Characterizing the Stability of Neuroimaging Analyses Through Perturbations in Experimental Design

Gregory Kiar
October 10th, 2018

Outline

This proposal follows the template established in Appendix A of the:

McGill University Department of Biological and Biomedical Engineering Guidelines

Summary of Research Proposal

In recent years there has been a growing focus on evaluating the reproducibility of scientific findings. With a large, confused, and growing lexicon of definitions in this space, reproducibility can briefly be described as providing sufficient information and resources that a researcher has the ability to successfully reproduce their own findings, while replicability refers to the ability of other researchers using similar means to arrive at the same conclusions [1]. Previous studies have shown across several domains of science [2]–[4] which have shown that a significant proportion of claims fail to replicate, posing a significant problem for the scientific community.

In neuroimaging, this issue has been explored with respect to the impact that analysis tools themselves have on claims and their replicability. In functional Magnetic Resonance Imaging (fMRI) selection of software tool has been shown to produce different results [5]. In structural MRI the stability of two software packages with respect to minor data perturbations has been evaluated [6], [7]. Currently, there lacks a characterization of stability in diffusion MRI, as well as the joint comparison of tool stability and its relative impact on downstream inferences.

For my thesis, I plan to characterize the stability of neuroimaging tools in the context of diffusion MRI. Leveraging the publicly-available Consortium of Reproducibility and Reliability (CoRR) dataset [8] consisting of similar test-retest datasets collected across multiple sites, I will evaluate the stability of commonly-used analysis pipelines for structural connectomics (FSL [9], MRTrix [10], Dipy [11]) with respect to perturbations in input data.

In the case of well-behaved functions, it is possible to generate a closed-form solution for their stability. However, the complexity of algorithms used in neuroimaging makes generating closed-form solutions for stability often intractable, requiring us to rely on estimates. Generating estimates of stability in this context will rely upon the ability to easily develop, launch, and consolidate analyses over datasets, noise levels, and tools. In order to operationalize the evaluation of tool stability I will explore pipeline components in addition to complete workflows.

The core outcome of this project will be the ability to characterize the stability of neuroimaging analyses and provide an accessible method for researchers to evaluate their scientific designs. The concrete contributions made throughout my Ph.D. will be:

- the creation of infrastructure for accessible, flexible and reproducible analysis,
- the definition and evaluation of stability for neuroimaging pipelines, and
- the exploration of individual processes and their relative impact on pipeline stability.

The successful completion of my proposed project has the potential to shed light on the effect that numerical instabilities have on neuroimaging analyses, and identify a dependence between scientific claims and the tools used to generate them. Understanding this will lead to the

production of more richly described, trustworthy scientific studies and tools. I will describe the progress and accomplishments to date and the plan for future work in the sections that follow.

Summary of Progress

The development and evaluation of methods assessing the stability of neuroimaging pipelines is dependent on a computational infrastructure that enables launching, comparing, exploring, and evaluating analyses on a variety of high performance computing (HPC) systems.

In order for the proposed platform, Clowdr, to satisfy the above requirements and be accessible for researchers, I developed it atop the shoulders of existing and emerging standards. In particular leveraging data and tool-description standards increases the portability of analyses to distinct datasets and interoperability with other existing platforms.

The Boutiques specification documents the command-line execution instructions of tools, and provides an interface for validating their description, execution, the existence of expected outputs, and more. I have contributed significantly to this project including leading the development of a Python package [12], and am one of two project maintainers. The advent of Boutiques in the context of pipeline deployment is that it enables clear documentation and repeatable execution of tools across platforms, making it an important backbone of the proposed projects. The Boutiques paper, for which I am the second author, was published in Gigascience this year [13]. Efforts are ongoing to integrate this standard more closely with tools for workflow management such as Nipype [14].

In the realm of data organization, the Brain Imaging Data Structure (BIDS) [15] has experienced tremendous adoption in the neuroimaging community. This standard prescribes an organization of folders and files on disk and serves as a low-barrier solution to unambiguously and accessibly share datasets. From this, along with others I worked on the development of the BIDS app initiative [16] which prescribes specific command-line arguments and inputs for tools to be run on BIDS datasets easily. Many neuroimaging tools have adopted and contributed to the development of this standard, including a pipeline for structural connectome estimation, which I developed: ndmg [17], [18]. These standards enable the rapid deployment of pipelines across disparate datasets.

I have completed the initial development of Clowdr [19] and released it on the Python package index. Clowdr allows users to develop Boutiques-described tools locally, deploy them on the cloud (i.e. Amazon) or clusters (i.e. Compute Canada), record rich provenance information about the execution, and easily monitor and share the results through the web. Clowdr was accepted for a poster and software demonstrations at the Organization for Human Brain Mapping (OHBM) 2018 conference, and the paper is currently in review at Frontiers in Neuroinformatics.

