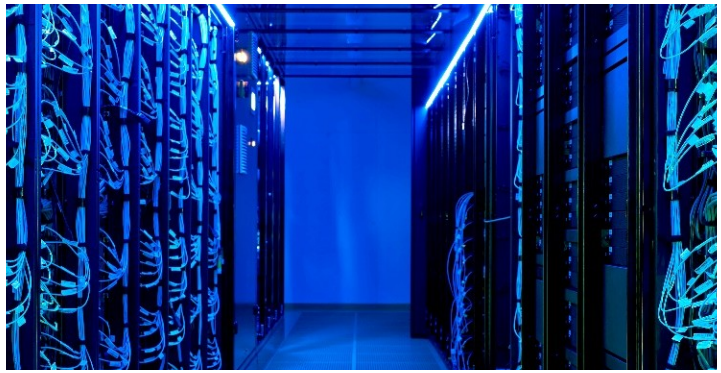
bw|HPC

KIT | NHR
National High-Performance Computing Center

Introduction to Jupyter at NHR@KIT

Leon Pascal Schuhmacher, SCC, KIT



Funding:

Baden-Württemberg
MINISTERIUM FÜR WISSENSCHAFT, FORSCHUNG UND KUNST

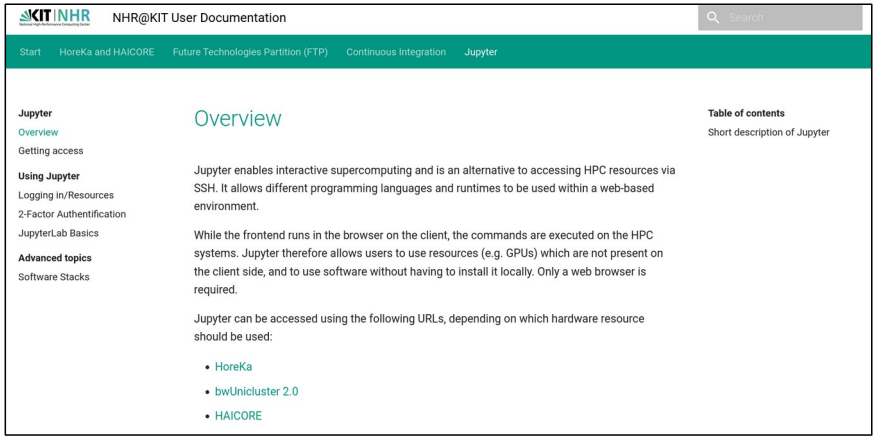
Bundesministerium
für Bildung
und Forschung

Outline

- Motivation
- Project Jupyter
- JupyterHub@HoreKa
- Software and Kernels
- Outlook
- Questions

Reference: Jupyter @ KIT

- Most information can be found at
  - bwHPC Wiki: https://wiki.bwhpc.de/e/Jupyter_at_SCC
  - NHR@KIT Wiki: https://www.nhr.kit.edu/userdocs/jupyter



Motivation

Why Jupyter?

HPC – “Classical”

- SSH
- **High Entry Hurdles**
  - Linux
  - Tools for connection and data transfer
  - Choice of resources
- Remote-Visualization
  - VNC? X11?

- State of the art for **advanced requirements!**

Why Jupyter?

HPC – Jupyter

- Web browser
  - No additional software
  - No data transfer for analysis
- **Low Entry Hurdles**

- **Intermediate** performance requirements
- **Interactive visualization** of data
- **Prototyping**

