BABS:
Reproducible, generalizable, and scalable software for BIDS App analysis of large datasets

Work in progress

PennLINC Lab Meeting

Chenying Zhao

Feb 28th, 2023

Overview – The challenge and proposed solutions

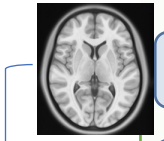
Convert into BIDS format + Data curations

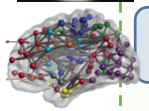
Pre- and post-processing using BIDS Apps

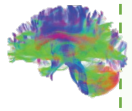
Group-level statistical analysis

Raw neuroimaging data →

BIDS-fy; CuBIDS







Structural MRI: FreeSurfer, …

Functional MRI: fMRIPrep, XCP-D, …

Diffusion MRI: QSIPrep, …

…

BABS: BIDS App Bootstrap

Voxel-wise data

Fixel-wise data

Surface data

Statistical tools

e.g., ModelArray

For reproducible image processing at scale

Outline
“BABS” – BIDS App Bootstrap

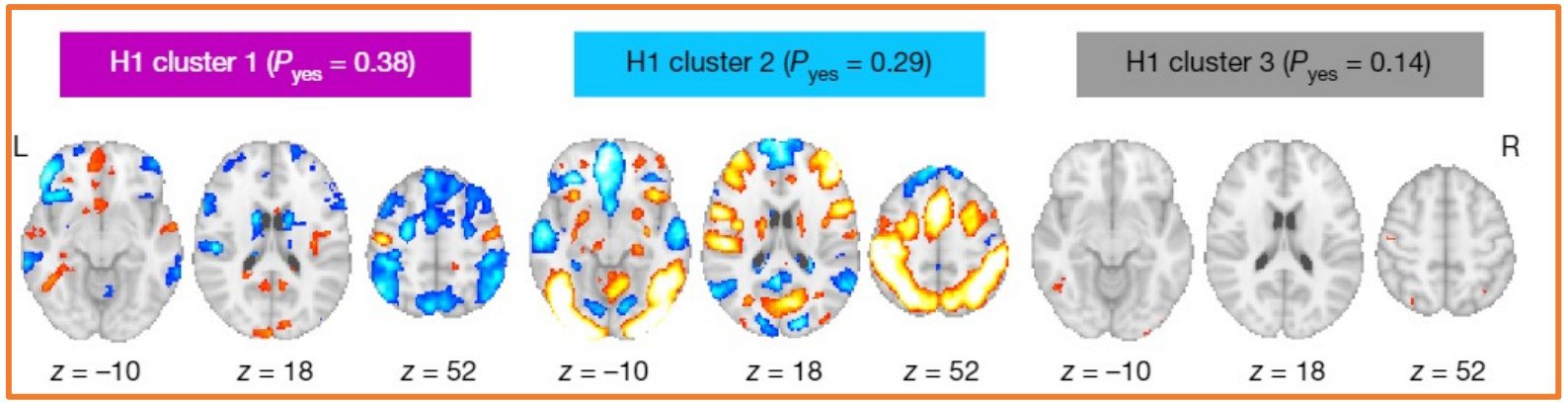
- Background, motivations, and introduction of existing effort
- Design
- Current progress
- Next steps

Background

Reproducibility crisis in neuroimaging studies

- Numerous choices in both image processing and statistical analysis
- Even with the same data, different analytical methods and tools can lead to discordant results:

Functional MRI study, same input dataset, answering same hypothesis by different teams:



Botvinik-Nezer et al., Nature 2020

large difference in whole-brain statistical results…

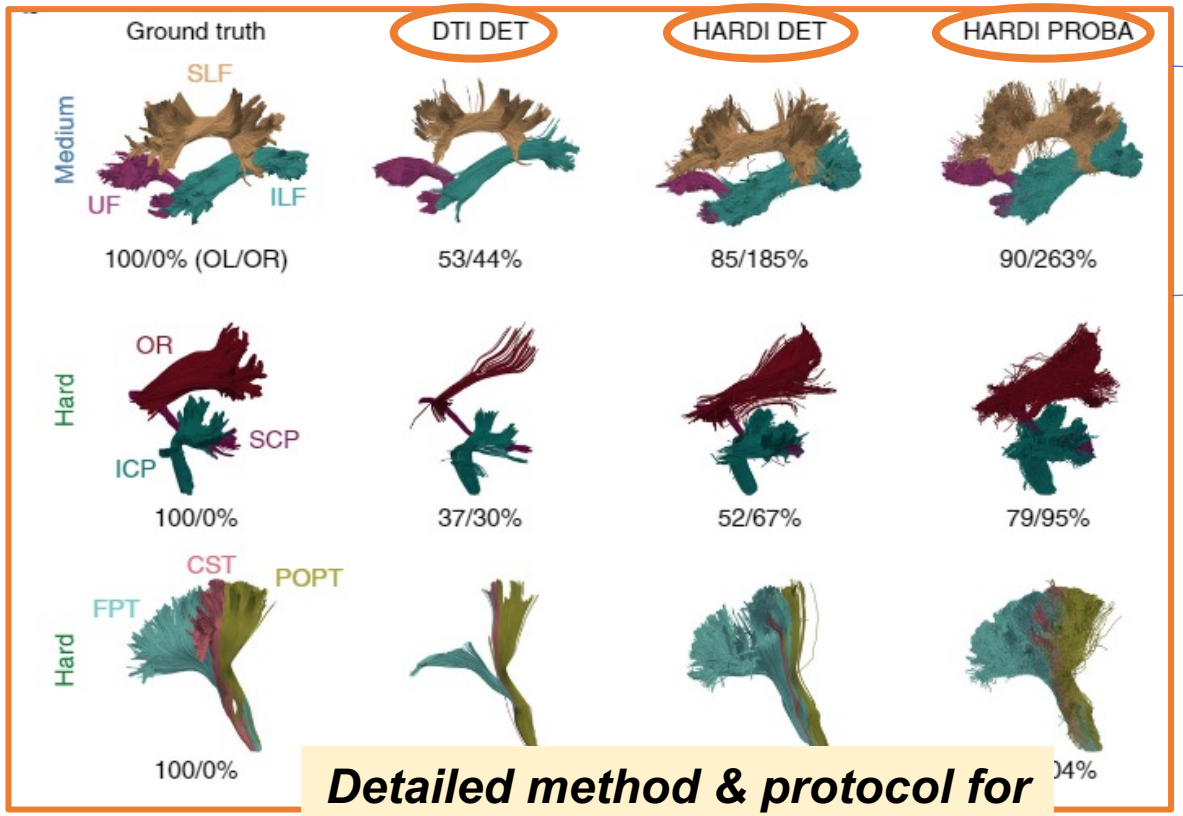
Background

Reproducibility crisis in neuroimaging studies

- Numerous choices in both image processing and statistical analysis
- Even with the same data, different analytical methods and tools can lead to discordant results:

Different reconstruction and tractography methods

Diffusion MRI:



Very different shapes of white matter tracts

Detailed method & protocol for replicating results!

Maier-Hein et al., Nat Comm 2017

Background

Challenges in reproducibly processing large-scale datasets and emerging solutions

- Human Connectome Project
  - Young adult (n=1,200)
  - Development (n=1,300)
- Philadelphia Neurodevelopmental Cohort (PNC, n=1,601)
- Healthy Brain Network (n=5,000)
- UK Biobank (n=40,000+)
- …

- Large-scale datasets:
  - ☺ enhance statistical power
  - ☹ complexity, heterogeneity, and huge sample size
- Emerging solutions:

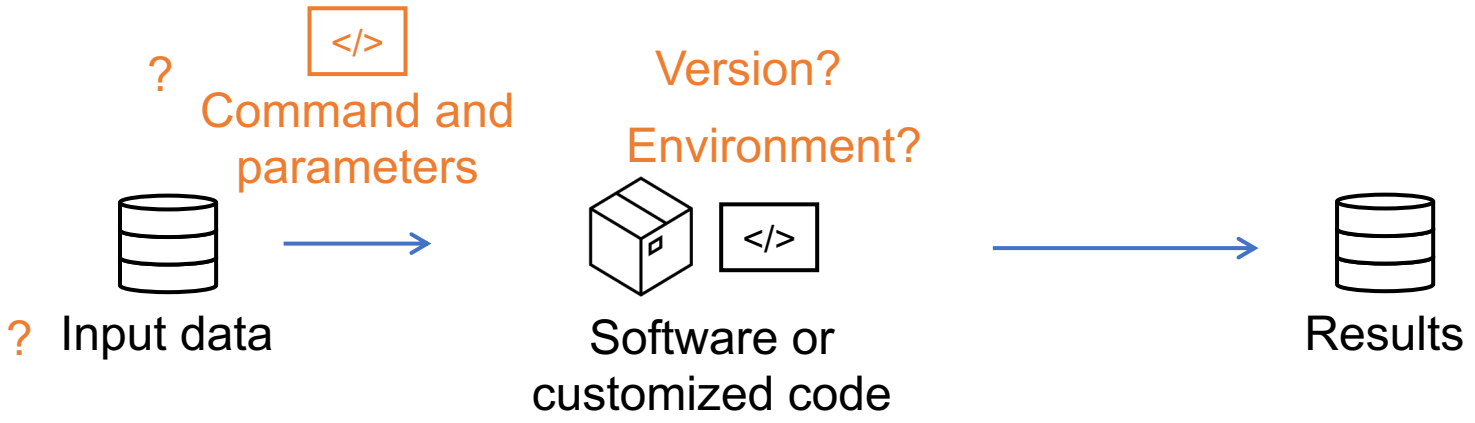
Code version control tools

Portable containers

High performance computing (HPC) clusters

Still, there are obstacles…

Complete reproducibility with automatic data provenance tracking



What we hope:

1. Use containerized software
2. *Automatically* track all provenance + scalable
3. User-friendly
4. Generalizable

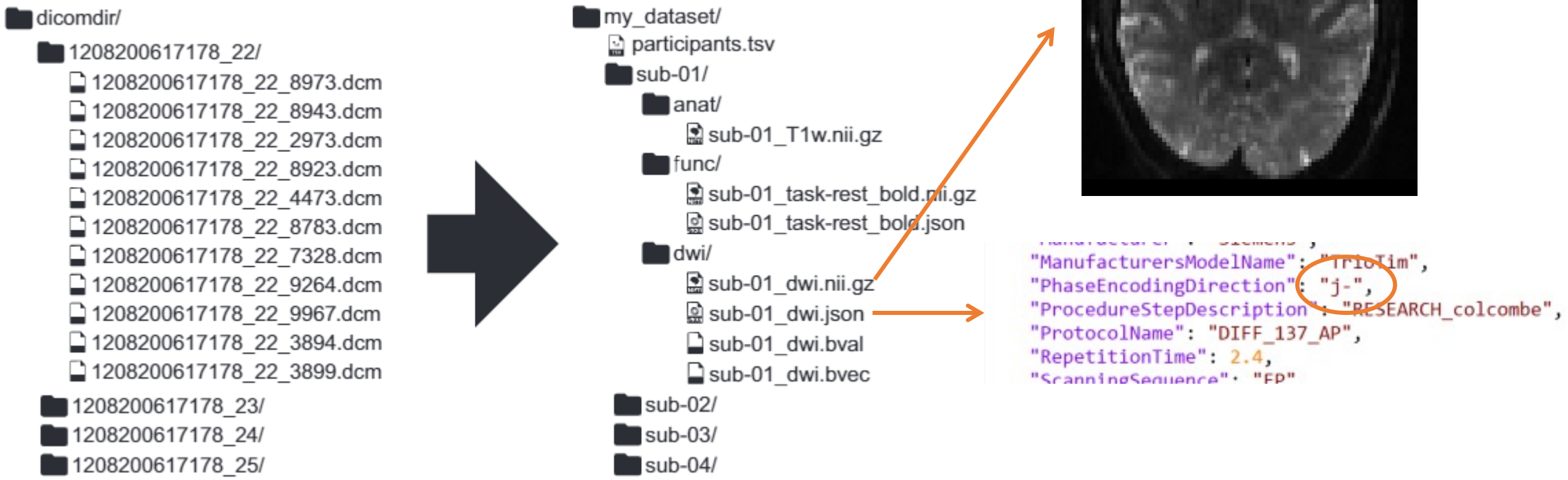
docker Singularity image

Containerized software

BIDS and BIDS Apps

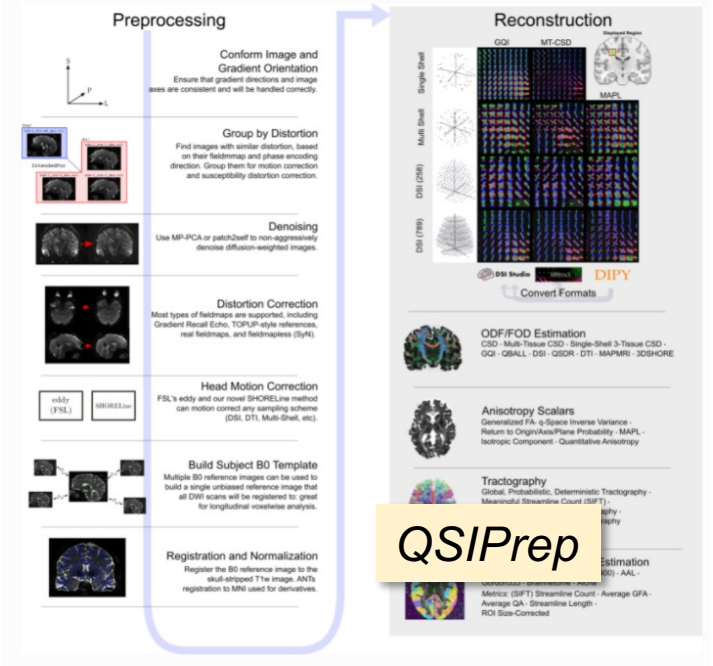
What is Brain Imaging Data Structure (BIDS) format?

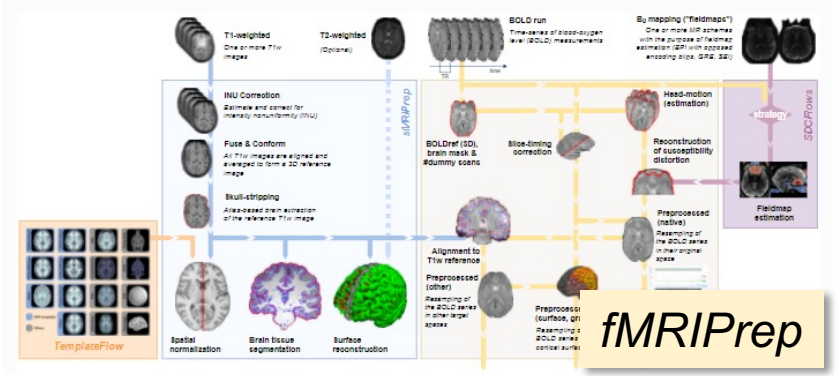
- BIDS format is a standard format to organize data;
- BIDS includes images + their metadata in sidecar JSON files:



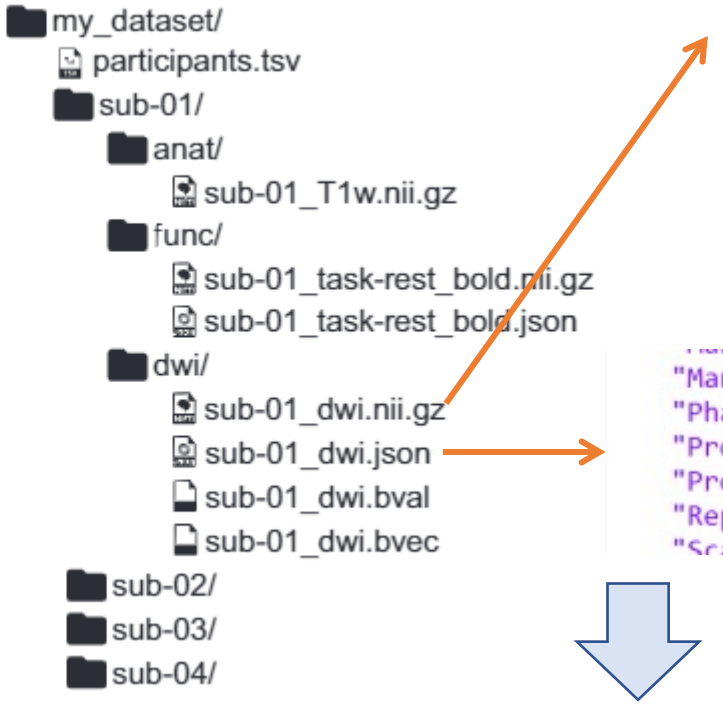
Gorgolewski et al., SciData 2016

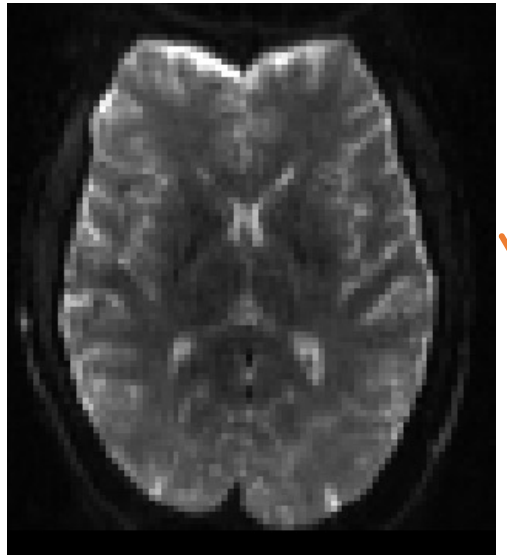
BIDS and BIDS Apps
What are BIDS Apps?



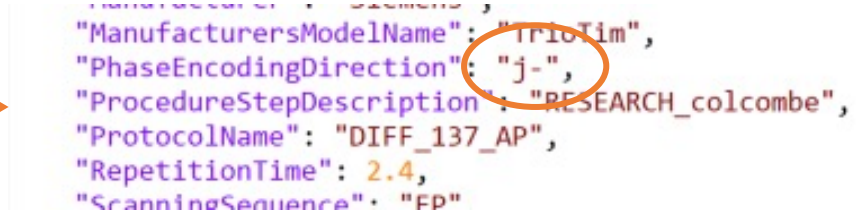


...





Phase encoding direction



BIDS Apps: ✓ Automatically configure

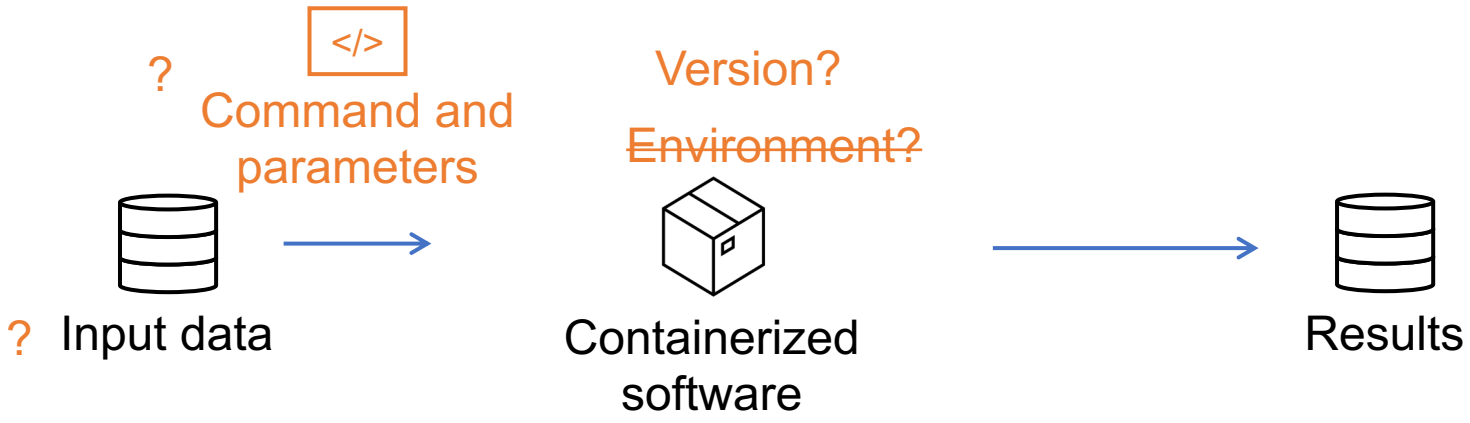
✓ Containerized

Cieslak et al., Nat Methods 2021; Esteban et al., Nat Methods 2019

Gorgolewski et al., Plos Compt Biol 2017

Still, there are obstacles…

Complete reproducibility with automatic data provenance tracking



What we hope:

1. ✓ Use containerized software: use BIDS and BIDS Apps
2. *Automatically* track all provenance + scalable
3. User-friendly
4. Generalizable

Portable containers

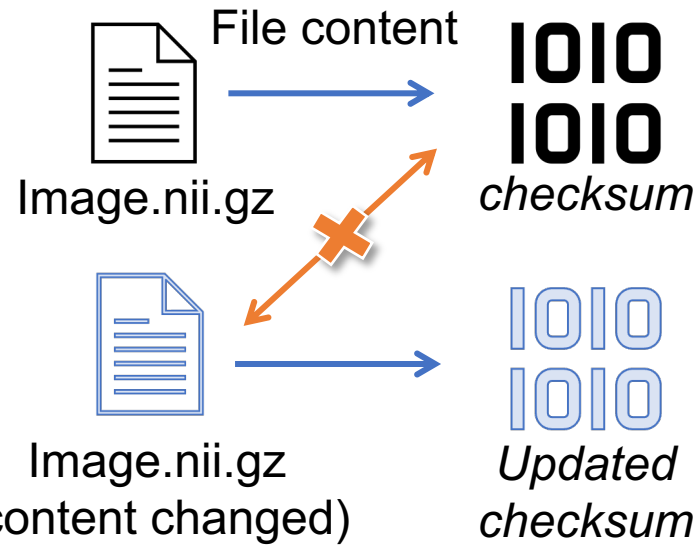
Method for data provenance tracking using DataLad
Data version control

git-annex

git

For data version control

Large? PHI?



- No matter how large the file is
- (Sensitive) contents in the file cannot be retrieved from the checksum

Method for data provenance tracking using DataLad

Functions in DataLad

- DataLad provides machine-readable, re-executable provenance record

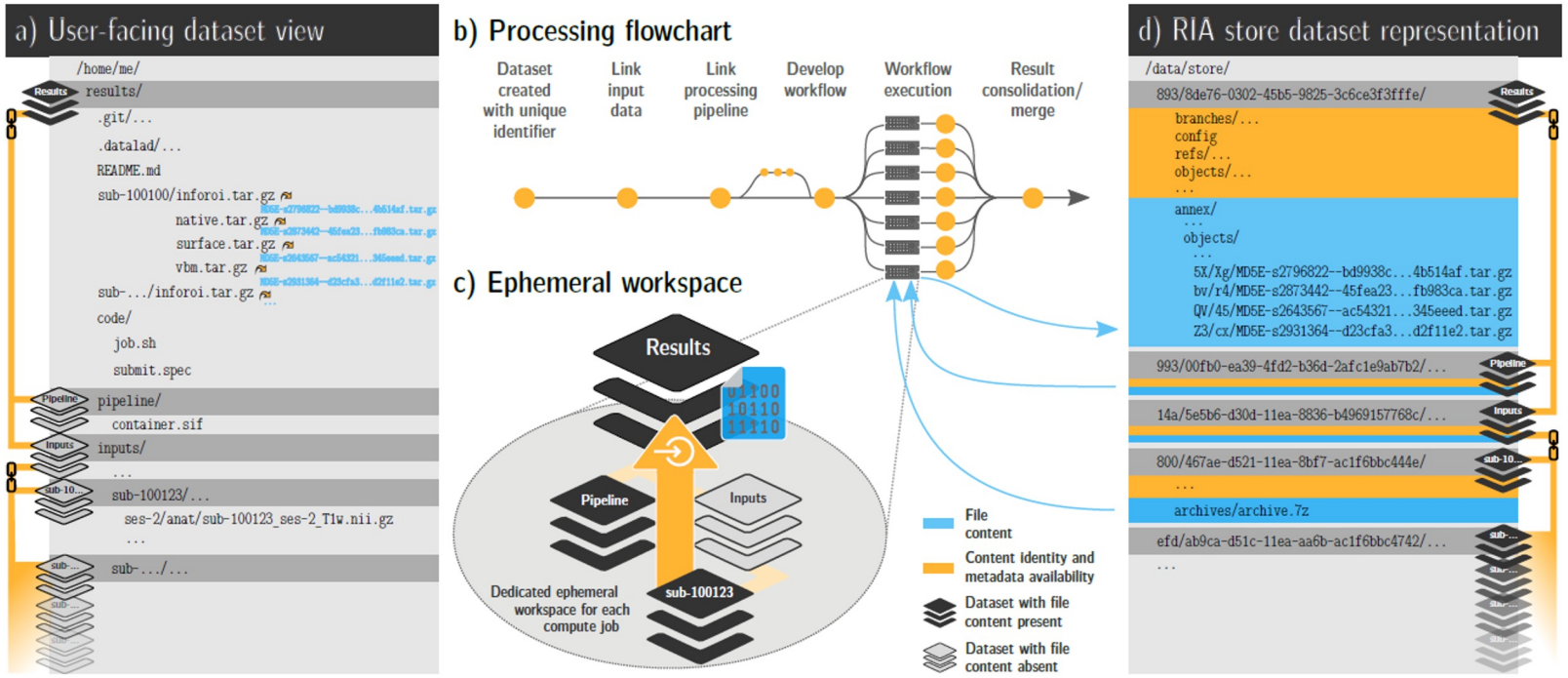
Example provenance record:

```
commit e035f896s45c9fac70cn7cc4dbd0dad43907755p
Author:     Jane Doe <j.doe@fz-juelich.de>
AuthorDate: Wed Feb 10 18:05:30 2021 +0100
Commit:     Jane Doe <j.doe@fz-juelich.de>
CommitDate: Wed Feb 10 18:05:30 2021 +0100

[DATALAD RUNCMD] Compute sub-6025043/ses-2

   === Do not change lines below ===
   {
    "chain": [],
    "cmd": "singularity exec -B {pwd} --cleanenv code/pipeline/.datalad/
            environments/cat/image sh -e -u -x -c  [...]'
    "dsid": "8938de76-0302-45b5-9825-3c6ce3f3fffe",
    "exit": 0,
    "extra_inputs": [
     "code/pipeline/.datalad/environments/cat/image"
    ],
    "inputs": [
     "inputs/ukb/sub-6025043/ses-2/anat/sub-6025043_ses-2_T1w.nii.gz",
     "code/cat_standalone_batch.txt",
     "code/finalize_job_outputs.sh"
    ],
    "outputs": [
     "sub-6025043/ses-2"
    ],
    "pwd": "."
   }
   ^^^ Do not change lines above ^^^
---
 sub-6025043/ses-2/inforoi.tar.gz | 1 +
 sub-6025043/ses-2/native.tar.gz  | 1 +
 sub-6025043/ses-2/surface.tar.gz | 1 +
 sub-6025043/ses-2/vbm.tar.gz     | 1 +
 4 files changed, 4 insertions(+)
```

Wagner et al., Sci Data 2022

Method: FAIRly big framework

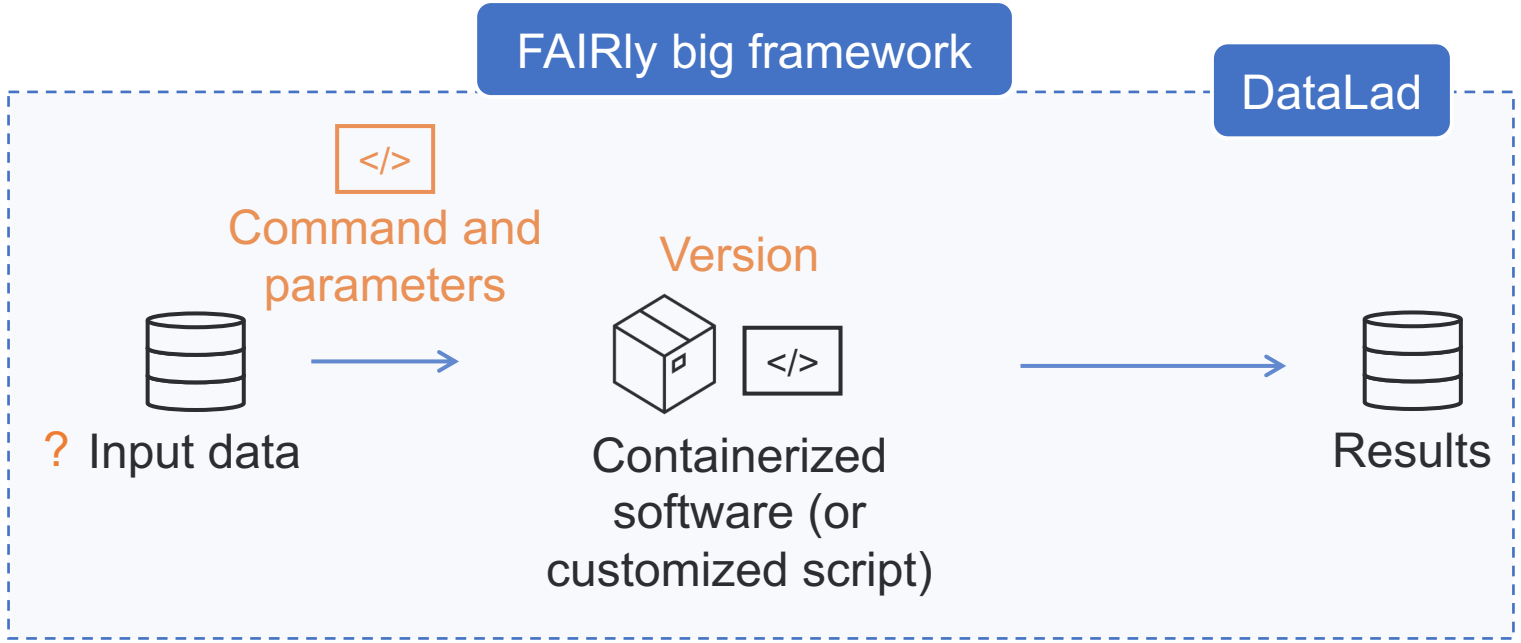
- FAIRly big framework: aka: “The Way”, “The lifestyle”
  - FAIR = findability, accessibility, interoperability, and reusability
  - FAIRly big is a DataLad-based framework for reproducible processing of large-scale datasets.



Wagner et al., Sci Data 2022.

Method: FAIRly big framework

- FAIRly big framework: aka: “The Way”, “The lifestyle”
  - FAIR = findability, accessibility, interoperability, and reusability
  - FAIRly big is a DataLad-based framework for reproducible processing of large-scale datasets.
  - This facilities a full audit trail for processing data at scale



Wagner et al., Sci Data 2022.

Still, there are obstacles…

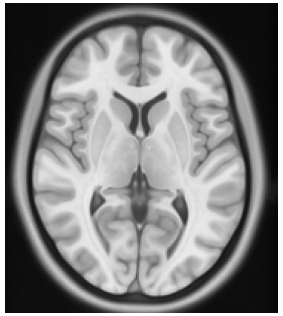
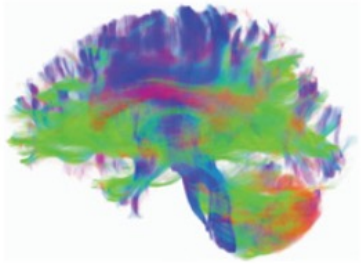
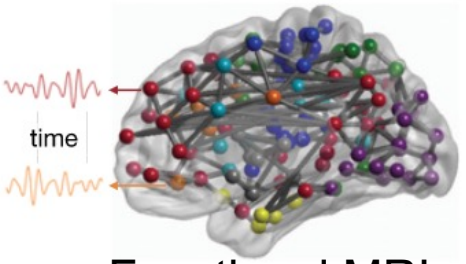
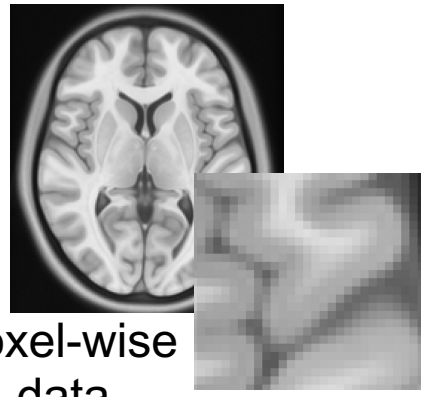
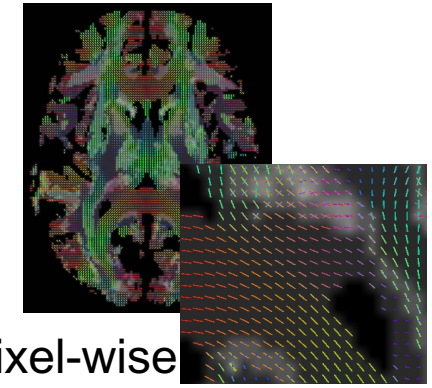
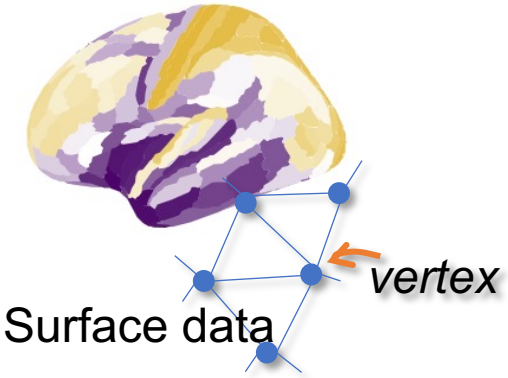
Complete reproducibility with automatic data provenance tracking

What we hope:

1. ✓ Use containerized software: use BIDS and BIDS Apps
2. ✓ *Automatically* track all provenance + scalable
3. User-friendly ?
4. Generalizable ?

Drawbacks in existing solutions

Poor generalizability of existing effort for various processing use cases

Image types	 Structural MRI	 Diffusion MRI	 Functional MRI	...
Data types	 Voxel-wise data	 Fixel-wise data (dMRI)	 Surface data	...
Processing methods and software	ANTs	QSIPrep	MRtrix3	...

Figures adapted from: Baum et al., PNAS 2020; Sydnor et al., Neuron 2021

Drawbacks in existing solutions

Poor generalizability of existing effort for various processing use cases

- One script per pipeline
- Often copied from another script
- Manual work, frequent mistakes
- Hard to version control

Datasets

Single-session vs. Multi-session

Raw BIDS data vs. BIDS derivatives

×

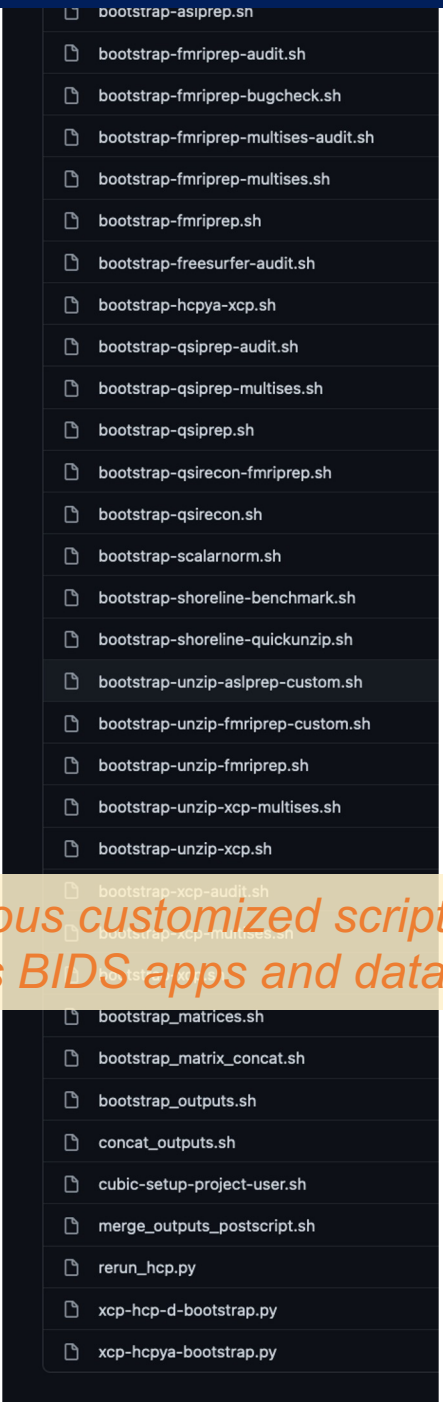
Image types and Processing BIDS apps

sMRI: FreeSurfer, ANTs, …

fMRI: fMRIPrep, XCP-D, …

dMRI: QSIPrep, MRtrix, …

…



Numerous customized scripts for various BIDS apps and datasets

Poor generalizability for use cases → Hard to version control…

Drawbacks in existing solutions

Poor generalizability of existing effort for various cluster systems

- Separate scripts for another cluster management systems

- There are various management systems for clusters, and the commands for job scheduling can vary across systems:

User Commands	SGE (e.g., Penn Medicine CUBIC cluster)	Slurm (e.g., U of Minnesota)
Job submission	qsub [script_file]	sbatch [script_file]
Job deletion	qdel [job_id]	scancel [job_id]
Job status by job	qstat -u * [-j job_id]	squeue [job_id]
…	…	…

Poor generalizability for different clusters → Hard to version control…

Table adapted from: https://srcc.stanford.edu/sge-slurm-conversion

Drawbacks in existing solutions

Not user-friendly

- Numerous steps and functions
- Hard to debug esp. to beginners
- Need to be very careful
- ⚠️⚠️ WARNING ⚠️⚠️

```
#!/bin/bash
# fail on any issue, show commands
set -e -u -x
# name arguments for readability
dssource="$1"
pushgitremote="$2"
subid="$3"

# obtain the analysis dataset, which
# also tracks the required inputs
datalad clone "${dssource}" ds
cd ds

# register location for result
# deposition, separate from the input
# source for performance reasons only
git remote add outputstore
  ↪ "$pushgitremote"

# all job results will be put into
# a job-specific, dedicated branch
git checkout -b "job-$JOBID"

# START OF APPLICATION-SPECIFIC CODE
# pull down input data manually,
# only needed for wildcard-based file
# selection in the next command
datalad get -n "inputs/ukb/${subid}"
```

```
# datalad containers-run executes
# the "cat" computational pipeline.
# specified inputs are auto-obtained,
# specified outputs are saved with
# provenance record
da
```

Part of the full scrip

Running Bids App pipelines

In general, we will need to apply an image processing workflow to the raw data we've curated in BIDS. This workflow is essentially the same for any of the preps, including fMRIPrep, QSIPrep, c-PAC and ASLPrep. In this section we show how to create an *analysis dataset* that contains the prep containers, the provenance of the prep runs, and the prep outputs. Then, we'll demonstrate how to run pipelines on your exemplars (and eventually the full dataset) using the "bootstrap" method.

- Preparing your containers
- Preparing the analysis dataset
  - ⚠️⚠️ WARNING ⚠️⚠️
- Editing the executable code
  - Keeping files from an example run
  - Making sure you're using the correct singularity image
- Running a test subject
  - When jobs are stuck in the "r" state
- After the pipeline runs
  - ⚠️⚠️ WARNING ⚠️⚠️
  - Accessing files created by a bootstrap script
  - Auditing Your Runs
  - Something Went Wrong in BIDS
- Running a bootstrap on the outputs of another bootstrap
- Run single subject testing on interactive node using bootstraps
  - Set up necessary variables
  - Clone and get the dataset content
  - Run the BIDS app on the chosen subject

Docs from our lab

Not user-friendly…

Wagner et al., Sci Data 2022

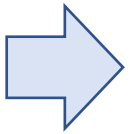
Background and Motivations

Goals and general scope of proposed software BABS

BIDS App Bootstrap ("BABS"): a user-friendly Python package for reproducible image processing at scale

What we hope:

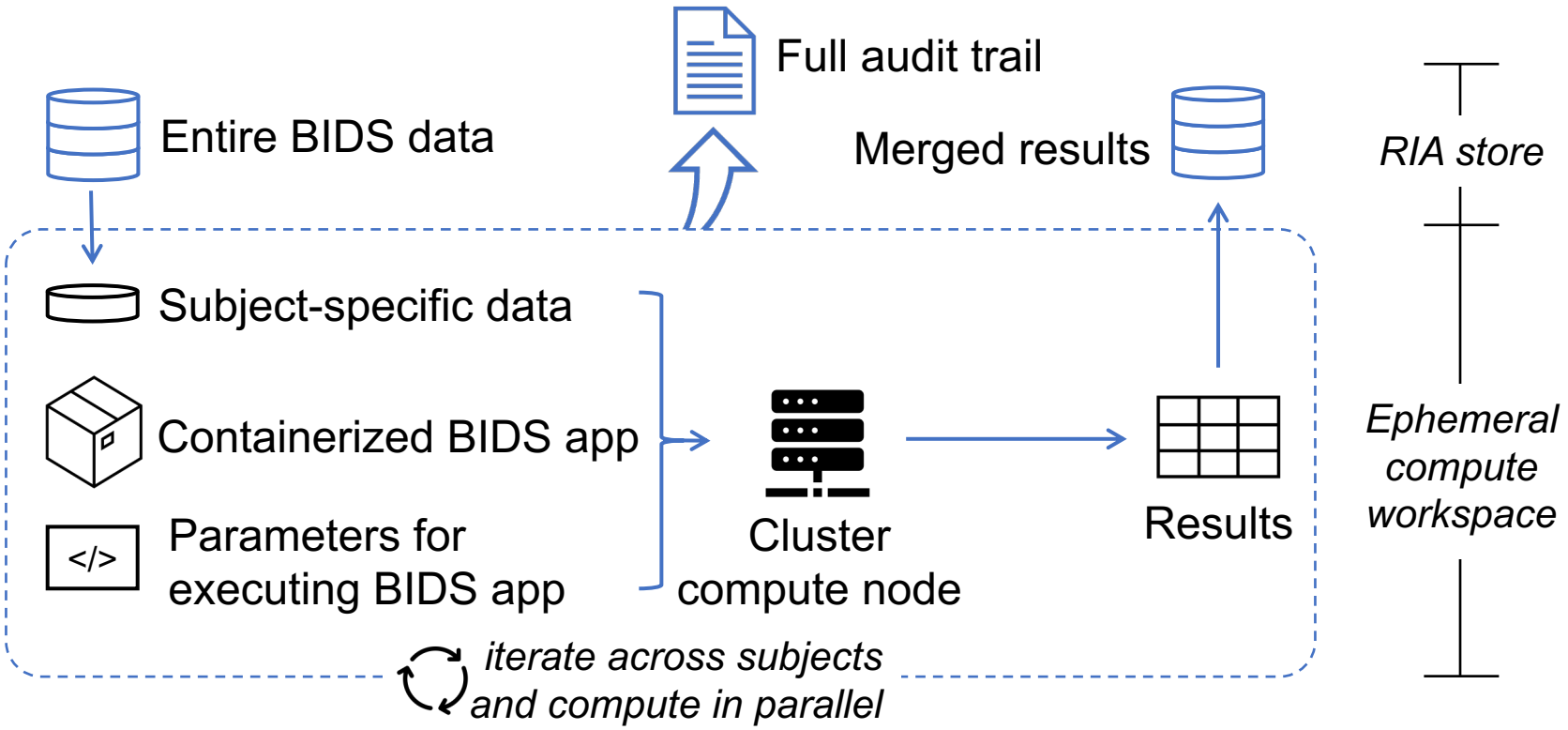
1. Use containerized software: use BIDS and BIDS Apps
2. *Automatically* track all provenance + scalable
3. User-friendly
4. Generalizable



- Reproducible:
  - All provenance tracked (DataLad)
  - Rerunnable (DataLad)
- Scalable:
  - FAIRly big workflow
- Generalizable: compatible with:
  - Various BIDS datasets
  - Various BIDS Apps
  - Popular cluster systems: SGE, Slurm
- User-friendly: BABS will automatically perform the "bootstrap"

"Bootstrap": Setting up a pipeline for these steps and executing it.

Design: BABS follows FAIRly big workflow



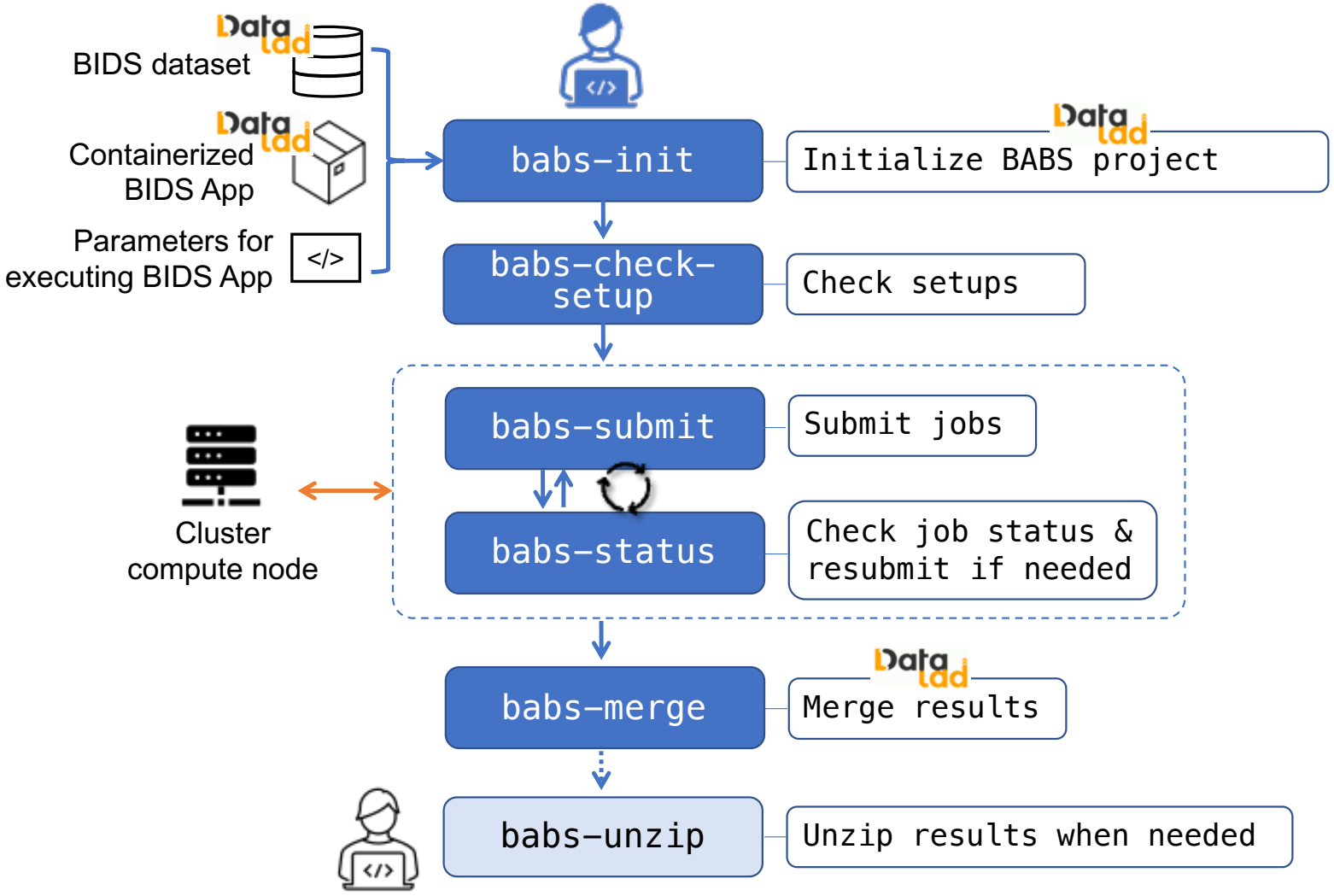
RIA store:

- Permanent store of tracked data
- Can be storage space on HPC
- Files often compressed

BABS developer-oriented workflow

Design: Functionality of BABS

Commands can be called from terminal!



BABS user-oriented workflow

Design: `babs-init` command-line interface

Example call for QSIPrep

```
babs-init \
    --where_project /path/to/my/research \
    --project_name test_babs_multi-ses_qsiprep \
    --input BIDS /path/to/input_BIDS_dataset \
    [--list-sub-file /path/to/initial_included_subject.csv \]
    --container_ds /path/to/qsiprep-container \
    --container_name qsiprep-0-16-0RC3 \
    --container_config_yaml_file /path/to/container_qsiprep_config.yaml \
    --type_session multi-ses \
    --type_system sge
```

Define a BABS project

Input dataset(s)

Container

Other configurations

- See here for example printed messages from `babs-init`
- See next slide for more about `--container-config-yaml-file`
- For more, see the documentation of BABS: https://pennlinc-babs.readthedocs.io/en/latest/

Design: `babs-init` command-line interface

Container's config YAML file

Container's config YAML file: let BABS know instructions for (running) the BIDS App

- Arguments in singularity run of the BIDS App
- Cluster resources needed to run the BIDS App
- Required files needed in each subject (or session)
- Alerting messages in log files that might indicate failure
- …

For more, see the documentation of BABS:
https://pennlinc-babs.readthedocs.io/en/latest/preparation_config_yaml_file.html

Example YAML file:

```yaml
babs_singularity_run:
    -w: "$BABS_TMPDIR"
    --n_cpus: 1
    --stop-on-first-crash: ""
    ...

cluster_resources:
   interpreting_shell: /bin/bash
   hard_memory_limit: 25G
   temporary_disk_space: 200G

required_files:
   $INPUT_DATASET_#1:
      - "func/*_bold.nii*"
      - "anat/*_T1w.nii*"

keywords_alert:
   o_file:
      - "fMRIPrep failed"
...
```

Design: `babs-init` command-line interface

Outputs: BABS project and its content

```
# Folders in BABS project directory:
.
├── analysis                # main folder
|   ├── CHANGELOG.md
|   ├── code
|   ├── containers          # container dataset
|   ├── inputs              # input dataset(s)
|   ├── logs
|   └── README.md
├── input_ria               # datalad sibling RIA of `analysis`
├── merge_ds                # generated after merging is done
└── output_ria              # outputs, also a datalad sibling RIA of `analysis`
```

Design: `babs-submit` and `babs-status` command-line interface

```
babs-submit \
    --project_root /path/to/my_BABS_project \
    [--count number_of_jobs_to_submit]
    [--all]    # submit all remaining jobs
    [--job <sub_id> <ses_id>]  # submit specific job, can be multiple
```

Different choices of job submission

Use one of them at a time

```
babs-status \
    --project_root /path/to/my_BABS_project \
    [--resubmit failed|pending \]
    [--resubmit-job <sub_id> <ses_id> \]  # resubmit specific job, can be multiple
    [--container_config_yaml_file /path/to/container_config.yaml]
    [--job_account]
```

Resubmit jobs if needed

Job auditing

For more, see the documentation of BABS:
https://pennlinc-babs.readthedocs.io/en/latest/

Example outputs from `babs-status`

```
$ babs-status \
    --project_root /path/to/my/BABS/project \
    --container_config_yaml_file /path/to/container.yaml \
    --job-account

Did not request resubmit based on job states (no `--resubmit`).
`--job-account` was requested; `babs-status` may take longer time...

Job status:
There are in total of 2565 jobs to complete.
2565 job(s) have been submitted; 0 job(s) haven't been submitted.
Among submitted jobs,
376 job(s) are successfully finished;
1900 job(s) are pending;
286 job(s) are running;
3 job(s) are failed.

Among all failed job(s):
1 job(s) have alert message: '.o file: fMRIPrep failed';
2 job(s) have alert message: 'BABS: No alert keyword found in log files.';

Among job(s) that are failed and don't have alert keyword in log files:
2 job(s) have job account of: 'qacct: failed: 37 : qmaster enforced h_rt, h_cpu, or h_vmem
limit';

All log files are located in folder: /path/to/my/BABS/project/analysis/logs
```

Job status summary

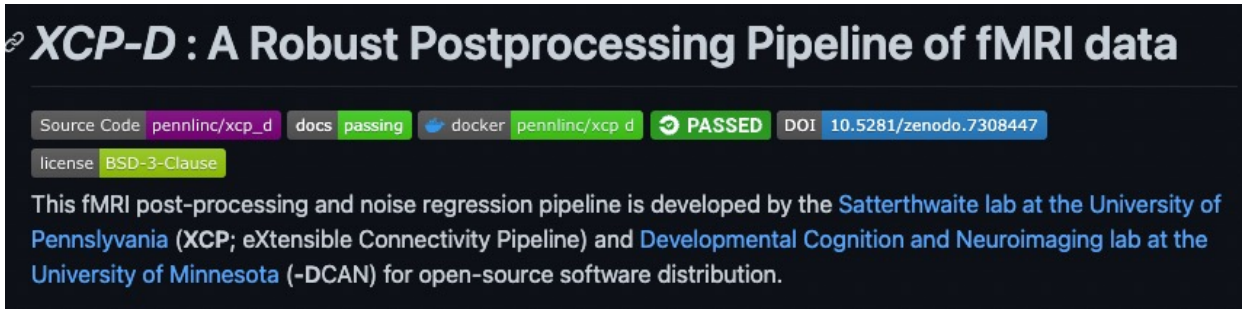
Failed job auditing

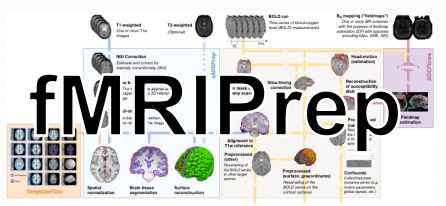
Current progress

- Tested on different cases:

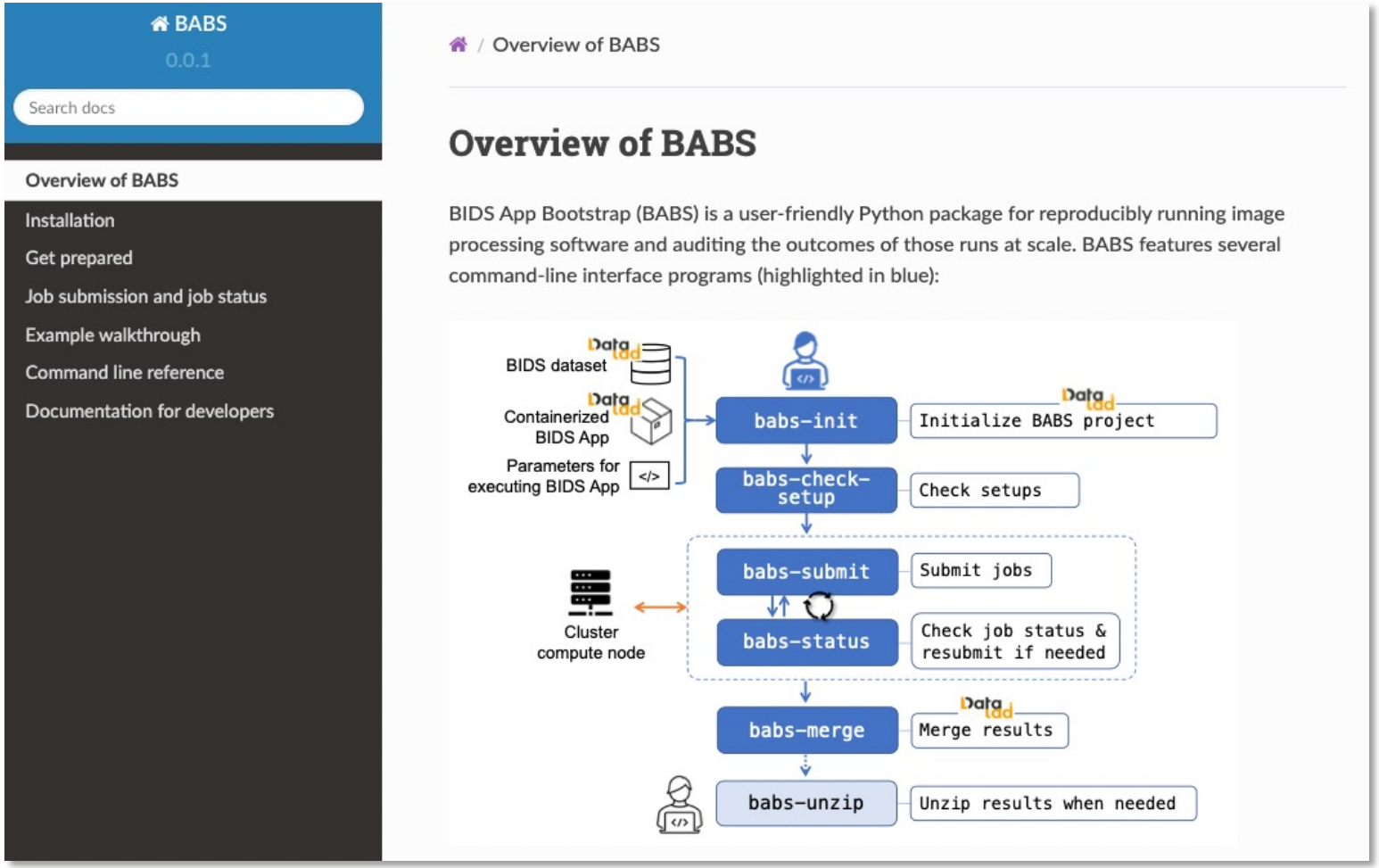
Use cases / BIDS Apps	1st input dataset	2nd input dataset
fMRIPrep (for fMRI)	Raw BIDS (unzipped)	N/A
fMRIPrep (with FreeSurfer results ingressed)	Raw BIDS (unzipped)	FreeSurfer results (BIDS derivatives, zipped)
QSIPrep (for dMRI)	Raw BIDS (unzipped)	N/A
XCP (for fMRI)	fMRIPrep results (BIDS derivatives, zipped)	N/A

- Tested on large-scale dataset (Healthy Brain Network [HBN], n=2,565) with fMRIPrep
- Made sure BABS works on SGE (Penn Medicine CBICA cluster)



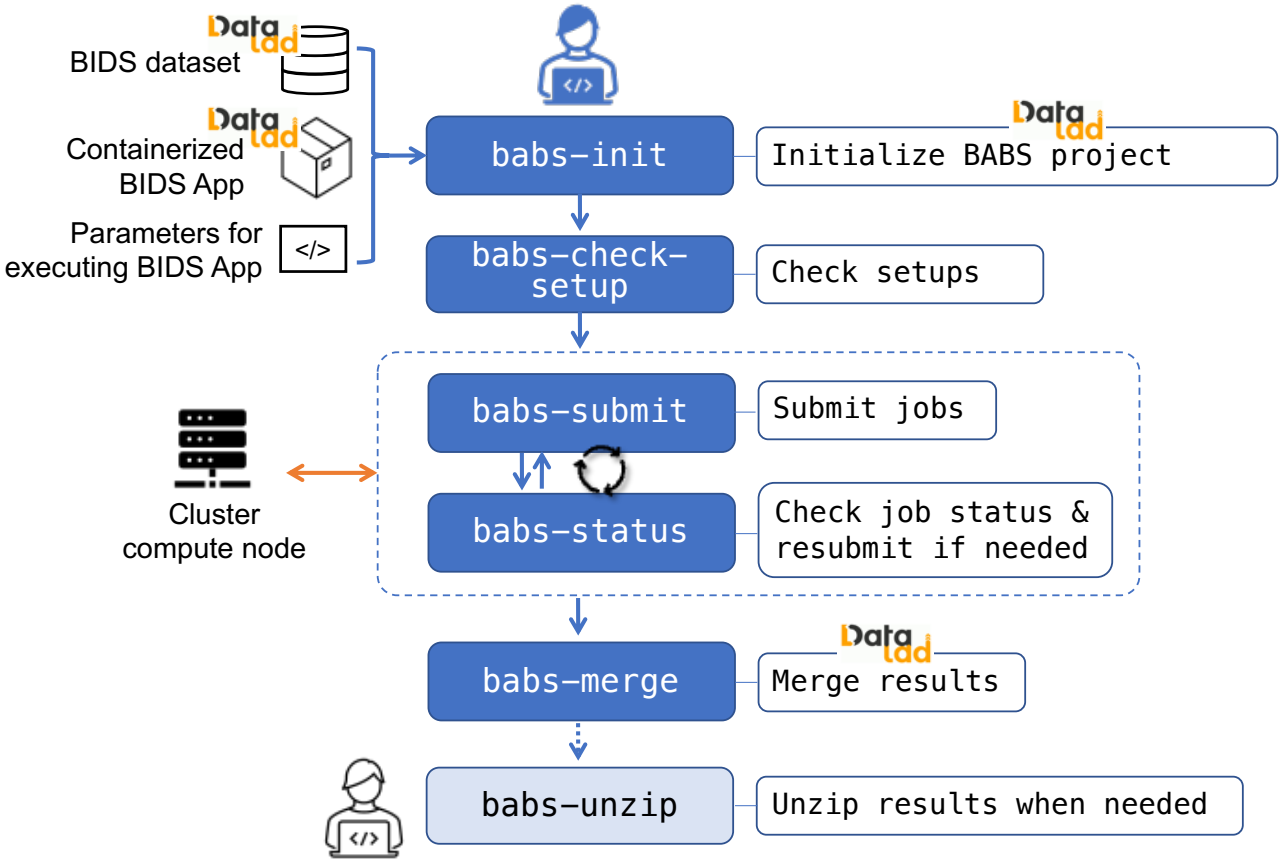


Current progress
Documentation: ongoing



Documentation: https://pennlinc-babs.readthedocs.io/en/latest/
Source code of BABS: https://github.com/PennLINC/babs

What's next?



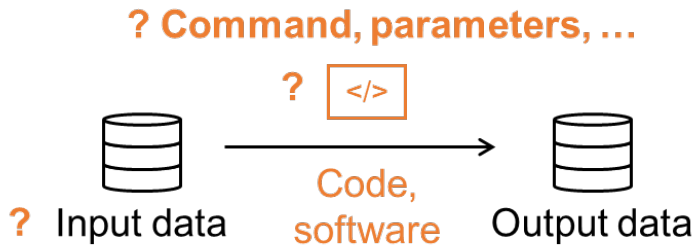
- Get SGE part done!
- Generalize BABS to Slurm
  - Collaborating with Ghosh group (MIT) and Halchenko group (Dartmouth)
- Set up automatic testing
- Finish documentation

- Also collaborating with Hanke group (aka DataLad group)!

Summary

Challenges

Large-scale datasets processing



Reproducibility

Large N

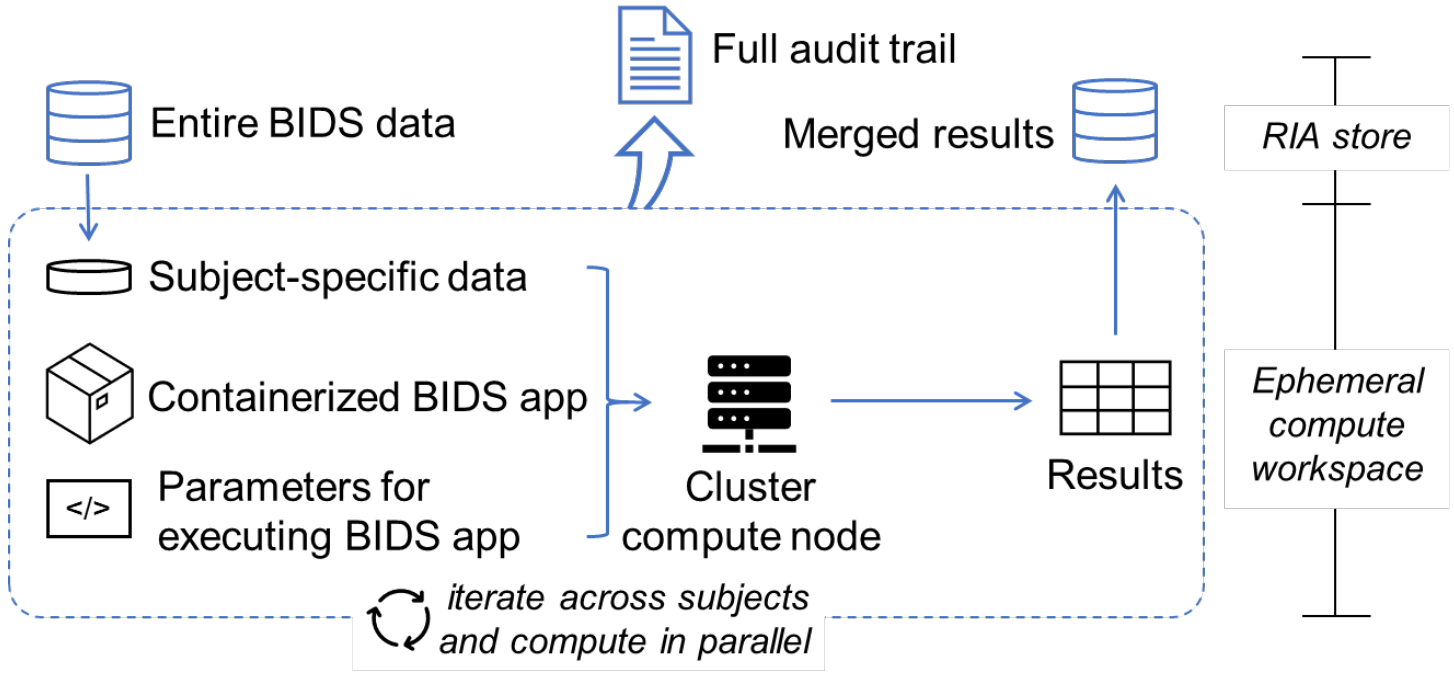
Scalability

- fMRI, dMRI, …
- Volume, surface data, …
- fMRIPrep, QSIPrep, …

Generalizability; User-friendly

Proposed solution

BIDS App Bootstrap (BABS)



For reproducibly processing BIDS dataset at scale:

✓ **Automatically tracking all data provenance**

✓ **User-friendly, generalizable software**

Acknowledgement

PennLINC

Ted Satterthwaite

Matt Cieslak

Sydney Covitz

Taylor Salo

Research Center Jülich, Germany

Michael Hanke

Felix Hoffstaedter

Simon B. Eickhoff

MIT

Satrajit Ghosh

Dorota Jarecka

Dartmouth College

Yaroslav O Halchenko

Austin S. Macdonald

Thank you !!!