Using Python, Conda and Pip on the Cluster

Introduction: This workshop will discuss how the various Python related tools work together on the cluster and introduce a series of best practices for managing these environments.

Course Goals:

- Understand where pip installs packages within a user's home folder with respect to different versions of Python.
- Understand how to use pip to install a package within a conda environment.
- Demonstrate how to create a conda environment, with pip installs, that can be shared across a project.
- Demonstrate creating a module file to *load* a conda/python environment.
- Suggested Best Practices.

This is not a workshop on learning the Python language, but on using Python, Pip and Conda explicitly on the ARCC clusters (not on desktops). Users should be familiar with python, using pip, and creating Conda environments.

Notes:

- The workshop modules work best in a sequential manner as a story introducing concepts and providing examples, but sections can be used separately to focus on a particular concept.
- If you have existing Conda environments, or have installed your own version of Anaconda, then the examples within this workshop might provide different results.
- You will need to *modify* username, project names, and folder locations to apply to yourself.

Sections

1. Python Pip Installs on the Cluster: Understand where pip installs packages within a user's home folder with respect to different versions of Python.
2. Pip Install within a Conda Environment: Understand how Conda's pip works with a User's Python pip package Installs.
3. Create a Shared TensorFlow Conda Environment: General process for creating and sharing a conda environment under a project.
4. Create a Module File to Load Your Conda Environment: Demonstrate creating a module file to *load* a Conda environment.
5. Extend Conda Environment to Jupyter Kernel: Demonstrate extending a Python related conda environment in to Juypter kernel.

6. Jupyter Python Packages and Issues: When using Jupyter how are python packages managed?
7. Conda and Pip Environments and Reproducibility: Introduce ideas and practices to assist in managing the reproducibility of environments created using Conda and Pip.
8. Python, Conda and Pip: Suggested Best Practices: Suggest best practices and summarize workshop.

Python Pip Installs on the Cluster

Goal: Understand where pip installs packages within a user's home folder with respect to different versions of Python.

- Python on the System
- Python as a Module
- Pip Install Numpy - Python 3.10.6
- Use Python 3.12.0 - Is numpy available?
- Python System Configuration
- What is the Installation Scheme?
- What is under the userbase?
- Why is they no lib/python3.12?
- Pip Install Numpy - Python 3.12.0
- What has changed under the .local/bin folder?

Python on the System

Python is available on the System (but not necessarily on the compute nodes due to keeping node images *lite*).

```
[salexan5@mblog2 ~]$ python --version
Python 3.9.18

[salexan5@mblog1 ~]$ which python
/usr/bin/python
```

Note:

- Version might update during cluster maintenance.
- Pip may/may note be installed.
- Recommended to not use to due potential updates.

Python as a Module

Different versions are available via loading a module

```
[salexan5@mblog2 ~]$ module spider python
----------------------------------------------------------------------------
  python:
----------------------------------------------------------------------------
     Versions:
        python/3.10.6
        python/3.12.0
     Other possible modules matches:
        python2

[salexan5@mblog2 ~]$ module load gcc/13.2.0 python/3.10.6
[salexan5@mblog2 ~]$ python --version
Python 3.10.6
[salexan5@mblog2 ~]$ which pip
/apps/u/spack/gcc/13.2.0/python/3.10.6-jh3qs5v/bin/pip
```

Note:

- Recommended to use a module version for reproducibility.
- Pip will be available.

Pip Install Numpy - Python 3.10.6

```
# Using python/3.10.6
[salexan5@mblog2 ~]$ pip install numpy
Defaulting to user installation because normal site-packages is not writeable
Collecting numpy
  Downloading numpy-2.0.0-cp310-cp310-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (60 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
60.9/60.9 kB 2.0 MB/s eta 0:00:00
Downloading numpy-2.0.0-cp310-cp310-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (19.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
19.3/19.3 MB 131.3 MB/s eta 0:00:00
Installing collected packages: numpy
Successfully installed numpy-2.0.0

[notice] A new release of pip is available: 24.0 -> 24.1
[notice] To update, run: pip install --upgrade pip
```

py_test.py

```
[salexan5@mblog2 ~]$ python py_test.py
```

```
Python: 3.10.6 (main, Apr 30 2024, 11:23:04) [GCC 13.2.0]
Numpy: 2.0.0
```

Use Python 3.12.0 - Is numpy available?

```
[salexan5@mblog2 ~]$ module purge
[salexan5@mblog2 ~]$ module load gcc/13.2.0 python/3.12.0
[salexan5@mblog2 ~]$ python --version
Python 3.12.0
[salexan5@mblog2 ~]$ which python
/apps/u/spack/gcc/13.2.0/python/3.12.0-ovfqpv2/bin/python

[salexan5@mblog2 ~]$ python py_test.py
Traceback (most recent call last):
  File "/cluster/medbow/home/salexan5/py_test.py", line 2, in <module>
    import numpy
ModuleNotFoundError: No module named 'numpy
```

But didn’t we previously install the `numpy` package?

Python System Configuration

sysconfig: The `sysconfig` module provides access to Python’s configuration information like the list of installation paths and the configuration variables relevant for the current platform.

```
[salexan5@mblog2 ~]$ module purge
[salexan5@mblog2 ~]$ module load gcc/13.2.0 python/3.10.6
[salexan5@mblog2 ~]$ python -m sysconfig
Platform: "linux-x86_64"
Python version: "3.10"
Current installation scheme: "posix_prefix"
Paths:
        ...
               scripts = "/apps/u/spack/gcc/13.2.0/python/3.10.6-jh3qs5v/bin"
        stdlib = "/apps/u/spack/gcc/13.2.0/python/3.10.6-
jh3qs5v/lib/python3.10"
...
Variables:
        ...
        py_version = "3.10.6"
        ...
        py_version_short = "3.10"
        ...
        userbase = "/home/salexan5/.local"
# Full list has over 700 lines!
```

What is the Installation Scheme?

Prefix scheme: The “prefix scheme” is useful when you wish to use one Python installation to perform the build/install (i.e., to run the setup script), but install modules into the third-party module directory of a different Python installation (or something that looks like a different Python installation). If this sounds a trifle unusual, it is—that’s why the user and home schemes come before. However, there are at least two known cases where the prefix scheme will be useful.

posix_prefix:

Path	Installation directory
stdlib	`prefix/lib/pythonX.Y`
scripts	`prefix/bin`
data	`prefix`

The prefix will be the `userbase` - so in this case: `/home/<username>/.local`

What is under the userbase?

```
[salexan5@mblog2 ~]$ ls .local
bin  lib  share

[salexan5@mblog2 ~]$ ls -al .local/bin
```

```
total 1
drwxr-xr-x 2 salexan5 salexan5 4096 Jun 24 11:16 .
drwxr-xr-x 2 salexan5 salexan5 4096 Jun 24 11:16 ..
-rwxr-xr-x 1 salexan5 salexan5  257 Jun 24 11:16 f2py
-rwxr-xr-x 1 salexan5 salexan5  257 Jun 24 11:16 numpy-config

[salexan5@mblog2 ~]$ ls .local/lib/
python3.10
[salexan5@mblog2 ~]$ ls .local/lib/python3.10/site-packages/
numpy  numpy-2.0.0.dist-info  numpy.libs

# ~/.local/lib/python3.10
```

Why is they no lib/python3.12?

Because you have not pip installed anything for this particular version.

When using python/3.12.0 it is looking under `~/.local/lib/python3.12` for pip installed packages.

Since we have not pip installed `numpy` using version 3.12.0, no package exists under this folder.

Pip Install Numpy - Python 3.12.0

```
[salexan5@mblog2 ~]$ module purge
[salexan5@mblog2 ~]$ module load gcc/13.2.0 python/3.12.0
[salexan5@mblog2 ~]$ pip install numpy==1.26.4
[salexan5@mblog2 ~]$ python py_test.py
Python: 3.12.0 (main, Apr 30 2024, 11:30:37) [GCC 13.2.0]
Numpy: 1.26.4

[salexan5@mblog2 ~]$ ls .local/lib/
python3.10  python3.12

[salexan5@mblog2 ~]$ ls .local/lib/python3.12/site-packages/
numpy  numpy-1.26.4.dist-info  numpy.libs
```

What has changed under the .local/bin folder?

```
# Previously when installing numpy for python/3.10.6
[salexan5@mblog2 ~]$ ls -al .local/bin
total 1
```

```
-rwxr-xr-x 1 salexan5 salexan5  257 Jun 24 11:16 f2py
-rwxr-xr-x 1 salexan5 salexan5  257 Jun 24 11:16 numpy-config
[salexan5@mblog2 ~]$ cat .local/bin/f2py
#!/apps/u/spack/gcc/13.2.0/python/3.10.6-jh3qs5v/bin/python
...

# Now after installing numpy for python/3.12.0
[salexan5@mblog2 ~]$ ls -al .local/bin/
-rwxr-xr-x 1 salexan5 salexan5  257 Jun 24 11:45 f2py
-rwxr-xr-x 1 salexan5 salexan5  257 Jun 24 11:16 numpy-config
[salexan5@mblog2 ~]$ cat .local/bin/f2py
#!/apps/u/spack/gcc/13.2.0/python/3.12.0-ovfqpv2/bin/python
...
```

The `python/3.10.6` `numpy` related files have been overridden.

Pip Install within a Conda Environment

Goal: Understand how Conda’s pip works with a User’s Python pip package Installs.

- Quick Note: Default Python Version with a Conda Environment
- Conda Environments and pip Installs
- Try Installing numpy into our Conda environment
- Confirm what is Currently Installed/Available
- Force a numpy update
- Create a Self-Contained Conda Environment
- Recommendation: Always set the PYTHONUSERBASE environment variable

---

Quick Note: Default Python Version with a Conda Environment

Miniconda does ship with a default version of Python.

```
[salexan5@mblog1 conda]$ cd /project/arcc/salexan5/conda
[salexan5@mblog1 conda]$ module load miniconda3/24.3.0
[salexan5@mblog1 conda]$ conda create -p py_env
## Package Plan ##
  environment location: /cluster/medbow/project/arcc/salexan5/conda/py_env

# To activate this environment, use
#     $ conda activate /cluster/medbow/project/arcc/salexan5/conda/py_env
```

```
[salexan5@mblog1 conda]$ conda activate
/cluster/medbow/project/arcc/salexan5/conda/py_env

(/cluster/medbow/project/arcc/salexan5/conda/py_env) [salexan5@mblog1 conda]$
python --version
Python 3.12.2

(/cluster/medbow/project/arcc/salexan5/conda/py_env) [salexan5@mblog1 conda]$
which python
/apps/u/opt/linux/miniconda3/24.3.0/bin/python
```

Conda Environments and pip Installs

Installing non-conda packages:

- If a package is not available from conda or Anaconda.org, you may be able to find and install the package via conda-forge or with another package manager like `pip`.
- Pip packages do not have all the features of conda packages and we recommend first trying to install any package with conda. If the package is unavailable through conda, try finding and installing it with conda-forge.
- If you still cannot install the package, you can try installing it with pip. The differences between pip and conda packages cause certain unavoidable limits in compatibility but conda works hard to be as compatible with pip as possible.
- **Note**:
  - Both pip and conda are included in Anaconda and Miniconda, so you do not need to install them separately.
  - It is possible to have pip installed outside a conda environment or inside a conda environment.

Not every Python package can be `conda install`-ed. Tensorflow is only available via `pip install`.

Try Installing numpy into our Conda environment

```
(/cluster/medbow/project/arcc/salexan5/conda/py_env) [salexan5@mblog1 conda]$
pip install numpy
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: numpy in
/home/salexan5/.local/lib/python3.12/site-packages (1.26.4)

(/cluster/medbow/project/arcc/salexan5/conda/py_env) [salexan5@mblog1 conda]$
python ~/py_test.py
```

```
Python: 3.12.2 | packaged by Anaconda, Inc. | (main, Feb 27 2024, 17:35:02)
[GCC 11.2.0]
Numpy: 1.26.4
```

What’s happened?

Conda’s pip will look under the userbase for existing packages.

In this case `~/.local/lib/python3.12/site-packages/` where it will find the previously installed version of `numpy/1.26.4`

Confirm what is Currently Installed/Available

Check for conda installed packages:

```
# This list should be empty since we have essentially a blank conda
environment
# since we have not conda installed anything.
(/cluster/medbow/project/arcc/salexan5/conda/py_env) [salexan5@mblog1 conda]$
conda list
# packages in environment at
/cluster/medbow/project/arcc/salexan5/conda/py_env:
#
# Name                    Version                   Build  Channel
```

Check for pip installed packages:

```
(/cluster/medbow/project/arcc/salexan5/conda/py_env) [salexan5@mblog1 conda]$
pip list -v
Package                 Version     Location
Installer
----------------------- ----------- -----------------------------------------
----------------------- ---------
anaconda-anon-usage     0.4.4
/apps/u/opt/linux/miniconda3/24.3.0/lib/python3.12/site-packages conda
archspec                0.2.3
/apps/u/opt/linux/miniconda3/24.3.0/lib/python3.12/site-packages conda
...
numpy                   1.26.4
/home/salexan5/.local/lib/python3.12/site-packages               pip
...
pip                     23.3.1
/apps/u/opt/linux/miniconda3/24.3.0/lib/python3.12/site-packages
...
```

Conda’s pip will check for package installs under the `~/.local/lib/PythonX.Y` folder.

If the python package already exists it will be used.

Force a numpy update

```
(/cluster/medbow/project/arcc/salexan5/conda/py_env) [salexan5@mblog1 conda]$
pip install numpy==2.0.0
Defaulting to user installation because normal site-packages is not writeable
...
Installing collected packages: numpy
  Attempting uninstall: numpy
    Found existing installation: numpy 1.26.4
    Uninstalling numpy-1.26.4:
      Successfully uninstalled numpy-1.26.4
Successfully installed numpy-2.0.0
(/cluster/medbow/project/arcc/salexan5/conda/py_env) [salexan5@mblog1 conda]$
pip list -v
Package                 Version     Location
Installer
----------------------- ----------- -----------------------------------------
----------------------- ---------
...
numpy                   2.0.0
/home/salexan5/.local/lib/python3.12/site-packages               pip
...
```

numpy has been updated, but still lives under: `~/.local/lib/python3.12/`

Create a Self-Contained Conda Environment

We can force our conda environment to only use what is *installed* within it, by setting the `PYTHONUSERBASE` environment variable:

```
(/cluster/medbow/project/arcc/salexan5/conda/py_env) [salexan5@mblog1 conda]$
echo $CONDA_PREFIX
/cluster/medbow/project/arcc/salexan5/conda/py_env

(/cluster/medbow/project/arcc/salexan5/conda/py_env_3.12.4) [salexan5@mblog2
conda]$ export PYTHONUSERBASE=$CONDA_PREFIX

(/cluster/medbow/project/arcc/salexan5/conda/py_env) [salexan5@mblog1 conda]$
pip list -v
Package                 Version  Location
Installer
----------------------- -------- --------------------------------------------
----------------------------------- ---------
...
```

```
numpy                   2.0.0
/cluster/medbow/project/arcc/salexan5/conda/py_env/lib/python3.12/site-
packages pip
...
```

Conda’s pip now only looks under, and installs under, it’s own site-packages folder:

```
(/cluster/medbow/project/arcc/salexan5/conda/py_env) [salexan5@mblog1 conda]$
pwd
/project/arcc/salexan5/conda

(/cluster/medbow/project/arcc/salexan5/conda/py_env) [salexan5@mblog1 conda]$
ls py_env/lib/python3.12/site-packages/
numpy  numpy-2.0.0.dist-info  numpy.libs
```

Recommendation: Always set the `PYTHONUSERBASE` environment variable

Sessions/Environments can get *confused* if you have previously set environment variables and have forgotten what has/hasn’t been set within a session.

Starting a ***new*** session, notice which version of `numpy` is being used by our conda environment before and after setting our `PYTHONUSERBASE` environment variable.

```
[salexan5@mblog2 ~]$ module load miniconda3/24.3.0
[salexan5@mblog2 ~]$ conda activate /project/arcc/salexan5/conda/py_env
(/project/arcc/salexan5/conda/py_env) [salexan5@mblog2 ~]$ pip list -v
Package                 Version     Location
Installer
----------------------- ----------- -----------------------------------------
----------------------- ---------
...
numpy                   2.0.0
/home/salexan5/.local/lib/python3.12/site-packages               pip
...

(/project/arcc/salexan5/conda/py_env) [salexan5@mblog2 ~]$ export
PYTHONUSERBASE=$CONDA_PREFIX
(/project/arcc/salexan5/conda/py_env) [salexan5@mblog2 ~]$ pip list -v
Package                 Version  Location
Installer
----------------------- -------- --------------------------------------------
-------------------- ---------
...
numpy                   2.0.0
/project/arcc/salexan5/conda/py_env/lib/python3.12/site-packages pip
...
```

Create a Shared TensorFlow Conda Environment

Goal: General process for creating and sharing a conda environment under a project.

- Use Case
- General Process
- Notice Where Related Packages Were Installed
- How is this Shared?

Use Case

We want to create a software installation of TensorFlow (an end-to-end platform for machine learning) that can be shared across a project by all users.

From the documentation (as of time of writing):

- The install documentation only details using a `pip install`.
- TensorFlow is tested and supported on the following 64-bit systems: Python 3.8–3.11

General Process

```
[salexan5@mblog2 tensorflow]$ module load miniconda3/24.3.0
[salexan5@mblog2 tensorflow]$ conda create -p 2.16 python=3.11
[salexan5@mblog2 tensorflow]$ conda activate
/cluster/medbow/project/arcc/software/tensorflow/2.16
(/cluster/medbow/project/arcc/software/tensorflow/2.16) [salexan5@mblog2
tensorflow]$ export PYTHONUSERBASE=$CONDA_PREFIX
(/cluster/medbow/project/arcc/software/tensorflow/2.16) [salexan5@mblog2
tensorflow]$ echo $PYTHONUSERBASE
/cluster/medbow/project/arcc/software/tensorflow/2.16
(/cluster/medbow/project/arcc/software/tensorflow/2.16) [salexan5@mblog2
tensorflow]$ pip install tensorflow
(/cluster/medbow/project/arcc/software/tensorflow/2.16) [salexan5@mblog2
tensorflow]$ python --version
Python 3.11.9

(/cluster/medbow/project/arcc/software/tensorflow/2.16) [salexan5@mblog2
tensorflow]$ python ~/tf_test.py
```

```
...
TensorFlow Version: 2.16.1
tf.Tensor(997.42975, shape=(), dtype=float32)
(/cluster/medbow/project/arcc/software/tensorflow/2.16) [salexan5@mblog2
tensorflow]$ conda deactivate
[salexan5@mblog2 tensorflow]$
```

tf_test.py

```
import tensorflow as tf
print("TensorFlow Version: " + str( tf.__version__))
print(tf.reduce_sum(tf.random.normal([1000, 1000])))
```

Full pip install:

```
(/cluster/medbow/project/arcc/software/tensorflow/2.16) [salexan5@mblog2
tensorflow]$ pip install tensorflow
Collecting tensorflow
  Downloading tensorflow-2.16.1-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (4.3 kB)
Collecting absl-py>=1.0.0 (from tensorflow)
  Downloading absl_py-2.1.0-py3-none-any.whl.metadata (2.3 kB)
Collecting astunparse>=1.6.0 (from tensorflow)
  Downloading astunparse-1.6.3-py2.py3-none-any.whl.metadata (4.4 kB)
Collecting flatbuffers>=23.5.26 (from tensorflow)
  Downloading flatbuffers-24.3.25-py2.py3-none-any.whl.metadata (850 bytes)
Collecting gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 (from tensorflow)
  Downloading gast-0.5.4-py3-none-any.whl.metadata (1.3 kB)
Collecting google-pasta>=0.1.1 (from tensorflow)
  Downloading google_pasta-0.2.0-py3-none-any.whl.metadata (814 bytes)
Collecting h5py>=3.10.0 (from tensorflow)
  Downloading h5py-3.11.0-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (2.5 kB)
Collecting libclang>=13.0.0 (from tensorflow)
  Downloading libclang-18.1.1-py2.py3-none-manylinux2010_x86_64.whl.metadata
(5.2 kB)
Collecting ml-dtypes~=0.3.1 (from tensorflow)
  Downloading ml_dtypes-0.3.2-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (20 kB)
Collecting opt-einsum>=2.3.2 (from tensorflow)
  Downloading opt_einsum-3.3.0-py3-none-any.whl.metadata (6.5 kB)
Collecting packaging (from tensorflow)
  Downloading packaging-24.1-py3-none-any.whl.metadata (3.2 kB)
Collecting
protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.2
0.3 (from tensorflow)
  Downloading protobuf-4.25.3-cp37-abi3-manylinux2014_x86_64.whl.metadata
(541 bytes)
Collecting requests<3,>=2.21.0 (from tensorflow)
  Downloading requests-2.32.3-py3-none-any.whl.metadata (4.6 kB)
Requirement already satisfied: setuptools in ./2.16/lib/python3.11/site-
packages (from tensorflow) (70.1.0)
Collecting six>=1.12.0 (from tensorflow)
  Using cached six-1.16.0-py2.py3-none-any.whl.metadata (1.8 kB)
Collecting termcolor>=1.1.0 (from tensorflow)
  Downloading termcolor-2.4.0-py3-none-any.whl.metadata (6.1 kB)
Collecting typing-extensions>=3.6.6 (from tensorflow)
  Downloading typing_extensions-4.12.2-py3-none-any.whl.metadata (3.0 kB)
Collecting wrapt>=1.11.0 (from tensorflow)
```

```
  Downloading wrapt-1.16.0-cp311-cp311-
manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.manylinux2014_x8
6_64.whl.metadata (6.6 kB)
Collecting grpcio<2.0,>=1.24.3 (from tensorflow)
  Downloading grpcio-1.64.1-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (3.3 kB)
Collecting tensorboard<2.17,>=2.16 (from tensorflow)
  Downloading tensorboard-2.16.2-py3-none-any.whl.metadata (1.6 kB)
Collecting keras>=3.0.0 (from tensorflow)
  Downloading keras-3.3.3-py3-none-any.whl.metadata (5.7 kB)
Collecting tensorflow-io-gcs-filesystem>=0.23.1 (from tensorflow)
  Downloading tensorflow_io_gcs_filesystem-0.37.0-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (14 kB)
Collecting numpy<2.0.0,>=1.23.5 (from tensorflow)
  Downloading numpy-1.26.4-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (61 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
61.0/61.0 kB 2.1 MB/s eta 0:00:00
Requirement already satisfied: wheel<1.0,>=0.23.0 in
./2.16/lib/python3.11/site-packages (from astunparse>=1.6.0->tensorflow)
(0.43.0)
Collecting rich (from keras>=3.0.0->tensorflow)
  Downloading rich-13.7.1-py3-none-any.whl.metadata (18 kB)
Collecting namex (from keras>=3.0.0->tensorflow)
  Downloading namex-0.0.8-py3-none-any.whl.metadata (246 bytes)
Collecting optree (from keras>=3.0.0->tensorflow)
  Downloading optree-0.11.0-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (45 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
45.4/45.4 kB 1.6 MB/s eta 0:00:00
Collecting charset-normalizer<4,>=2 (from requests<3,>=2.21.0->tensorflow)
  Downloading charset_normalizer-3.3.2-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (33 kB)
Collecting idna<4,>=2.5 (from requests<3,>=2.21.0->tensorflow)
  Downloading idna-3.7-py3-none-any.whl.metadata (9.9 kB)
Collecting urllib3<3,>=1.21.1 (from requests<3,>=2.21.0->tensorflow)
  Downloading urllib3-2.2.2-py3-none-any.whl.metadata (6.4 kB)
Collecting certifi>=2017.4.17 (from requests<3,>=2.21.0->tensorflow)
  Downloading certifi-2024.6.2-py3-none-any.whl.metadata (2.2 kB)
Collecting markdown>=2.6.8 (from tensorboard<2.17,>=2.16->tensorflow)
  Downloading Markdown-3.6-py3-none-any.whl.metadata (7.0 kB)
Collecting tensorboard-data-server<0.8.0,>=0.7.0 (from
tensorboard<2.17,>=2.16->tensorflow)
  Downloading tensorboard_data_server-0.7.2-py3-none-
manylinux_2_31_x86_64.whl.metadata (1.1 kB)
Collecting werkzeug>=1.0.1 (from tensorboard<2.17,>=2.16->tensorflow)
  Downloading werkzeug-3.0.3-py3-none-any.whl.metadata (3.7 kB)
Collecting MarkupSafe>=2.1.1 (from werkzeug>=1.0.1->tensorboard<2.17,>=2.16-
>tensorflow)
  Downloading MarkupSafe-2.1.5-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (3.0 kB)
Collecting markdown-it-py>=2.2.0 (from rich->keras>=3.0.0->tensorflow)
  Downloading markdown_it_py-3.0.0-py3-none-any.whl.metadata (6.9 kB)
Collecting pygments<3.0.0,>=2.13.0 (from rich->keras>=3.0.0->tensorflow)
  Downloading pygments-2.18.0-py3-none-any.whl.metadata (2.5 kB)
```

```
Collecting mdurl~=0.1 (from markdown-it-py>=2.2.0->rich->keras>=3.0.0-
>tensorflow)
  Downloading mdurl-0.1.2-py3-none-any.whl.metadata (1.6 kB)
Downloading tensorflow-2.16.1-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (589.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
589.8/589.8 MB 12.5 MB/s eta 0:00:00
Downloading absl_py-2.1.0-py3-none-any.whl (133 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
133.7/133.7 kB 6.2 MB/s eta 0:00:00
Downloading astunparse-1.6.3-py2.py3-none-any.whl (12 kB)
Downloading flatbuffers-24.3.25-py2.py3-none-any.whl (26 kB)
Downloading gast-0.5.4-py3-none-any.whl (19 kB)
Downloading google_pasta-0.2.0-py3-none-any.whl (57 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
57.5/57.5 kB 2.0 MB/s eta 0:00:00
Downloading grpcio-1.64.1-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (5.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5.6/5.6
MB 107.4 MB/s eta 0:00:00
Downloading h5py-3.11.0-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (5.4 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5.4/5.4
MB 104.4 MB/s eta 0:00:00
Downloading keras-3.3.3-py3-none-any.whl (1.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.1/1.1
MB 37.0 MB/s eta 0:00:00
Downloading libclang-18.1.1-py2.py3-none-manylinux2010_x86_64.whl (24.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
24.5/24.5 MB 89.4 MB/s eta 0:00:00
Downloading ml_dtypes-0.3.2-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (2.2 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.2/2.2
MB 60.9 MB/s eta 0:00:00
Downloading numpy-1.26.4-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (18.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
18.3/18.3 MB 91.6 MB/s eta 0:00:00
Downloading opt_einsum-3.3.0-py3-none-any.whl (65 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
65.5/65.5 kB 2.9 MB/s eta 0:00:00
Downloading protobuf-4.25.3-cp37-abi3-manylinux2014_x86_64.whl (294 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
294.6/294.6 kB 15.6 MB/s eta 0:00:00
Downloading requests-2.32.3-py3-none-any.whl (64 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
64.9/64.9 kB 2.6 MB/s eta 0:00:00
Using cached six-1.16.0-py2.py3-none-any.whl (11 kB)
Downloading tensorboard-2.16.2-py3-none-any.whl (5.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5.5/5.5
MB 86.2 MB/s eta 0:00:00
Downloading tensorflow_io_gcs_filesystem-0.37.0-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (5.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5.1/5.1
MB 85.7 MB/s eta 0:00:00
```

```
Downloading termcolor-2.4.0-py3-none-any.whl (7.7 kB)
Downloading typing_extensions-4.12.2-py3-none-any.whl (37 kB)
Downloading wrapt-1.16.0-cp311-cp311-
manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.manylinux2014_x8
6_64.whl (80 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
80.7/80.7 kB 3.2 MB/s eta 0:00:00
Downloading packaging-24.1-py3-none-any.whl (53 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
54.0/54.0 kB 2.1 MB/s eta 0:00:00
Downloading certifi-2024.6.2-py3-none-any.whl (164 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
164.4/164.4 kB 7.5 MB/s eta 0:00:00
Downloading charset_normalizer-3.3.2-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (140 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
140.3/140.3 kB 6.1 MB/s eta 0:00:00
Downloading idna-3.7-py3-none-any.whl (66 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
66.8/66.8 kB 2.7 MB/s eta 0:00:00
Downloading Markdown-3.6-py3-none-any.whl (105 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
105.4/105.4 kB 4.5 MB/s eta 0:00:00
Downloading tensorboard_data_server-0.7.2-py3-none-manylinux_2_31_x86_64.whl
(6.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.6/6.6
MB 102.1 MB/s eta 0:00:00
Downloading urllib3-2.2.2-py3-none-any.whl (121 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
121.4/121.4 kB 4.7 MB/s eta 0:00:00
Downloading werkzeug-3.0.3-py3-none-any.whl (227 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
227.3/227.3 kB 10.1 MB/s eta 0:00:00
Downloading namex-0.0.8-py3-none-any.whl (5.8 kB)
Downloading optree-0.11.0-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (312 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
312.0/312.0 kB 15.7 MB/s eta 0:00:00
Downloading rich-13.7.1-py3-none-any.whl (240 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
240.7/240.7 kB 12.1 MB/s eta 0:00:00
Downloading markdown_it_py-3.0.0-py3-none-any.whl (87 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
87.5/87.5 kB 3.5 MB/s eta 0:00:00
Downloading MarkupSafe-2.1.5-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (28 kB)
Downloading pygments-2.18.0-py3-none-any.whl (1.2 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.2/1.2
MB 47.0 MB/s eta 0:00:00
Downloading mdurl-0.1.2-py3-none-any.whl (10.0 kB)
Installing collected packages: namex, libclang, flatbuffers, wrapt, urllib3,
typing-extensions, termcolor, tensorflow-io-gcs-filesystem, tensorboard-data-
server, six, pygments, protobuf, packaging, numpy, mdurl, MarkupSafe,
markdown, idna, grpcio, gast, charset-normalizer, certifi, absl-py, werkzeug,
requests, optree, opt-einsum, ml-dtypes, markdown-it-py, h5py, google-pasta,
astunparse, tensorboard, rich, keras, tensorflow
Successfully installed MarkupSafe-2.1.5 absl-py-2.1.0 astunparse-1.6.3
certifi-2024.6.2 charset-normalizer-3.3.2 flatbuffers-24.3.25 gast-0.5.4
google-pasta-0.2.0 grpcio-1.64.1 h5py-3.11.0 idna-3.7 keras-3.3.3 libclang-
18.1.1 markdown-3.6 markdown-it-py-3.0.0 mdurl-0.1.2 ml-dtypes-0.3.2 namex-
0.0.8 numpy-1.26.4 opt-einsum-3.3.0 optree-0.11.0 packaging-24.1 protobuf-
4.25.3 pygments-2.18.0 requests-2.32.3 rich-13.7.1 six-1.16.0 tensorboard-
2.16.2 tensorboard-data-server-0.7.2 tensorflow-2.16.1 tensorflow-io-gcs-
filesystem-0.37.0 termcolor-2.4.0 typing-extensions-4.12.2 urllib3-2.2.2
werkzeug-3.0.3 wrapt-1.16.0
```

Notice Where Related Packages Were Installed

Since we set `export PYTHONUSERBASE=$CONDA_PREFIX` there is nothing within the home folder under: `.local/lib/` i.e. no `Python3.11` folder:

```
[salexan5@mblog1 ~]$ ls .local/lib/
python3.10  python3.12
```

Everything is self contained within the conda environment itself:

```
(/cluster/medbow/project/arcc/software/tensorflow/2.16) [salexan5@mblog2
tensorflow]$ pwd
/project/arcc/software/tensorflow

(/cluster/medbow/project/arcc/software/tensorflow/2.16) [salexan5@mblog2
tensorflow]$ ls 2.16/lib/python3.11/site-packages/
absl                                grpc                            mdurl-
0.1.2.dist-info       pkg_resources
tensorboard_data_server-0.7.2.dist-info
absl_py-2.1.0.dist-info             grpcio-1.64.1.dist-info         ml_dtypes
protobuf-4.25.3.dist-info          tensorflow
astunparse                          h5py
ml_dtypes-0.3.2.dist-info   __pycache__                        tensorflow-
2.16.1.dist-info
astunparse-1.6.3.dist-info          h5py-3.11.0.dist-info           namex
pygments                           tensorflow_io_gcs_filesystem
certifi                             h5py.libs                       namex-
0.0.8.dist-info       pygments-2.18.0.dist-info
tensorflow_io_gcs_filesystem-0.37.0.dist-info
...
```

How is this Shared?

Since the environment was created under a *project*, any one part of this project can also access/*activate* this environment.

```
[salexan5@mblog2 tensorflow]$ pwd
/project/arcc/software/tensorflow
[salexan5@mblog2 tensorflow]$ ls -al
...
drwxr-sr-x 2 salexan5 arcc 4096 Jun 24 16:33 2.16
```

The user `brewer` is part of the `arcc` project.

```
[brewer@mblog2 ~]$ module load miniconda3/24.3.0
[brewer@mblog2 ~]$ conda activate /project/arcc/software/tensorflow/2.16/
(/project/arcc/software/tensorflow/2.16) [brewer@mblog2 ~]$ python -c "import
tensorflow as tf; print(\"TensorFlow Version: \" + str( tf.__version__))"
...
TensorFlow Version: 2.16.1
(/project/arcc/software/tensorflow/2.16) [brewer@mblog2 ~]$ conda deactivate
[brewer@mblog2 ~]$
```

Create a Module File to Load Your Conda Environment

Goal: Demonstrate creating a module file to *load* a Conda environment.

- Remember what a module is
- Module File Basic Template
- Use Your Module Files
- Using Your Module
- Notice the sysconfig
- Running our TF Code

Remember what a module is

We use the LMOD module system to setup our environment by loading modules of compilers, languages, libraries and applications…

Although typically the module are setup by ARCC, you can create your own module files for software and environments that you have installed (within home/project). You can create your own module files that expose and load these.

Module File Basic Template

An Introduction to Writing Modulefiles: Module files are scripted using Lua and define core environment elements required to setup and use *something*.

Example:

```
[salexan5@mblog2 ~]$ cd /project/arcc/software
[salexan5@mblog2 software] mkdir -p modules/tensorflow
[salexan5@mblog2 software] cd modules/tensorflow
[salexan5@mblog2 tensorflow] vim 2.16.lua
# 2.16.lua
whatis(" Name: TensorFlow ")
whatis(" Version : 2.16 ")
whatis(" Short Description: An end-to-end platform for machine learning.")
prepend_path("PATH","/project/arcc/software/tensorflow/2.16/bin/")
```

Use Your Module Files

To expose your module files you need to tell the module system where to look using the `module use <path>` command.

```
[salexan5@mblog1 ~]$ module avail
-------------- /apps/s/lmod/mf/opt/linux-rhel9-x86_64/containers -------------
---
   stress-ng/0.17.08
...
# Expose your module files.
[salexan5@mblog1 ~]$ module use /project/arcc/software/modules/
[salexan5@mblog1 ~]$ module avail
------------------------ /project/arcc/software/modules ----------------------
---
   tensorflow/2.16
-------------- /apps/s/lmod/mf/opt/linux-rhel9-x86_64/containers -------------
---
   stress-ng/0.17.08
...
[salexan5@mblog2 ~]$ module spider tensorflow
-----------------------------------------------------------------------------
  tensorflow: tensorflow/2.16
-----------------------------------------------------------------------------
    This module can be loaded directly: module load tensorflow/2.16
```

Using Your Module

Loading a local module has the same effect as using a System module.

Your environment will be updated by setting appropriate environment variables.

```
[salexan5@mblog1 ~]$ salloc -A arcc -t 10:00
salloc: Granted job allocation 845851
salloc: Nodes mbcpu-001 are ready for job
[salexan5@mbcpu-001 ~]$ module use /project/arcc/software/modules/
[salexan5@mbcpu-001 ~]$ module load tensorflow/2.16
[salexan5@mbcpu-001 ~]$ python --version
Python 3.11.9

[salexan5@mblog2 tensorflow]$ which python
/project/arcc/software/tensorflow/2.16/bin/python
```

Notice the sysconfig

After loading our module and exposing the conda environment, notice how the `sysconfig` is based around this environment.

```
[salexan5@mblog2 tensorflow]$ python -m sysconfig
Platform: "linux-x86_64"
Python version: "3.11"
Current installation scheme: "posix_prefix"
Paths:
        data = "/project/arcc/software/tensorflow/2.16"
        ...
        stdlib = "/project/arcc/software/tensorflow/2.16/lib/python3.11"
Variables:
        ...
        userbase = "/cluster/medbow/project/arcc/software/tensorflow/2.16"
```

Notice that the `userbase` is pointing to where our conda environment was installed.

Running our TF Code

All the conda and pip package installs within this conda environment are available.

```
[salexan5@mbcpu-001 ~]$ python -c "import tensorflow as tf;
print(\"TensorFlow Version: \" + str( tf.__version__))"
...
TensorFlow Version: 2.16.1
[salexan5@mbcpu-001 ~]$ exit
exit
salloc: Relinquishing job allocation 845851
```

Extend Conda Environment to Jupyter Kernel

Goal: Demonstrate extending a Python related conda environment in to Juypter kernel.

- General Process
- Install the ipykernel package
- Configure Your Jupyter Environment
- Start Jupyter
- Within a Notebook

General Process

The general process involves updating the conda environment to include kernel related packages, and then configuring the kernel spec to allow it to be picked up by the Jupyter service.

1. Activate you Python Conda Environment.
2. Conda install the `ipykernel` package.
3. Deactivate your Conda environment.
4. Copy the created `kernelspec` into your home `.local/share/jupyter/kernels/` folder.
5. Update the kernel.json.

The process creates a kernel for an individual user from the previously created shared Conda environment.

Each individual from the shared space will need to follow this process to create their own kernel.

If anyone updated the packages within this shared Conda environment, then the update will effect everyone. Be warned.

Install the `ipykernel` package

```
[salexan5@mblog1 ~]$ module load miniconda3/24.3.0
[salexan5@mblog1 ~]$ conda activate
/cluster/medbow/project/arcc/software/tensorflow/2.16
(/cluster/medbow/project/arcc/software/tensorflow/2.16) [salexan5@mblog1 ~]$
export PYTHONUSERBASE=$CONDA_PREFIX
```

```
(/cluster/medbow/project/arcc/software/tensorflow/2.16) [salexan5@mblog1 ~]$
conda install ipykernel
...
(/cluster/medbow/project/arcc/software/tensorflow/2.16) [salexan5@mblog1 ~]$
conda deactivate
```

Created kernel related folder

Installing the ipykernel package will create a kernel spec related folder that we can use.

This can be found under the Conda environment location, under: `share/jupyter/kernels/` named `python3`

```
[salexan5@mblog1 ~]$ ls
/project/arcc/software/tensorflow/2.16/share/jupyter/kernels/python3/

[salexan5@mblog1 ~]$ cd /project/arcc/software/tensorflow/2.16
[salexan5@mblog1 2.16]$ cd share/jupyter/kernels/
[salexan5@mblog1 kernels]$ ls
python3

[salexan5@mblog1 kernels]$ ls python3
kernel.json  logo-32x32.png  logo-64x64.png  logo-svg.svg
```

Rename the folder to something more appropriate:

```
[salexan5@mblog1 kernels]$ mv python3 TF2.16
```

Configure Your Jupyter Environment

If you haven’t used the Jupyter service, then you might not have, and thus will need to create the following folders:

```
.local/share/jupyter/kernels/
```

Copy the created `TF2.16` folder into your home:

```
[salexan5@mblog1 kernels]$ cp -r TF2.16/ ~/.local/share/jupyter/kernels/
```

Update the `kernel.json` file to:

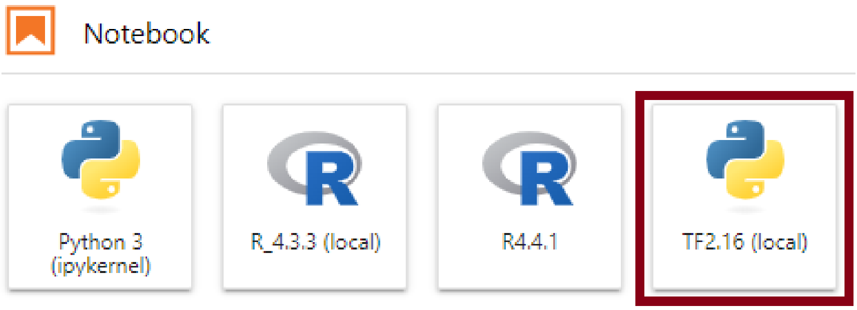
- give this kernel a unique **display name**.

```
[salexan5@mblog1 kernels]$ ~/.local/share/jupyter/kernels/TF2.16/
[salexan5@mblog1 tf2.16]$ cat kernel.json
{
 "argv": [
  "/cluster/medbow/project/arcc/software/tensorflow/2.16/bin/python",
  "-m",
  "ipykernel_launcher",
  "-f",
  "{connection_file}"
 ],
 "display_name": "TF2.16 (local)",
 "language": "python",
 "metadata": {
  "debugger": true
 }
}
```

Start Jupyter

From OnDemand start a Jupyter session.

Notice how the newly configured kernel is now available.



Within a Notebook

Within a cell try:

```
import tensorflow as tf;
print("TensorFlow Version: " + str( tf.__version__))
TensorFlow Version: 2.16.1
```

Jupyter Python Packages and Issues

Goal: When using Jupyter how are python packages managed?

- Where are Python Packages Installed During a Jupyter Session?

- Can I pip install?
- Can I Use Conda Within My Notebook?
- Jupyter Sessions
- Issue: Jupyter Doesn’t Start?
- Issue: Jupyter Doesn’t Start: Example

Where are Python Packages Installed During a Jupyter Session?

You can use `pip list -v` and `conda list` to inspect what is in your environment.

Example using the *default* Python kernel:

```
# Cell
import sys
print(sys.version)
3.12.3 | packaged by Anaconda, Inc. | (main, May  6 2024, 19:46:43) [GCC
11.2.0]
```

pip list -v

```
Package                          Version      Location
Installer
------------------------- ----------- ---------------------------------------
------------------- ----------
anyio                          4.2.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
argon2-cffi                    21.3.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
argon2-cffi-bindings           21.2.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
asttokens                      2.0.5
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
async-lru                      2.0.4
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
attrs                          23.1.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
Babel                          2.11.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
beautifulsoup4                 4.12.2
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
bleach                         4.1.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
Brotli                         1.0.9
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
certifi                        2024.2.2
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
cffi                           1.16.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
charset-normalizer             2.0.4
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
```

```
comm                      0.2.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
debugpy                   1.6.7
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
decorator                 5.1.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
defusedxml                0.7.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
executing                 0.8.3
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
fastjsonschema            2.16.2
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
idna                      3.7
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
ipykernel                 6.28.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
ipython                   8.20.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
ipywidgets                8.1.2
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jedi                      0.18.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
Jinja2                    3.1.3
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
json5                     0.9.6
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jsonschema                4.19.2
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jsonschema-specifications 2023.7.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jupyter                   1.0.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jupyter_client            8.6.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jupyter-console           6.6.3
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jupyter_core              5.5.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jupyter-events            0.8.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jupyter-lsp               2.2.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jupyter_server            2.10.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jupyter_server_terminals  0.4.4
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jupyterlab                4.0.11
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jupyterlab-pygments       0.1.2
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jupyterlab_server         2.25.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
jupyterlab-widgets        3.0.10
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
MarkupSafe                2.1.3
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
```

```
matplotlib-inline          0.1.6
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
mistune                    2.0.4
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
nbclient                   0.8.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
nbconvert                  7.10.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
nbformat                   5.9.2
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
nest-asyncio               1.6.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
notebook                   7.0.8
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
notebook_shim              0.2.3
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
numpy                      2.0.0
/home/salexan5/.local/lib/python3.12/site-packages         pip
overrides                  7.4.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
packaging                  23.2
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
pandocfilters              1.5.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
parso                      0.8.3
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
pexpect                    4.8.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
pip                        24.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages
platformdirs               3.10.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
ply                        3.11
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
prometheus-client          0.14.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
prompt-toolkit             3.0.43
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
psutil                     5.9.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
ptyprocess                 0.7.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
pure-eval                  0.2.2
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
pycparser                  2.21
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
Pygments                   2.15.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
PyQt5                      5.15.10
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
PyQt5-sip                  12.13.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
PySocks                    1.7.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
python-dateutil            2.9.0.post0
/home/salexan5/.local/lib/python3.12/site-packages         pip
```

```
python-json-logger        2.0.7
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
pytz                      2024.1
/home/salexan5/.local/lib/python3.12/site-packages         pip
PyYAML                    6.0.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
pyzmq                     25.1.2
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
qtconsole                 5.5.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
QtPy                      2.4.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
referencing               0.30.2
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
requests                  2.31.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
rfc3339-validator         0.1.4
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
rfc3986-validator         0.1.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
rpds-py                   0.10.6
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
Send2Trash                1.8.2
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
setuptools                69.5.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages
sip                       6.7.12
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
six                       1.16.0
/home/salexan5/.local/lib/python3.12/site-packages         pip
sniffio                   1.3.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
soupsieve                 2.5
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
stack-data                0.2.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
terminado                 0.17.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
tinycss2                  1.2.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
tornado                   6.3.3
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
traitlets                 5.7.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
typing_extensions         4.11.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
tzdata                    2024.1
/home/salexan5/.local/lib/python3.12/site-packages         pip
urllib3                   2.2.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
wcwidth                   0.2.5
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
webencodings              0.5.1
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
websocket-client          0.58.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
```

```
wheel                     0.43.0
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages
widgetsnbextension        4.0.10
/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages conda
Note: you may need to restart the kernel to use updated packages.
```

Notice there are two paths:

1. The library path of the conda environment that this default kernel is running from: `/apps/s/jupyterlab/miniconda3/lib/python3.12/site-packages`
2. Your local home library path related to the version of Python within this kernel: `/home/salexan5/.local/lib/python3.12/site-packages`

Can I `pip install`?

Yes. And this will installed into your local home under the appropriate library/version folder.

```
# Before pip install:
[salexan5@mblog2 ~]$ ls .local/lib/python3.12/site-packages/
dateutil  numpy-2.0.0.dist-info  __pycache__                            pytz
six-1.16.0.dist-info  tzdata
numpy     numpy.libs             python_dateutil-2.9.0.post0.dist-info  pytz-
2024.1.dist-info  six.py               tzdata-2024.1.dist-info
# Within Cell:
pip install nltk
```

Cell Result

```
Defaulting to user installation because normal site-packages is not writeable
Collecting nltk
  Using cached nltk-3.8.1-py3-none-any.whl.metadata (2.8 kB)
Collecting click (from nltk)
  Using cached click-8.1.7-py3-none-any.whl.metadata (3.0 kB)
Collecting joblib (from nltk)
  Using cached joblib-1.4.2-py3-none-any.whl.metadata (5.4 kB)
Collecting regex>=2021.8.3 (from nltk)
  Using cached regex-2024.7.24-cp312-cp312-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (40 kB)
Collecting tqdm (from nltk)
  Downloading tqdm-4.66.5-py3-none-any.whl.metadata (57 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
57.6/57.6 kB 1.5 MB/s eta 0:00:00
Using cached nltk-3.8.1-py3-none-any.whl (1.5 MB)
Using cached regex-2024.7.24-cp312-cp312-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (790 kB)
Using cached click-8.1.7-py3-none-any.whl (97 kB)
Using cached joblib-1.4.2-py3-none-any.whl (301 kB)
Downloading tqdm-4.66.5-py3-none-any.whl (78 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
78.4/78.4 kB 2.9 MB/s eta 0:00:00
Installing collected packages: tqdm, regex, joblib, click, nltk
```

```
Successfully installed click-8.1.7 joblib-1.4.2 nltk-3.8.1 regex-2024.7.24
tqdm-4.66.5
Note: you may need to restart the kernel to use updated packages.
# After Installation:
# Notice the new packages installed.
[salexan5@mblog2 ~]$ ls .local/lib/python3.12/site-packages/
click                  joblib-1.4.2.dist-info  numpy-2.0.0.dist-info
pytz                       six-1.16.0.dist-info   tzdata
click-8.1.7.dist-info  nltk                    numpy.libs
pytz-2024.1.dist-info      six.py                 tzdata-2024.1.dist-info
dateutil               nltk-3.8.1.dist-info    __pycache__
regex                      tqdm
joblib                 numpy                   python_dateutil-
2.9.0.post0.dist-info  regex-2024.7.24.dist-info  tqdm-4.66.5.dist-info
```

Remember: Any previously install Python packages under this location will be available to your notebook.

Any installs/updates within the notebook will thus also affect any other Python environments you have that use this location and packages.

This could cause issues with dependencies and versions.

Can I Use Conda Within My Notebook?

Yes and no…

conda list

```
# packages in environment at /apps/s/jupyterlab/miniconda3:
#
# Name                    Version                   Build  Channel
_libgcc_mutex             0.1                        main
_openmp_mutex             5.1                       1_gnu
anyio                     4.2.0           py312h06a4308_0
...
zlib                      1.2.13               h5eee18b_1
zstd                      1.5.5                hc292b87_2
Note: you may need to restart the kernel to use updated packages.
```

conda search nltk

```
Loading channels: done
# Name                       Version           Build  Channel
nltk                           3.2.1          py27_0  conda-forge
nltk                           3.2.1          py34_0  conda-forge
...
nltk                           3.8.1  py39h06a4308_0  pkgs/main
nltk                           3.8.1    pyhd8ed1ab_0  conda-forge
Note: you may need to restart the kernel to use updated packages.
```

conda install nltk

```
Retrieving notices: ...working... done
```

```
Channels:
 - conda-forge
 - defaults
Platform: linux-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##
  environment location: /apps/s/jupyterlab/miniconda3
  added / updated specs:
    - nltk

The following packages will be downloaded:
    package                    |            build
    ---------------------------|-----------------
    certifi-2024.7.4           |     pyhd8ed1ab_0         156 KB  conda-forge
    nltk-3.8.1                 |     pyhd8ed1ab_0         1.0 MB  conda-forge
    regex-2023.10.3            |  py312h5eee18b_0         386 KB
    tqdm-4.66.5                |     pyhd8ed1ab_0          87 KB  conda-forge
    ------------------------------------------------------------
                                           Total:         1.6 MB

The following NEW packages will be INSTALLED:
  click              conda-forge/noarch::click-8.1.7-unix_pyh707e725_0
  colorama           conda-forge/noarch::colorama-0.4.6-pyhd8ed1ab_0
  joblib             conda-forge/noarch::joblib-1.4.2-pyhd8ed1ab_0
  nltk               conda-forge/noarch::nltk-3.8.1-pyhd8ed1ab_0
  regex              pkgs/main/linux-64::regex-2023.10.3-py312h5eee18b_0
  tqdm               conda-forge/noarch::tqdm-4.66.5-pyhd8ed1ab_0

The following packages will be UPDATED:
  ca-certificates    pkgs/main::ca-certificates-2024.3.11-~ --> conda-
forge::ca-certificates-2024.7.4-hbcca054_0
  certifi            pkgs/main/linux-64::certifi-2024.2.2-~ --> conda-
forge/noarch::certifi-2024.7.4-pyhd8ed1ab_0
  libgcc-ng          pkgs/main::libgcc-ng-11.2.0-h1234567_1 --> conda-
forge::libgcc-ng-14.1.0-h77fa898_0
  libgomp              pkgs/main::libgomp-11.2.0-h1234567_1 --> conda-
forge::libgomp-14.1.0-h77fa898_0
  openssl              pkgs/main::openssl-3.0.13-h7f8727e_1 --> conda-
forge::openssl-3.3.1-h4bc722e_2

The following packages will be SUPERSEDED by a higher-priority channel:
  _libgcc_mutex           pkgs/main::_libgcc_mutex-0.1-main --> conda-
forge::_libgcc_mutex-0.1-conda_forge
  _openmp_mutex          pkgs/main::_openmp_mutex-5.1-1_gnu --> conda-
forge::_openmp_mutex-4.5-2_gnu
...
Preparing transaction: done
Verifying transaction: failed

EnvironmentNotWritableError: The current user does not have write permissions
to the target environment.
  environment location: /apps/s/jupyterlab/miniconda3
  uid: 10275911
  gid: 10275911
```

```
Note: you may need to restart the kernel to use updated packages.
```

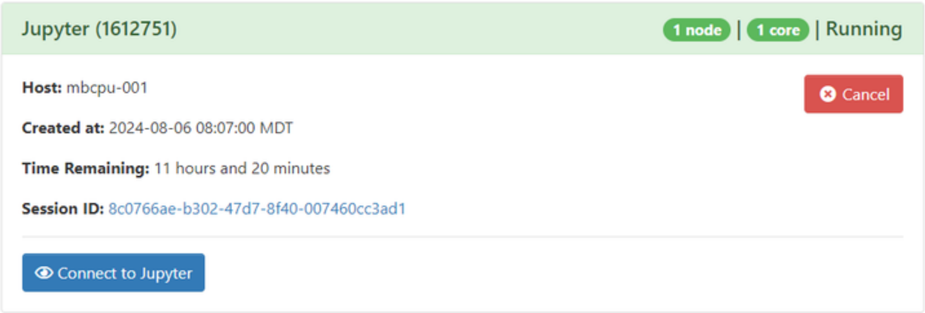
You can use `conda` commands within cells to inspect and search.

You will not have permissions to `conda install` since you do not own this folder, so do not have permissions to install/update anything within it.

But, you should have permissions if *you* have created the kernel, own it, and thus have permissions.

Jupyter Sessions

When you start a Jupyter session via OnDemand, you get a **Session ID**, which also creates a corresponding folder within your home:



Jupyter (1612751) 1 node | 1 core | Running

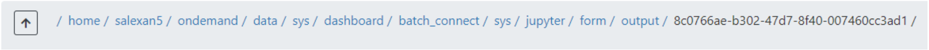
Host: mbcpu-001

Created at: 2024-08-06 08:07:00 MDT

Time Remaining: 11 hours and 20 minutes

Session ID: 8c0766ae-b302-47d7-8f40-007460cc3ad1

Clicking on the Session ID link will open up a new tab:



/ home / salexan5 / ondemand / data / sys / dashboard / batch_connect / sys / jupyter / form / output / 8c0766ae-b302-47d7-8f40-007460cc3ad1 /

Or, you can navigate to it via the command-line:

```
[salexan5@mblog2 ~]$ ls
ondemand/data/sys/dashboard/batch_connect/sys/jupyter/form/output/8c0766ae-
b302-47d7-8f40-007460cc3ad1/
after.sh  before.sh  config.py  connection.yml  job_script_content.sh
job_script_options.json  output.log  script.sh  user_defined_context.json
```

You can view the `output.log` file under this folder to track the session.

Issue: Jupyter Doesn’t Start?

Sometimes when you try and start a Jupyter session it immediately fails to open.

This is typically caused by something that you have done/run/installed that has installed a python package under the library/version folder that is causing a dependency issue with a Python package installed under the kernel..

If you do not have the Session ID, then list the child session folders, in date order, under `~/ondemand/data/sys/dashboard/batch_connect/sys/jupyter/form/output/` and look at the `output.log` file under the last child folder created:

```
[salexan5@mblog2 ~]$ ls -altr
~/ondemand/data/sys/dashboard/batch_connect/sys/jupyter/form/output/
...
drwxr-xr-x 2 salexan5 salexan5 4096 Aug  6 08:07 8c0766ae-b302-47d7-8f40-
007460cc3ad1
```

Issue: Jupyter Doesn’t Start: Example

This example is from an older cluster/version of OnDemand, but the principle still applies.

The `fakeusername` user had the following error:

Full Trace:

```
[fakeusername@blog2 d248fc5c-162c-42c6-b96d-e2ea46cbf3c1]$ cat output.log
...
TIMING - Starting jupyter at: Fri Jul 26 13:20:12 MDT 2024
Traceback (most recent call last):
  File "/apps/s/jupyterlab/miniconda3/bin/jupyter-lab", line 6, in <module>
    from jupyterlab.labapp import main
  File "/apps/s/jupyterlab/miniconda3/lib/python3.9/site-
packages/jupyterlab/__init__.py", line 7, in <module>
    from .labapp import LabApp
  File "/apps/s/jupyterlab/miniconda3/lib/python3.9/site-
packages/jupyterlab/labapp.py", line 18, in <module>
    from jupyterlab_server import WORKSPACE_EXTENSION, LabServerApp, slugify
  File "/apps/s/jupyterlab/miniconda3/lib/python3.9/site-
packages/jupyterlab_server/__init__.py", line 4, in <module>
    from .app import LabServerApp
  File "/apps/s/jupyterlab/miniconda3/lib/python3.9/site-
packages/jupyterlab_server/app.py", line 11, in <module>
    from .handlers import LabConfig, add_handlers
  File "/apps/s/jupyterlab/miniconda3/lib/python3.9/site-
packages/jupyterlab_server/handlers.py", line 19, in <module>
    from .translations_handler import TranslationsHandler
  File "/apps/s/jupyterlab/miniconda3/lib/python3.9/site-
packages/jupyterlab_server/translations_handler.py", line 16, in <module>
    from .translation_utils import get_language_pack, get_language_packs,
is_valid_locale, translator
  File "/apps/s/jupyterlab/miniconda3/lib/python3.9/site-
packages/jupyterlab_server/translation_utils.py", line 15, in <module>
```

```
    import pkg_resources
  File "/home/fakeusername/.local/lib/python3.9/site-
packages/pkg_resources/__init__.py", line 96, in <module>
    from jaraco.text import (
  File "/home/fakeusername/.local/lib/python3.9/site-
packages/setuptools/_vendor/jaraco/text/__init__.py", line 12, in <module>
    from jaraco.context import ExceptionTrap
  File "/home/fakeusername/.local/lib/python3.9/site-
packages/setuptools/_vendor/jaraco/context.py", line 17, in <module>
    from backports import tarfile
ImportError: cannot import name 'tarfile' from 'backports'
(/apps/s/jupyterlab/miniconda3/lib/python3.9/site-
packages/backports/__init__.py)
Timed out waiting for Jupyter Notebook server to open port 40276!
TIMING - Wait ended at: Fri Jul 26 13:21:19 MDT 2024
Cleaning up...
```

Two things to notice.

1. The `ImportError` caused by a dependency issue: `ImportError: cannot import name 'tarfile' from 'backports'`
2. The location of the package, that has been installed, where this is originating from: `/home/fakeusername/.local/lib/python3.9/site-packages/pkg_resources`

Unfortunately, the simplest way to resolve is to remove this folder.

How this effects your environment/pipeline you will have to figure out.

Unfortunately, the more complicated you try and make a single environment, the more likely this is too occur.

Suggestion: Maybe consider creating a separate Conda environment/kernel.

Conda and Pip Environments and Reproducibility

Goal: Introduce ideas and practices to assist in managing the reproducibility of environments created using Conda and Pip.

- Create an Example Environment
- What is Installed? conda list
- What is Installed? pip list -v
- What is Installed? conda env export

Create an Example Environment

To help demonstrate the various concepts, lets create the following environment:

```
[salexan5@mblog2 ~] cd /project/arcc/salexan5/conda
[salexan5@mblog2 conda]$ module load miniconda3/24.3.0
[salexan5@mblog2 conda]$ conda create -p py_conda_pip_env python=3.12
[salexan5@mblog2 conda]$ conda activate
/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env
(/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env)
[salexan5@mblog2 conda]$ python --version
Python 3.12.4
(/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env)
[salexan5@mblog2 conda]$ which python
/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env/bin/python
# Create a self contined environment i.e don't install anything into the
local home folder.
(/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env)
[salexan5@mblog2 conda]$ export PYTHONUSERBASE=$CONDA_PREFIX
# Use conda to install: numpy
(/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env)
[salexan5@mblog2 conda]$ conda install numpy
# Use pip to install: nltk
# has dependencies: click, joblib, regex and tqdm
(/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env)
[salexan5@mblog2 conda]$ pip install nltk
```

What is Installed? `conda list`

The conda list command will provide details of all packages as well as conda environment libraries:

```
(/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env)
[salexan5@mblog2 conda]$ conda list
...
click                     8.1.7                    pypi_0    pypi
joblib                    1.4.2                    pypi_0    pypi
nltk                      3.8.1                    pypi_0    pypi
numpy                     2.0.1           py312h1103770_0    conda-forge
regex                     2024.7.24                pypi_0    pypi
tqdm                      4.66.4                   pypi_0    pypi
```

full conda list

```
(/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env)
[salexan5@mblog2 conda]$ conda list
# packages in environment at
/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env:
#
# Name                    Version                   Build  Channel
_libgcc_mutex             0.1                 conda_forge    conda-forge
_openmp_mutex             4.5                       2_gnu    conda-forge
```

```
bzip2                     1.0.8                h4bc722e_7    conda-forge
ca-certificates           2024.7.4             hbcca054_0    conda-forge
click                     8.1.7                    pypi_0    pypi
joblib                    1.4.2                    pypi_0    pypi
ld_impl_linux-64          2.40                 hf3520f5_7    conda-forge
libblas                   3.9.0           23_linux64_openblas    conda-forge
libcblas                  3.9.0           23_linux64_openblas    conda-forge
libexpat                  2.6.2                h59595ed_0    conda-forge
libffi                    3.4.2                h7f98852_5    conda-forge
libgcc-ng                 14.1.0               h77fa898_0    conda-forge
libgfortran-ng            14.1.0               h69a702a_0    conda-forge
libgfortran5              14.1.0               hc5f4f2c_0    conda-forge
libgomp                   14.1.0               h77fa898_0    conda-forge
liblapack                 3.9.0           23_linux64_openblas    conda-forge
libnsl                    2.0.1                hd590300_0    conda-forge
libopenblas               0.3.27          pthreads_hac2b453_1    conda-forge
libsqlite                 3.46.0               hde9e2c9_0    conda-forge
libstdcxx-ng              14.1.0               hc0a3c3a_0    conda-forge
libuuid                   2.38.1               h0b41bf4_0    conda-forge
libxcrypt                 4.4.36               hd590300_1    conda-forge
libzlib                   1.3.1                h4ab18f5_1    conda-forge
ncurses                   6.5                  h59595ed_0    conda-forge
nltk                      3.8.1                    pypi_0    pypi
numpy                     2.0.1           py312h1103770_0    conda-forge
openssl                   3.3.1                h4bc722e_2    conda-forge
pip                       24.0               pyhd8ed1ab_0    conda-forge
python                    3.12.4          h194c7f8_0_cpython    conda-forge
python_abi                3.12                    4_cp312    conda-forge
readline                  8.2                  h8228510_1    conda-forge
regex                     2024.7.24                pypi_0    pypi
setuptools                71.0.4             pyhd8ed1ab_0    conda-forge
tk                        8.6.13          noxft_h4845f30_101    conda-forge
tqdm                      4.66.4                   pypi_0    pypi
tzdata                    2024a                h0c530f3_0    conda-forge
wheel                     0.43.0             pyhd8ed1ab_1    conda-forge
xz                        5.2.6                h166bdaf_0    conda-forge
```

What is Installed? `pip list -v`

The pip list command will only list the explicitly python packages.

Look at the Installer column to identify what was installed via `conda install`, `pip install`, by default.

```
(/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env)
[salexan5@mblog2 conda]$ pip list -v
Package    Version   Location
Installer
---------- --------- -------------------------------------------------------
-------------------------------- ---------
```

```
click      8.1.7
/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env/lib/python3.12/s
ite-packages pip
joblib     1.4.2
/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env/lib/python3.12/s
ite-packages pip
nltk       3.8.1
/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env/lib/python3.12/s
ite-packages pip
numpy      2.0.1
/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env/lib/python3.12/s
ite-packages conda
pip        24.0
/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env/lib/python3.12/s
ite-packages
regex      2024.7.24
/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env/lib/python3.12/s
ite-packages pip
setuptools 71.0.4
/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env/lib/python3.12/s
ite-packages
tqdm       4.66.4
/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env/lib/python3.12/s
ite-packages pip
wheel      0.43.0
/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env/lib/python3.12/s
ite-packages
```

Check under the python related `lib` folder.

List what's in: /project/arcc/salexan5/conda/py_conda_pip_env/lib/python3.12/site-packages/

```
(/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env)
[salexan5@mblog2 conda]$ ls
/project/arcc/salexan5/conda/py_conda_pip_env/lib/python3.12/site-packages/
click                     joblib                  numpy
pkg_resources              setuptools                        wheel
click-8.1.7.dist-info     joblib-1.4.2.dist-info  numpy-2.0.1.dist-info
README.txt                 setuptools-71.0.4-py3.12.egg-info  wheel-
0.43.0.dist-info
_distutils_hack           nltk                    pip
regex                      tqdm
distutils-precedence.pth  nltk-3.8.1.dist-info    pip-24.0-py3.12.egg-info
regex-2024.7.24.dist-info  tqdm-4.66.4.dist-info
```

What is Installed? `conda env export`

The output of the `conda env export` command you can use to recreate.

Identify what was install via `conda install` vs `pip install`:

```
(/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env)
[salexan5@mblog2 conda]$ conda env export > env.yml
```

```
(/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env)
[salexan5@mblog2 conda]$ cat env.yml
name: /cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env
...
dependencies:
...
  - numpy=2.0.1=py312h1103770_0
...
  - python=3.12.4=h194c7f8_0_cpython
...
  - pip:
      - click==8.1.7
      - joblib==1.4.2
      - nltk==3.8.1
      - regex==2024.7.24
      - tqdm==4.66.4
prefix: /cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env
```

full output:

```
(/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env)
[salexan5@mblog2 conda]$ conda env export > env.yml
(/cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env)
[salexan5@mblog2 conda]$ cat env.yml
name: /cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env
channels:
  - conda-forge
  - defaults
dependencies:
  - _libgcc_mutex=0.1=conda_forge
  - _openmp_mutex=4.5=2_gnu
  - bzip2=1.0.8=h4bc722e_7
  - ca-certificates=2024.7.4=hbcca054_0
  - ld_impl_linux-64=2.40=hf3520f5_7
  - libblas=3.9.0=23_linux64_openblas
  - libcblas=3.9.0=23_linux64_openblas
  - libexpat=2.6.2=h59595ed_0
  - libffi=3.4.2=h7f98852_5
  - libgcc-ng=14.1.0=h77fa898_0
  - libgfortran-ng=14.1.0=h69a702a_0
  - libgfortran5=14.1.0=hc5f4f2c_0
  - libgomp=14.1.0=h77fa898_0
  - liblapack=3.9.0=23_linux64_openblas
  - libnsl=2.0.1=hd590300_0
  - libopenblas=0.3.27=pthreads_hac2b453_1
  - libsqlite=3.46.0=hde9e2c9_0
  - libstdcxx-ng=14.1.0=hc0a3c3a_0
  - libuuid=2.38.1=h0b41bf4_0
  - libxcrypt=4.4.36=hd590300_1
  - libzlib=1.3.1=h4ab18f5_1
  - ncurses=6.5=h59595ed_0
  - numpy=2.0.1=py312h1103770_0
  - openssl=3.3.1=h4bc722e_2
  - pip=24.0=pyhd8ed1ab_0
  - python=3.12.4=h194c7f8_0_cpython
  - python_abi=3.12=4_cp312
  - readline=8.2=h8228510_1
  - setuptools=71.0.4=pyhd8ed1ab_0
  - tk=8.6.13=noxft_h4845f30_101
  - tzdata=2024a=h0c530f3_0
  - wheel=0.43.0=pyhd8ed1ab_1
  - xz=5.2.6=h166bdaf_0
  - pip:
      - click==8.1.7
      - joblib==1.4.2
      - nltk==3.8.1
      - regex==2024.7.24
      - tqdm==4.66.4
prefix: /cluster/medbow/project/arcc/salexan5/conda/py_conda_pip_env
```

Python, Conda and Pip: Suggested Best Practices

Goal: Suggest best practices and summarize workshop.

- Suggested Best Practices
- Summary

Suggested Best Practices

- If not using Miniconda, use a module version of Python. Do not rely on a the general System/compute node version.
- Install packages within the Conda environment. Do not rely or try and manage packages installed under `~/./local/lib/pythonX.Y`.
- Be aware if relying on what is under `~/./local/lib/pythonX.Y` then updates to packages could effect other Conda environments that use them.
- After creating/activating an environment to setup/update, set the `PYTHONUSERBASE` environment variable to force to only using what is installed directly within the Conda environment.
- For individual research tasks, create self-contained Conda environments to enable reproducibility and sharing.

Summary

Looked at:

- How using different versions of Python pip installs into different locations.

- How Conda environments and pip installs interplay together and using the `PYTHONUSERBASE` environment variable to force installs into the Conda environment.
- A general process for creating a shared Conda environment with pip installs within a project.
- The basics on creating a module file to expose a Conda environment for general use across a project.

Use the following link to provide feedback on this training: https://forms.gle/7otURqe8uyvxzXiTA or use the QR code below.