

Reproducible, generalizable, and scalable analytic software for large neuroimaging datasets

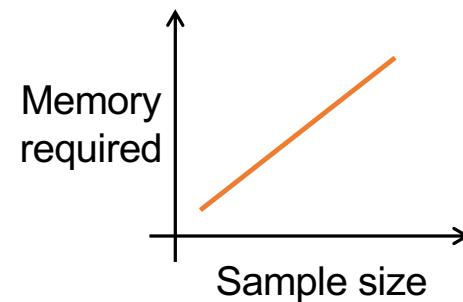
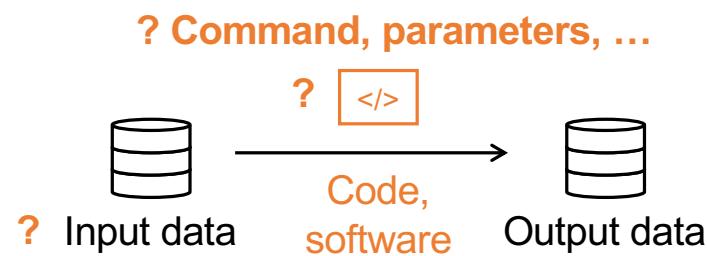
Dissertation Proposal Defense

Chenying Zhao

April 1st, 2022

Overview – The challenge: Reproducibility in large-scale neuroimaging research

- Reproducibility crisis in neuroscience and neuroimaging, esp. with large-scale datasets
 - Fully reproducible research requires a full audit trail of data processing; however existing tools are not user-friendly
 - Memory intensive for e.g., mass-univariate hypothesis testing
 - Analytic tools are often not generalizable for various image types, data types and processing methods.



- fMRI, dMRI, ...
- Volume, surface data, ...
- QSIPrep, ASLPrep, ...

Reproducibility

Scalability

Generalizability

Overview – Proposed solutions

**To develop reproducible, generalizable, and scalable analytic software
for large neuroimaging datasets**

- **Aim 1:** To develop and release an open-source software package for reproducible image processing at scale.
- **Aim 2:** To develop and release a generalizable, memory-efficient, open-source R package for mass-univariate hypothesis testing of large neuroimaging datasets.

Outline

- Aim 1: Software for reproducible image processing at scale
- Aim 2: Memory-efficient, generalizable software for statistical analysis
- Software development

Outline

- Aim 1: Software for reproducible image processing at scale
- Aim 2: Memory-efficient, generalizable software for statistical analysis
- Software development

Outline: Approach for Aim 1

To develop and release an open-source software package for reproducible image processing at scale

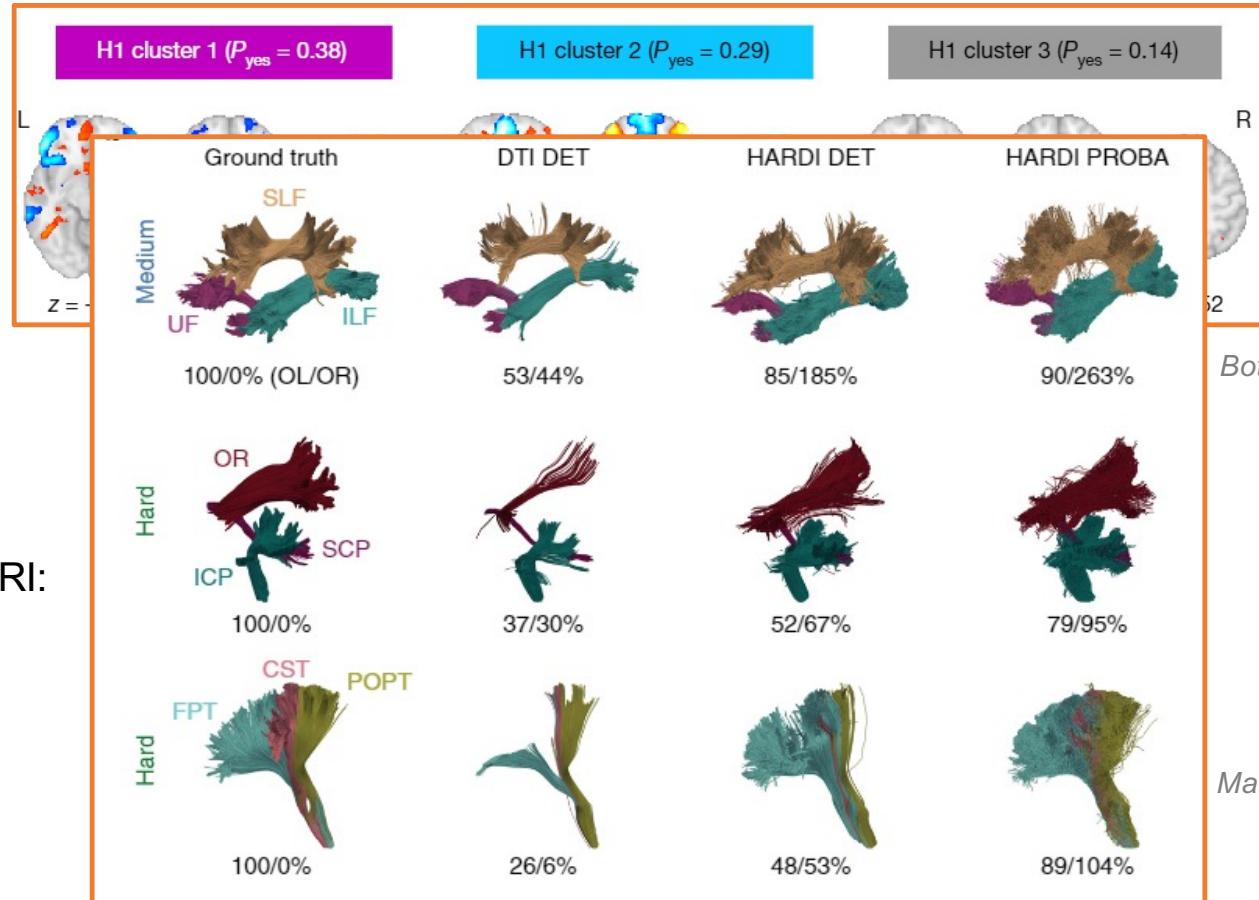
- Approach for **Aim 1:**
 - Background
 - BIDS and BIDS apps
 - Method for data provenance tracking using DataLad
 - Workflow and functionality of proposed software
 - Compatibility with computing cluster management systems
 - Evaluation data and details

Aim 1 – Background

Reproducibility crisis in neuroimaging studies

- Even with the same data, different analytical methods and tools can lead to discordant results:

Functional
MRI:



Botvinik-Nezer et al., Nature 2020

Diffusion MRI:

Maier-Hein et al., Nat Comm 2017

Aim 1 – Background

Large-scale neuroimaging datasets can enhance statistical power

- 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+)
- ...

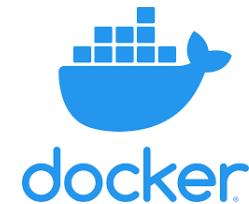
Aim 1 – Background

Challenges in reproducibly processing large-scale datasets and emerging solutions

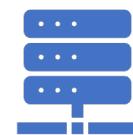
- Large-scale datasets:
 - ☺ enhance statistical power
 - ☹ complexity, heterogeneity, and huge sample size
- Emerging solutions:



Code version control tools



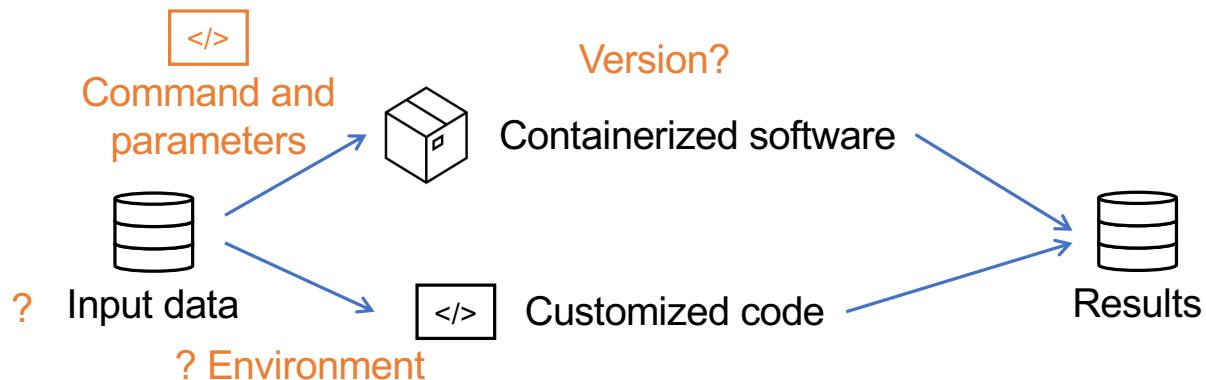
Portable containers



High performance computing (HPC) clusters

Aim 1 – Still, there are obstacles...

Complete reproducibility with automatic data provenance tracking



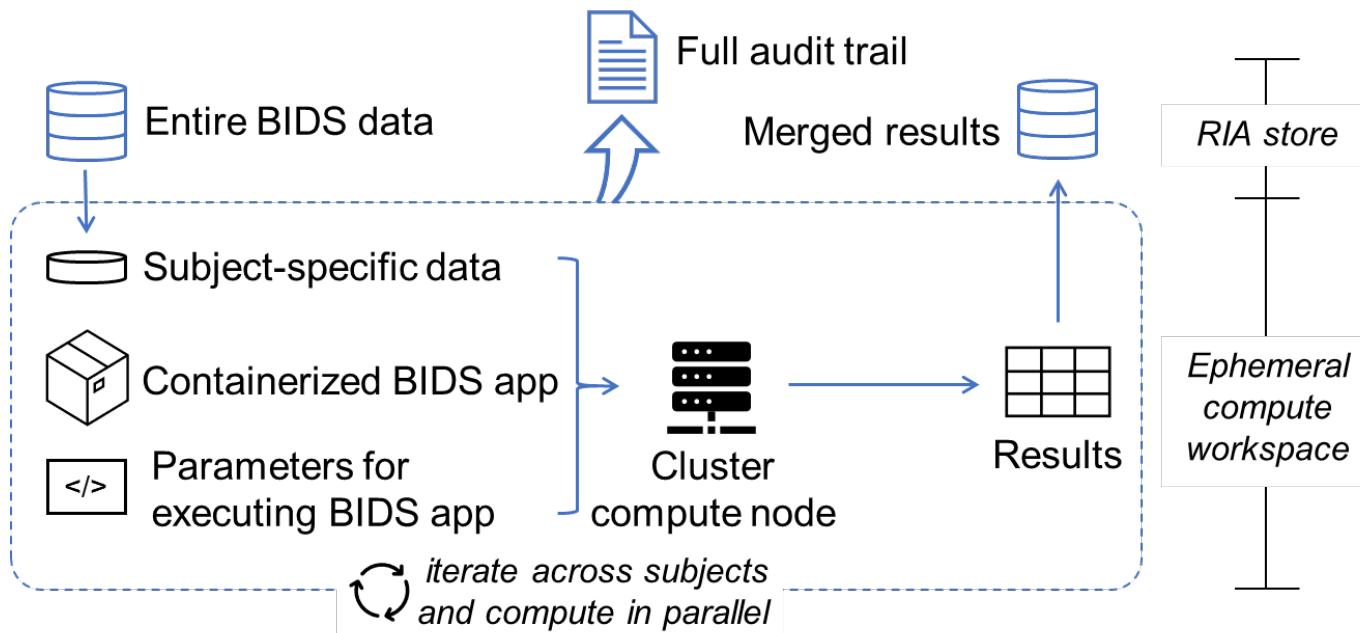
- ✗ Manually track data provenance
- ✓ Automatically track data provenance
 - DataLad¹
 - FAIRly big workflow²

- However, not practical for most academic investigators
 - Numerous steps and DataLad functions
 - Highly customized code

¹Halchenko et al., J Open Source Softw 2021; ²Wagner et al., bioRxiv 2021

Aim 1 – Proposed software: BABS

Proposed software package: BIDS-App Bootstrap (BABS)



→ Automatically tracking all data provenance

→ User-friendly, generalizable software

Aim 1 – BIDS and BIDS apps

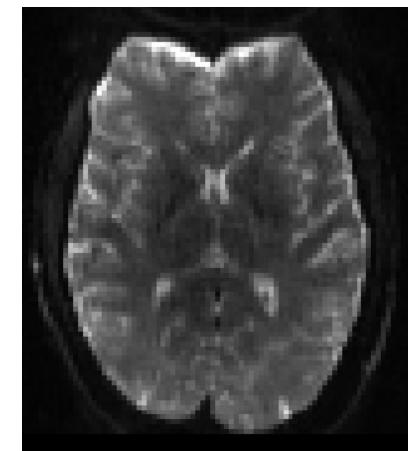
What is Brain Imaging Data Structure (BIDS) format?

