Installing Software Packages in Virtual Environments on Sapelo2

Georgia Advanced Computing Resource Center (GACRC)

UNIVERSITY OF GEORGIA
Georgia Advanced Computing Resource Center

Agenda

- **Virtual environment basics**
  - Benefits of using virtual environments for software management
  - Virtual environment architecture
  - Conda versus Python environments
  - Necessary software modules

- **Practice creating virtual environments**
  - Setting up environment
  - Activating/deactivating + using environment
  - How to install software packages
  - Best practices
  - Common Issues
  - Resources

Virtual Environment Basics

What is a virtual environment?
A virtual environment (venv) is a self contained space that allows you to keep packages installed there isolated from the global software environment, allowing management of various software packages within the virtual environment

Inherent benefits of virtual environments include:

- Prevents package conflict between projects
- Allows version control/management of packages
- Packages in a virtual environment do not impact system outside
- Isolates specific dependencies for a project, allowing for reproducibility

Virtual Environment Basics - Architecture

What is /home/MyID/env-name/ ?

- This is the absolute path to your virtual environment. Typically, your virtual environments would be kept in your /home/MyID directory.
- The ‘env-name’ is a placeholder for whatever name you give your future environments.

Where do executables and packages go after they are installed in a venv?

- Executables (binaries) are installed in /home/MyID/env-name/bin
- Python packages can be found in /home/MyID/env-name/lib/python3.x/site-packages/
- R packages can be found in /home/MyID/env-name/lib/R/library/
- Perl packages can be found in /home/MyID/env-name/lib/perl5/site_perl/5.xx.x/
- All other libraries/packages can be found under the lib/ directory in your virtual environment

Virtual Environment Basics – Conda vs Python

Conda virtual environments:

- Created from Miniforge3 software module
- Can install any package available in a conda package repository
  - This includes a number of Python, R, and other various software packages
  - Some conda software packages are available from specific “channels”

Python virtual environments:

- Created from a Python software module
- Can install Python packages
  - Will default to installing packages from the Python Package Index (PyPI)

A Note on Conda Environments

Conda environments on the Sapelo2 cluster formerly made use of the Miniconda3 software module, however, new Anaconda licensing policies have compelled us to consider other options for maintaining Conda environments. Our current solution is to use Miniforge3 software modules for all Conda environments moving forward.

Interactive Job

- Before creating or using your virtual environments, please start an interactive job first. You can start an interactive job from a login/submit node with the command interact
    - **Please DO NOT create or use virtual environments from a login/submit node (ss-sub#)**

Creating an environment - Conda

Step 1: Set up environment

Load Miniforge3 module: `ml Miniforge3/24.11.3-0`

Create the virtual environment: `conda create -p /home/MyID/env-name`

Step 2: Activate the environment: `source activate /home/MyID/env-name`

Step 3: Install software packages into environment: `conda install numpy`

Step 4: Deactivate the environment: `conda deactivate`

- See currently installed packages: `conda list`
- To install from a specific channel (ex. install python-dateutil from conda-forge channel): `conda install -c conda-forge python-dateutil`
- To install a specific version: `conda install scipy=1.14.1`
- Can also pip install in a conda environment: `pip install astropy`

Note: Python version will be latest unless you specify a Python version in creation command: `conda create -p /home/MyID/env-name python=3.6`

Creating an environment - Python

Step 1: Set up environment

Load Python module: ml Python/3.10.4-GCCcore-11.3.0

Create the virtual environment: python -m venv /home/MyID/env-name

Step 2: Activate the environment: . /home/MyID/env-name/bin/activate

Step 3: Install software packages into environment: pip install pandas

Step 4: Deactivate the environment: deactivate

- See currently installed packages: pip list
- To specify a version for a package (single = for conda, double == for pip):pip install scipy==1.14.1
- To install multiple packages at once: pip install scipy astropy

Note: Python version will be the same as the Python module used to create the environment