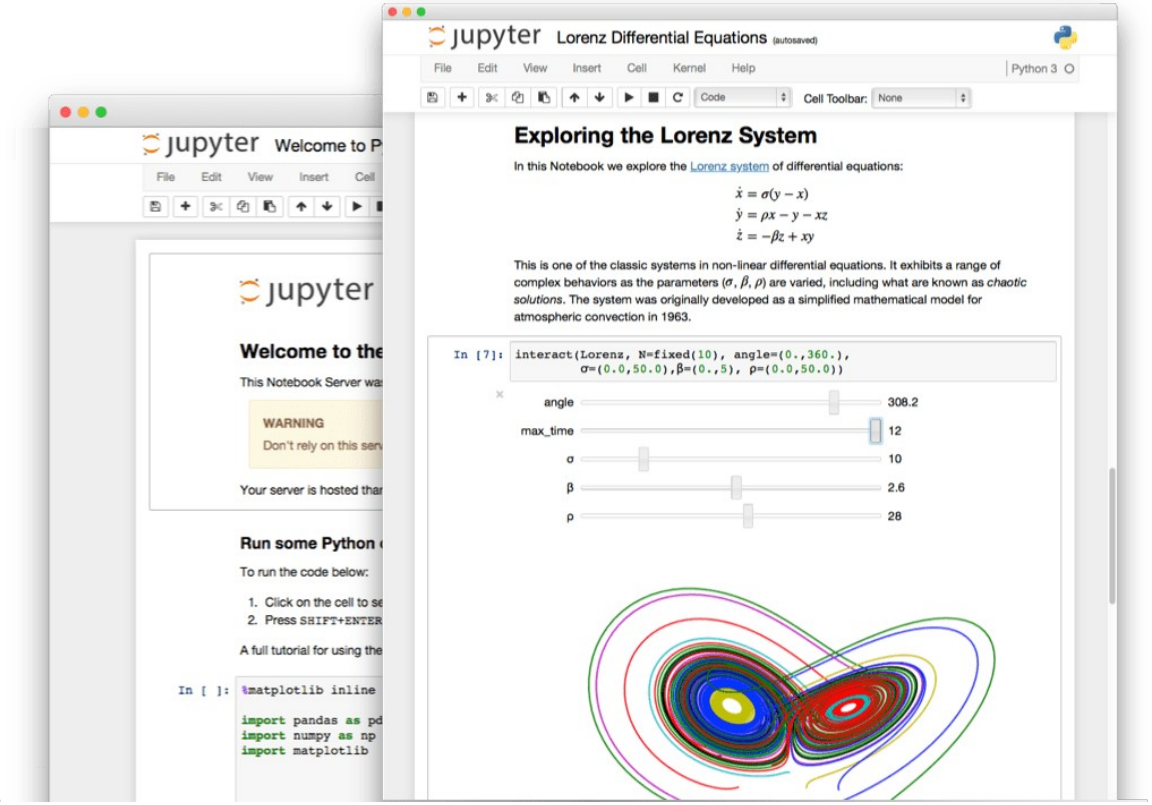
Project Jupyter

Project Jupyter

- Spin-off from project IPython
- Jupyter: Core languages
  - **Ju**lia
  - **Pyt**hon not**e**book
  - **R**
- **Language agnostic**
- Jupyter kernels
  - IPython
  - IJulia
  - IRKernel
  - >100 other kernels

Jupyter Notebook

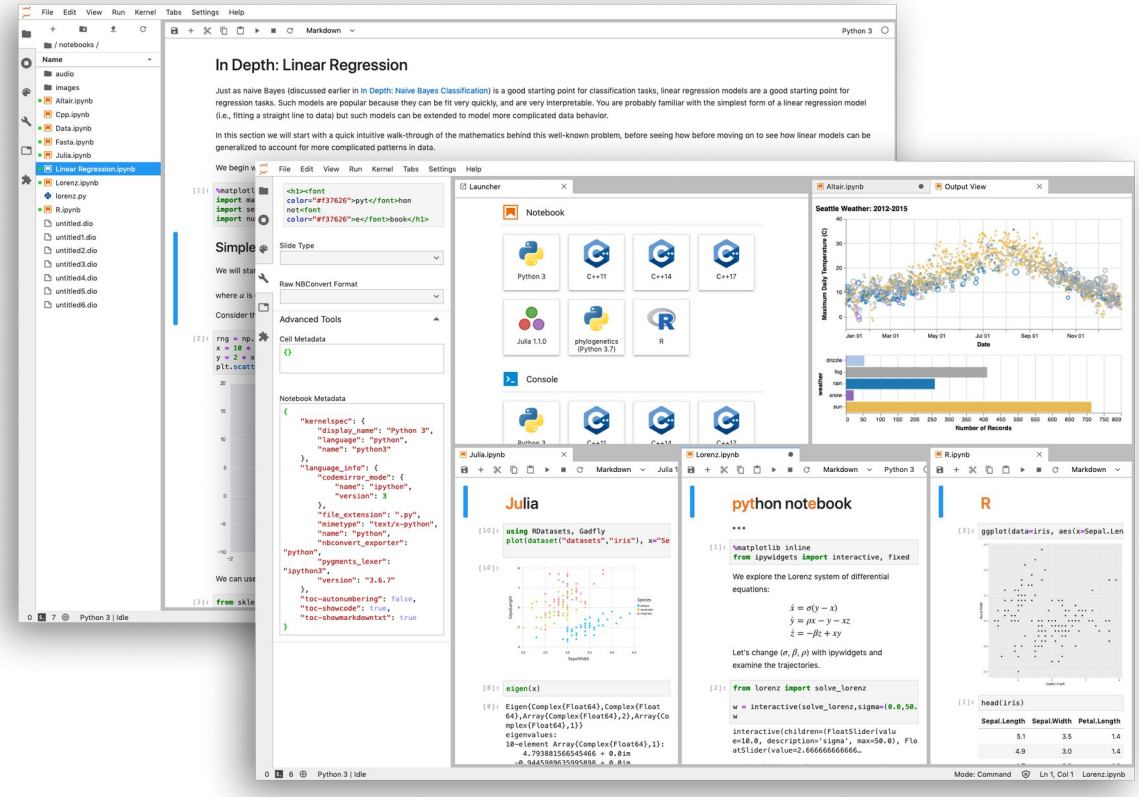
- Open-source web application
- Create and share documents
  - Live code
  - Equations
  - Visualizations
  - Narrative text
- **Execute code in browser**
  - … on HoreKa
- .ipynb file
  - JSON document



https://jupyter.org/

JupyterLab

- User interface for Project Jupyter
- Arrange documents/activities in tabs/blocks
  - Notebook
  - Terminal
  - Text editor
  - File browser
  - Rich outputs
  - …



https://jupyter.org/

JupyterHub

- Multi-user server for Jupyter Notebooks
- User management and authentication
- Spawning and proxying
- HPC context
  - Choice of resources
  - Slurm integration
  - Authentication

Jupyter: How-To

Local Jupyter

- Requirements:
  - Python / Anaconda (Windows users)
  - (nodejs + npm: JupyterLab extensions)
- Install locally

```
python -m venv env
source env/bin/activate
pip install jupyterlab
```

- Start
  - Jupyter Notebook: `jupyter notebook`
  - JupyterLab: `jupyter lab`
- Use
  - Open http://127.0.0.1:8888 in web browser

Live Demo

- Install

  ```
  python -m venv env
  source env/bin/activate
  pip install jupyterlab
  ```

  Additional Packages

  ```
  pip install matplotlib ipympl
  jupyter labextension install jupyter-matplotlib
  ```

- Start
  - Notebook

    ```
    jupyter notebook
    ```

  - JupyterLab

    ```
    jupyter lab
    ```

- Get the notebook:

  ```
  git clone https://github.com/pinae/BresenhamLidar
  ```

Jupyter@HoreKa

- Login to HoreKa, start **interactive job**

```
ssh <userID>@hk.scc.kit.edu
salloc -p accelerated --gres=gpu:4 --time=30:00
```

- Wait till job is running, remember compute node hostname (nodeID) and install and/or **start** Jupyter Notebook or JupyterLab …

```
module load jupyter/base
jupyter notebook --no-browser --port=8888 --ip 0.0.0.0
```

- From your local terminal: Establish **SSH tunnel** to compute node

```
ssh -L 8888:<nodeID>:8888 hk.scc.kit.edu
```

- Open in web browser: http://127.0.0.1:8888

JupyterHub@HoreKa

Registration Process – HoreKa

- **Registration @ HoreKa**
  - Online proposal form (Jards)
  - Peer reviewed proposal
  - HoreKa access form for each coworker
  - Web registration

- Set **service password**
  - FeLS → HoreKa → Set service password

- Register a software or hardware token (alias **2FA**)
  - FeLS → My Tokens
  - KIT users: https://my.scc.kit.edu/token

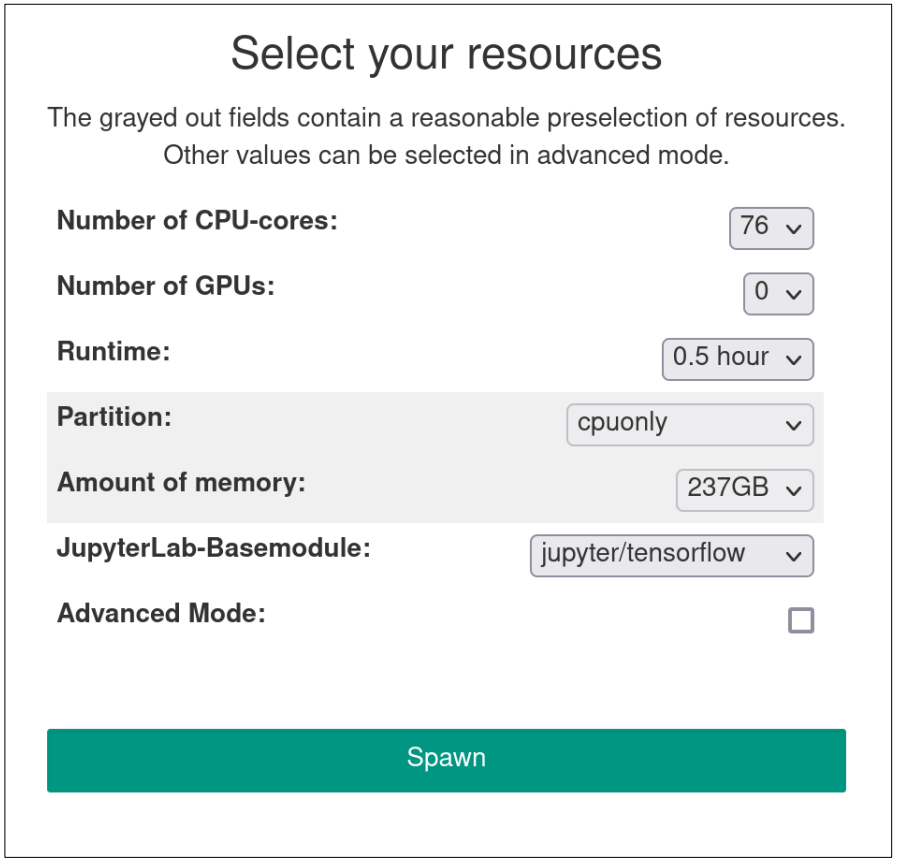
Accessing HoreKa

- Only within **network**
  - … of **KIT**
  - … of your **home institution**
- … otherwise establish **VPN** connection

- **SSH**
  - `ssh <userid>@hk.scc.kit.edu`
  - TOTP prompt (first)
  - Service password (second)

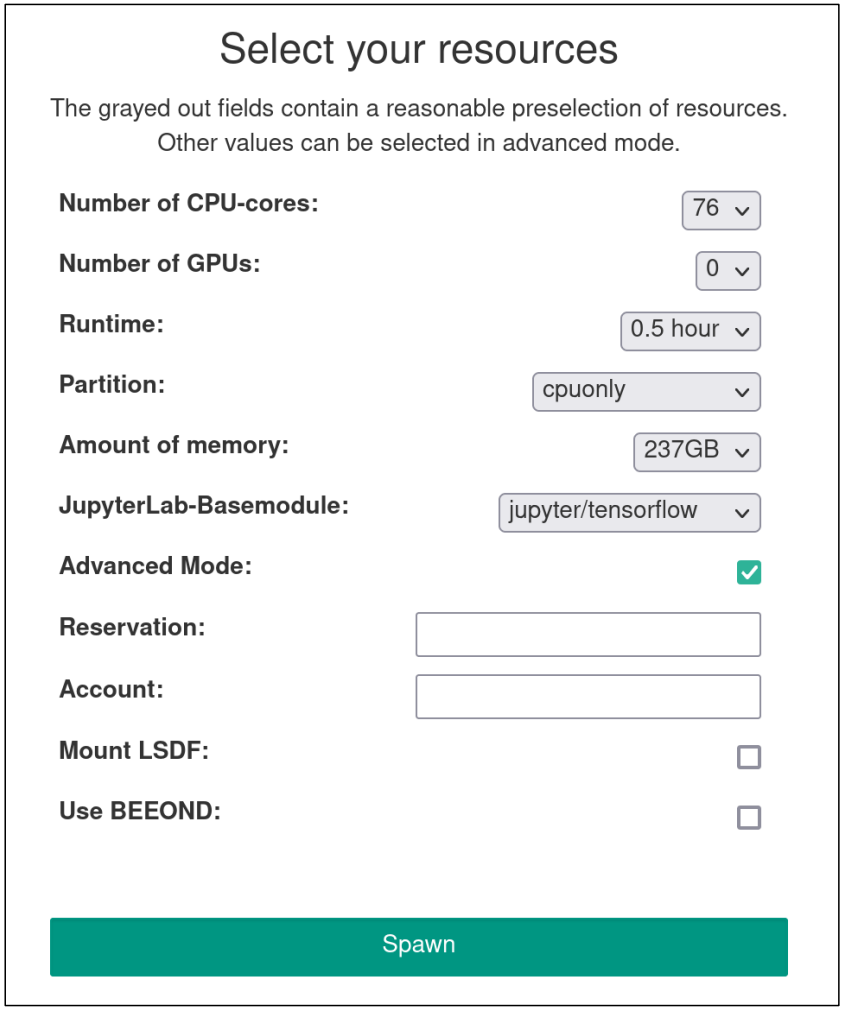
- **Jupyter**
  - (modern) Web browser: https://hk-jupyter.scc.kit.edu
  - **ONE successful login via SSH required**
    - … otherwise there is no `$HOME`
    - … spawning will fail (timeout)

Selection of Resources – Normal



- "**Normal**" mode
  - Number of CPU cores **OR** GPUs
  - Runtime
  - Jupyter Basemodule
  - Grayed out fields: Sane pre-selection of resources
- **Spawn**
  - Starts JupyterLab in interactive Slurm session
  - Connects/proxies to that session

Selection of Resources – Advanced



- „Advanced“ mode
  - Free choice of resources
  - No grayed out fields
  - No auto reservation
- Reservation
- Account
- LSDF
- BEEOND

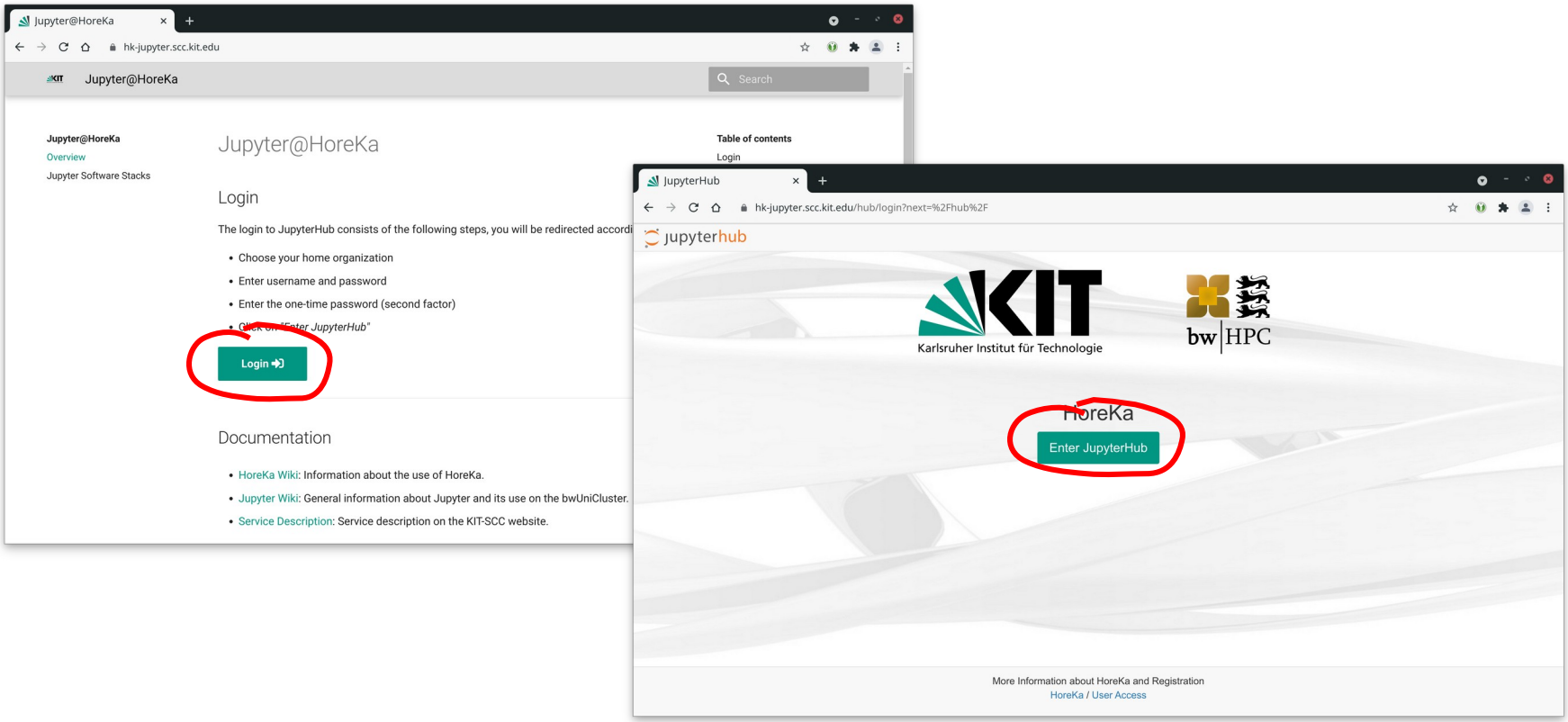
Jupyter Software Stacks

- Lmod modules
  - `jupyter/minimal`
  - `jupyter/base`
  - `jupyter/tensorflow`
- JupyterLab lives inside venv
  - `--system-site-packages` enabled/visible
  - Possible **interference** with `pip --user` installs (!)
- Access via
  - Drop-down menu in JupyterHub: "**JupyterLab-Basemodule**"
  - `module load jupyter/base` or `jupyter/tensorflow`

Login: Step-by-Step

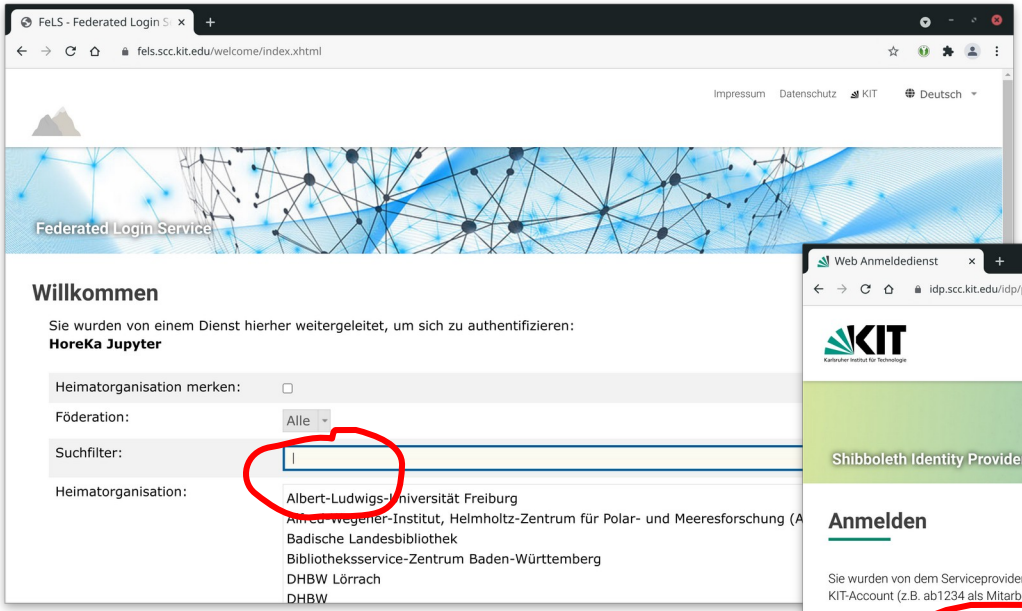
Step-by-Step Jupyter@HoreKa (1/4)

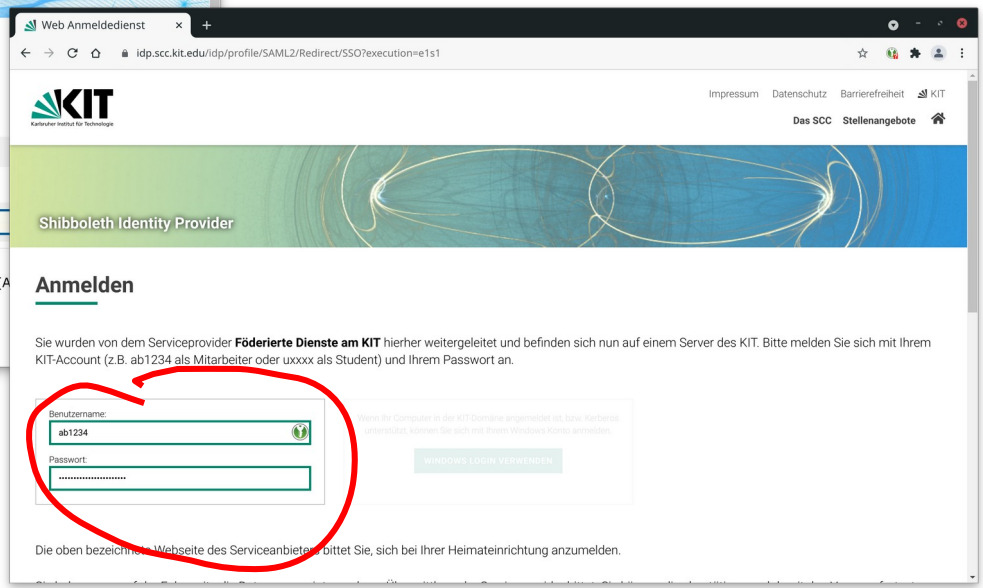
- Go to https://hk-jupyter.scc.kit.edu and click on „**Login**“
  - ...or go directly to https://hk-jupyter.scc.kit.edu/hub/login
- Click “**Enter JupyterHub**“



Step-by-Step Jupyter@HoreKa (2/4)

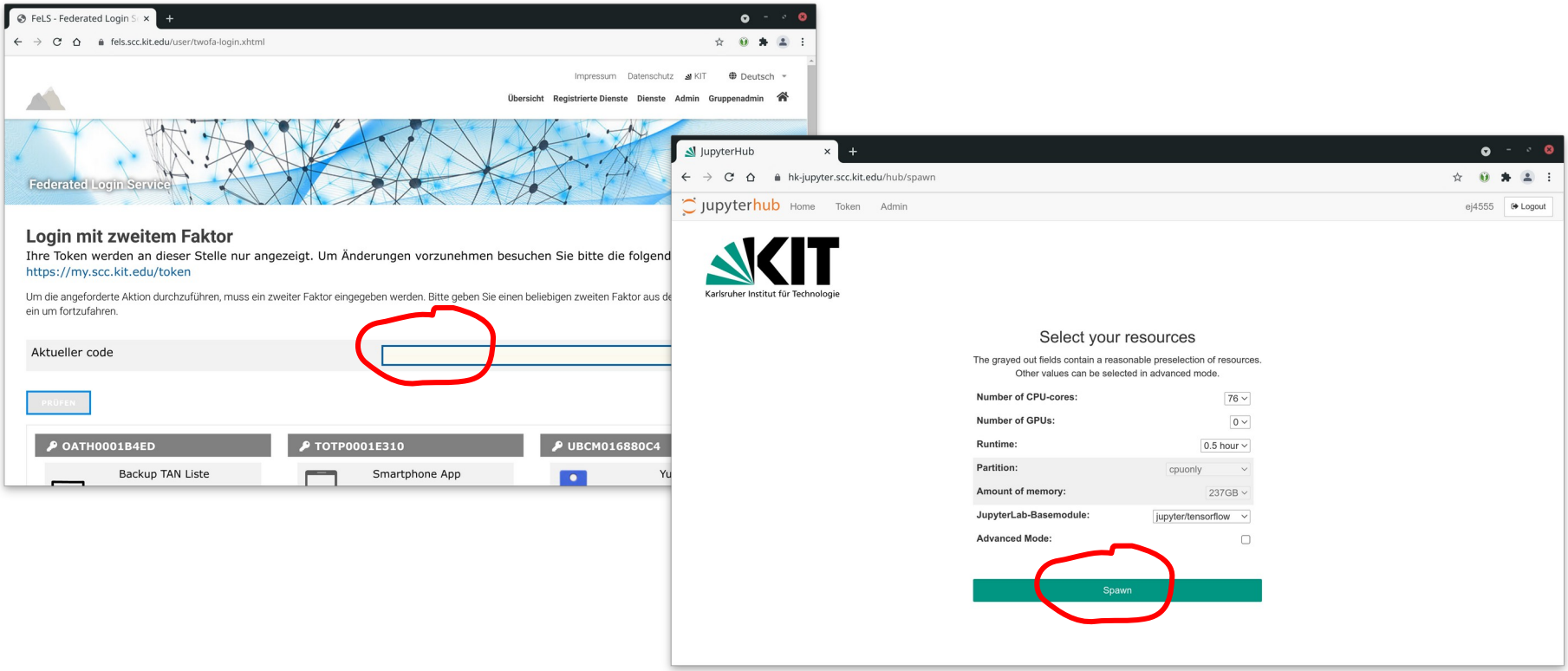
- Choose your **home organization** and continue
- Login to your home organization
  - **Username + password**





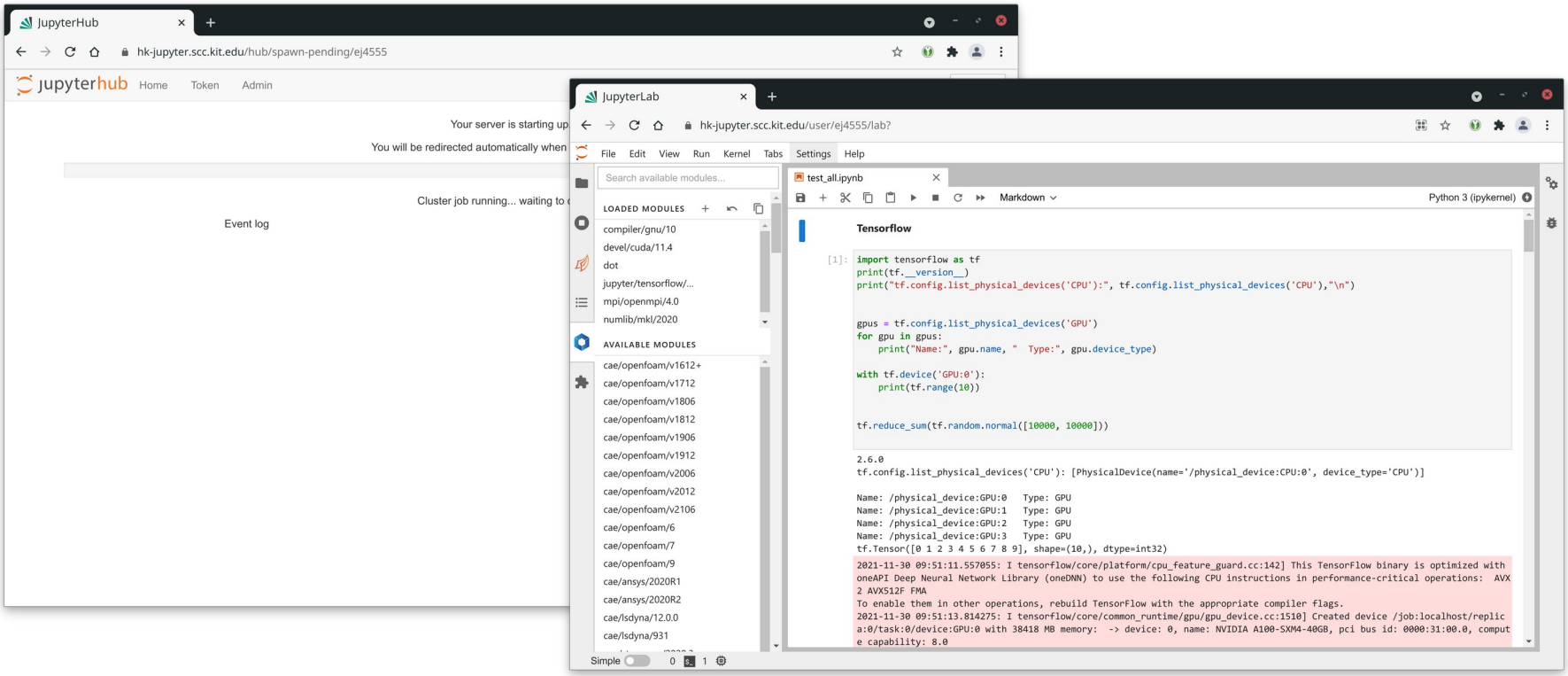
Step-by-Step Jupyter@HoreKa (3/4)

- Login to your home organization
  - **2FA**
- Select resources and click “**Spawn**”



Step-by-Step Jupyter@HoreKa (4/4)

- Spawning may take a while
  - … timeout after 10 minutes
- JupyterLab runs **on compute node** on HoreKa



Hands-On: Login

~10min

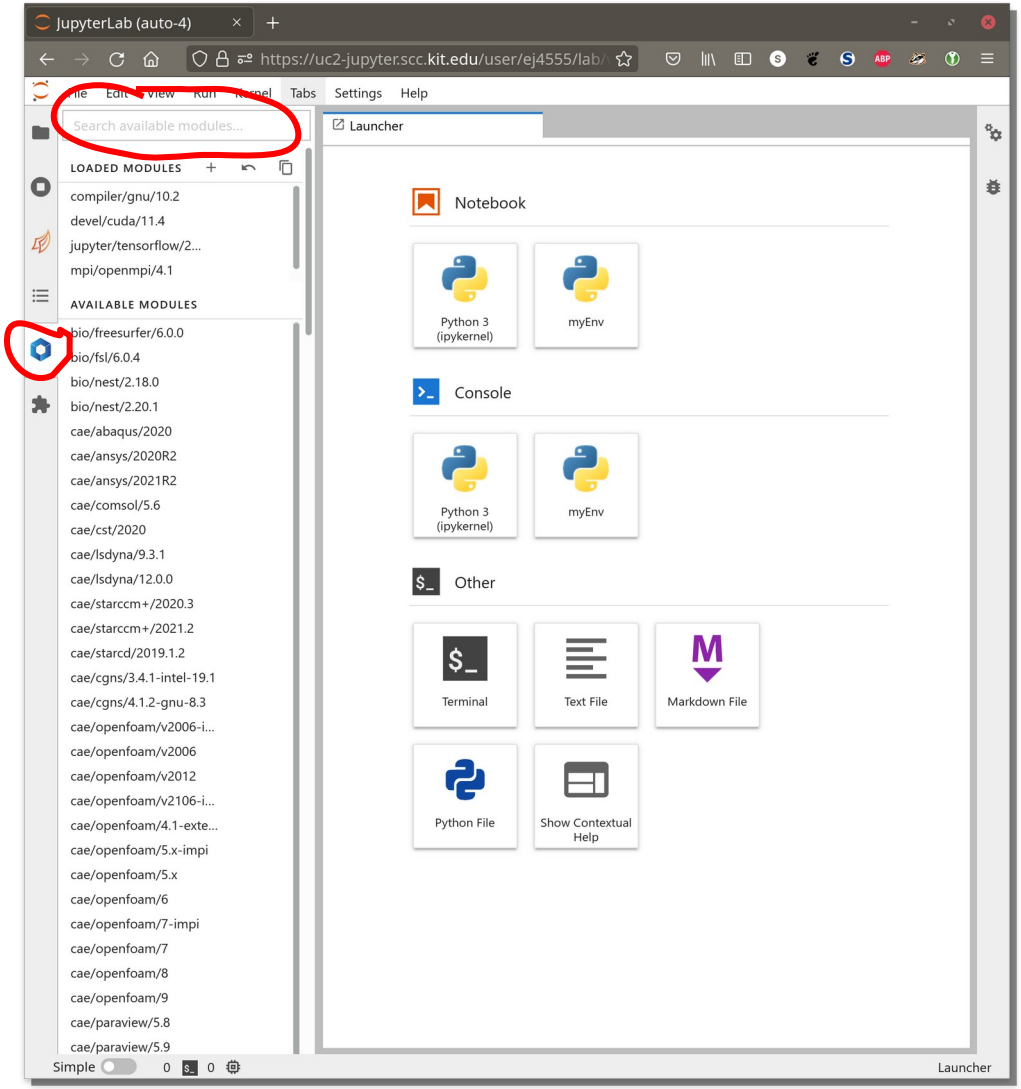
Hands-On: Login

- Start a session at https://hk-jupyter.scc.kit.edu
  - Hint:
    choose “accelerated” or “dev_*” partitions

Software and Kernels

Select Software

- Activate Lmod software modules
  → blue button
  → search field
- Kernel restart required



Add Python Packages + Custom Kernel

- Python: Use virtual environments

```
python -m venv myEnv
source myEnv/bin/activate
pip install <myPackage>
```

- Install kernel → IPython docs

```
python -m ipykernel install \
      --user \
      --name myEnv \
      --display-name "Python (myEnv)"
```

- You will get this:

R and Julia Kernel

You want to use R

- Load module `math/R`
- Open terminal
- `R`
  - `install.packages('IRkernel')`
  - `IRkernel::installspec()`

You want to use Julia

- Load module `devel/julia/1.6.2`
- Open terminal
- `julia`
  - `]`
  - `add IJulia`

Resetting everything

`rm -r …`

- Jupyter
  - … `~/.local/share/jupyter/kernels`
  - … `~/.jupyter`
- Python packages
  - … `~/.local/lib/python3.*`
- R
  - … `~/R`
- Julia
  - … `~/.julia`

Check state

- Bash
  - `~/.bashrc`

Hands-On: First Steps
~10min

Hands-On: First Steps

- Run the c't example on HoreKa
  - Hint 1:
    `git clone https://github.com/pinae/BresenhamLidar`
  - Hint 2:
    Replace `%matplotlib notebook` by `%matplotlib widget`

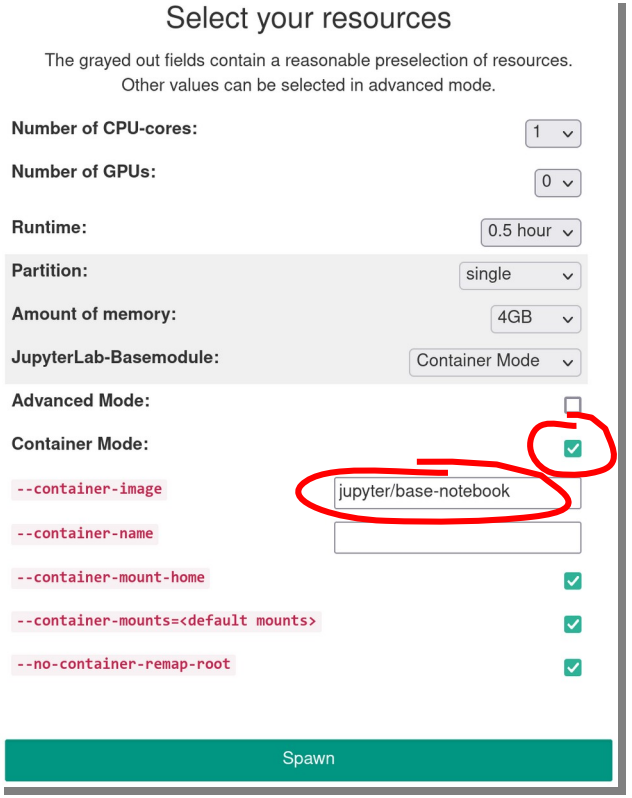
- Install a Julia Kernel
  - Compute `1+1` in a Julia Notebook
  - Try out some examples, e.g.: https://rosettacode.org/wiki/Factorial#Julia

Outlook

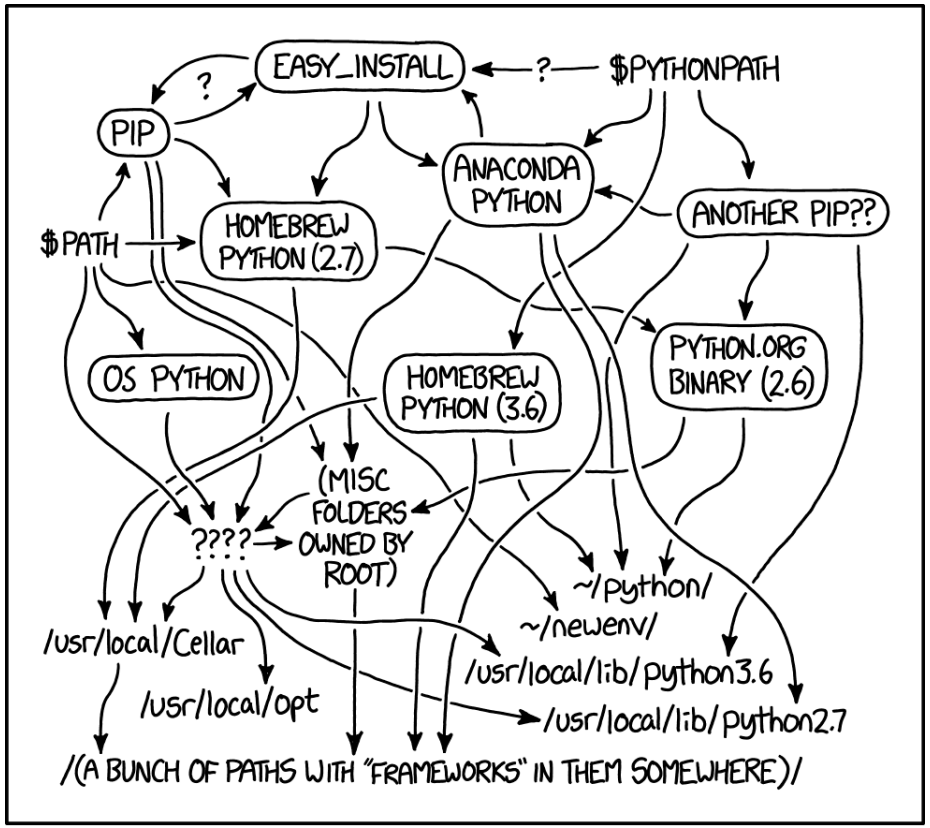
WIP: BYO Jupyter Container

- Connect **containerized** Jupyter with JupyterHub@HoreKa
- **Docker images** from any registry
  - For complicated/intrusive software stacks
  - Optimized software stacks
    - Intel, e.g.
      `intel/intel-optimized-tensorflow`
    - Nvidia, e.g.
      `nvcr.io#nvidia/tensorflow:21.10-tf2-py3`
    - AMD, e.g.
      `rocm/tensorflow:rocm4.3.1-tf2.6-dev`
- Possible **root access** (sic!)
  - **Yes, you can**

```
sudo apt-get install <myPackage>
```



Thank you for your attention!
Questions?



https://xkcd.com/1987

MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED
THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.