As mentioned above, I have yet to undertake considerable work on the stability portion of my project as the Clowdr tool is a prerequisite for the planned experiments. I will expand upon these experiments and next steps for my project in the following sections.

Research Proposal

Overview

As was summarized previously, the eventual objective of my work is to develop a simple mechanism for evaluating the stability of neuroimaging pipelines and components, ideally leading to improved pipeline selection, development, and more powerful scientific discoveries.

The first chapter in my thesis covers the development of Clowdr, a lightweight tool which increases the accessibility to perform complex analyses in high performance computing environments, record rich provenance records, sweep over design parameters, and share the results. As will be discussed, this chapter is complete and the manuscript summarizing it is currently in review at Frontiers in Neuroinformatics and attached as an Appendix to this proposal.

The second chapter of my thesis concerns the development of a method for empirically estimating the stability of neuroimaging analysis, in particular, applied to diffusion connectomics. This will start from the definition of stability in numerical analysis through the concept of "conditioning," where I will then generate a method for estimating the stability of an analysis by perturbing input properties and observing the impact on downstream behaviour. This process will leverage the tool developed in Chapter 1, and evaluate the three most commonly used pipelines for structural connectome estimation on CoRR datasets. This work will ultimately assess the stability of results across datasets and tools, and the stability of tools with respect to several types of injected noise.

The third and final chapter of my thesis aims to explore the role of individual pipeline components in the overall stability as observed in the pipelines assessed in Chapter 2. This will involve the same experimental paradigm as developed in the previous chapter, with the key difference being that noise will be injected and derivatives compared from individual pipeline components rather than treating pipelines as "black boxes." This will enable the comparison of similar methods across tools, such as image registration, and inform the construction of maximally stable pipelines. This will also allow us to explore the relationship between the stability of processing steps and the composite pipeline stability.

Together these projects involve the development of methods for empirically estimating the stability of analyses, and the application of these methods to characterize tool stability over a variety of independent variables. The successful completion of this thesis will be an important stepping stone towards understanding the depth of the current reproducibility crisis in neuroimaging and can be used as a basis for evaluation in future tool development.

Chapter 1: Scalable and Provenance Rich Pipeline Deployment (Clowdr)

The backbone of many neuroimaging analyses and claims relies on two key components: the data being processed, and the tool doing the processing. External conditions, such as where the tools were being deployed, the format of the data, which hyperparameters were used, while of considerable importance are rarely sufficiently recorded or documented in order for studies to be adequately reproduced, let alone meta-analyzed or staged alongside similar findings [20]. Here we will explore the case in which this metadata is recorded sufficiently for replication.

Given that researchers produced a replicable study, we still suffer from a clear understanding of what role each design decision, data point, or tool setting played on the resulting picture. There currently exists no known methods (other than ad-hoc) which easily enables studying the effect of perturbations on findings, by allowing iterative modification of both the data being processed and the tool settings.

The tool proposed here, Clowdr, will build reproducibility into analyses and give researchers the ability to easily perform perturbations on analyses. Clowdr will serve as a micro-environment for deploying and recording executions and scientific analyses. The primary objective of Clowdr is to increase the accessibility, reproducibility, and permutability of scientific analyses.

With respect to accessibility, Clowdr enables users to develop and execute tools locally and seamlessly transition to executing these same tools either on high performance computing clusters or commercial clouds while monitoring their progress, recording execution information, and enabling the sharing the results. This paradigm will shorten the feedback and development process for scientific software developers, and easily enable meta-analyses, such as hyperparameter sweeps and optimizations with a low barrier to entry.

There exist many platforms which aid in scientific computing, including CBRAIN [21], BrainCODE [22], OpenNeuro [23], and LONI [24] in the neuroimaging space alone. Each of these tools enables neuroscientists to interact with datasets and established computational tools from the comfort of their web browsers, and have been an incredible asset for hundreds of scientists. A key difference with these science-as-a-service infrastructures and Clowdr is that the analysis tools available to users are restricted to “production-level” workflows and are not conducive to tool development, where Clowdr was designed for this purpose.

These platforms often rely on an underlying structure for representing how tools are defined and will be executed; in the CBRAIN case, this is the Boutiques [13] command-line descriptive framework. The definitions enforce standards on the tools being executed, bringing clarity and consistency to their execution. While Clowdr adopts this approach and the Boutiques framework for standardizing command-line tool representation, other standards such as Nipype [14] or PSOM [25] address this and enable the construction of workflows for Python and MATLAB/Octave, respectively.

Clowdr operates as a “microservice,” meaning that no persistent server must be running and that it can be very easily used with minimal installation or configuration. It has been tested and

developed to run locally (macOS, Linux), on clusters (Compute Canada; SLURM environment), and commercial clouds (Amazon Web Services). The ability to flexibly deploy and modify executions through Clowdr allows permutation and perturbation tests to be performed accessibly on tools and data, and perform hyperparameter tuning. Clowdr also records rich provenance records and allows for their exploration accessibly via the web (Figure 1).

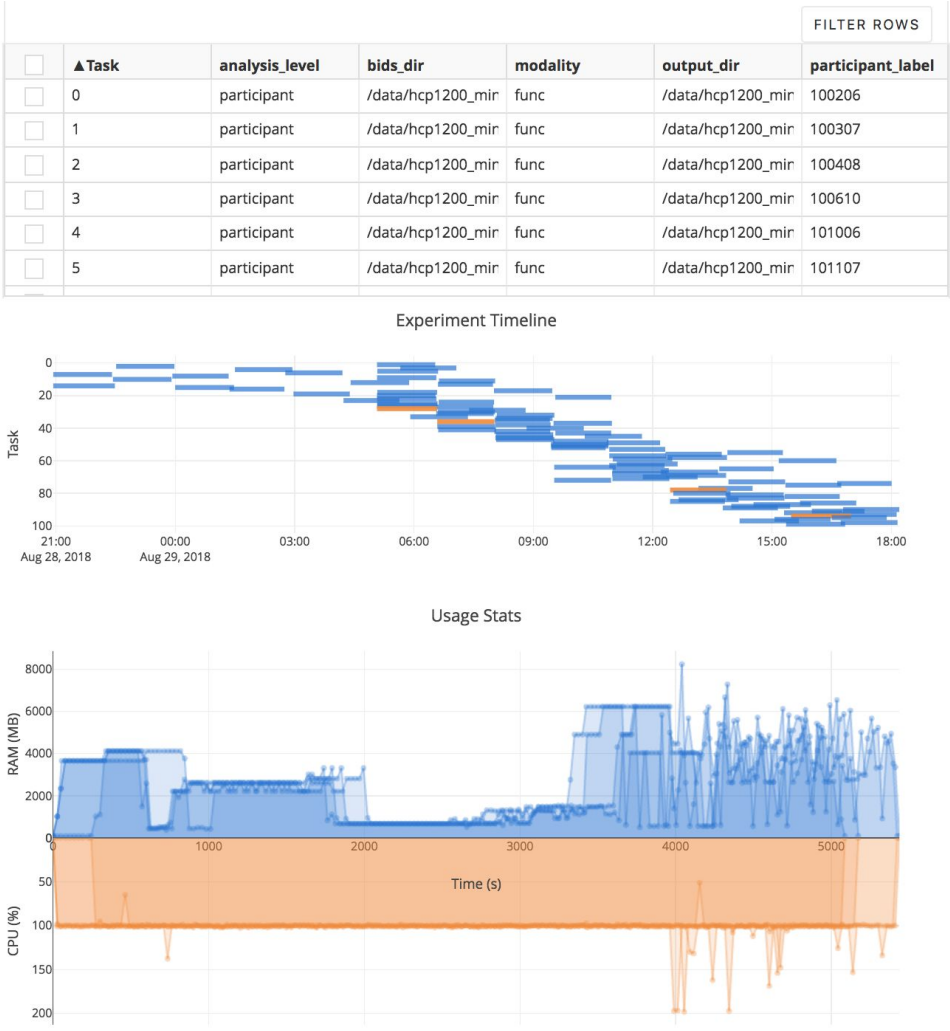

Figure 1: Clowdr share portal which provides summary statistics and allows filtering of results based on features such as resource consumption, exit status, and input values such as the tool, data, and hyperparameters used.

I have developed Clowdr for special consideration to neuroimaging, as it has underlying support for the BIDS data organization, which, when paired with Boutiques, dramatically reduce the burden on scientists when specifying the executions they wish to perform. Because of this, the combination of Clowdr and Boutiques have become somewhat of a native execution environment for BIDS apps, which immediately opens the door to using it with a variety of tools such as FSL [9], MRtrix [10], Dipy [11], and many others.

The Clowdr Python package is open-source on Github, https://github.com/clowdr/clowdr, hosted on the Python Package Index, https://pypi.python.org/pypi/clowdr, and was presented as a poster and software demonstration at the OHBM 2018 annual meeting, and is currently under review at Frontiers in Neuroinformatics. The submitted manuscript is attached.

While development will continue on this project, the initial development is complete and further work will go into extending the functionality of this tool. Clowdr has the potential to lower the barrier to entry for testing in scientific tool development, and result in higher quality and better characterized pipelines, and ultimately more impactful scientific claims.