- BIDS includes images + their metadata in sidecar JSON files:

```
dicomdir/
  1208200617178_22/
    1208200617178_22_8973.dcm
    1208200617178_22_8943.dcm
    1208200617178_22_2973.dcm
    1208200617178_22_8923.dcm
    1208200617178_22_4473.dcm
    1208200617178_22_8783.dcm
    1208200617178_22_7328.dcm
    1208200617178_22_9264.dcm
    1208200617178_22_9967.dcm
    1208200617178_22_3894.dcm
    1208200617178_22_3899.dcm
  1208200617178_23/
  1208200617178_24/
  1208200617178_25/
```



```
my_dataset/
  participants.tsv
  sub-01/
    anat/
      sub-01_T1w.nii.gz
    func/
      sub-01_task-rest_bold.nii.gz
      sub-01_task-rest_bold.json
    dwi/
      sub-01_dwi.nii.gz
      sub-01_dwi.json
      sub-01_dwi.bval
      sub-01_dwi.bvec
  sub-02/
  sub-03/
  sub-04/
```

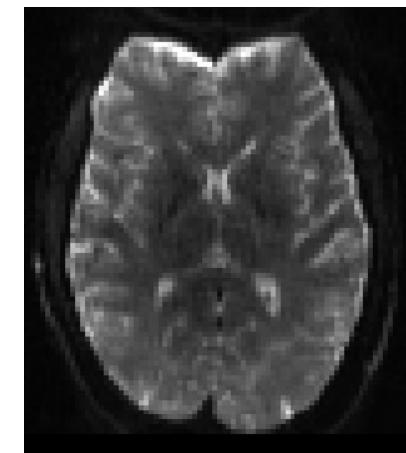
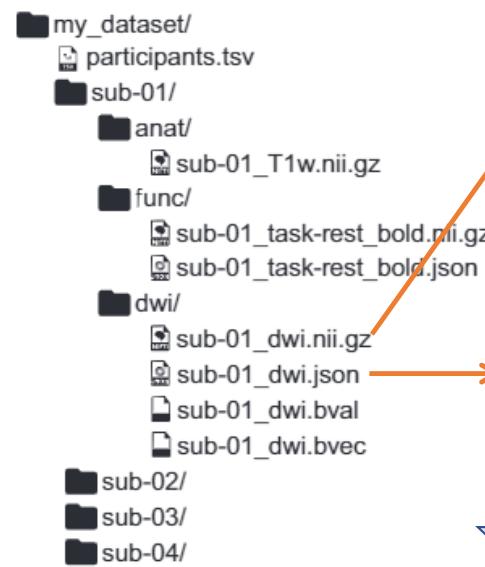
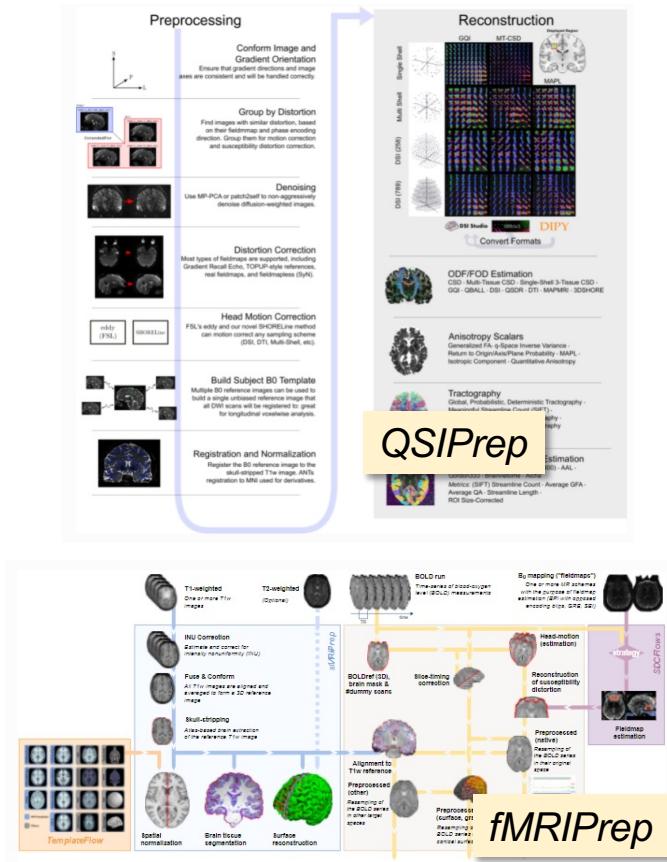


Phase encoding direction

```
"ManufacturersModelName": "TrioTim",
"PhaseEncodingDirection": "j-",
"ProcedureStepDescription": "RESEARCH_colcombe",
"ProtocolName": "DIFF_137_AP",
"RepetitionTime": 2.4,
"ScanningSequence": "FP"
```

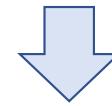
Aim 1 – BIDS and BIDS apps

What are BIDS apps?



Phase encoding direction

```
"ManufacturersModelName": "TricTim",
"PhaseEncodingDirection": "j-",
"ProcedureStepDescription": "RESEARCH_colcombe",
"ProtocolName": "DIFF_137_AP",
"RepetitionTime": 2.4,
"ScanningSequence": "EP"
```

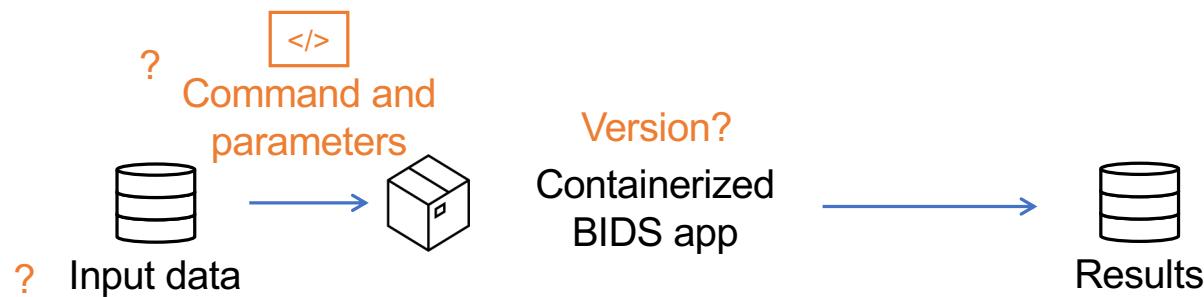


BIDS Apps: ✓ Automatically configure
✓ Containerized

Aim 1 – BIDS and BIDS apps

What is lacked for complete reproducibility?

Applying containerized BIDS apps enhances reproducibility, however...



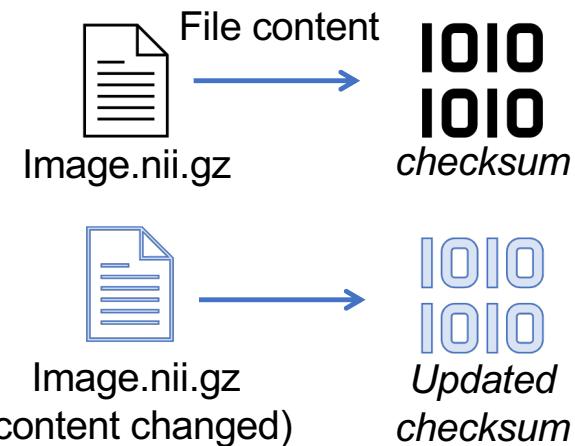
BABS will fill this gap and maximize reproducibility

Aim 1 – Method for data provenance tracking using DataLad

Data version control



For data version control



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

Aim 1 – 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

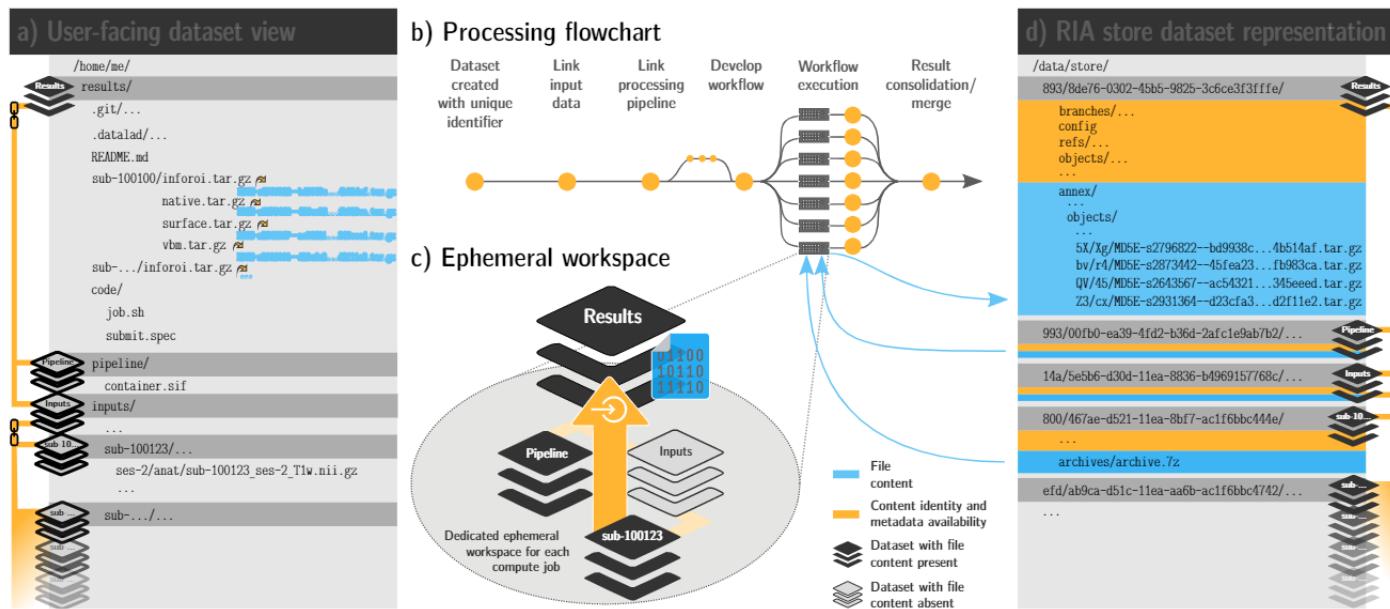
[DATALAD RUNCMD] Compute sub-6025043/ses-2

==== Do not change lines below ====
{
  "chain": [],
  "cmd": "singularity exec -B {pwd} --cleanenv code/pipeline/.datalad/
          environments/cat/image sh -e -u -x -c [...]"
  "dsid": "8938de76-0302-45b5-9825-3c6ce3f3ffe",
  "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(+)
```

Aim 1 – Workflow of BABS

- BABS will follow the workflow of FAIRly big framework
 - FAIR = findability, accessibility, interoperability, and reusability
 - FAIRly big is a DataLad-based framework for reproducible processing of large-scale datasets.
 - This facilitates a full audit trail for processing data at scale



Background – Still, there are obstacles... Impractical adoption of existing effort

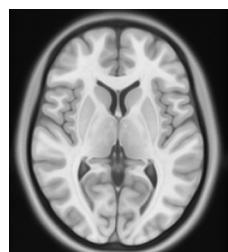
```
1#!/bin/bash
2# fail on any issue, show commands
3set -e -u -x
4# name arguments for readability
5dssource="$1"
6pushgitremote="$2"
7subid="$3"
8
9# obtain the analysis dataset, which
10# also tracks the required inputs
11datalad clone "${dssource}" ds
12cd ds
13
14# register location for result
15# deposition, separate from the input
16# source for performance reasons only
17git remote add outputstore
18  "$pushgitremote"
19# all job results will be put into
20# a job-specific, dedicated branch
21git checkout -b "job-$JOBID"
22
23# START OF APPLICATION-SPECIFIC CODE
24# pull down input data manually,
25# only needed for wildcard-based file
26# selection in the next command
27datalad get -n "inputs/ukb/${subid}"
28# datalad containers-run executes
29# the "cat" computational pipeline.
30# specified inputs are auto-obtained,
31# specified outputs are saved with
32# provenance record
33datalad containers-run \
34  -m "Compute subject ${subid}" \
35  -n cat \
36  --explicit \
37  --output "${subid}" \
38  --input
39  ↪ "inputs/ukb/${subid}/*Tiw.nii.gz"
40# END OF APPLICATION-SPECIFIC CODE
41
42# push result file content to the
43# configured "storage-remote"
44datalad push --to storage-remote
45
46# push branch with provenance records
47# needs a global lock to prevent
48# write conflicts
49flock "$DSLOCKFILE" git push
50  ↪ outputstore
51# log entry to mark non-error exit
52echo SUCCESS
```

Part of the full script

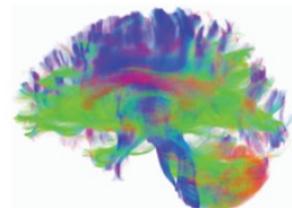
- FAIRly big framework is not user-friendly
- Long scripts
 - Numerous steps ...
 - Numerous DataLad functions ...

Background – Still, there are obstacles...
Poor generalizability of existing effort

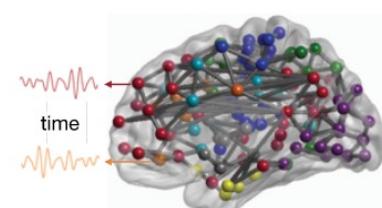
Image types



Structural MRI



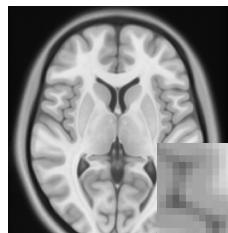
Diffusion MRI



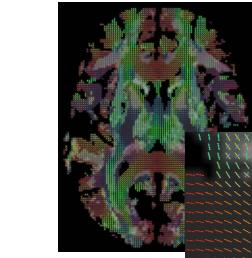
Functional MRI

...

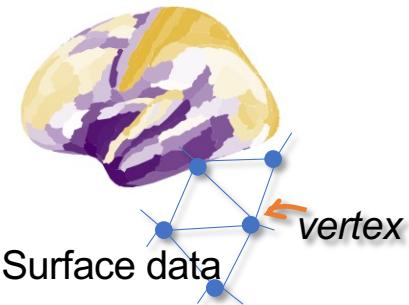
Data types



Voxel data



Pixel data (dMRI)



Surface data

...

Processing methods
and software

ANTs

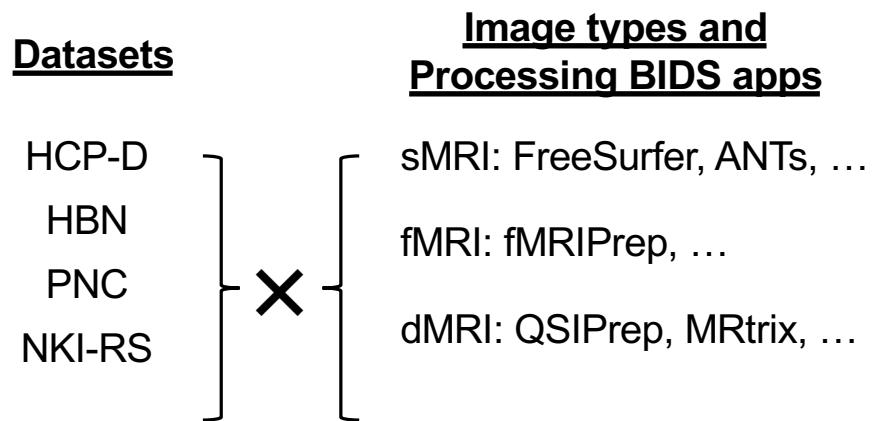
QSIPrep

MRtrix3

...

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

Aim 1 – Still, there are obstacles...
Poor generalizability of existing effort



```
..  
bootstrap-aslprep.sh  
bootstrap-fmriprep-audit.sh  
bootstrap-fmriprep-bugcheck.sh  
bootstrap-fmriprep-multisес-audit.sh  
bootstrap-fmriprep-multisес.sh  
bootstrap-fmriprep.sh  
bootstrap-freesurfer-audit.sh  
bootstrap-hcpya-xcp.sh  
bootstrap-qsiprep-audit.sh  
bootstrap-qsiprep-multisес.sh  
bootstrap-qsiprep.sh  
bootstrap-qsiрecon-fmriprep.sh  
bootstrap-qsiрecon.sh  
bootstrap-scalarnorm.sh  
bootstrap-shoreline-benchmark.sh
```

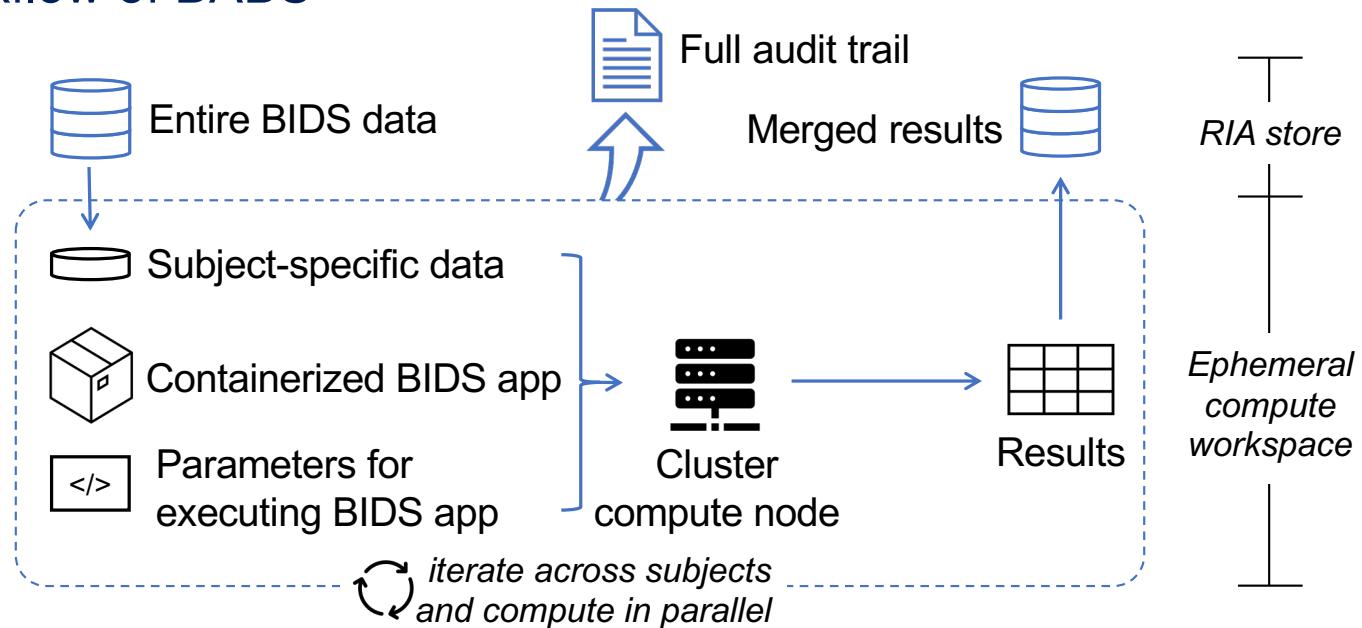
How FAIRly big workflow works currently in practice...



Numerous customized scripts for various BIDS apps and datasets

BABS will be user-friendly, generalizable software

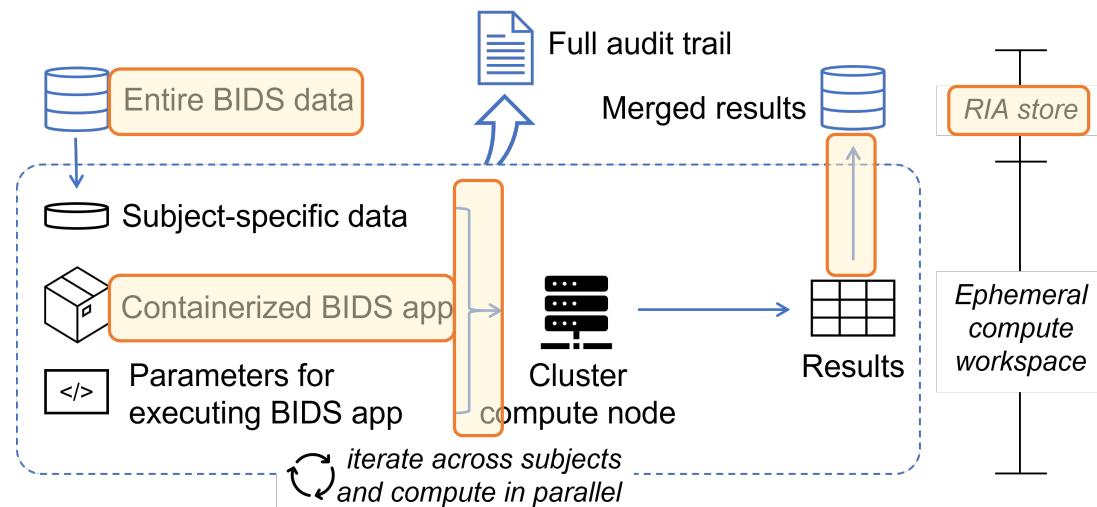
Aim 1 – Workflow of BABS



- ✓ All provenance tracked
- ✓ Fully reproducible rerun
- ✓ Generalizability: not depending on specific information of BIDS dataset and app

Aim 1 – Functionality of BABS

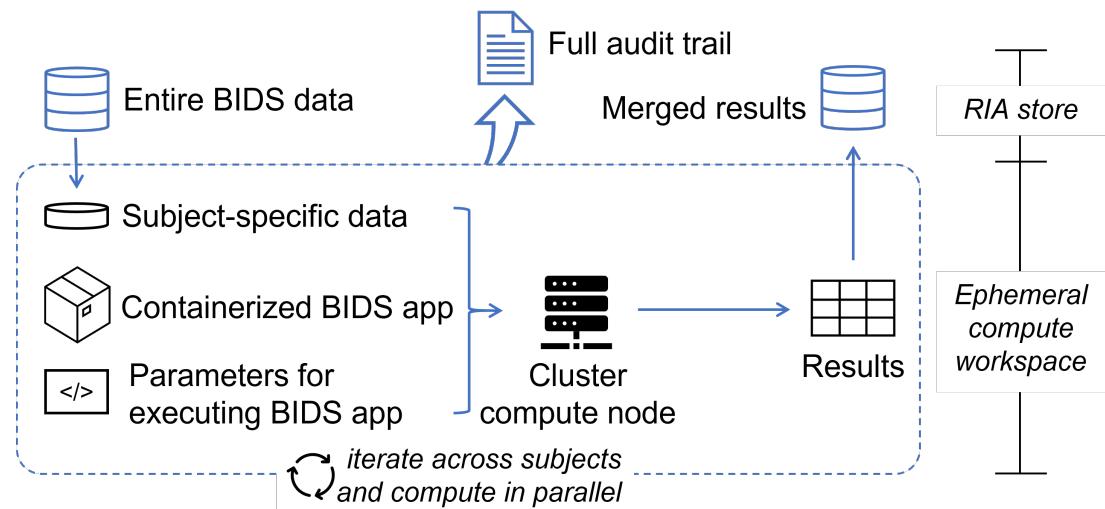
- babs-init: initialize necessities for each run
 - Create RIA store
 - Ask DataLad to start to track data provenance
 - Automatically generate the scripts that will later be used internally
- babs-check-setup
- babs-submit
- babs-status
- babs-merge
- babs-unzip



Proposed BABS workflow

Aim 1 – Functionality of BABS

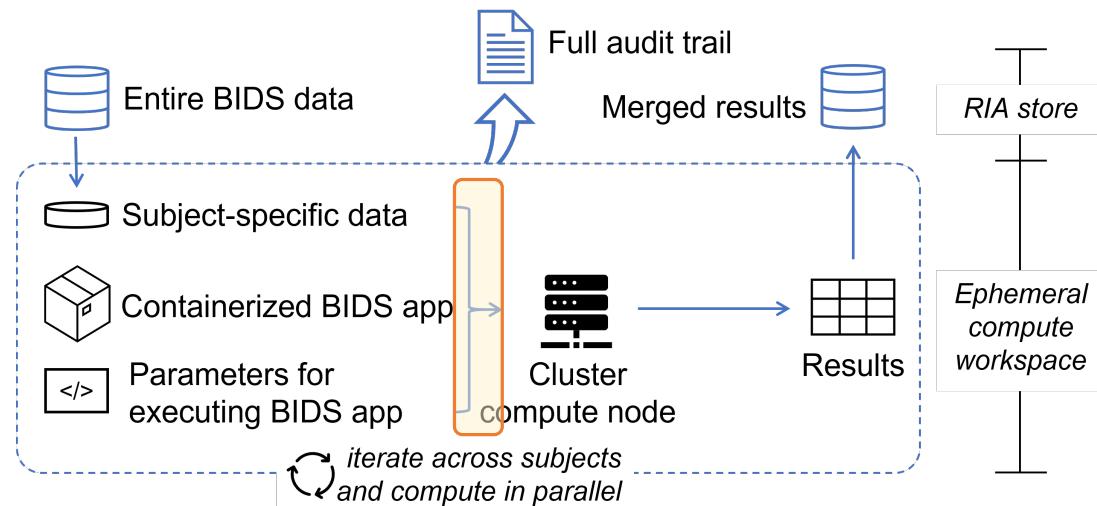
- babs-init: initialize necessities for each run
- babs-check-setup: check all necessities are correctly set up by babs-init
- babs-submit
- babs-status
- babs-merge
- babs-unzip



Proposed BABS workflow

Aim 1 – Functionality of BABS

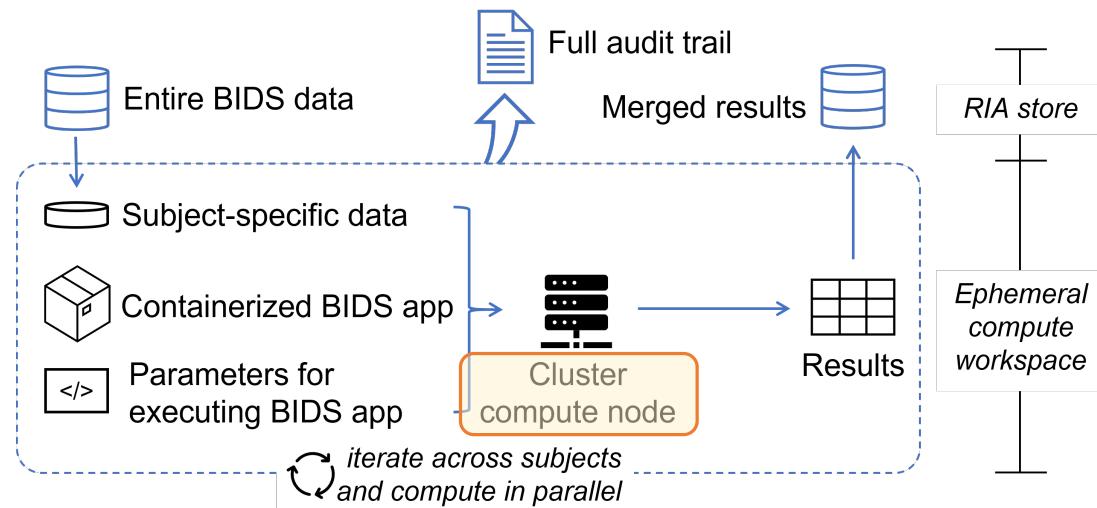
- babs-init: initialize necessities for each run
- babs-check-setup: verify that data and container configuration is appropriate
- babs-submit: submit the jobs to be executed in parallel on cluster compute nodes. Options include:
 - submitting only one subject for testing
 - submitting all subjects but excluding the subjects with successful outputs
- babs-status
- babs-merge
- babs-unzip



Proposed BABS workflow

Aim 1 – Functionality of BABS

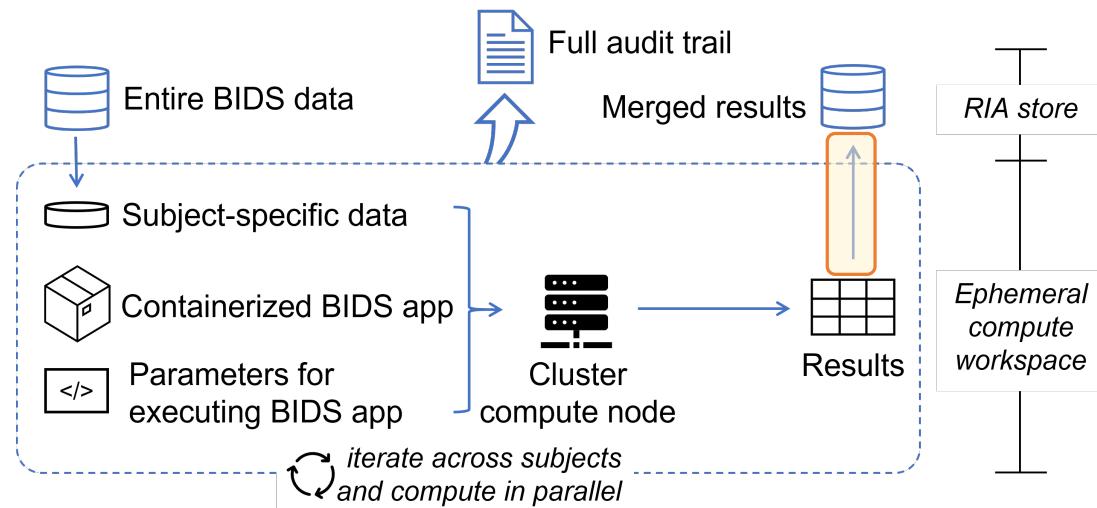
- babs-init: initialize necessities for each run
- babs-check-setup: verify that data and container configuration is appropriate
- babs-submit: submit the jobs to be executed in parallel on cluster compute nodes
- babs-status: easily check all jobs' status, and it allows users to automatically resubmit the jobs if failed or stalled
- babs-merge
- babs-unzip



Proposed BABS workflow

Aim 1 – Functionality of BABS

- babs-init: initialize necessities for each run
- babs-check-setup: verify that data and container configuration is appropriate
- babs-submit: submit the jobs to be executed in parallel on cluster compute nodes
- babs-status: easily check all jobs' status and automatically resubmit if needed
- babs-merge: merge results and provenance from all jobs. Results are now compressed at the RIA store.
- babs-unzip

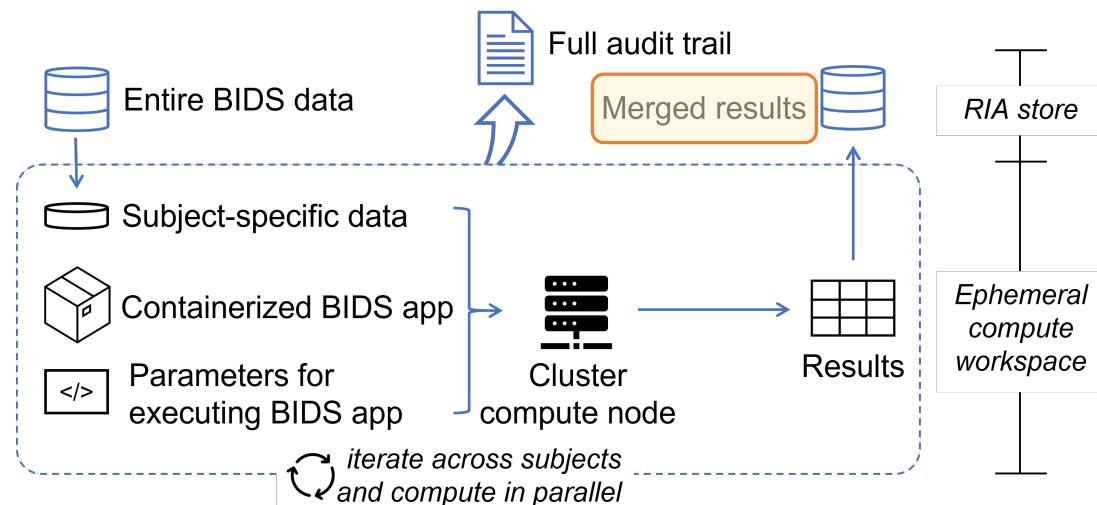


Proposed BABS workflow

Aim 1 – Functionality of BABS

- babs-init: initialize necessities for each run
- babs-check-setup: verify that data and container configuration is appropriate
- babs-submit: submit the jobs to be executed in parallel on cluster compute nodes
- babs-status: easily check all jobs' status and automatically resubmit if needed
- babs-merge: merge results and provenance from all jobs
- babs-unzip: allow users to decompress the zipped data and only get the requested files

- ✓ Parsimonious set of commands
- ✓ Generalizable functionality of BABS
 - any BIDS dataset
 - any BIDS-app
- ✓ Fully reproducible way with complete provenance tracking



Proposed BABS workflow

Aim 1 – Compatibility with computing cluster management systems

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



User Commands	SGE	Slurm
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]
...

- Design of BABS: will be compatible with Slurm and SGE
- Test of BABS: on Slurm and SGE systems, using
 - Slurm cluster through collaborations with University of Minnesota
 - SGE-based CUBIC system at Penn Medicine

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

Aim 1 – Evaluation data and details

BIDS data and app

- Input BIDS data: large-scale diffusion MRI (dMRI) from Reproducible Brain Charts (RBC) effort
 - Four datasets with different dMRI acquisition protocols
 - In total n=6,372 dMRI

Datasets	dMRI			
	Sample size (n)	Resolution	b-values (s/mm ²)	Number of directions
Philadelphia Neurodevelopmental Cohort (PNC)¹	1409	1.875x1.875x2 mm	1000	64
Healthy Brain Network (HBN)²	2151	1.8mm isotropic	1000, 2000	64
NKI Rockland Sample (NKI-RS)³	2169	2mm isotropic	1500	137
Human Connectome Project - Development (HCP-D)⁴	643	1.5mm isotropic	1500, 3000	185
Total	6372			

¹Satterthwaite et al., *NeuroImage* 2014;

²Alexander et al., *Sci Data* 2017;

³Nooner et al., *Front Neurosci* 2012;

⁴Somerville et al., *NeuroImage* 2018

Aim 1 – Evaluation data and details

BIDS data and app

- Containerized BIDS app: QSIPrep
 - For dMRI preprocessing + reconstruction
 - Can process the dMRI with nearly all sampling schemes

Preprocessed data will also be used in Aim 2

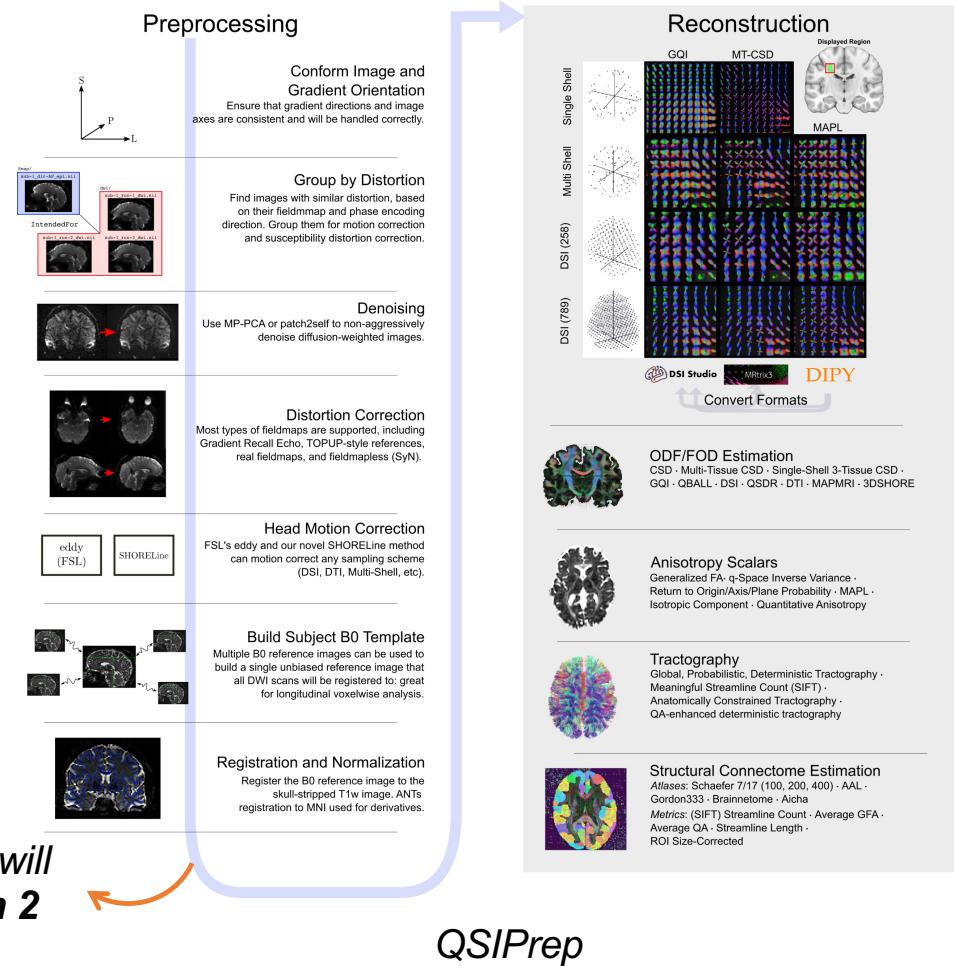
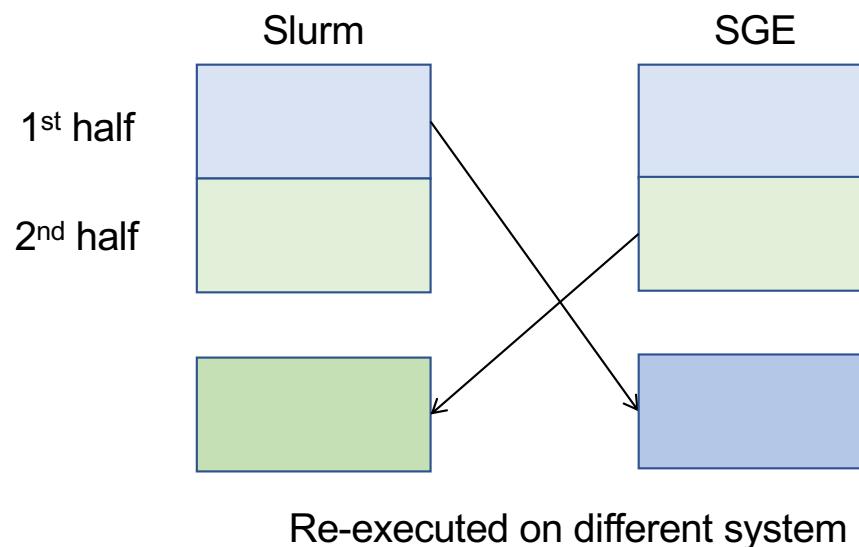


Figure adapted from: Cieslak et al., Nat Methods 2021

Aim 1 – Evaluation data and details

Evaluation details

1. **Compatibility** with both cluster systems: BABS will be applied to all datasets on both Slurm and SGE systems
2. **Reproducibility and capability** of capturing full process provenance even for dataset at scale:
 - To perform a split-half cross-validation:



Aim 1 – Evaluation data and details

Evaluation details: split-half cross-validation

2. To demonstrate the **reproducibility and capability** of capturing full process provenance even for dataset at scale: To perform a split-half cross-validation:
 - To compare the differences between the output connectivity matrices from original run and rerun
 - Expect that there is minimal difference

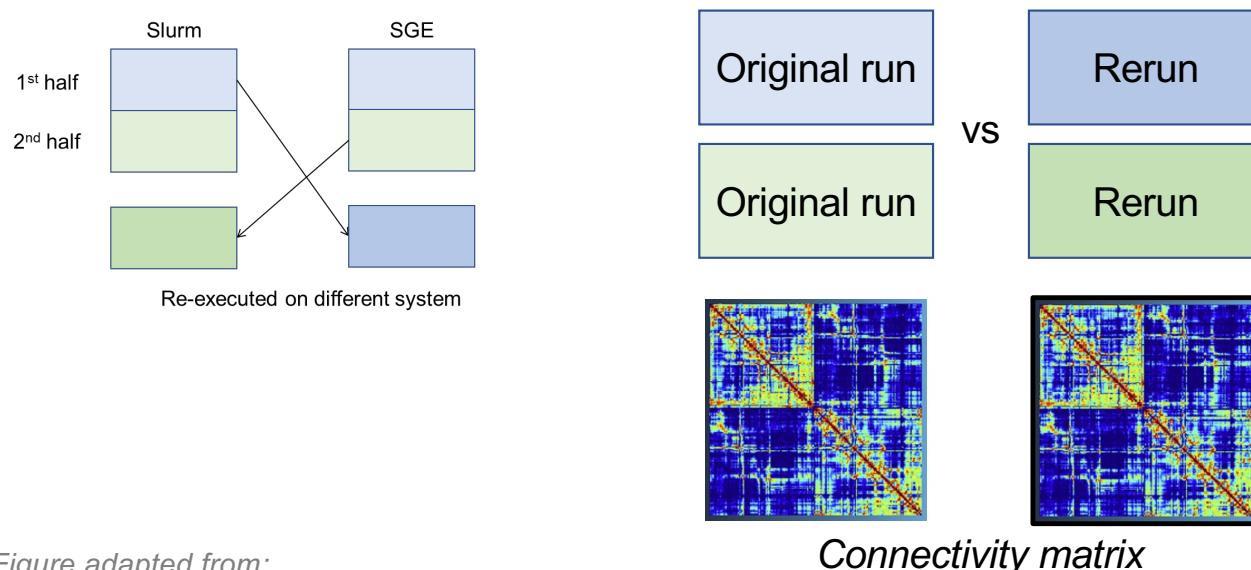


Figure adapted from:
Baum et al., *NeuroImage* 2018

Outline

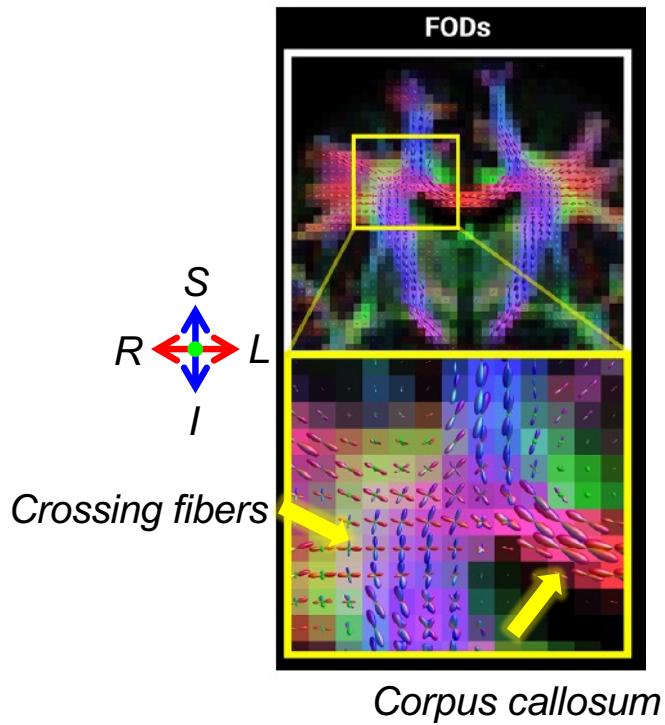
- Aim 1: Software for reproducible image processing at scale
- Aim 2: Memory-efficient, generalizable software for statistical analysis
- Software development

Outline: Approach for Aim 2

To develop and release a generalizable, memory-efficient, open-source R package for mass-univariate hypothesis testing of large neuroimaging datasets

- Approach for **Aim 2:**
 - Background
 - Method for efficient memory
 - Workflow and functionality of proposed software
 - Evaluation data and details
 - Application of GAMs to developmental datasets
 - Generalizability and extensibility

Aim 2 – Background: What is fixel data?

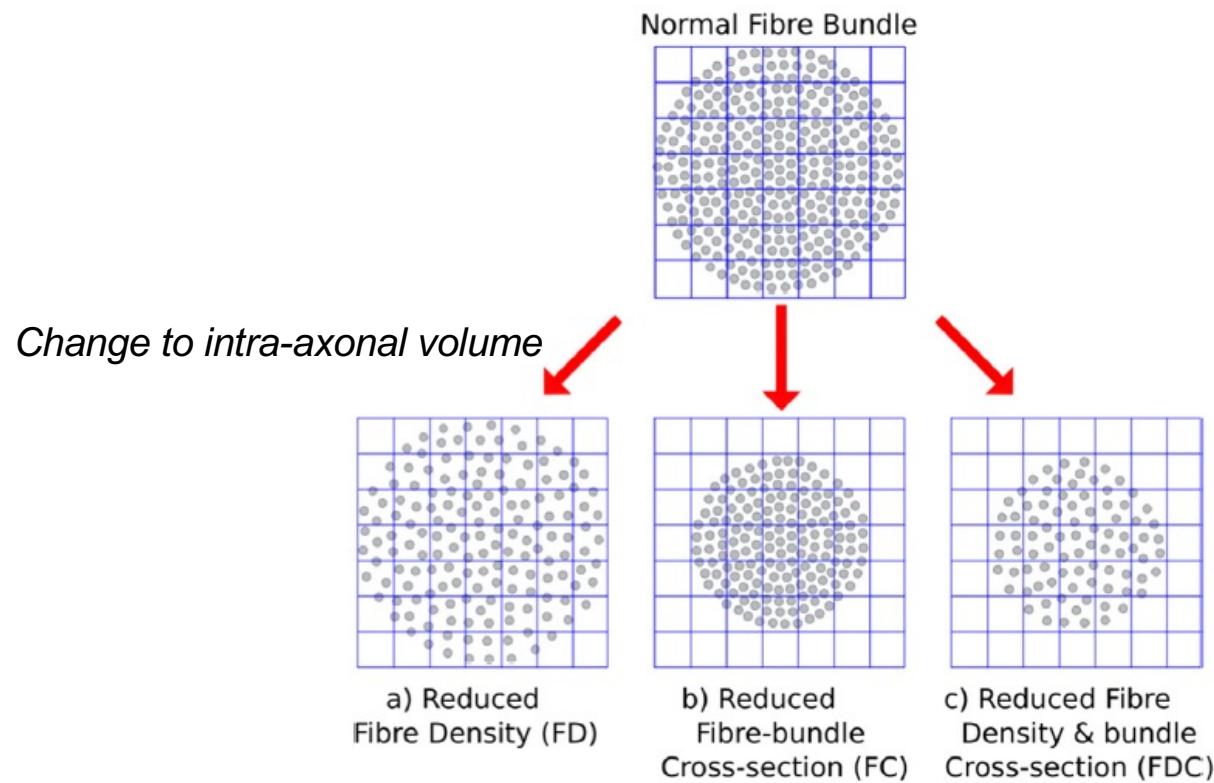


*FOD, fiber orientation distribution
FD, fiber density*

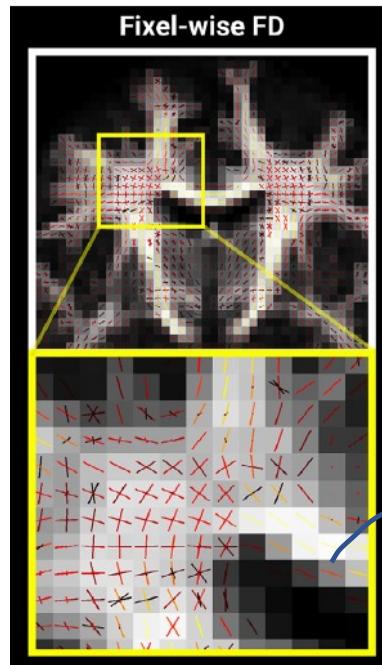
- “Fixel”: an individual fiber population within a voxel

*Dhollander et al., NeuroImage 2021
Raffelt et al., NeuroImage 2015*

Aim 2 – Background: What is fixel data? Fixel-wise metrics



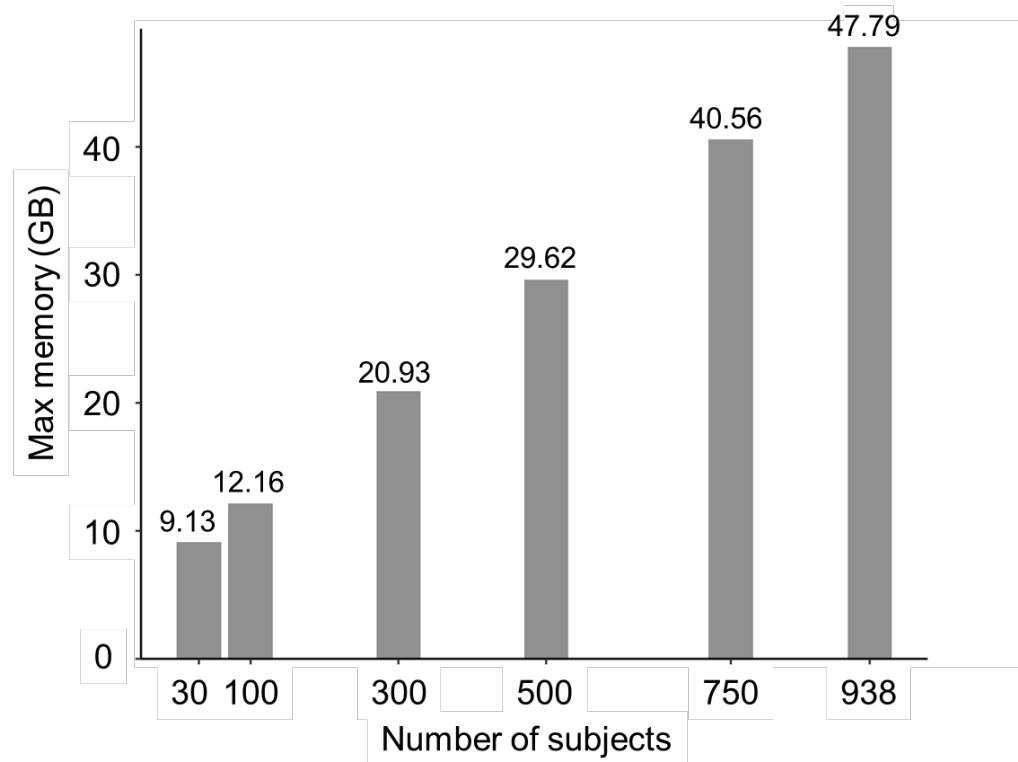
Aim 2 – Background: Mass-univariate hypothesis testing



Does fixel-wise FD change over development?
→ To perform a mass-univariate hypothesis testing:
@ each fixel, test $FD \sim age$

Aim 2 – Background

Challenge #1: Intensive memory requirement for large-scale dataset



*Software MRtrix, for fixel-wise
linear model: FDC ~ age*

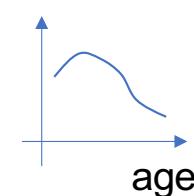
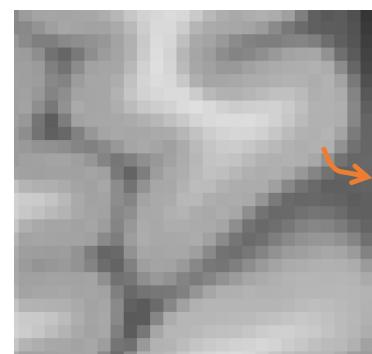
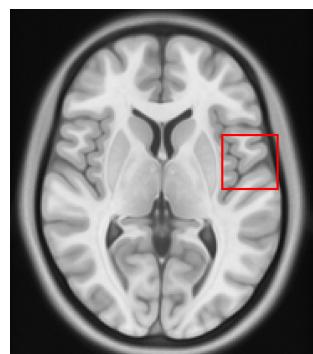


Using HCP cluster?

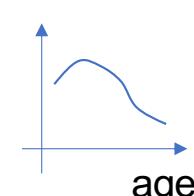
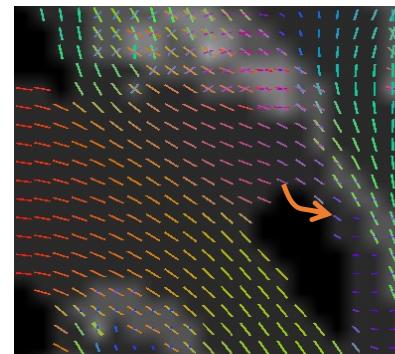
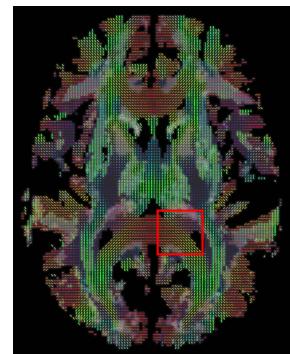
Aim 2 – Background

Challenge #2: No consistent statistical analysis tool generalized across image types

Voxel data



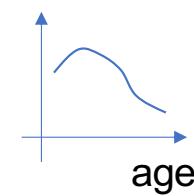
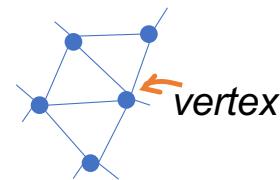
Pixel data



Surface vertex data



Sydnor et al., Neuron 2021

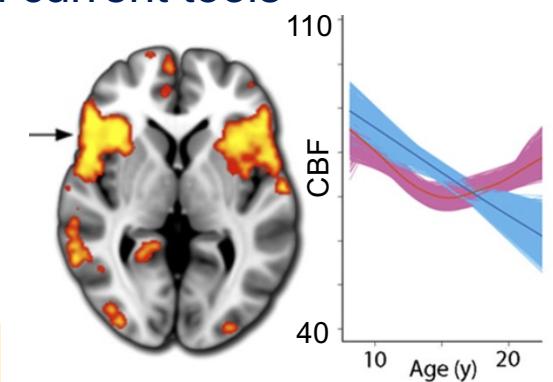


Consistent statistical
analysis tool?

Aim 2 – Background

Challenge #3: Users cannot easily model nonlinear age effects with current tools

- Lifespan age effects are often nonlinear
- Current tools often only support generalized linear model (GLM):
 - Limited flexibility: not supporting GAM with penalized splines
 - Generating design matrix and contrast matrix for GLM is not as easy as writing a formula in R



	GLM			Formula in R		
	<u>Design matrix</u>		<u>Contrast matrix</u>			
Linear regression	intercept	sex	age			<code>lm(): FD ~ age + sex</code>
	1	1	21.75			
	1	1	20.50			
	1	0	20.50			
	1	1	20.92	0	0	
	1	0	20.92		1	
	1	0	21.50			
	...					
GAM		???				<code>gam(): FD ~ s(age) + sex</code>

Aim 2 – Proposed software: ModelArray

Current challenges in mass-univariate hypothesis testing for large-scale datasets:

1. Scalability: Massive memory resources needed by large datasets
2. Generalizability: No consistent statistical analysis tool generalized across data types
3. User-friendly: Users cannot easily model nonlinear age effects with current tools

To address these challenges, I propose to develop **ModelArray**, a generalizable, memory-efficient, open-source R package for mass-univariate hypothesis testing of large neuroimaging datasets.

Aim 2 – Method for efficient memory

- HDF5 file format:
 - An HDF5 file (.h5) stores large dataset hierarchically and provides fast data access.
- R package DelayedArray:
 - Accessing and analyzing data **without** loading the entire data into memory

File size on the disk:

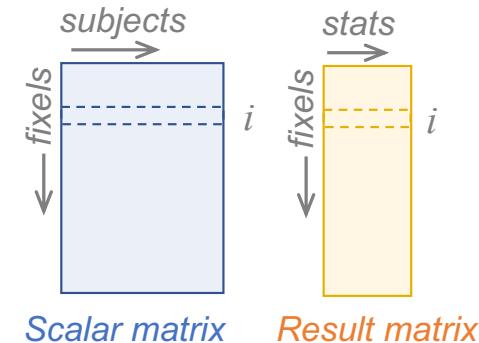


2.5 GB

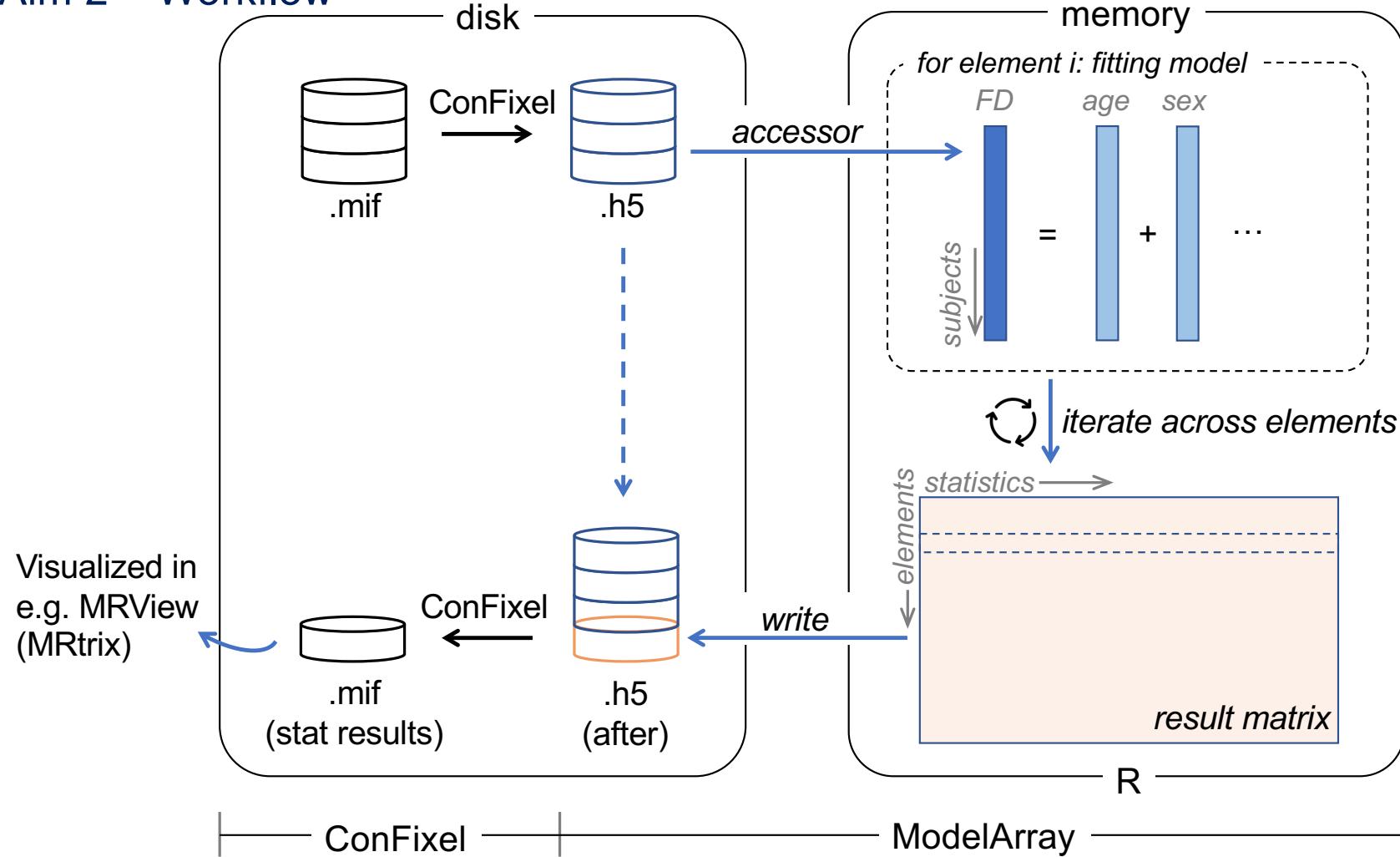
```
# in-memory size in R:  
> object.size(fixelarray)  
> 15584 bytes
```

An example .h5 file for fixel data:

Information on fixel data	
	/fixels /voxels /scalars /FDC /ids /values
Scalar matrix	602229 x 5 433909 x 4 1 x 938 602229 x 938
Result matrix	/results /lm /results_matrix 602229 x 13



Aim 2 – Workflow



Aim 2 – Functionality of ModelArray

Example script of the whole procedure of running ModelArray in R

```
> library(ModelArray)

> filename <- "example.h5"
> modelarray <- ModelArray(filename, scalar_types = "FDC")
> phenotypes <- read.csv("example.csv")

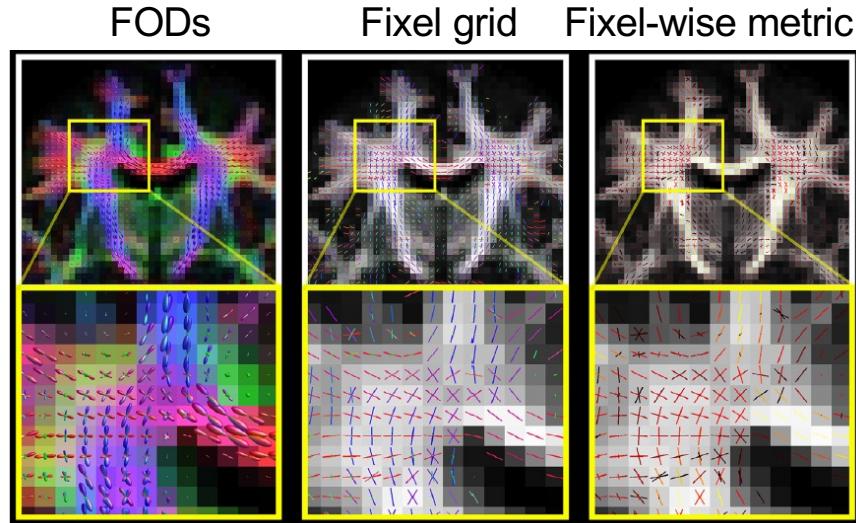
# Linear model:
> lm.outputs <- ModelArray.lm(FDC~age, modelarray, phenotypes, "FDC")
# Generalized additive model (GAM):
> gam.outputs <- ModelArray.gam(FDC~s(age)+sex+motion, modelarray, phenotypes, "FDC")

> writeResults(filename, df.output=lm.outputs, analysis_name="lm")
> writeResults(filename, df.output=gam.outputs, analysis_name="gam")
```



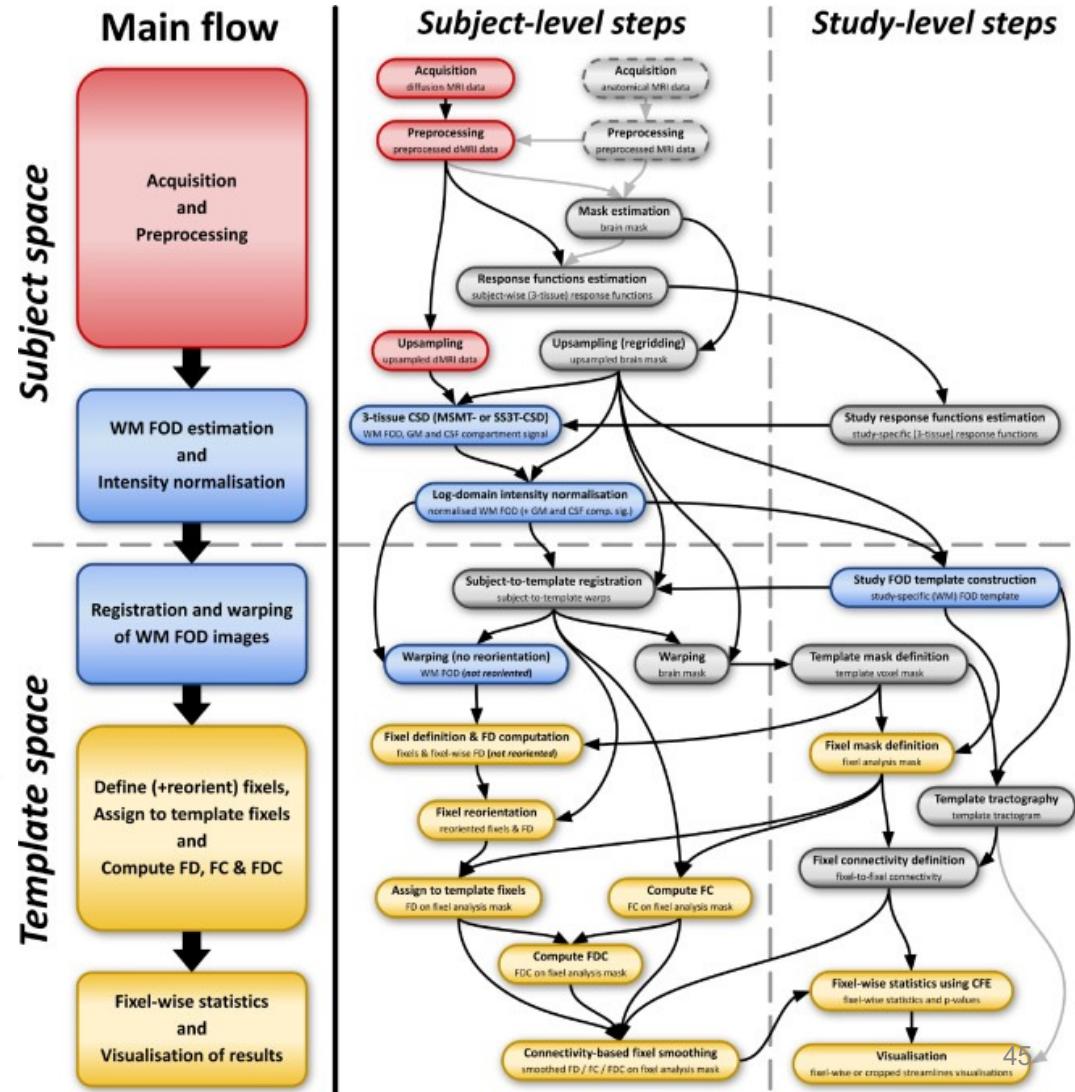
Aim 2 – Evaluation of ModelArray

Data: dMRI fixel data from RBC



Pixel-wise metric: FDC
(combination of FD and FC, more sensitive)

Figures adapted from Dhollander et al., *NeuroImage* 2021



Aim 2 – Evaluation of ModelArray

- Evaluations:
 - Memory profiling
 - Runtime profiling
- Evaluation details:
 - Will compare to primary existing tool for fixel-wise statistical analysis: the function “fixelcfestats” in MRtrix¹.
 - Will evaluate on a standalone computer with Linux system to avoid interference from other users.
 - Will use simple linear model: FDC ~ age

¹Raffelt et al., *NeuroImage* 2015

Aim 2 – Evaluation of ModelArray

Memory profiling

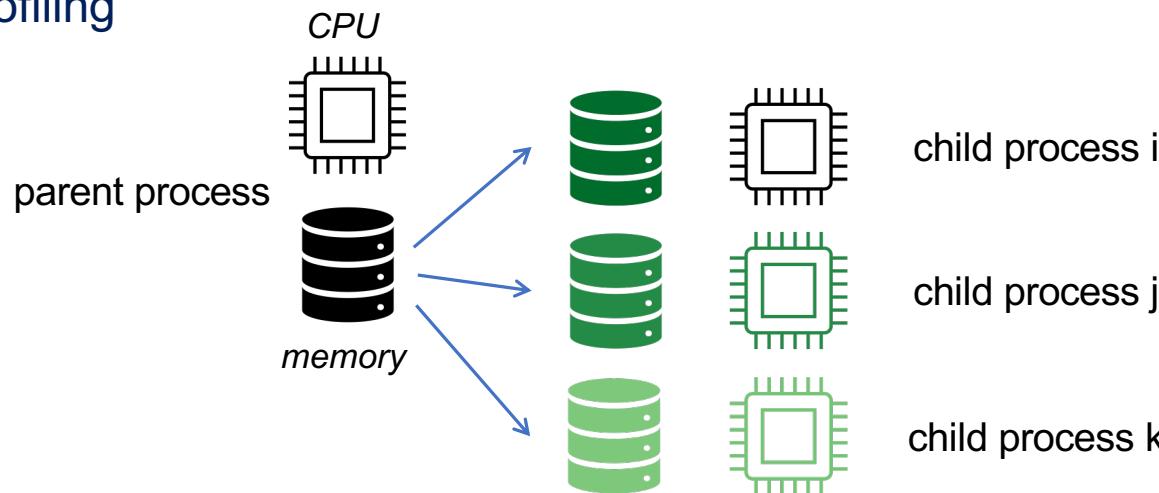


Figure adapted from :
https://www.blasbenito.com/post/02_parallelizing_loops_with_r/

\$ htop

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
10652	chenying	20	0	1135M	471M	9568	R	99.8	0.9	1:25.39	R --no-echo --no-r
10658	chenying	20	0	1135M	471M	9560	R	99.8	0.9	1:25.40	R --no-echo --no-r
10656	chenying	20	0	1135M	471M	9560	R	99.2	0.9	1:25.43	R --no-echo --no-r
3566	chenying	20	0	1081M	439M	2148	S	0.0	0.8	0:09.83	R --no-echo --no-r
2445	chenying	20	0	4148M	300M	82168	S	0.0	0.5	0:14.12	gnome-shell
2968	chenying	20	0	50.7G	253M	106M	S	0.0	0.5	0:44.26	/usr/share/code/co

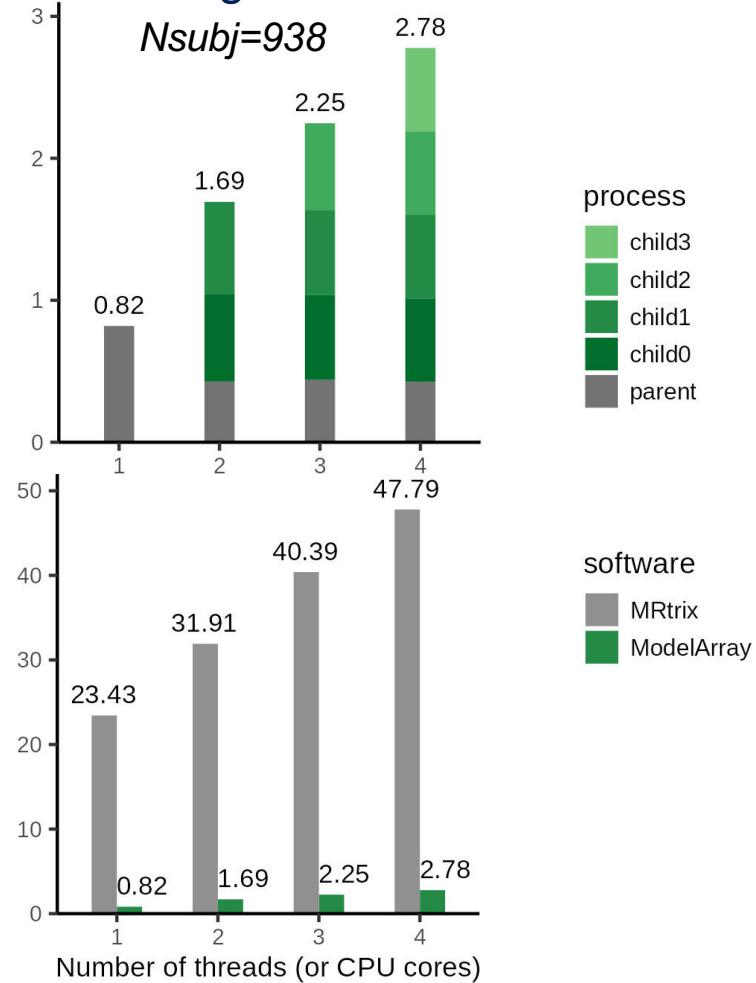
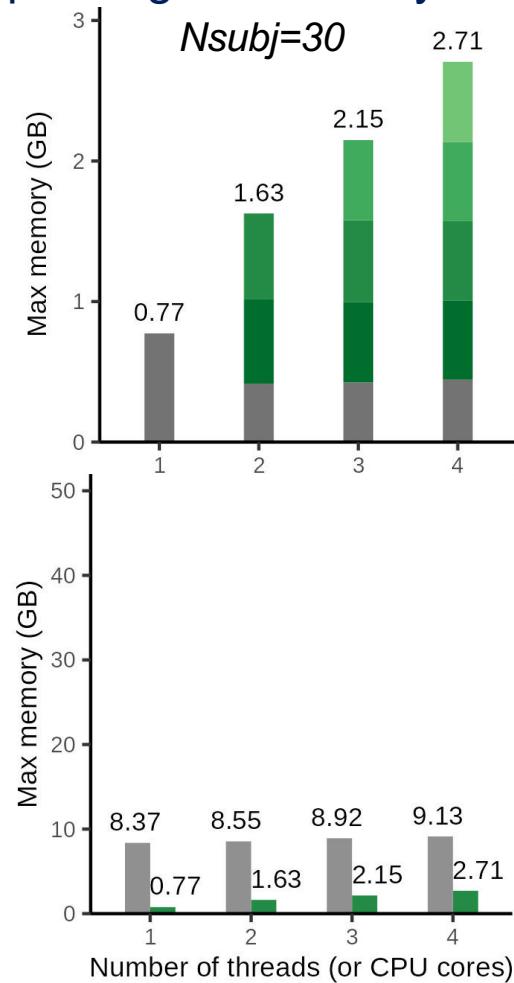
child processes
parent process →

Memory usage will be captured by Working Set Size (WSS) Tools for Linux

<https://www.brendangregg.com/wss.html>

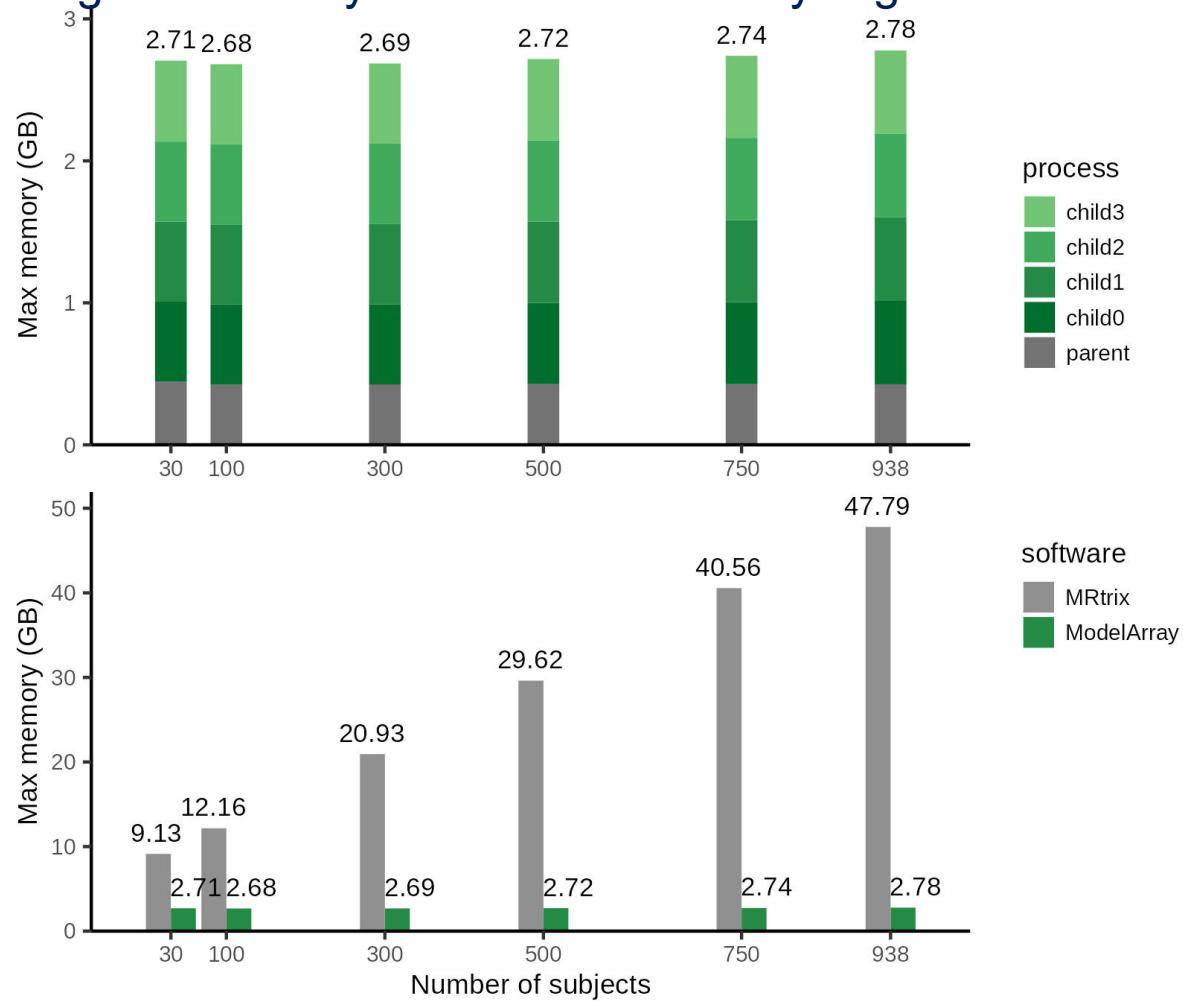
Aim 2 – Evaluation of ModelArray

Memory profiling: Preliminary results – when using different number of threads (or CPU cores)



Aim 2 – Evaluation of ModelArray

Memory profiling: Preliminary results – when analyzing different number of subjects



parallelization factor = 4

- 4 CPU cores requested for ModelArray
- 4 threads requested for MRtrix

Aim 2 – Application of GAMs to developmental datasets

Methods

- I will also demonstrate the use of GAMs in ModelArray for modelling nonlinear developmental effects with penalized splines:

$$FDC \sim s(Age) + sex + motion$$

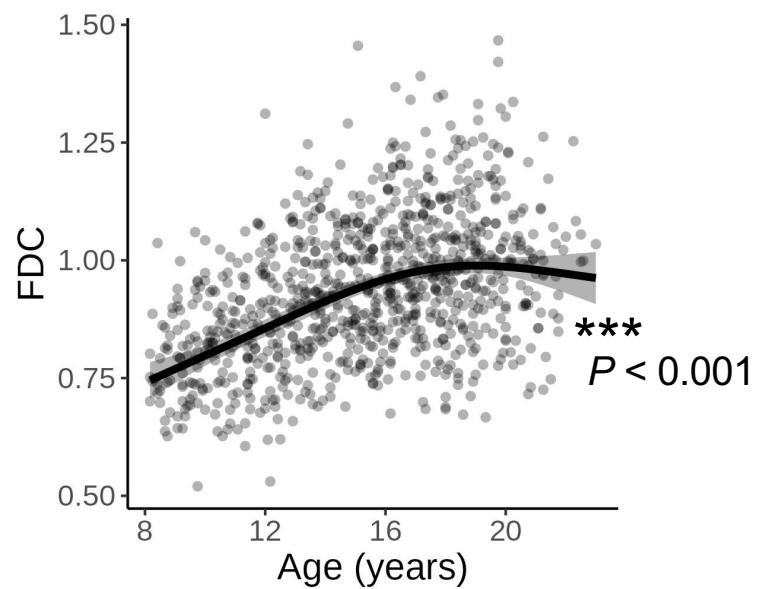
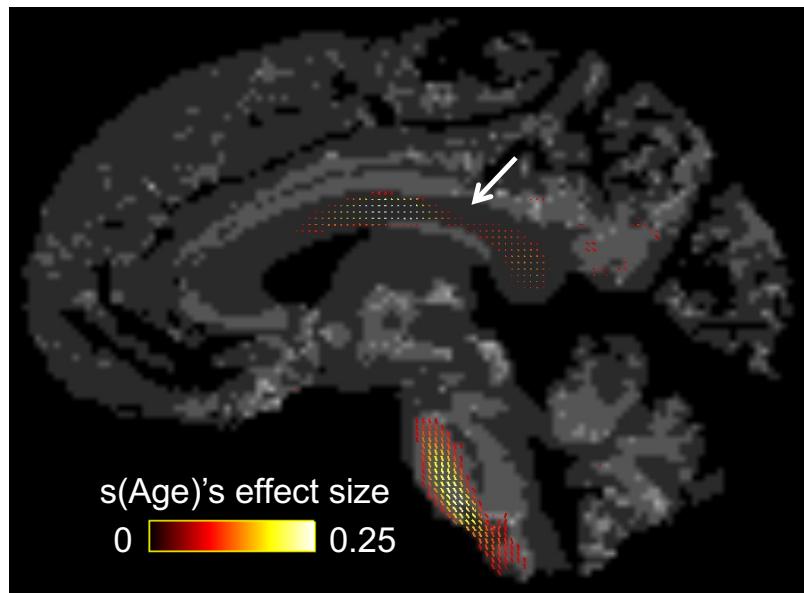
- motion*: mean relative volume-to-volume displacement of 7 b=0 images
- Effect size* = $R^2_{full} - R^2_{reduced}$



$$FDC \sim sex + motion$$

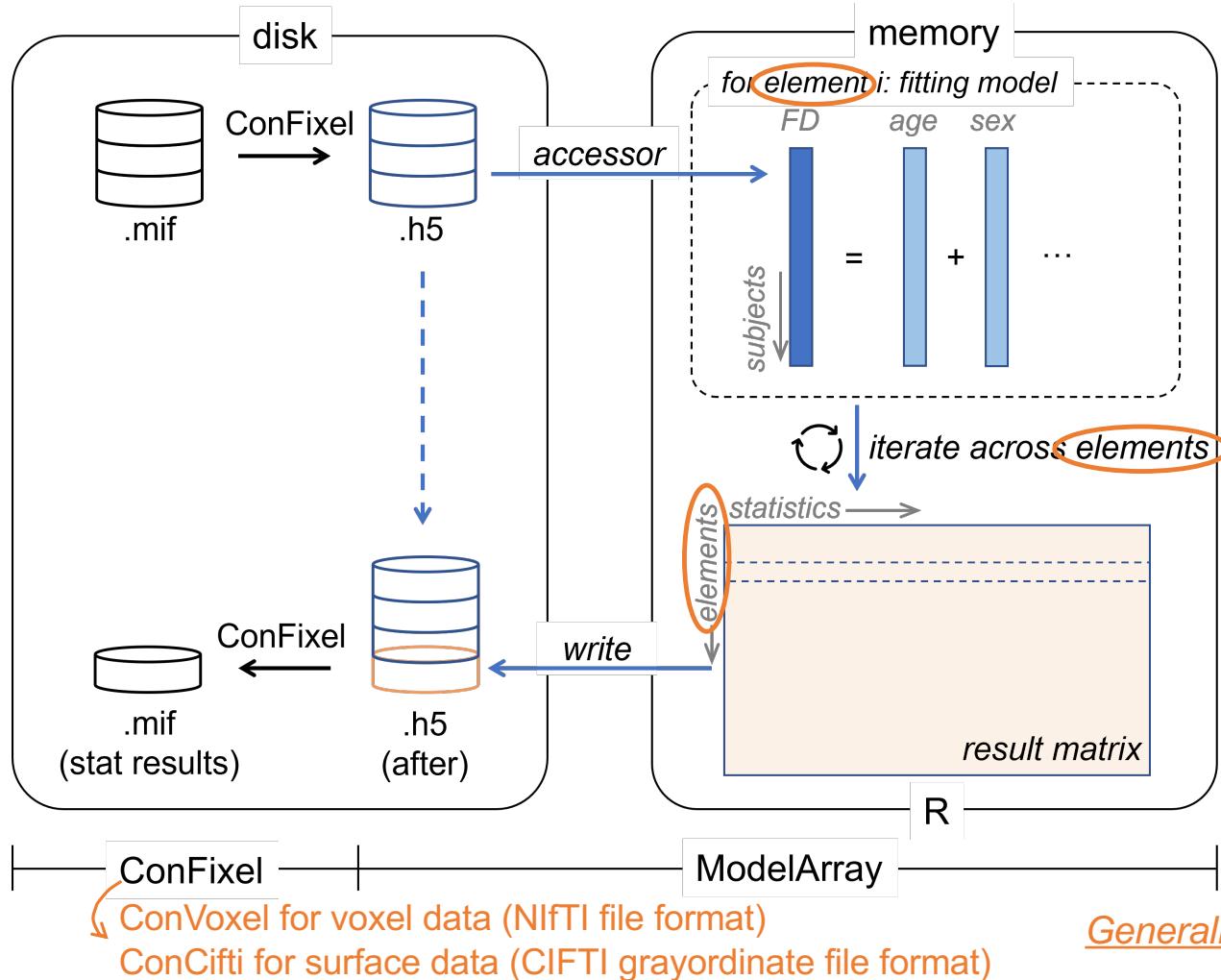
Aim 2 – Application of GAMs to developmental datasets

Preliminary data: nonlinear age effect in adolescent development



- Data: PNC dataset ($n=938$), aged 8-22 years
- Threshold: P value of $s(\text{Age}) < 1 \times 10^{-15}$
- GAM Formula: $FDC \sim s(\text{Age}) + \text{sex} + \text{motion}$

Aim 2 – Generalizability and extensibility of ModelArray



Extensibility:

Model fitting functions in ModelArray:

- `ModelArray.lm()` for linear model
- `ModelArray.gam()` for GAM
- `ModelArray.new()` for ?

Outline

- Aim 1: Software for reproducible image processing at scale
- Aim 2: Memory-efficient, generalizable software for statistical analysis
- **Software development**

Software development

Version control and CI testing



A screenshot of a GitHub repository page for PennLINC/ModelArray. The repository is public, has 14 issues, 1 pull request, and 1 project. The main branch is 'main' with 15 branches and 0 tags. The code tab is selected. On the right, there's a list of commits. An orange arrow points from the 'All checks have passed' message in the commit list to the green checkmark icon in the commit details. The commit details show a merge pull request from Zhao (9e735b9) on Feb 4, which includes 275 commits. The commit list also shows other recent commits related to trixStats package updates and workflow changes.

Commit	Date	Details
9e735b9 on Feb 4	275 commits	Zhao (Merge pull request #41 from PennLINC/enh/partialRsq_sse)
trixStats package	3 months ago	Zhao (Workflow: buildR)
	3 months ago	Zhao (Workflow: buildR)
er; minor update...	last month	Zhao (Workflow: buildR)
er; minor update...	last month	Zhao (Workflow: buildR)
analyse.R to util...	2 months ago	Zhao (Workflow: buildR)

CI = Continuous Integration

Software development

Version control and CI testing



Screenshot of a GitHub repository page for PennLINC/ModelArray. The repository is public and has 14 issues and 1 pull request. It contains 15 branches and 0 tags. A merge pull request from zhao-cy is shown.

The .circleci directory contains a configuration file for CI testing. The CI status shows "All checks have passed" with 3 successful checks:

- pkgdown / pkgdown (push) - Successful
- build_test_deploy - Successful in 8m — 1 job
- ci/circleci: develR — Your tests passed on 2023-09-11T14:44:23Z

A modal window titled "ModelArray" displays CI pipeline history:

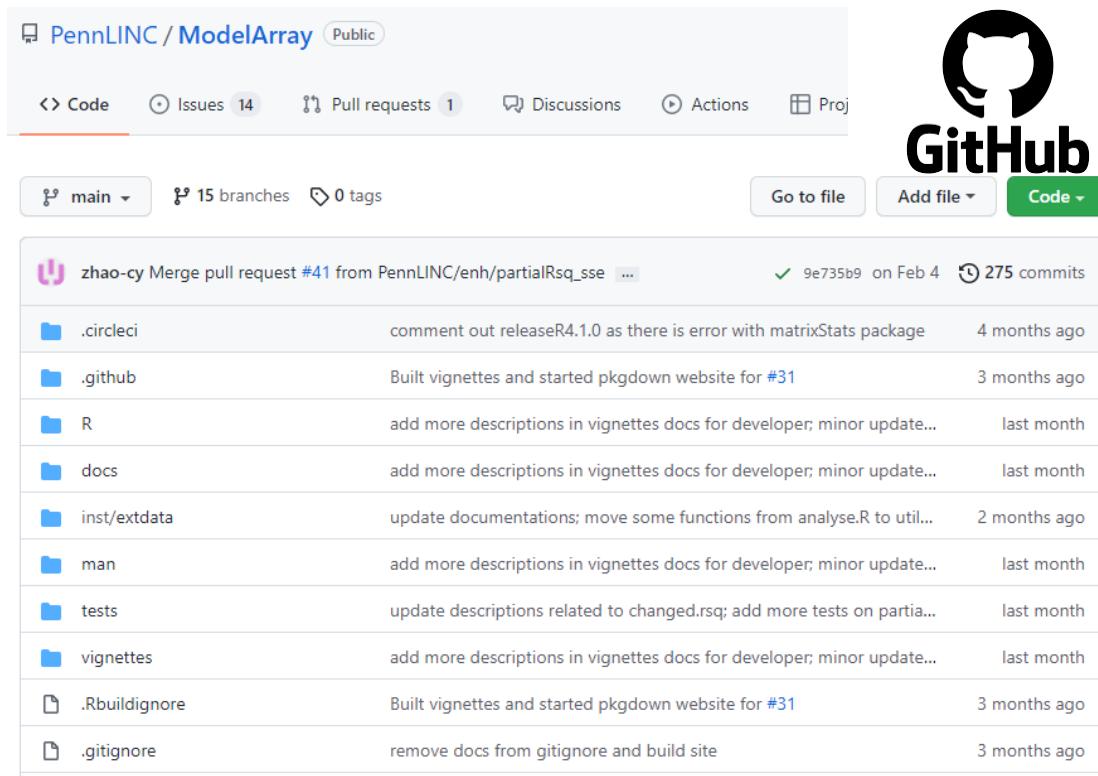
Pipeline	Status	Workflow
ModelArray 174	Failed	build_test_deploy
ModelArray 173	Success	build_test_deploy
ModelArray 172	Success	build_test_deploy
ModelArray 1/1	Success	build_test_deploy

The pipeline "ModelArray 174" failed due to a "develR" job error. The modal also lists "Jobs" for each pipeline run.

This assures software is functional without error after updates.

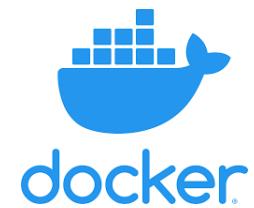
Software development

Containerization and release



Author	Commit Message	Date
zhao-cy	Merge pull request #41 from PennLINC/enh/partialRsq_sse	on Feb 4
.circleci	comment out releaseR4.1.0 as there is error with matrixStats package	4 months ago
.github	Built vignettes and started pkgdown website for #31	3 months ago
R	add more descriptions in vignettes docs for developer; minor update...	last month
docs	add more descriptions in vignettes docs for developer; minor update...	last month
inst/extdata	update documentations; move some functions from analyse.R to util...	2 months ago
man	add more descriptions in vignettes docs for developer; minor update...	last month
tests	update descriptions related to changed.rsq; add more tests on partia...	last month
vignettes	add more descriptions in vignettes docs for developer; minor update...	last month
.Rbuildignore	Built vignettes and started pkgdown website for #31	3 months ago
.gitignore	remove docs from gitignore and build site	3 months ago

Source code openly available on GitHub



- Will build a container to enhance portability
- Will release on DockerHub

This workflow ensures both version control and frictionless portability.

Software development

Comprehensive documentations

The screenshot shows a web-based documentation site for the `ModelArray` package. At the top, there's a navigation bar with links for `ModelArray` 0.0.0.1, Reference, Articles, and a search bar. The main content area has a header "ModelArray" and a sub-section "Run GAM for element-wise data". On the left, there are sections for "ModelArray features" (with bullet points) and "Insta" (with a note about dev has). Below these are two code snippets in a box labeled "foo@ba" and "If you g". On the right, there's a "Developer documentation" section with a target audience note, a general overview of the package structure, and a list of steps for contributing. A yellow box highlights the "Documentation for developers" section at the bottom.

ModelArray 0.0.0.1 Reference Articles ▾

Search for

ModelArray

Run GAM for element-wise data

ModelArray features

- Easy
- Low
- At p

'ModelArray.gam' fits gam model for each of elements requested, and returns a tibble data.frame of requested mode

Usage

```
ModelArray.gam(  
  formula,  
  data,  
  phenotypes,  
  scalar,  
  element.subset  
  full.outputs =  
  var.smoothTerms  
  var.parametric  
  var.model = c(  
    changed.rsq.ter  
    correct.p.value  
    correct.p.value  
  verbose = TRUE  
  pbar = TRUE,  
  n_cores = 1,  
  ...  
)
```

Developer documentation

Target audience: developers, anyone who hopes to contribute to `ModelArray` to enhance functionality, fix bugs, or tailor for a specific usage.

In this documentation, we walk through important methods used in `ModelArray` package, and the structure of the package and its functions.

If you hope to improve functions or fix a bug in our package, please follow the following steps:

- fork the [github repository](#)
- modify the scripts
- update the tests and/or add necessary new tests (see folder `test`)
- to test the package and related documentations and websites can be built: run `devtools::check()`, `devtools::build()`, `devtools::build_vignettes()`, and finally `pkgdown::build_site()`. Please check whether there is no error along the process, and the final built sites look good.
- if everything looks good, please submit a pull request to our github repository.

Overview

`ModelArray` R package sup

Documentation for developers

HDF5Array, etc R packages, so that statistical analysis can be performed without loading original data in HDF5 file into memory. This makes `ModelArray` very efficient in memory usage. For details, please see section "ModelArray Construction" below.

Thank you !!!