

JupyterHub on bwUniCluster & DACHS

JupyterHub on bwUniCluster 3.0 and DACHS

1/2

- 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:

The screenshot shows the Jupyter@UC3 login page. On the left, there's a sidebar with a menu icon and the text 'Jupyter@UC3'. Below it, there are two banners for 'Föderierte Dienste am KIT'. The main content area is titled 'Select your resources' and contains a form for selecting resources. The form includes fields for 'Number of CPU-cores' (set to 1), 'Number of GPUs' (set to 0), 'Runtime' (set to 0.5 hour), 'Partition' (set to 'cpu_il'), 'Amount of memory' (set to 4GB), 'JupyterLab-Basemodule' (set to 'jupyter/ai'), 'Auto-Reservation' (checked), 'Advanced Mode' (unchecked), and 'Container Mode' (unchecked). There is a 'Spawn' button at the bottom right. On the left side of the form, there is a section for 'Login with second factor' which includes a 'Current code' input field and a 'Check' button. Below this, there is a link for 'Login with other identity providers'.

Jupyter@UC3

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: 1

Good availability

Number of GPUs: 0

Runtime: 0.5 hour

Partition: cpu_il

Amount of memory: 4GB

JupyterLab-Basemodule: jupyter/ai

Auto-Reservation: ☒

Advanced Mode: ☐

Container Mode: ☐

Spawn

Login with second factor

To perform the requested action, a second factor is required.

Current code

Check

Login with other identity providers:

The screenshot shows the 'Resource Selection' page. At the top, there is a yellow banner with the text 'Make sure you can access DACHS via SSH.' and a red arrow pointing to the 'SSH' link. Below the banner, the title 'Resource Selection' is displayed. The main content area is titled 'Select your preferred resources:' and contains a form for selecting resources. The form includes fields for 'Number of CPU cores' (set to 1), 'Memory' (set to 16 GB), 'Allocate One GPU (48 GB VRAM)?' (unchecked), 'Runtime' (set to 30 min), 'Load module' (set to 'jupyter/ai'), and 'Use reservation' (checked). There is a 'Start' button at the bottom. Below the form, there is a table titled 'Currently reserved resources for JupyterHub:'.

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

JupyterHub: First steps

1/3

■ First view (without existing Notebook Untitled.ipynb)

Files

Controlling Tabs

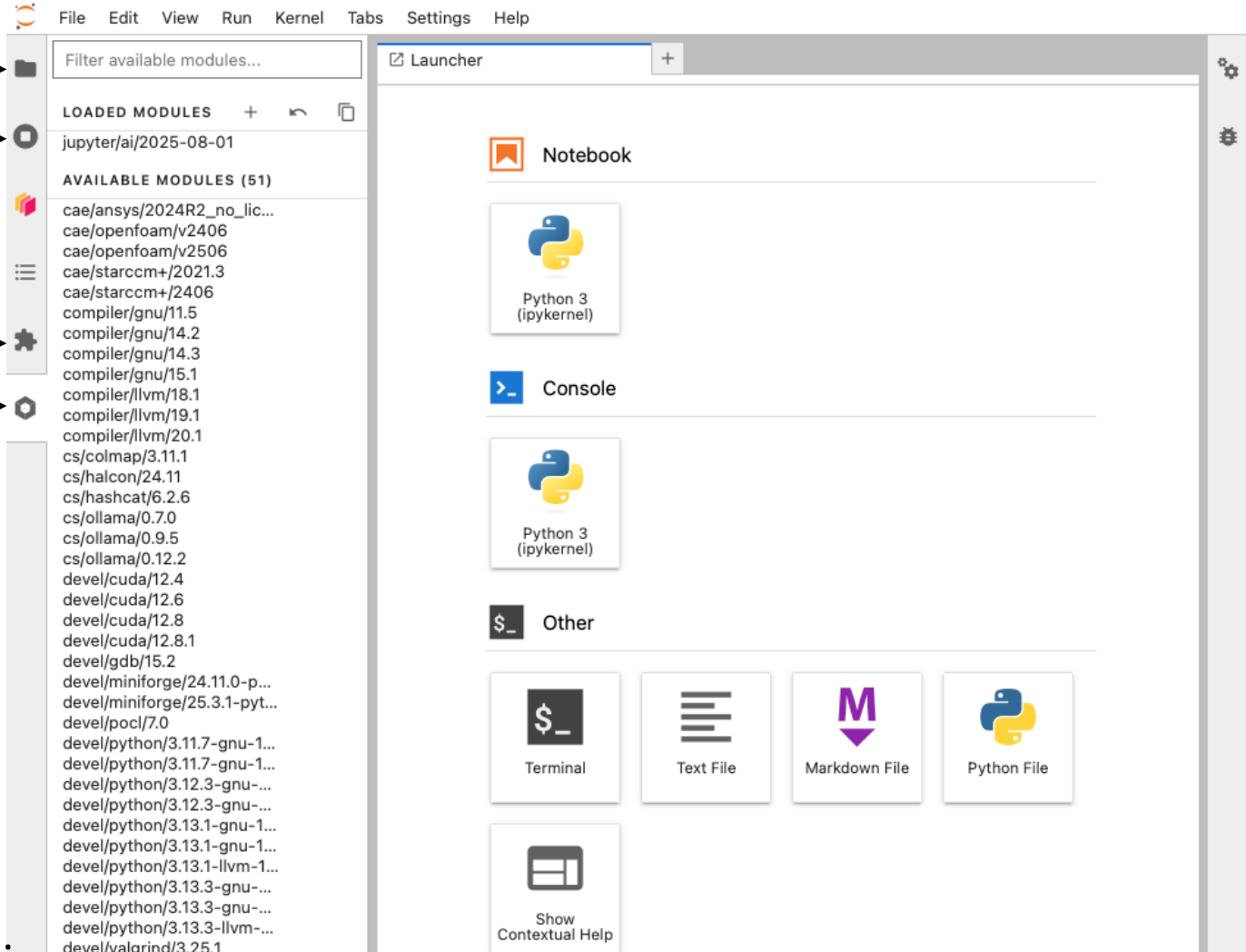
Jupyter Extensions, like `jupyter-matplotlib`

Loaded & Available Modules pane:

The default module `jupyter/ai` loads:

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

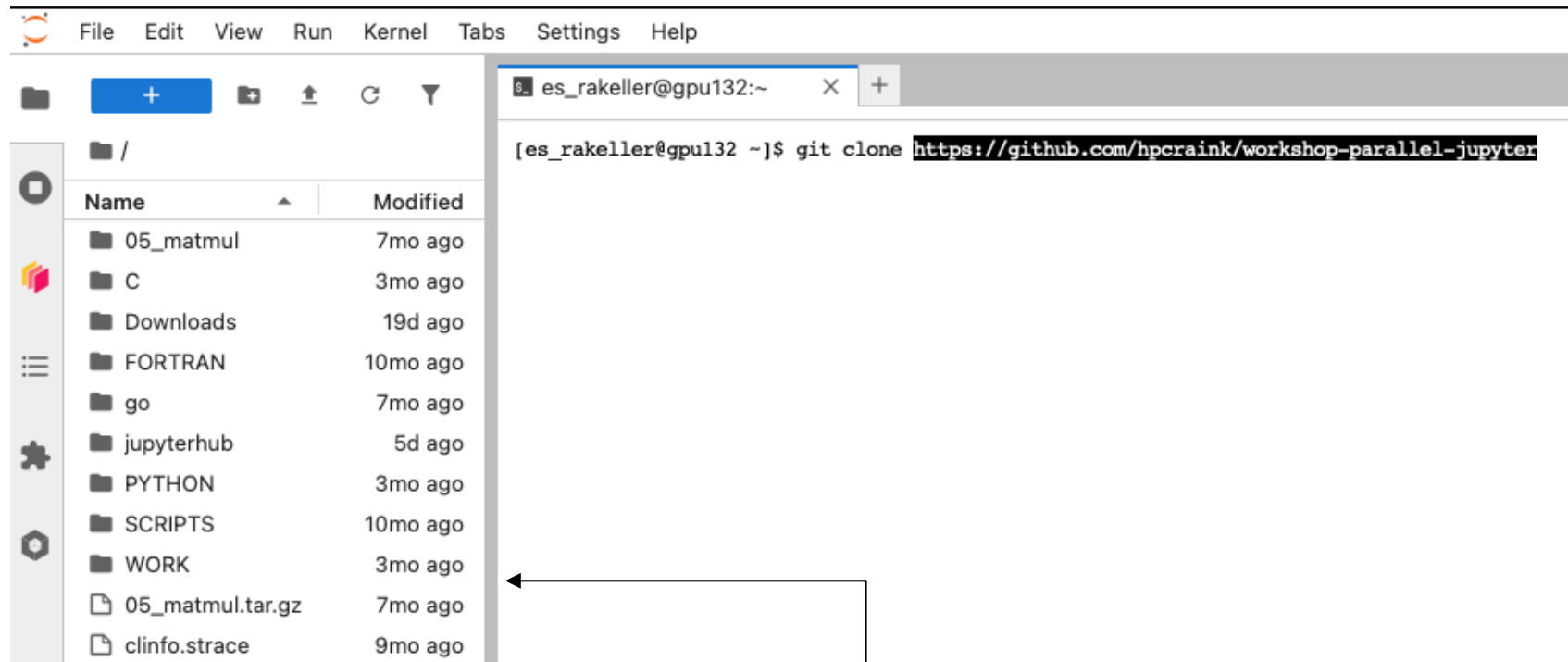
For other modules: [Wiki](#).



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:



- 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

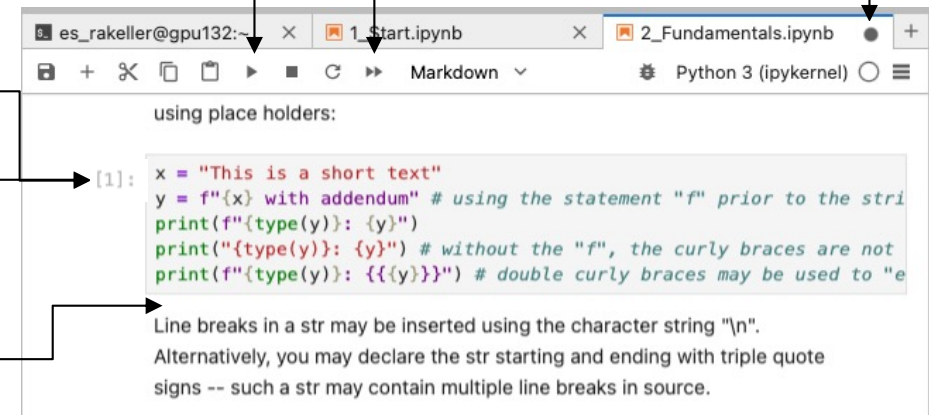
Execute current statement

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



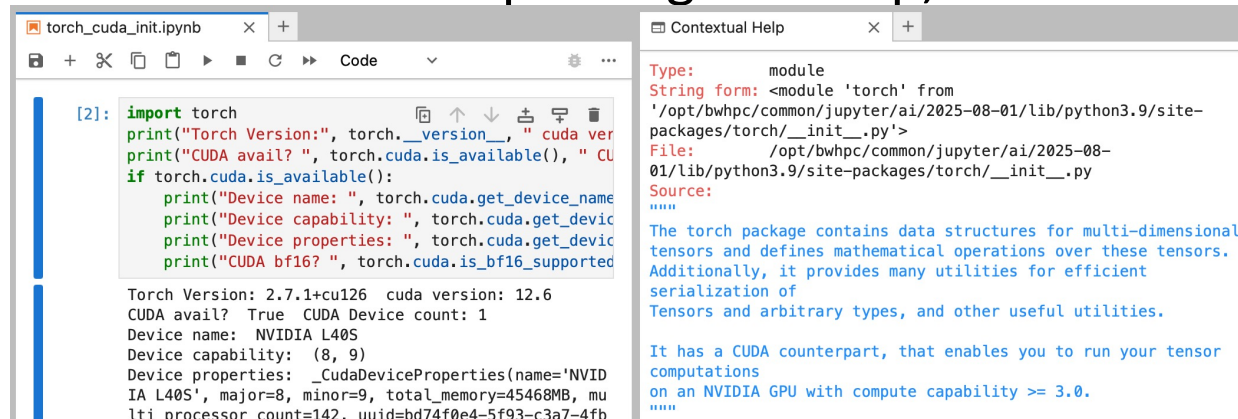
- `2_Fundamentals` explains Python programming
- `3_Numpy` explains basic Matrix operations using NumPy
- `4_Pandas` shows how Pandas works with large Parquet files
- `5_Machine_Learning` introduces SciKit Learn with Visualization

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:

