



Installing software on Alpine with Conda and Mamba

Installing software on Alpine with Conda and Mamba

Instructor: Brandon Reyes

- Research Computing
- Website: www.rc.colorado.edu
- Documentation: <https://curc.readthedocs.io>
- Helpdesk: rc-help@colorado.edu
- Survey: <http://tinyurl.com/curc-survey18>



Meet the User Support Team



Layla
Freeborn



Brandon
Reyes



Andy
Monaghan



Michael
Schneider



John
Reiland



Dylan
Gottlieb



Mohal
Khandelwal



Ragan
Lee

Slides

https://github.com/ResearchComputing/alpine_conda_mamba_primer



Session Overview

- Installing software on CURC systems
- What are Conda and Mamba?
- Using Conda and Mamba on CURC systems
 - Creating environments
 - Installing packages
 - Useful commands
 - Batch jobs
- Useful strategies for complex environments

Building Software on CURC Systems

- There are numerous ways to install software on Alpine
 - Grab pre-compiled binaries
 - Compile from source
 - Using containers (Apptainer/Singularity)
 - Using a package manager for HPC systems (Spack)
 - **Within virtual environments (via Conda, Miniconda, Mamba)**

Logging into CU Research Computing

Login to CURC via your terminal:

```
ssh <username>@login.rc.colorado.edu
```

Or login to CURC via your browser:

- Navigate to <https://ondemand-rmacc.rc.colorado.edu>
- Once logged in, select **Clusters → Alpine shell**

What are Conda and Mamba?

- They are a package (software) management system
 - Installs, runs, and updates packages and their dependencies
 - Creates, saves, loads, and switches between virtual environments
 - Created for Python programs, but can package and distribute software for any language

Why would we use Mamba?

- Mamba is a faster and more robust package manager in comparison to Conda
 - It is fully compatible with Conda packages
 - Supports most of Conda's commands
 - In most cases, Mamba can be used as a drop-in replacement for Conda
 - i.e. replace “conda” with “mamba” in commands

For a more detailed overview of Mamba's capabilities, please see: <https://mamba.readthedocs.io>

What is a `.condarc` file?

- The file “`.condarc`” is a special file that specifies configurations for Conda and Mamba
 - Specifies items such as where to store installed packages and environments
 - Located in your Home directory, `/home/$USER/.condarc`
 - We create it for you, if it doesn’t exist, with the content:

```
pkgs_dirs:  
  - /projects/$USER/.conda_pkgs  
envs_dirs:  
  - /projects/$USER/software/anaconda/envs
```

Getting access to Conda and Mamba

On CURC systems, Conda and Mamba are made available through modules

- **We highly recommend using these modules**
 - Redirects output produced by Conda and Mamba
 - Creates useful variables
 - Creates the “.condarc” file, if it doesn’t exist
 - They are easier than installing it yourself!

Conda and Mamba modules

- You must be on a compute node to get access to modules!
- Conda is accessible via

```
$ module load anaconda
```

- Mamba is accessible via

```
$ module load miniforge
```

NOTE: We will be retiring our current “mambaforge” module as it is depreciated, please use “miniforge”. Keep an eye out for name changes and updates around Conda and Mamba.

Demo time!

Start a session and load Conda

Start a session on an Alpine compute node with **acompile**:

```
[johndoe@login11 ~]$ acompile --help
[johndoe@login11 ~]$ acompile --ntasks=4
...
[johndoe@c3cpu-a5-u28-1 ~]$ module load anaconda
(base) [johndoe@c3cpu-a5-u28-1 ~]$
```

Note: when you login to CURC you'll be on a **login** node. You'll need to be on a **compute** node to use anaconda. The **acompile** command allows you to quickly start an interactive job on a compute node.

Create your first Conda environment!

- Environments are created and programs are installed in a few simple steps

```
(base) [johndoe@c3cpu-a5-u28-1 ~]$ conda create -n my_first_env python==3.10
(base) [johndoe@c3cpu-a5-u28-1 ~]$ conda activate my_first_env
(my_first_env) [johndoe@c3cpu-a5-u28-1 ~]$ python
```

Don't install packages in your base environment!

Installing packages

- Packages are installed within **activated** environments using **conda install**
- Install latest version available:

```
(my_first_env) [johndoe@c3cpu-a5-u28-1 ~]$ conda install pandas
```

- Install a specific version:

```
(my_first_env) [johndoe@c3cpu-a5-u28-1 ~]$ conda install pandas==2.2.0
```

Installing packages with “pip”

- pip installs should be done within **activated** environments

Using **pip** to install latest version:

```
(my_first_env) [johndoe@c3cpu-a5-u28-1 ~]$ pip install --no-cache-dir pandas
```

--no-cache-dir may be crucial on CURC systems

Useful Conda Commands

```
conda env list                      # list all environments  
conda list                          # list packages in active env  
conda env remove -n <envname> --all  # remove an environment  
conda config --show channels        # view configured channels  
conda deactivate                    # deactivate environment  
conda create --name <clonedenv> /  
    --clone <envtoclone>            # clone an environment
```

Useful Conda file paths on Alpine

```
# location of python libraries  
/projects/$USER/software/anaconda/<env>/lib/python3.10/site-packages
```

```
# location of package executables  
/projects/$USER/software/anaconda/<env>/bin
```

```
# location of .condarc file  
/home/$USER/.condarc
```

Running Alpine batch jobs with Conda

```
[johndoe@login11 ~]$ nano runconda.sh #Step 1: open new job script in editor
```

```
#!/bin/bash
# job script name: runconda.sh

#SBATCH --partition=amilan
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --time=10:00

module purge
module load anaconda
conda activate my_first_env
python my_python_code.py
```

Step 2: Write job script
<https://curc.readthedocs.io/en/latest/running-jobs/batch-jobs.html>

```
[johndoe@login11 ~]$ sbatch runconda.sh #Step 3: Schedule job
```

Using environments in Open OnDemand

- In Open OnDemand users can utilize their environments in Jupyter sessions. This can be done using two different methods:
 - Specifying the environment name in the “Conda environment” field
 - Allows you to launch the notebook from within your environment
 - Needed for some Jupyter extensions
 - Create a Jupyter Kernel
 - Allows you to switch between different environments while in a notebook
 - May prevent some Jupyter extensions from working
- Both methods are described at
https://curc.readthedocs.io/en/latest/open_ondemand/jupyter_session.html

Strategies for complex environments

- You may have to use different channels and change channel order
- Conflicts can arise when iteratively installing packages. If this happens create a new environment and install the package that causes conflicts first.
- Use Mamba to accelerate installations

Thank you!

Survey and feedback

<http://tinyurl.com/curc-survey18>



Slides

https://github.com/ResearchComputing/alpine_conda_mamba_primer

