IDIA Inter-University Institute for Data Intensive Astronomy

Ilifu Online Training

Session 2: Software environments

9 September 2025

Jeremy Smith, ilifu Operations Manager
University of Cape Town

University of Cape Town · iYunivesithi yaseKapa · Universiteit van Kaapstad

UNIVERSITY of the WESTERN CAPE

NRF National Research Foundation | SARAO South African Radio Astronomy Observatory

Topics

- Software environments
  - Python virtual environments
  - Using R and RStudio
- Advanced Slurm usage
  - Interactive jobs in Slurm
  - Advanced Slurm commands

Getting help

- Support contact

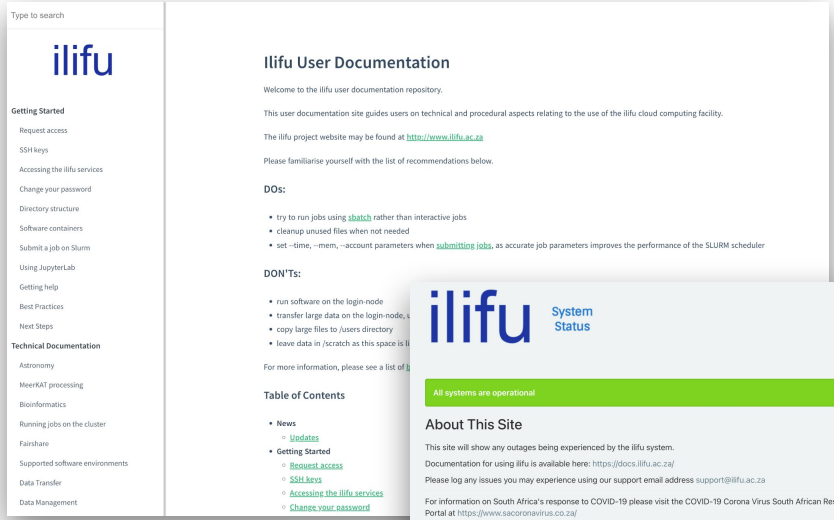
  support@ilifu.ac.za
- User documentation

  http://docs.ilifu.ac.za
- Ilifu System Status

  https://status.ilifu.ac.za
- Training videos

  https://www.ilifu.ac.za/latest-training
- Training tutorials

  https://github.com/ilifu/ilifu_user_training



Python Virtual Environments

virtualenv

- Availability anywhere
- https://virtualenv.pypa.io/en/latest/
- Isolated Python environment
- Less risk of conflicts occurring with pip install --user
- Similar to venv (python -m venv)
- Can customize which os python is used: python2.7, python3+
- Installable packages limited by os libraries

Python Virtual Environments

```
module load python/3.12.3*
virtualenv --help
virtualenv /path/to/virtual_environment
--python                    The Python interpreter to use
--system-site-packages      Gives the virtual environment
                            access to the global
                            site-packages
```

* Use a module to create a virtual environment, rather than system Python, as a change to the OS can break venvs created with the system Python.

Python Virtual Environments

Create a new Python virtual environment:

```
virtualenv /path/to/virtual/environment
```

Example:

```
$ virtualenv venv/tutenv
$ source venv/tutenv/bin/activate        #to enter
(tutenv)$ which python
   /…/venv/tutenv/bin/python
(tutenv)$ python -m pip install scikit-learn
(tutenv)$ deactivate                     #to exit
```

Python Virtual Environments

Python virtualenv as a Jupyter kernel

Once the virtual environment is active:

```
python -m pip install ipykernel
ipython kernel install --name "<kernel_name>" --user
```

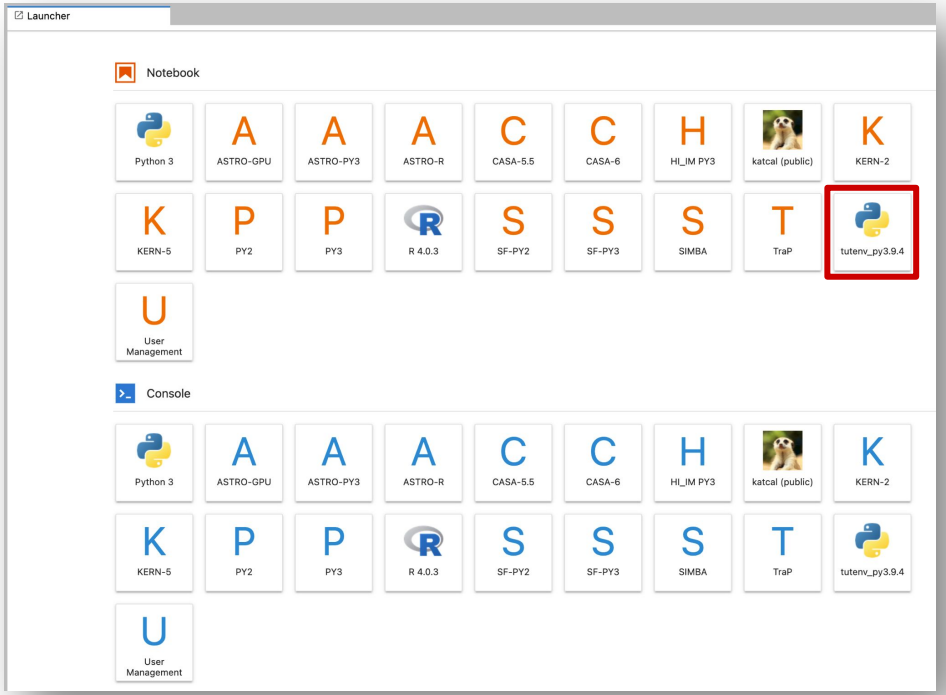
Example:

```
source venv/tutenv/bin/activate
python -m pip install ipykernel
ipython kernel install --name "tutenv_py3.9.4" --user
Installed kernelspec jupkernel in ~/.local/share/jupyter/kernels/tutenv_py3.9.4
```

