

JupyterHub

■ JupyterHub on bwUniCluster & DACHS

- JupyterHub is a Web-frontend to Jupyter Notebooks (**Julia, Python & R**) providing *interactive* Python w/ formatting in a Web-Browser (saved as JSON)
- With an account – and having logged in once to create Your home – use:
 - <https://uc3-jupyter.scc.kit.edu> or
 - <https://dachs-jupyter.hs-esslingen.de>
- Login & Select Your resources:

Jupyter@UC3

Föderierte Dienste am KIT

Select your resources

The grayed out fields contain a reasonable preselection of resources.
Other values can be selected in advanced mode.

Number of CPU-cores:
Good availability 1

Number of GPUs: 0

Runtime: 0.5 hour

Partition: cpu_il

Amount of memory: 4GB

JupyterLab-Basemode: jupyter/ai

Auto-Reservation:

Advanced Mode:

Container Mode:

Check

Spawn

Login with other identity providers:

Make sure you can access DACHS via [SSH](#).

Resource Selection

Select your preferred resources:

Number of CPU cores: 1

Memory: 16 GB

Allocate One GPU (48 GB VRAM)?

Runtime: 30 min

Load module: jupyter/ai

Use reservation:

Start

Currently reserved resources for JupyterHub:

Node	CPUs (Free/Total)	Memory GB (Free/Total)	GPU available?
gpu103	48/48	384/384	yes
gpu104	48/48	384/384	yes
gpu132	48/48	384/384	yes
gpu133	48/48	384/384	yes
gpu134	48/48	384/384	yes

A word about resources:

- This submits the `jupyterhub-spawner` on Your behalf
- If You select GPU: **one whole node** for yourself, otherwise shared!

For this Workshop today, we have 20 nodes reserved for now...

- Please be considerate: at Resource selection, check # of free nodes
Remember: *Interactive* use is the *least* efficient usage!
- If You need a number of nodes for a lecture at specific times/dates:
Please write an email to dachs-admin@hs-esslingen.de
- Currently, there's no interactive Jupyter access to multi-GPU nodes

■ First view (without existing Notebook Untitled.ipynb)

The screenshot shows the JupyterHub interface. On the left, there's a sidebar titled "Files" with a "Controlling Tabs" section. Below it, under "Jupyter Extensions, like jupyter-matplotlib", is a "Loaded & Available Modules pane". The main area is the "Launcher" which lists notebooks, a console, and other file types.

Files

Controlling Tabs

Jupyter Extensions, like jupyter-matplotlib

Loaded & Available Modules pane:

The default module jupyter/ai loads:

- CUDA
- Torch & Tensorflow
- Pandas
- SciKit-Learn
- Seaborn and more

For other modules: [Wiki](#).

File Edit View Run Kernel Tabs Settings Help

Filter available modules...

LOADED MODULES + ↻ 🔍

jupyter/ai/2025-08-01

AVAILABLE MODULES (51)

- cae/ansys/2024R2_no_lic...
- cae/openfoam/v2406
- cae/openfoam/v2506
- cae/starccm+/2021.3
- cae/starccm+/2406
- compiler/gnu/11.5
- compiler/gnu/14.2
- compiler/gnu/14.3
- compiler/gnu/15.1
- compiler/llvm/18.1
- compiler/llvm/19.1
- compiler/llvm/20.1
- cs/colmap/3.11.1
- cs/halcon/24.11
- cs/hashcat/6.2.6
- cs/ollama/0.7.0
- cs/ollama/0.9.5
- cs/ollama/0.12.2
- devel/cuda/12.4
- devel/cuda/12.6
- devel/cuda/12.8
- devel/cuda/12.8.1
- devel/gdb/15.2
- devel/miniforge/24.11.0-p...
- devel/miniforge/25.3.1-pyt...
- devel/pocl/7.0
- devel/python/3.11.7-gnu-1...
- devel/python/3.11.7-gnu-1...
- devel/python/3.12.3-gnu-...
- devel/python/3.12.3-gnu-...
- devel/python/3.13.1-gnu-1...
- devel/python/3.13.1-gnu-1...
- devel/python/3.13.1-llvm-1...
- devel/python/3.13.3-gnu-...
- devel/python/3.13.3-gnu-...
- devel/python/3.13.3-llvm-...
- devel/valgrind/3.25.1

Launcher +

Notebook

Python 3 (ipykernel)

Console

Python 3 (ipykernel)

Other

Terminal

Text File

Markdown File

Python File

Show Contextual Help

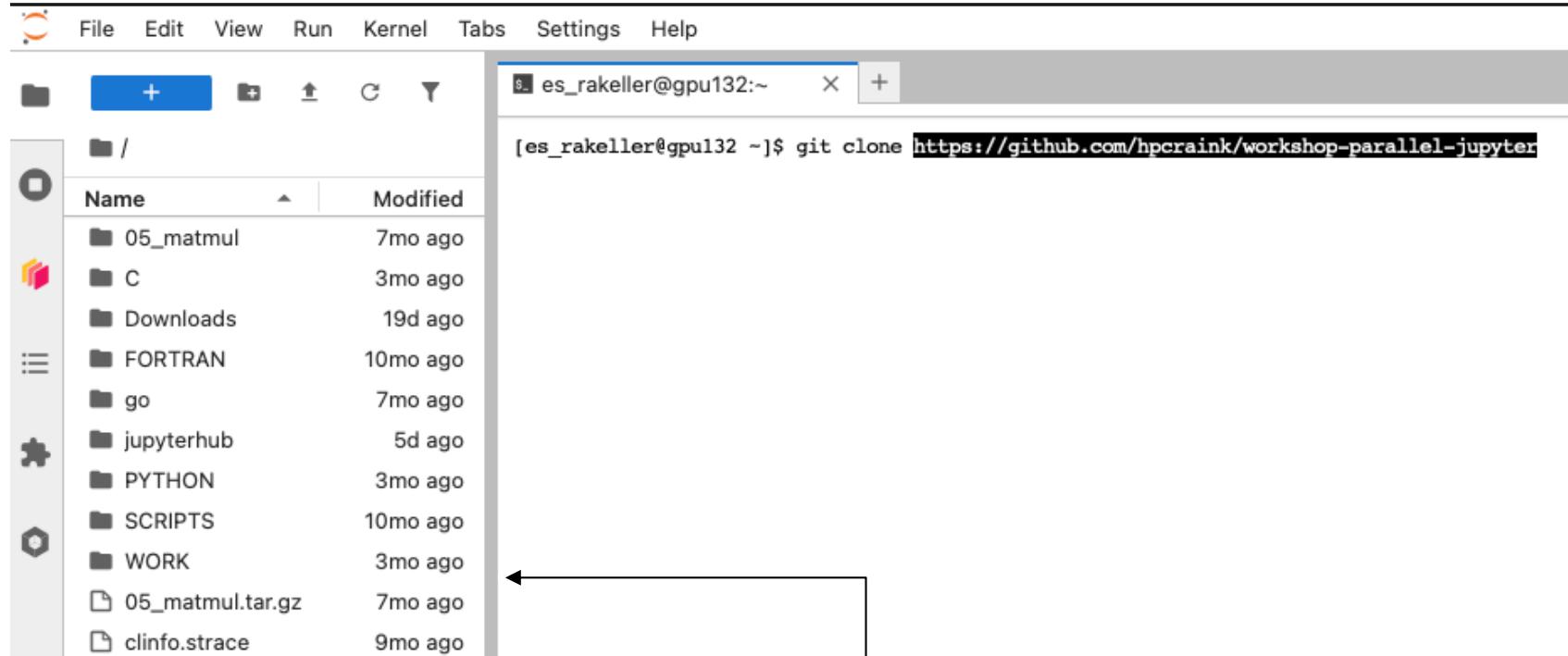
JupyterHub: First steps

2/3

- For example for best practice (on Jupyter, Python, Visualization, ...)

```
git clone https://github.com/hpcraink/workshop-parallel-jupyter
```

- In a new terminal:



The screenshot shows a terminal window with the following content:

```
[es_rakeller@gpu132 ~]$ git clone https://github.com/hpcraink/workshop-parallel-jupyter
```

To the left of the terminal is a file explorer window showing a directory structure:

Name	Modified
05_matmul	7mo ago
C	3mo ago
Downloads	19d ago
FORTRAN	10mo ago
go	7mo ago
jupyterhub	5d ago
PYTHON	3mo ago
SCRIPTS	10mo ago
WORK	3mo ago
05_matmul.tar.gz	7mo ago
clinfo.strace	9mo ago

- It will show up **here** in a few seconds
- Then click on the cloned directory and open `1_Start.ipynb`

JupyterHub: First steps

3/3

- The Interactive Python Notebook File (.ipynb) contains Markup, Python code and executed results and stores visualization to be displayed in the Browser:

Execute **all** statements

Execute current statement

While a statement executes, the block will be marked as [*]:

Once it has finished, it will be numbered consecutively [1]:

The results will be inserted here

File changed, needs saving

The screenshot shows a Jupyter Notebook interface with three tabs: '1_Start.ipynb' (active), '2_Fundamentals.ipynb', and '3_Fundamentals.ipynb'. In the '1_Start.ipynb' tab, a code cell is executing, indicated by the red asterisk marker [*]. The code cell contains:

```
[1]: x = "This is a short text"
y = f"{x} with addendum" # using the statement "f" prior to the stri
print(f'{type(y)}: {y}')
print("{type(y)}: {y}") # without the "f", the curly braces are not
print(f'{type(y)}: {{(y)}}') # double curly braces may be used to "e
```

The output of the cell is:

Line breaks in a str may be inserted using the character string "\n".
Alternatively, you may declare the str starting and ending with triple quote signs -- such a str may contain multiple line breaks in source.

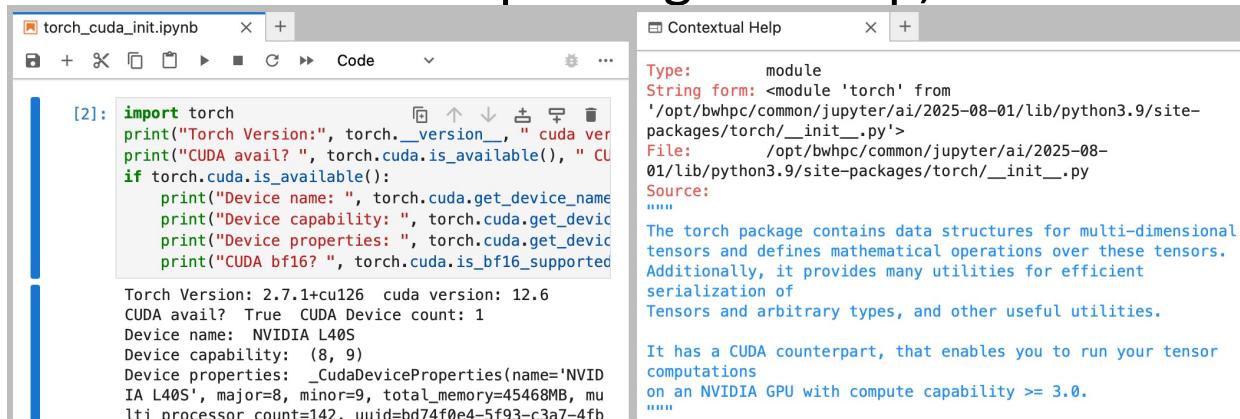
- 2_Fundamentals explains Python programming
- 3_Numpy explains basic Matrix operations using NumPy
- 4_Pandas shows how Pandas works with large Parquet files
- 5_Machine_Learning introduces SciKit Learn with Visualization

JupyterHub: Best practices

- Using a GPU node, check in Terminal the output of nvidia-smi...
- When using PyTorch, check the output of:

```
import torch
print("Torch Version: ", torch.__version__)
print("CUDA version: ", torch.version.cuda)
print("CUDA avail? ", torch.cuda.is_available())
if torch.cuda.is_available():
    print("Device count: ", torch.cuda.device_count())
    print("Device name: ", torch.cuda.get_device_name())
    print("Device capability: ", torch.cuda.get_device_capability())
    print("Device properties: ", torch.cuda.get_device_properties())
    print("CUDA bf16? ", torch.cuda.is_bf16_supported())
```

- The Contextual Help is of great help, when editing (right mouse click)



JupyterHub: on bwUniCluster Please Stop Server

- On bwUniCluster Prior to logging out, end session to free resource:

