Installing and Running Python Packages on ITaP Community Clusters

Amiya K Maji

June 5, 2020

Sr. Computational Scientist
ITaP Research Computing
Purdue University
Contents

- Fun with Python
- System Python vs. Anaconda module
- Challenges for installing Python packages
- Conda environments
- rcac-conda-env
- Install cartopy
- Installing packages in a group shared directory
- Miniconda
- Troubleshooting
- Questions
Fun with Python

Source: https://xkcd.com/1987/
System Python vs. Anaconda module

- Default Python is located at `/bin/python` (v2.7.5)
  - No scientific packages are pre-installed
- Always load the anaconda module
  - `module load anaconda`
- Stick to a Python/Aprotoconda version
  - `module load anaconda/5.1.0-py36`
- Challenges for installing additional packages
  - Insufficient permission
  - Mismatched dependencies for various packages
  - Refer Slide 3 for a clear picture.
Installing Python packages

- Use **conda environments** (for additional packages)
- Use **conda install** (in your personal environment)
- Keep it simple and self-contained
- It is important to keep track of what you installed with conda/pip
- Run **conda list** to see what packages are available
- **Shortcomings of conda environments**
  - Using installed packages is painful
  - `source activate mypackage` does not work in tcsh
  - `conda activate` needs `conda init` which inserts BAD CODE in your `~/.bashrc`
  - **NEVER RUN** `conda init`
  - `conda activate` followed by `conda deactivate` can destroy your environment.
rcac-conda-env: A wrapper for using conda env

- Workflow for installing Python packages
  1. Create an environment and its corresponding module file
  2. Load the modules
    - Alternative for `conda activate`
  3. Install packages with conda or pip
  4. Import installed packages
    - Make sure you have the modules loaded
rcac-conda-env: Features

- **Run** `rcac-conda-env --help`
- **create**
  - Create a minimal anaconda environment
  - Python must match with base Python
- **delete**
  - Delete an existing environment
- **module**
  - Create/update module file for an existing environment
- **kernel**
  - Create Jupyter kernel for an existing environment
  - The environment must have ipython and ipykernel installed
- **Let’s install some packages!**
Install *cartopy* using Conda

- `rcac-conda-env create -n cartopy`
- Answer the prompts
- Note down the instructions for loading the *cartopy* environment
- Load necessary modules
- `conda install cartopy`
- `which python`
- `conda list`
- Let’s try to load *cartopy*
- What went wrong?
  - `!@#$%^&*`
Test cartopy

- Use the Python inside the environment
  - rcac-conda-env module -n cartopy --local-python
- Now try to import cartopy
  - Success!!!
- Run some more examples
Install packages with pip

- List which modules are loaded
- `pip install pipdeptree`
- `pipdeptree`
- `pipdeptree --graph-output png`
- `pip install graphviz`
- `pipdeptree --graph-output png > dep.png`
- `display dep.png`
Install **cartopy** for your research group

- **Motivations**
  - Share a single lab-wide installation
  - Installations in $HOME consume space

- `rcac-conda-env create -p /depot/mylab/apps/cartopy -m /depot/mylab/etc/modules --local-python`

- **Load the modules**
- `conda install cartopy`
- `conda list`
- `which python`
- **Run example codes**
Working with Miniconda

- Download miniconda from

- Install

- Set `PATH` to miniconda installation

- Advantages
  - No need to use the anaconda module
  - Totally isolated
  - You can install any Python version that you want

- Disadvantage
  - You must manage your own installation
Python IDEs

- Spyder
- Pycharm
- Jupyter
Caveats

- Do not install packages with `pip install --user`
- Do not mix channels, create separate environments instead
- Watch out for dependencies across packages
- Watch for disk usage in your home directory
  - `myquota`
- **Do not load Python/Anaconda in** `~/.bashrc`
- **Do not use** `conda init`
Troubleshooting

● Always be mindful of your runtime environment
  ○ module list
  ○ echo $PYTHONPATH
  ○ echo $PATH
  ○ echo $LD_LIBRARY_PATH

● When in doubt, clean up directories where Python installs packages
  mv ~/.conda ~/.conda.bak
  mv ~/.local ~/.local.bak
  mv ~/.cache ~/.cache.bak

● Other configuration locations
  ~/.jupyter ~/.ipython ~/.config

● Read the user guide
  ○ https://www.rcac.purdue.edu/knowledge/scholar/run/examples/apps/python/packages
Questions