Data Management and Machine-Actionable Reproducibility for HPC

NEC User Group Meeting, Osaka, Japan

2025-05-14 // Andreas Knüpfer, Timothy J. Callow

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

www.casus.science

HZDR HELMHOLTZ ZENTRUM DRESDEN ROSSENDORF
TECHNISCHE UNIVERSITÄT DRESDEN
CBG Max Planck Institute of Molecular Cell Biology and Genetics
HELMHOLTZ CENTRE FOR ENVIRONMENTAL RESEARCH – UFZ
Uniwersytet Wrocławski

SPONSORED BY THE
Federal Ministry of Education and Research
STAATSMINISTERIUM FÜR WISSENSCHAFT UND KUNST
Freistaat SACHSEN

Contents

Center for Advanced Systems Understanding (CASUS)

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

- Started 2019 in Görlitz / Saxony
- Founded by 5 partner organizations

CBG
Max Planck Institute of Molecular Cell Biology and Genetics

HZDR
HELMHOLTZ ZENTRUM DRESDEN ROSSENDORF

HELMHOLTZ CENTRE FOR ENVIRONMENTAL RESEARCH – UFZ

TECHNISCHE UNIVERSITÄT DRESDEN

Uniwersytet Wrocławski

- Funding by

SPONSORED BY THE
Federal Ministry of Education and Research

STAATSMINISTERIUM FÜR WISSENSCHAFT UND KUNST

Freistaat SACHSEN

Center for Advanced Systems Understanding (CASUS)

- CASUS hosts no large research equipment like most Helmholtz institutes
- … not even the big computing clusters (for good reasons)
- Diverse topics, from biology/ecology to high energy physics and astronomy

Science icons created by Freepik - Flaticon, https://www.flaticon.com

- Computational Science is the connecting element

CASUS combines domain expertise and method expertise

- There are groups with application research in their names
- … and some with methods like "Machine Learning" or "Mathematical"
- And there is Scientific Computing Core (SCC) for Computational Science

Research Data Version Control

Research Data Version Control: Motivation

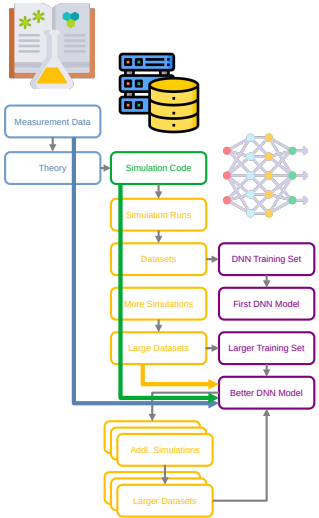
DNN surrogate models from HPC simulation results

- Extensive simulation runs with HPC over months
- … produce labeled data for DNN training

- Start first DNN training with available data early
- … determine feasibility
- … hyperparameter optimization
- … find out how much training data needed

- Iterate for several DNN versions and flavors
- Spot spurious error in some data $\mapsto$ fix in code $\mapsto$ re-run some simulations $\mapsto$ replace affected data

- Which DNN version was trained with which exact subset of the data? Affected by faulty data points?
- Exact versions of the simulation code or the DNN training script used? Track git commit id!



https://www.flaticon.com/free-icon/science_6969627
https://www.flaticon.com/free-icon/database-storage_2906274
https://www.flaticon.com/free-icon/deep-learning_8637092

… we solved it for Software Versioning

- well established and most well known
- Clear steps: commit, push/pull, merge, …
- Command line + GUI + IDE integrations
- **But git is not good for binary data or very large files**

- Central repositories and web GUI
- Authentication and authorization
- Several alternatives like github, Gogs, Gitea, Gin, …

… we solved it for Software Versioning

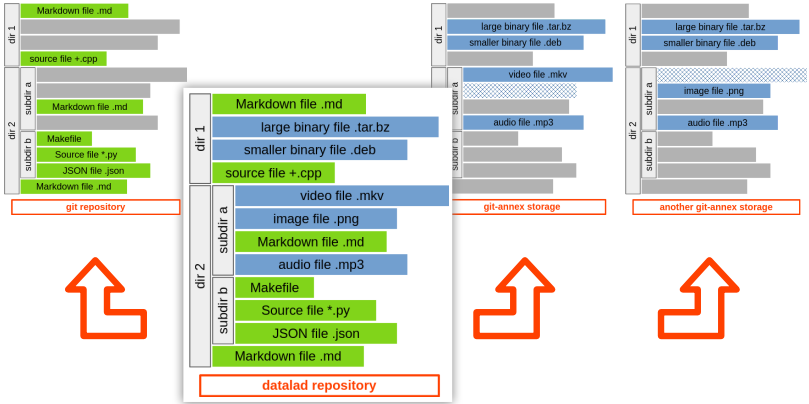
- well established and most well known
- Clear steps: commit, push/pull, merge, …
- Command line + GUI + IDE integrations
- **But git is not good for binary data or very large files**

- Central repositories and web GUI
- Authentication and authorization
- Several alternatives like github, Gogs, Gitea, Gin, …

- git plugin `git annex`
- new sub-commands
- Manage large binary files via git, yet don't store them in git
- Special git-annex storages instead
- Works with usual sub-commands, GUIs, IDEs, …

git-annex: git for large / binary data

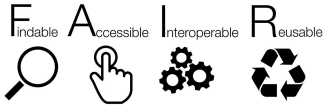
git + git-annex: Text files and binary files go to separate storages:



The F.A.I.R. Criteria and Versioning

- Data versioning is **not** part of the F.A.I.R. criteria
  - Forgotten when they were designed in 2014?
  - Or considered too much to ask?
- Rather few publications dare to ask for fully fledged version control for research data
- Dataset publication with version numbers as the best state-of-the-art approach?
  - Make few data publications, once per year +/-
  - Large sets of archive files (all or nothing)
  - Redundant data between versions, errata lists [a]
- Dataset publication as appendix to papers
  - One point in time per paper
  - Redundant data for successive papers

[a] `https://abcdstudy.org/scientists/data-sharing-archive/`



SangyaPundir, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=53414062

- F1. Unique and persistent identifier
- F2. Rich metadata
- F3. Metadata include the identifier
- F4. (Meta)data searchable

- A1. (Meta)data retrievable by identifier via standardised protocol
- A2. Metadata accessible, even when data no longer available

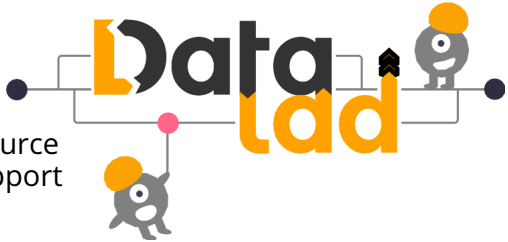
- I1. Formal, accessible, shared, and broadly applicable language
- I2. (Meta)data use FAIR vocabularies
- I3. Qualified references to other (meta)data

- R1.1. Data usage license
- R1.2. Detailed provenance
- R1.3. Meet domain-relevant community standards

Machine-Actionable Reproducibility with DataLad

Machine-Actionable Reproducibility with DataLad

- Distributed data management[1]
- On top of git and git-annex
- Academic background, free and Open Source
- Active community provides hints and support
- simplifies some tasks
- additional functionality



[1] `https://helmholtz.software/software/datalad`, `https://handbook.datalad.org`

Machine-Actionable Reproducibility with `datalad run` and `rerun`

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

Record processing steps:

- Use `datalad run` to execute scripts or binaries, preferably with scripts or code sources also in the repository
- Specify input and output files, preferably all part of repository
- See also `datalad containers-run`

```
$>datalad run -m "Resize image" \
  --input git_commit_2x.png \
  --output git_commit_2x.small.png \
  "convert {inputs} -geometry \
   439x439 {outputs}"
```

```
knue@fwu082:~/data/datalad-demo/images > git log -1
commit 4f7a1e6d301e3bdd03aeb6c9e355fb5f2ceb9a23 (HEAD -> mas
Author: Andreas Knüpfer <knue@posteo.de>
Date:   Fri Jul 5 13:19:01 2024 +0200

    [DATALAD RUNCMD] Resize image

    === Do not change lines below ===
    {
     "chain": [],
     "cmd": "convert {inputs} -geometry 439x439 {outputs}",
     "dsid": "be528a88-4333-49cf-9705-0be6e3f84a23",
     "exit": 0,
     "extra_inputs": [],
     "inputs": [
      "git_commit_2x.png"
     ],
     "outputs": [
      "git_commit_2x.small.png"
     ],
     "pwd": "images"
    }
    ^^^ Do not change lines above ^^^
```

Machine-Actionable Reproducibility with `datalad run` and `rerun`

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

Work with existing Datalad repository:

- Clone the repository
- All text files get downloaded but annexed files not yet

Get selected / all annexed files: `$> datalad get ...`

Reproduce processing steps: `$> datalad rerun <git commit id>`

- Previous example: `$> datalad rerun 4f7a1e6d`
- Execute same command again, commit results if something changed
- Run with current versions of input files and scripts / binaries
- If necessary do `datalad get` for input files first

- **Added value** when doing all steps the DataLad way

Challenges on HPC and how we solved it

Challenges on HPC

The DataLad reproducibility scheme is very much incompatible with HPC :-(

1. A git repository must not be accessed concurrently: **critical conflict!**
   - Changing files inside in parallel is fine, concurrent git commands are not
2. All `git` / `datalad` calls would be sequential inside the Slurm job
3. Slurm job script not a good reproducibility record if `datalad` commands inside
   - … would need to change `datalad run` to `datalad rerun`

The state of the art workaround: separate clone per job [1]

- For all potentially concurrent jobs in the same repository
- Not necessarily all large / binary files in every clone
- … but all small git bookkeeping files – would your parallel file system like it?

It would not be accepted in the HPC community and by the HPC centers

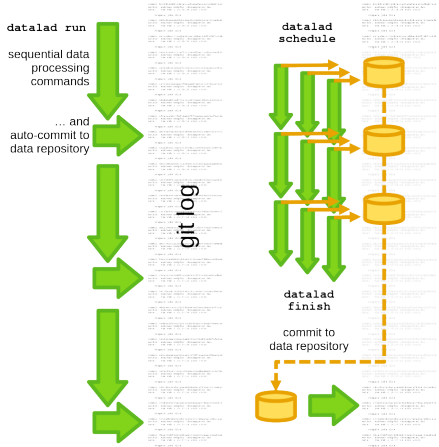
[1] Adina S. Wagner et al. “FAIRly big: A framework for computationally reproducible processing of large-scale data”. In: Scientific data 9.1 (2022), p. 80.

Solution: The DataLad extension for Slurm

DataLad commands around Slurm jobs:

1. Prepare and submit jobs
   - Get annexed files if needed
   - Submit job to Slurm, don't wait
   - New `datalad slurm-schedule` command
2. Finalize jobs and commit of result files
   - After job(s) finished
   - Check and commit files to repository
   - New `datalad slurm-finish` command
3. Re-run jobs
   - `datalad slurm-reschedule <commit-id>`

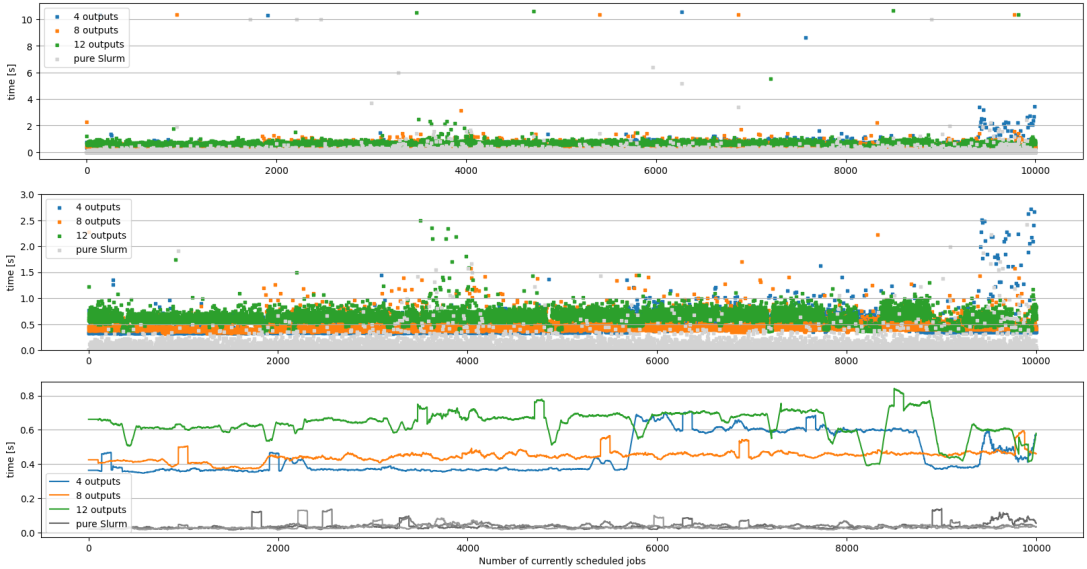
- Call DataLad commands sequentially
- Jobs can run concurrently
- Slurm job script untouched



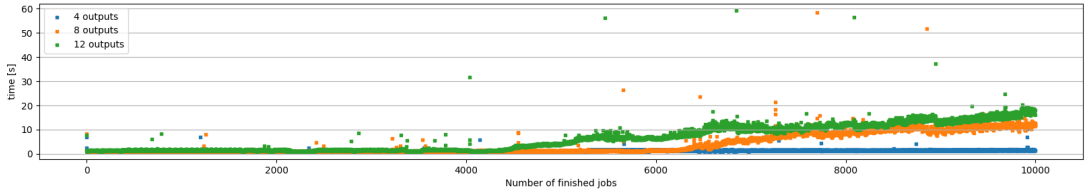
Evaluation

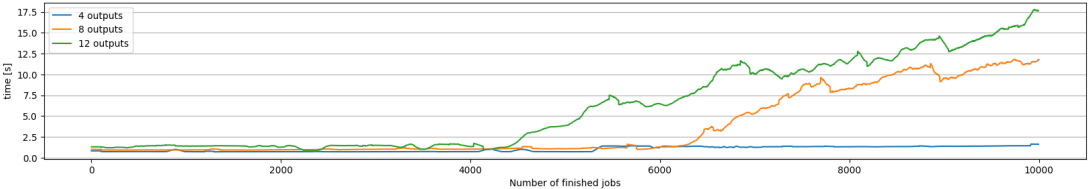
- Evaluation runs with large repositories (number of files)
- Compare to pure Slurm cases

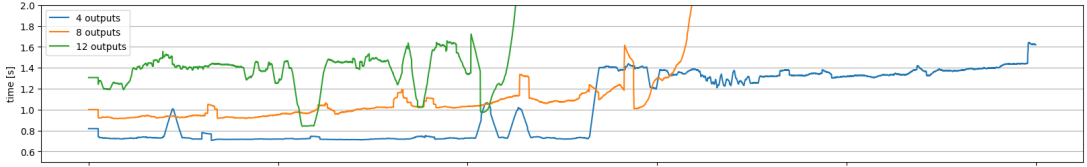
Evaluation: datalad slurm-schedule



Evaluation: datalad slurm-finish







Evaluation Results

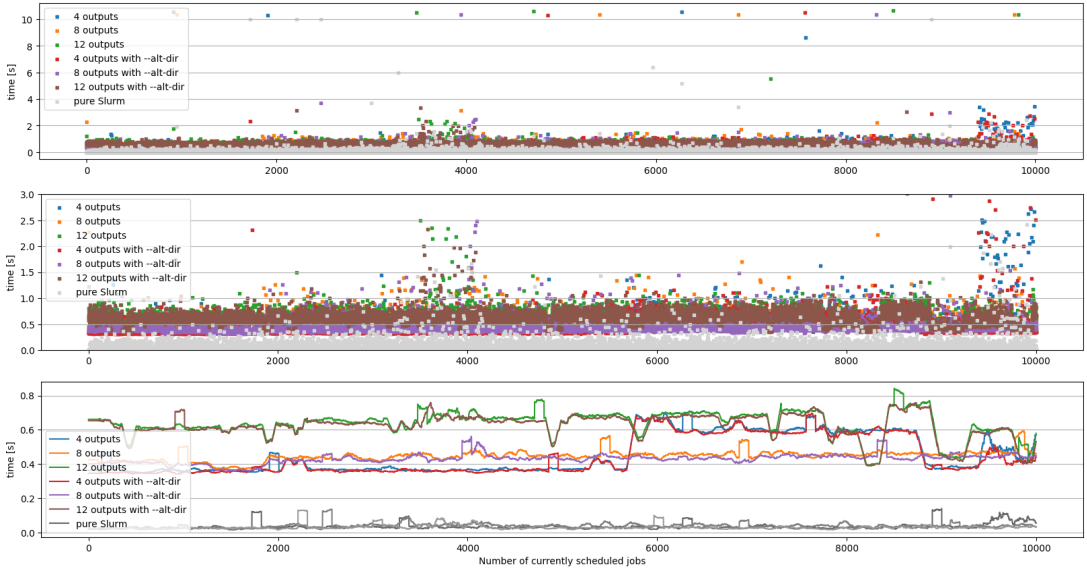
- Some overhead of 0.5*s* … 1.5*s* per job over pure Slurm
- … cost of data version management and machine-actionable reproducibility

