DEMYSTIFYING PYTHON PACKAGE INSTALLATION WITH `CONDA-ENV-MOD`

AMIYA MAJI, LEV GORENSTEIN, GEOFFREY LENTNER

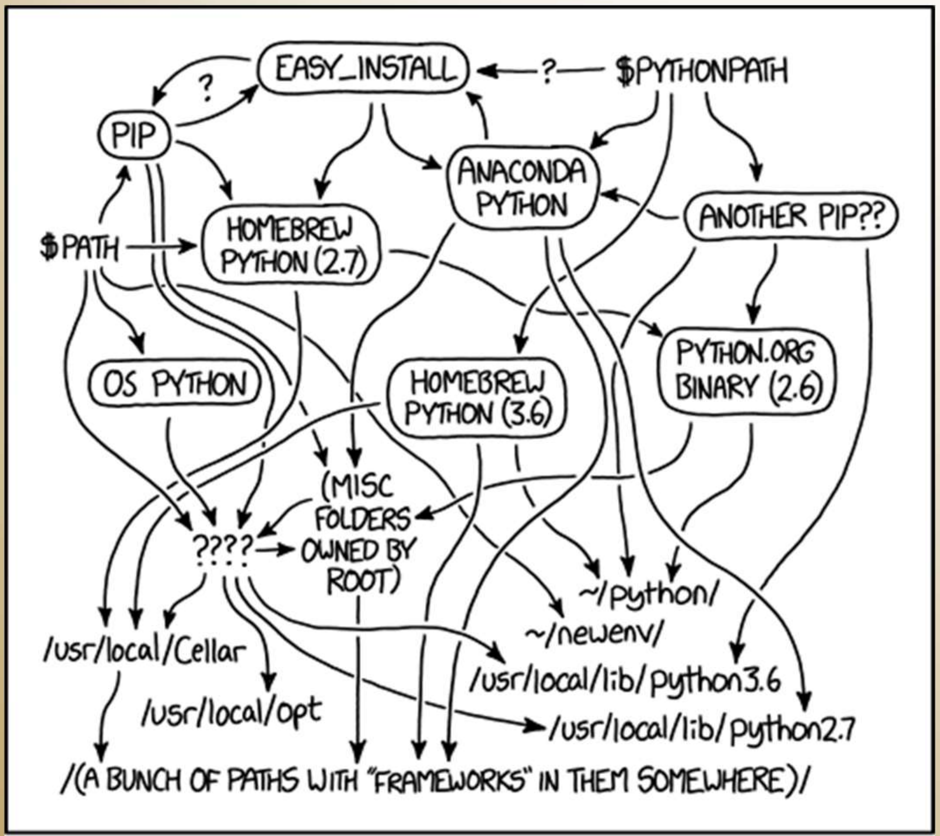
Seventh Annual Workshop on HPC User Support Tools (HUST'20)

ITaP Research Computing
Purdue University

CONTENTS

- Fun with Python
- Challenges for installing Python packages
- Contributions
- `conda-env-mod`
- Success stories
- Evaluation
- Demo
- Questions

FUN WITH PYTHON



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

Source: https://xkcd.com/1987/

MOTIVATION

- Growing use of Python in HPC
  - Deep Learning, Bioinformatics, Data Science, …
- Challenges for installing Python packages on HPC clusters
  - Insufficient permission
  - Mismatched dependencies for various packages
  - Refer to Slide 3 for a ~~clear~~ picture.
  - Poor documentation

TYPICAL APPROACHES

- Install in user mode (pip install --user)
  - May not work when shared across clusters
  - May break Jupyter kernels
- Use VirtualEnv
- Use Conda environments
- Shortcomings
  - `source activate myenv` does not work in tcsh
  - `conda init` inserts code in `~/.bashrc`
  - `conda deactivate` can corrupt your environment
  - Does not allow activating multiple environments (VirtualEnv, Conda)

MORE CHALLENGES

- How can I share my Python environment with a colleague
- How can I use my environment in JupyterHub
- How can I upgrade a single package without breaking the environment

- Managing and using virtual environments become too daunting for novice users

CONTRIBUTIONS

- Simplify and streamline installation of Python packages
  - **Simplify** management of virtual environments
  - **Non-interference** with existing packages
  - **Flexible** activation
- Empower interactive Python users

CONDA-ENV-MOD: INSTALL PYTHON PACKAGES

create myenv

- conda create
- create module
- create kernel

module load

- pip install
- conda install

CONTRIBUTIONS

- Jupyter Kernels
  - conda-env-mod installs required packages
  - Generates a Kernel specification file
- Standalone module generation
- Standalone kernel generation

OTHER CONSIDERATIONS

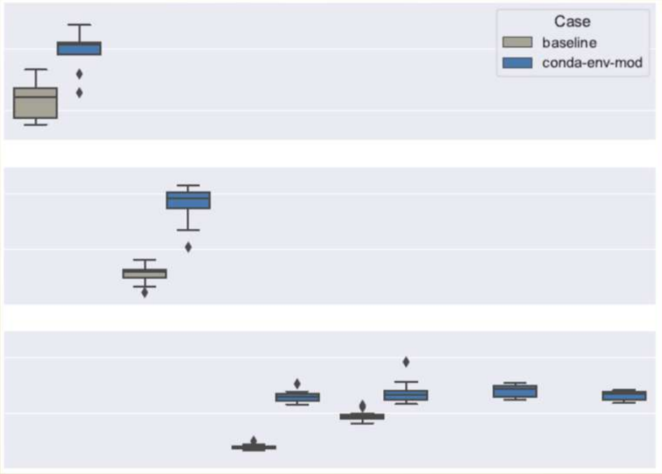
- Module file location
- Python version
  - Matched with the base Anaconda
- Which Python to use
  - Base vs. environment

SUCCESS STORIES

- Deep Learning package installation
  - 9 applications (multiple versions)
  - 2 Python versions
  - CPU and GPU version
- Shared Python environment for teaching
  - Instructor uses conda-env-mod to install packages
  - Students load modules and Jupyter Kernels
  - 12+ course, 1500+ students (since Fall 2019)
  - Data Science, Atmospheric Science, Molecular Chemistry, Library Science
- Shared Python environment for research groups
  - New feature requests

EVALUATIONS

- Overhead for `conda-env-mod`
  - One time cost
  - More details in paper
- Usage (June-Aug 2020)
  - 40379 module loads
  - 178 unique modules (environments)
  - **161 unique users**
- Deep Learning modules usage (June-Aug 2020)
  - 34109 module loads
  - 54 unique modules
  - **64 unique users**



CONCLUSION

- `conda-env-mod` simplifies Python package installation and use
  - Offload *management of virtual environments*
- Help interactive Python use
  - JupyterHub
  - Open OnDemand
- Download
  - `https://github.com/amaji/conda-env-mod`

QUESTIONS

INSTALL `CARTOPY` FOR YOUR RESEARCH GROUP

- Motivations
  - Share a single lab-wide installation
  - Installations in `$HOME` consume space
- `conda-env-mod 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