HPC.NRW

NATIONALES HOCHLEISTUNGS RECHNEN

EDIH Rheinland

WEST AI KI-Servicezentrum

# PPCES: Machine and Deep Learning

## Execution Options for ML / DL software

DAS KOMPETENZNETZWERK FÜR HOCHLEISTUNGSRECHNEN.

# Running ML / DL Software (Mostly Python based)

HPC.NRW

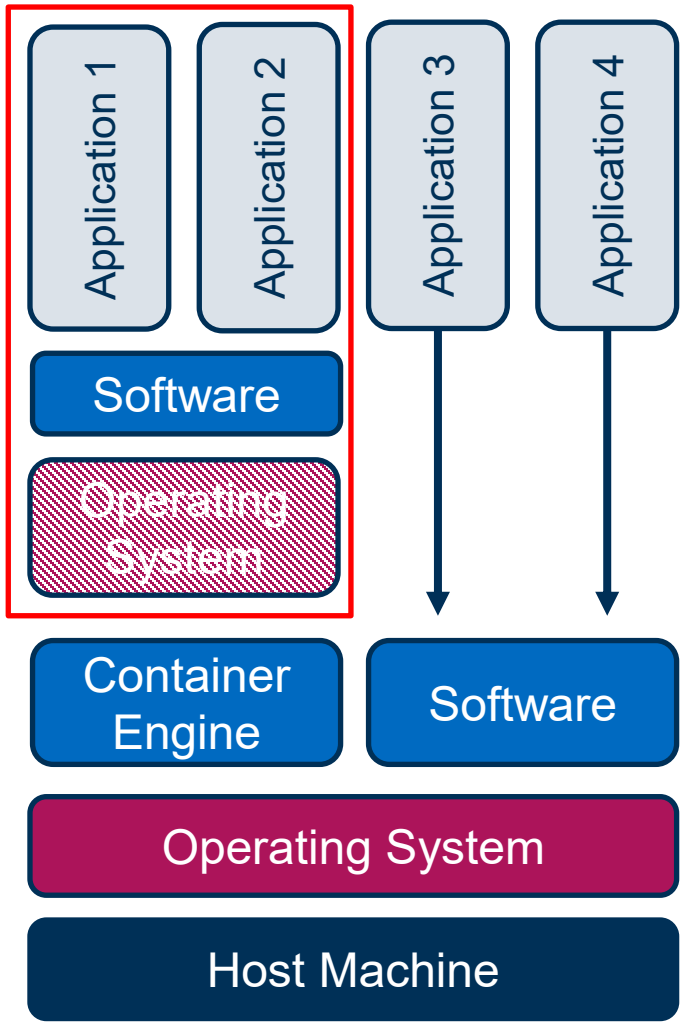
## Approach 1: Virtual Environments

- Solutions available
  - Python virtualenv (pip / uv pip)
  - Conda (conda forge)
- Can also be used locally on your desktop
- Highly flexible (especially during prototyping)
  - You can install and manage your package according to your needs
- Every user has its own installations
  - GBs and 35k+ files per environment. High pressure on shared cluster file system / slow
  - Less reproducibility when working in teams Packages may differ in versions

## Approach 2: Containers (recommended)

- Virtualized OS encapsulating software with all dependencies to make it available on other machines
- Predefined containers available for certain software / frameworks
- Can be used by many users
  - Reproducibility (development + production)
  - Less pressure on file system (only 1 copy and appears as single file)
- Once build they can't be modified
  - How to handle missing packages?

# More Information about Containers




- The container engine (e.g., Docker or Apptainer / Singularity), virtualizes parts of the operating system (process space, user namespace, network namespace)
- Containerized and native applications share the same host OS kernel. You can build it on one machine and use it somewhere else
- Negligible performance overhead when using containers

# Available Containers on CLAIX

HPC.NRW

- **Question:** What containers are available on CLAIX?

```
# show available software on the cluster
module avail

# available containers on the cluster
JupyterDataScienceNotebook/7.5.5    PyTorch/nvcr-24.01-py3         TensorFlow/nvcr-24.01-tf2-py3
                                    PyTorch/nvcr-25.02-py3 (D)     TensorFlow/nvcr-25.02-tf2-py3 (D)
```

- Standard Apptainer containers for most common frameworks
  - Regular updates of container versions
  - Based on Docker images from DockerHub or NVIDIA
- Support for user containers
  - Build and run your own containers
  - Possibility to convert Docker images to Apptainer images
  - Requests for standard containers → IT service desk (mailto:servicedesk@itc.rwth-aachen.de)

APPTAINER

Documentation

# Containers – Handling Missing Packages

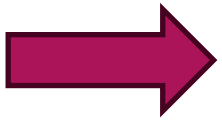
HPC.NRW

- **Reminder:** Containers are read-only after they are built
- **Question:** What happens if packages are missing in container?

→ Review Guide

- Sample Code:

```
#!/usr/bin/env python3
import matplotlib.pyplot as plt
import numpy as np

# calculation or other workload
t = np.arange(0.0, 2.0, 0.01)
s = 1 + np.sin(2 * np.pi * t)

# visually plot results
fig, ax = plt.subplots()
ax.plot(t, s)
ax.grid()
fig.savefig("testplot.png")
print("I'm done!")
```



