

- **Checking current and available conda environment**
  
  ```
  conda info --envs
  ```

- **Check installed packages**
  
  ```
  conda list
  ```

- **Installing, upgrade and uninstall packages inside a conda environment**
  
  ```
  conda install -c channel_name package1=version package2=version
  conda upgrade/update package1 package2
  conda uninstall/remove package1 package2
  ```

- **Remove a conda environment**
  
  ```
  conda env remove -n env_name
  ```

- **Combining pip with conda**
Some useful links for conda

- Conda documentation
  https://docs.conda.io/en/latest/
- Anaconda
  https://anaconda.org/
Exercise

- **Install numpy 1.20.1 in /work/your_username/test by using pip**
  - Check if numpy 1.20.1 is successfully installed with “pip show numpy”
  - Check if the path to f2py is your /work directory with “which f2py”

- **Create a conda environment with python 3.9.5**
  - Activate the conda environment
  - Check the python version with “python --version”
  - Search the available versions of numpy and install one version
Next Week Training

- **Weekly trainings during regular semester**
  - Wednesdays “9:00am-11:00am” session

- **Workshop**
  - End of May

- **Keep an eye on our webpage:** www.hpc.lsu.edu
HPC@LSU User Services

- **Contact user services**
  - Email Help Ticket: sys-help@loni.org
  - Telephone Help Desk: +1 (225) 578-0900