# Towards easily accessible interactive big data analysis on supercomputers

Martin Claus, Katharina Höflich, Willi Rath (GEOMAR)
Dorian Krause, Benedikt von St. Vieth, Kay Thust (JSC / FZ Jülich)
Nikolay Koldunov (AWI)

Exascale Earth System Modeling WP4 | https://github.com/ExaESM-WP4

ExaESM
Pilot Lab Exascale Earth System Modelling

HELMHOLTZ
RESEARCH FOR GRAND CHALLENGES

GEOMAR
Helmholtz Centre for Ocean Research Kiel

JÜLICH
FORSCHUNGSZENTRUM

AWI

# Classical HPC Jobs ⚡ Interactive Workloads

HPC systems / policies designed around batch tasks ← *complex simulations*

- Access via terminal sessions
- Curated software environments in modules
- Testing on limited but highly available batch resources
- Multi-node queued batch production tasks
- Desired / Feasible time to insight: Days to weeks

Increasing demand from interactive workloads ← *data analysis*

- Terabytes to petabytes of data stored
- Highly intermittent and bursty / variable resource demands
- GUI (browser-based, VNC, X11, …)
- Desired time to insight: creative timescales ~ minutes

# Project goals

Objectives

- Optimize existing world-class HPC systems, e.g. JUWELS, in terms of **accessibility**, **usability**, and **interactivity**.
- Inform decision making about future systems.

Leading questions

- How to provide convenient (and graphical) **remote access** to HPC?
- How to **reconcile** highly **variable resource demands** of interactive sessions and requirement for **high utilization** of providers?
- How to provide **machine optimized defaults** and still allow for **re-usable working environments** across many different platforms?

# Improve accessibility of HPC systems

Assessment of Jupyter at JSC and other HPC centers:

- Evaluate robustness, configurability, documentation, security, etc.
  https://github.com/ExaESM-WP4/JupyterHub-Evaluation-Whitepaper

**Preliminary results:**

→ Compute environment management is hard!

→ Separation of Jupyter frontend and backend is leaky.

→ Do we really need a JupyterHub? Or just HTTP to the big machine?

# Improve interactive analysis user experience

Parallelism of analysis software: Contribute to Dask for HPC
https://github.com/dask/dask-jobqueue

- increase configurability: *done / doing*
- implement heterogeneous clusters: *planned*

Optimize batch scheduling policy:

- job pre-emption for classical jobs / for analysis jobs
- resilience and elasticity of Dask clusters
- shared node access?
- towards shared Dask clusters?
  https://github.com/dask/dask-gateway

# Improve container use on HPC

Support climate scientists in using container workflows across:

- Laptop or dedicated data analysis computer (with Docker or Singularity)
- HPC clusters (with Singularity)
- Community-run cloud resources (https://mybinder.org)
- Cloud-based clusters sized for serious work (https://pangeo.io)

Container-based analysis builds expertise for classical HPC applications:

- Containerized MITgcm tests with Singularity
- Containerized FESOM2 environment across HPC centres and laptop (not public, AWI partners)