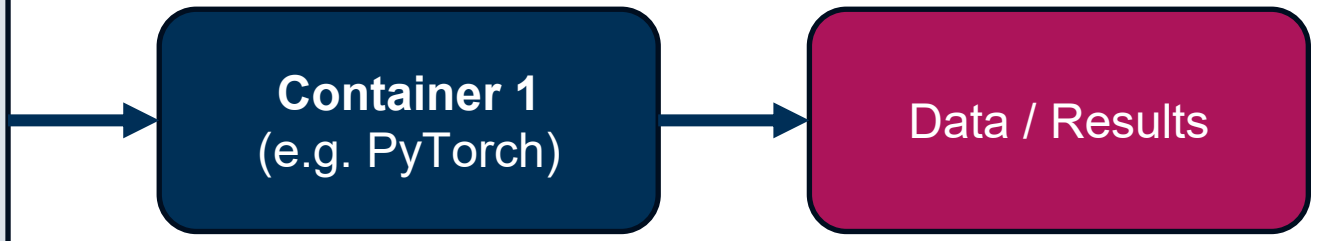
```
# loading the tensorflow container
module load TensorFlow/nvcr-24.01-tf2-py3

# run a shell in the container
apptainer shell -e ${TENSORFLOW_IMAGE}

# execute the sample in the container
Apptainer> python3 sample.py

Traceback (most recent call last):
  File "sample.py", line 3, in <module>
    import matplotlib.pyplot as plt
ModuleNotFoundError: No module named 'matplotlib'
```

# Containers – Handling Missing Packages

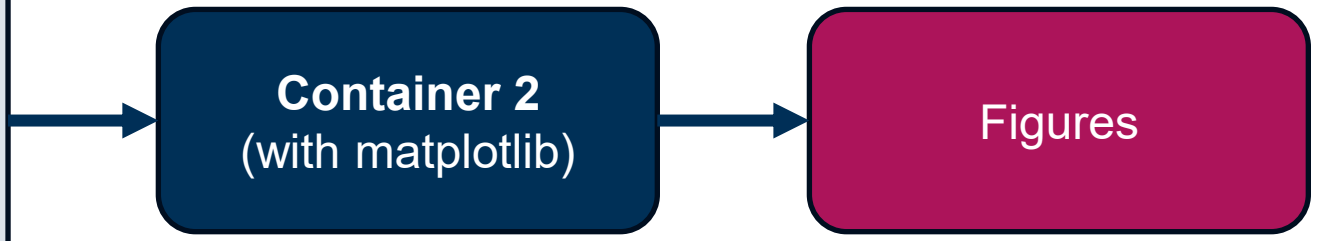
HPC.NRW

– **Solution 1:** Clear separation of concerns. Different containers for different tasks

```python
#!/usr/bin/env python3
import numpy as np

# calculation or other workload (e.g. PyTorch)
t = np.arange(0.0, 2.0, 0.01)
s = 1 + np.sin(2 * np.pi * t)

# Then: write to disk
```




```python
#!/usr/bin/env python3
import matplotlib.pyplot as plt

# Here: read / parse data

# visually plot the results
fig, ax = plt.subplots()
ax.plot(t, s)
ax.grid()
fig.savefig("testplot.png")
print("I'm done!")
```



# Containers – Handling Missing Packages

HPC.NRW

- **Solution 2:** Make additional packages available to container (User Space Installation)
- How does that work?
  - Install additional required packages in user space

```
pip install --user matplotlib
```

  - Saved in `$HOME/.local/lib/python<VER>/site-packages`

**Note:** Can lead to inconsistencies e.g., if Python versions differ or used from different containers with different depenencies

```
# loading the tensorflow container
module load TensorFlow/nvcr-24.01-tf2-py3

# run a shell in the container
apptainer shell -e ${TENSORFLOW_IMAGE}

# install missing package in users home directory
pip install --user matplotlib

# execute the sample in the container
Apptainer> python3 sample.py
I'm done!
```

# Containers – Handling Missing Packages

HPC.NRW

- **Solution 3:** Make additional packages available to container (Storage Overlay)
- **Remember:** Typically, containers (and inner file system) are immutable after build
  - **Overlay:** Possibility to add / extend files in a container (e.g. Python packages)
  - Clean: Creates separate overlay for each container
  - Efficient: Overlay is a single file avoiding file count quotas
  - Flexible: Multiple overlays possible for different containers and purposes

```
# loading the tensorflow container
module load TensorFlow/nvcr-24.01-tf2-py3

# Create an empty overlay filesystem
apptainer overlay create --size 512 overlay.img

# run a shell in the container and attach the overlay
apptainer shell -e --overlay overlay.img ${TENSORFLOW_IMAGE}

# install missing package inside the container
pip install matplotlib

# execute the sample in the container
Apptainer> python3 sample.py
I'm done!

apptainer exec --overlay overlay.img ${TENSORFLOW_IMAGE} \
python3 sample.py
```

# Backup Slides

# Examples: Virtual Environments (venv)

HPC.NRW

```
# load default python module
module load Python

# make sure virtualenv is installed
python3 -m pip install --user virtualenv

# create a new environment named myenv1 (will be created in current working directory)
python3 -m virtualenv myenv1

# activate that environment. Note: after activation environment name is usually shown
source myenv1/bin/activate

# install packages in that environment
python3 -m pip install python-dev-tools matplotlib
pip3 install python-dev-tools matplotlib

# list installed packages in environment
python3 -m pip list -v
pip list -v

# deactivate environment again
deactivate
```

# Examples: Virtual Environments (conda)

HPC.NRW

- Download and install coda-forge (* Anaconda is not allowed and blocked on our infrastructure)
  - Download and Installation: https://conda-forge.org/download/

```
# create a new environment named myenv1 (conda saves envs at a central location)
conda env create -n myenv1
conda config --set pip_interop_enabled True # allow installing packages with pip
conda env list                              # list existing environments

# activate that environment. Note: after activation environment name is usually shown
conda activate myenv1

# install packages in that environment
conda install python-dev-tools matplotlib
pip install python-dev-tools matplotlib

# list installed packages in environment
conda list

# deactivate again
conda deactivate
```