Creates the kernel.json file at:

```
~/.local/share/jupyter/kernels/tutenv_py3.9.4/kernel.json
```

Python Virtual Environments



R and RStudio with slurm

https://docs.ilifu.ac.za/#/tech_docs/software_environments?id=running-rstudio-server

When logged in via ssh:

```
jeremy@slurm-login:~$ srun --nodes=1 --tasks=1 --mem=8g --time
08:00:00 --job-name="rstudio test" --pty bash
jeremy@compute-001:~$ module add R/RStudio2023.06.1-524-R4.4.1
jeremy@compute-001:~$ rstudio
The environment variable RSTUDIO_PASSWORD was not set, so your
password has been chosen for you. It's: ********************
Running rserver on port 40739
To connect to this server run this on your local machine:
    ssh -A jeremy@compute-001 -o "ProxyCommand=ssh
jeremy@slurm.ilifu.ac.za nc compute-001 22" -L8081:localhost:40739
then visit http://localhost:8081 in your browser and use the username
"jeremy" to login with the password "********************"
(You may need to choose a different port (other than 8081), so
remember to change this in both the ssh and browser)
```

R and Studio with slurm

https://docs.ilifu.ac.za/#/tech_docs/software_environments?id=running-rstudio-server

On your local machine:

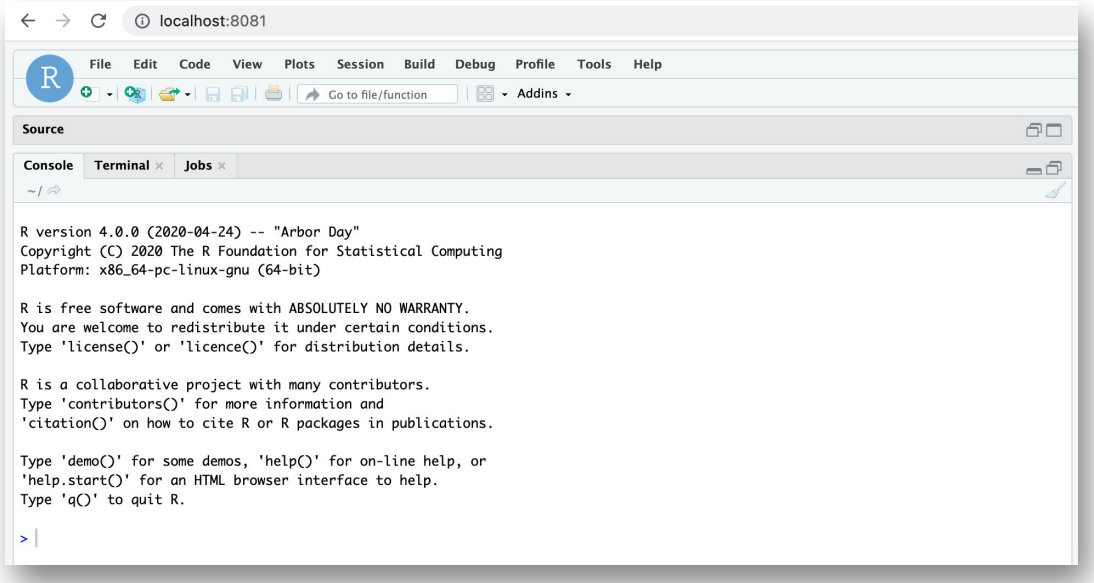
```
jeremy:~$ ssh -A jeremy@compute-001 -o "ProxyCommand=ssh
jeremy@slurm.ilifu.ac.za nc compute-001 22" -L8081:localhost:40739
```

Go to: http://localhost:8081 in your local browser

R and Studio with slurm

https://docs.ilifu.ac.za/#/tech_docs/software_environments?id=running-rstudio-server

Now you can access RStudio through you web browser:



Software Environment summary

virtual environment

- Good for prototyping and rapid development
- User created and managed
- Can be used by a group but needs to be in appropriate folder
- Limited by available os libraries

Modules

- Variety of languages, bioinformatics and utility software
- Useful for software that doesn't have a lot of dependencies
- Some modules execute containers more conveniently

Containers

- Best for reproducibility and sharing
- Best for software that requires libraries/dependencies
- Can be used by anyone with the path

Singularity Registry - container repository