Best Practices

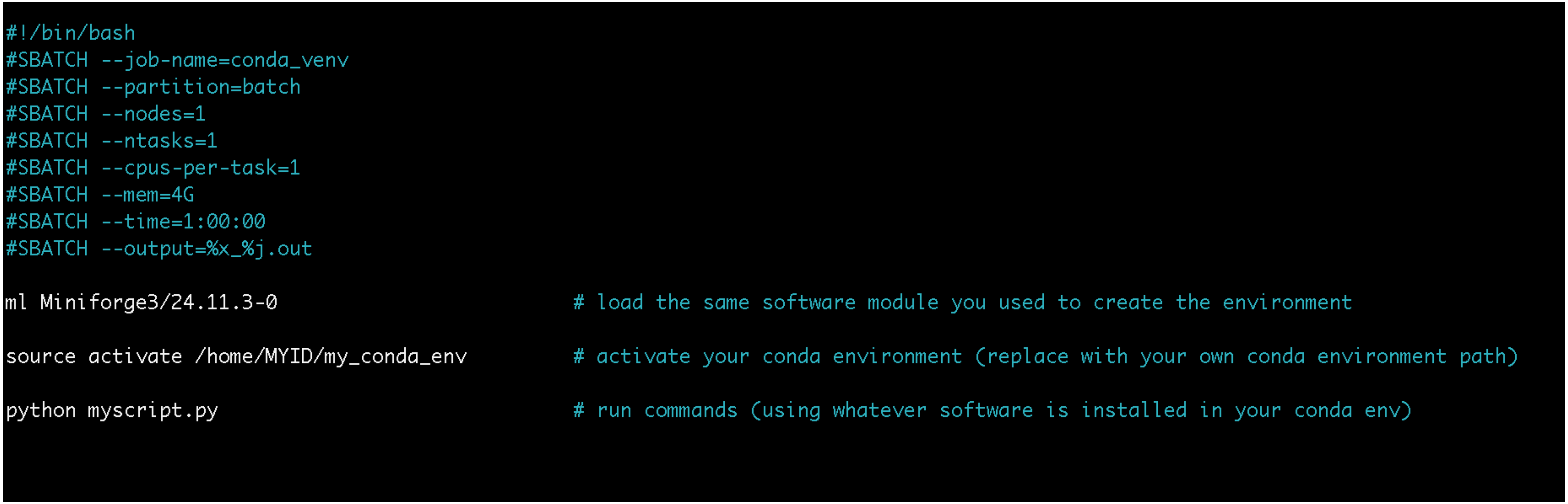
- Create virtual environments in your home directory (/home/MyID)
- Start an interactive job before creating a virtual environment
- Use the full module name of Miniforge3 or Python when loading (eg. Miniforge3/24.11.3-0)
- Name your virtual environments something descriptive so you can easily identify its purpose or the project it goes to
- Create different environments for different projects in order to avoid conflicts
- When you no longer have use for a virtual environment, please delete it
    - (rm -rf /home/MyID/env-name)
- Anytime you want to use the packages in your virtual environment, simply activate it
    - When you want to activate a virtual environment, be sure to load the same module you used to create the environment
    - Be sure to deactivate it when you are done.

Common Issues and Resources

- For “command not found” errors
  - Check that your environment is activated (and *only* that environment)
  - Check that you have the same module loaded you used to create the environment when you go to activate it (and *only* that module unless you know the modules you have loaded aren’t causing the problem)
- For conflict errors when installing packages into an environment
  - Try not specifying any version when you install the package
  - Consider downgrading the Python version of your environment
  - Try installing all desired packages at once rather than one at a time,
    - E.g. ”conda/pip install package1 package2 package3”

- For information on installing Conda environments on Sapelo2
  - https://wiki.gacrc.uga.edu/wiki/Installing_Applications_on_Sapelo2#How_to_install_Conda_packages
- For information on installing Python environments on Sapelo2
  - https://wiki.gacrc.uga.edu/wiki/Installing_Applications_on_Sapelo2#How_to_install_Python_packages
- How to use a Conda environment within a Jupyter-Notebook
  - https://wiki.gacrc.uga.edu/wiki/Using_a_Conda_environment_in_Jupyter
- How to use a Python environment within a Jupyter-Notebook
  - https://wiki.gacrc.uga.edu/wiki/Using_a_Python_environment_in_Jupyter

Sample Submission Script – Conda Env



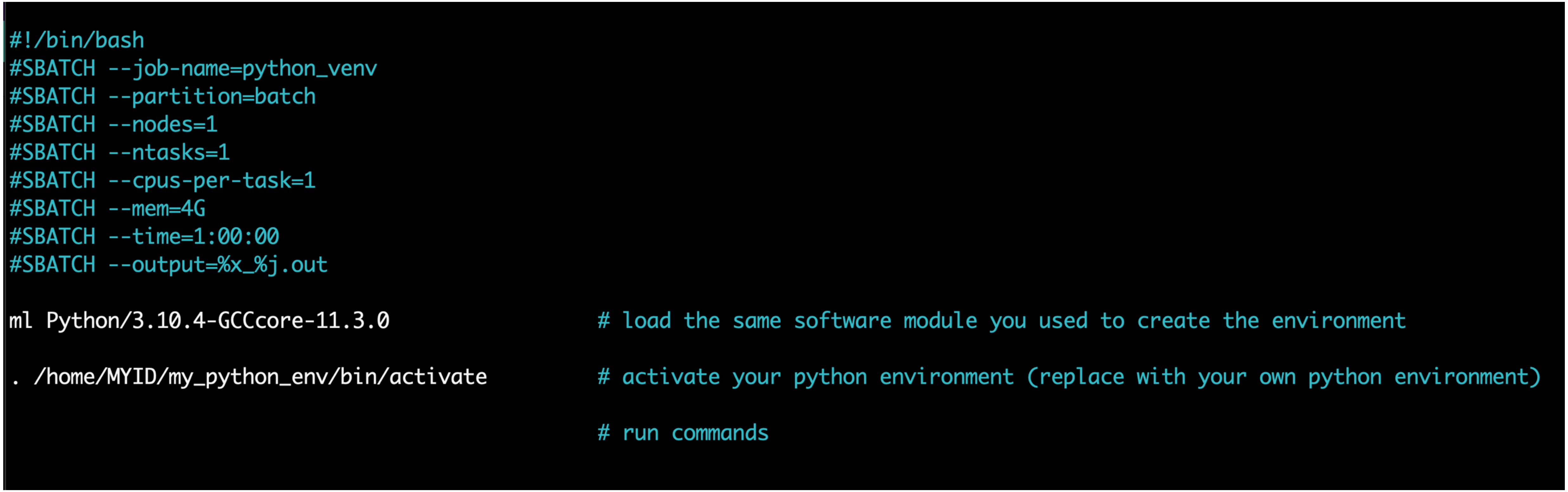
```
#!/bin/bash
#SBATCH --job-name=conda_venv
#SBATCH --partition=batch
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=1
#SBATCH --mem=4G
#SBATCH --time=1:00:00
#SBATCH --output=%x_%j.out

ml Miniforge3/24.11.3-0                         # load the same software module you used to create the environment

source activate /home/MYID/my_conda_env         # activate your conda environment (replace with your own conda environment path)

python myscript.py                              # run commands (using whatever software is installed in your conda env)
```

Sample Submission Script – Python Env



```
#!/bin/bash
#SBATCH --job-name=python_venv
#SBATCH --partition=batch
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=1
#SBATCH --mem=4G
#SBATCH --time=1:00:00
#SBATCH --output=%x_%j.out

ml Python/3.10.4-GCCcore-11.3.0                 # load the same software module you used to create the environment

. /home/MYID/my_python_env/bin/activate         # activate your python environment (replace with your own python environment)

                                                # run commands
```