- Adverse interaction of git with parallel file system (here GPFS)
- Increasingly slow for large repositories with $\geq$ 50 000 files
- Does not happen on local file systems

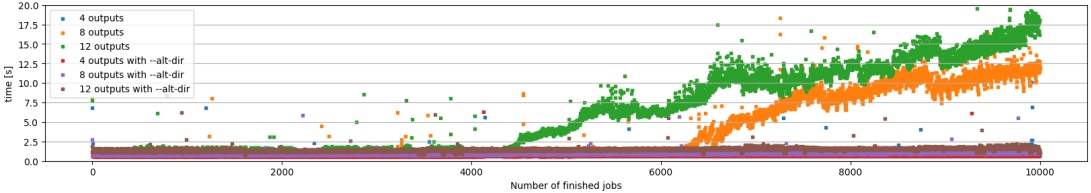
Introduce the `--alt-dir` option

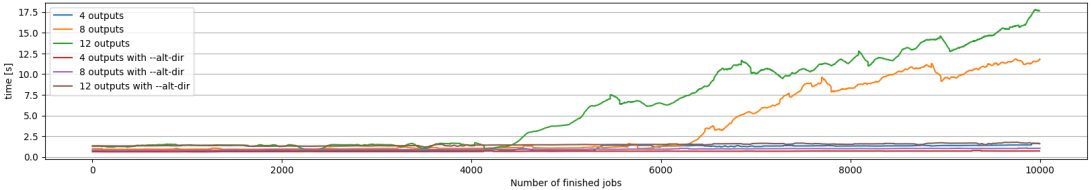
- Data repository on a local file system on a login node
- DataLad copies all input files to parallel file system as part of `slurm-schedule`
- … and copy all output files back during `slurm-finish`

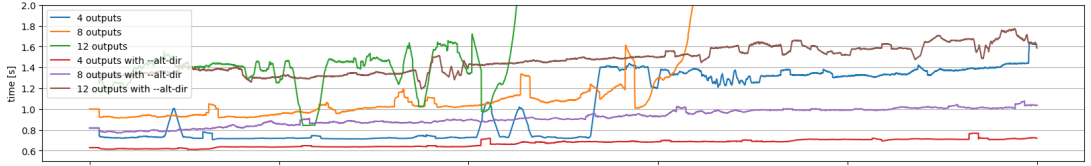
Evaluation: datalad slurm-schedule –alt-dir



Evaluation: datalad slurm-finish –alt-dir







Evaluation: datalad slurm-finish with –alt-dir

- Resonable overhead with `--alt-dir`
- basically the cost of new functionality

- Future work: Extensive evaluation on Lustre and Weka
- Future work: Evaluate with different file sizes

Summary & Outlook

Summary
- Data version control solution DataLad
- 100% git look&feel, keep existing know-how and command line / GUI / IDE tools
- Now compatible with HPC

More
- DataLad Slurm extension forbids conflicting jobs (details see paper)

Future work
- Compare with `git worktree` on parallel file systems, find out which is favorable for different scenarios
- Evaluate S3 as DataLad annex storage in HPC environments

CASUS

CENTER FOR ADVANCED
SYSTEMS UNDERSTANDING

www.casus.science

HZDR
HELMHOLTZ ZENTRUM DRESDEN ROSSENDORF

TECHNISCHE UNIVERSITÄT DRESDEN

CBG
Max Planck Institute of Molecular Cell Biology and Genetics

HELMHOLTZ CENTRE FOR ENVIRONMENTAL RESEARCH – UFZ

Uniwersytet Wrocławski

SPONSORED BY THE
Federal Ministry of Education and Research

STAATSMINISTERIUM FÜR WISSENSCHAFT UND KUNST
Freistaat SACHSEN