Chapter 2: Evaluating the Stability of Neuroimaging Pipelines & Analyses

In numerical analysis, the stability of a function can be characterized by a condition number [26]. In concept a condition number can be interpreted as the maximum ratio of the relative errors between the output and inputs of a function; thus, a higher value indicates a less stable, or ill-conditioned, function where a value near 1 is said to be well-conditioned. For a more detailed explanation of the derivation of condition numbers for matrices, univariate, or multivariate functions, please see [26].

Condition numbers have been calculated in diffusion MRI tensor reconstruction previously, and shown that in the case of generating apparent diffusion coefficient (ADC) estimates, more well-conditioned transformation matrices resulted in more stable results when faced with noise in both simulated and real data [27] (Figure 2). The authors explored multiple algorithms for tensor estimation and characterized the bias and standard deviation of resulting fractional anisotropy values, comparing them to the theoretical condition.

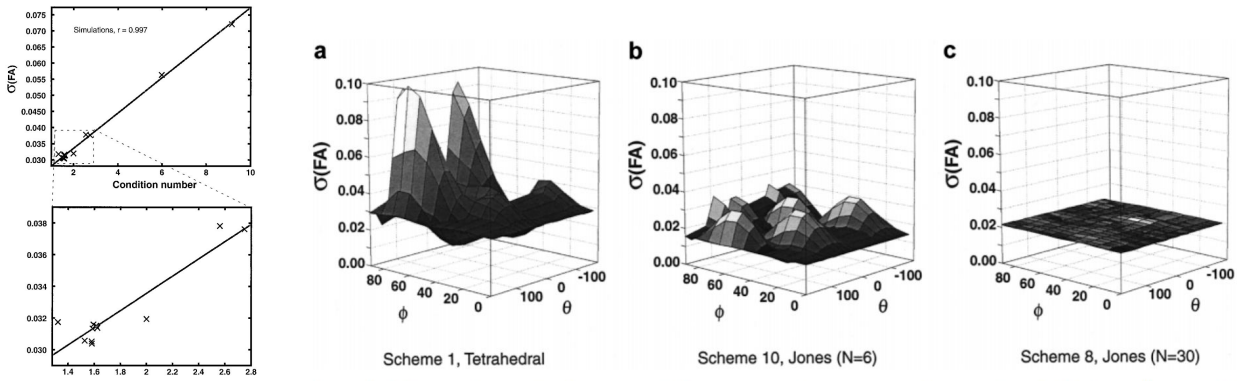

Figure 2: Left (Figure 1 from [27]): relationship between conditioning of various tensor estimation algorithms and the standard deviation of the fractional anisotropy values of the resulting tensors when faced with noise. This shows a strong relationship between theoretical conditioning of the model and the empirical stability. Right (Figure 3 from [27]): For three tensor estimation algorithms, shows the standard deviation dependent on the tensor orientation. Plot **a** corresponds to the highest condition number, where both **b** and **c** have similar condition values near 1. This figure highlights non-linearities of the instability presented by each algorithm, and importantly, that while the two rightmost algorithms have similar stabilities they have

different behaviour.

Doing this on both simulated and real data, the authors show a strong relationship between observed variation and the expected condition of algorithms. The right panels of Figure 2 importantly show, however, that the variation of these algorithms is not uniform. In panel **a**, an algorithm with high conditioning can be shown to exhibit highly variable variance, in particular around where angle ϕ approaches 90 degrees, as the conditioning of a nonlinear function is dependent on input data. In panels **b** and **c**, algorithms with almost identical conditioning values (1.5826 and 1.5945, respectively) have different appearances. Looking closely we can see that the plane in **c** is much flatter but is at a slightly higher variance than the lowest points of **b**, but **b** peaks higher. While understanding this second-order variance was not explored in detail in [27] and will not be a primary focus here, it is an exciting avenue for future work.

Table 1: Summarized experiment plan for the work performed in Chapter 2 of the proposed thesis.

Purpose	Characterizing the instability and variability of analyses		
Outcomes	❏ Metric for evaluating the stability of analyses with respect to dependent experimental variables ❏ Exploration of the variability introduced in Diffusion MRI experiments by dataset, noise, and tool selection		
Datasets	**Modality**	**Derivatives**	**Tools**
Consortium of Reproducibility and Reliability	Diffusion MRI, Structural MRI (Functional MRI)*	Structural Connectomes (Functional Activation Maps)*	Dipy, FSL, MRtrix (SPM, AFNI, FSL)*

Experiment	❏ Partial replication of [27] comparing conditioning to observed variance
	❏ Determine a space-independent proxy for conditioning
	❏ Process CoRR datasets using default Dipy, FSL, and MRtrix pipelines
	❏ Reprocess CoRR datasets with:
	  ❏ 1-voxel perturbation
	  ❏ Rician noise
	  ❏ Gaussian noise
	❏ Calculate conditioning across:
	  ❏ Noise (fixed tool and dataset)
	  ❏ Datasets (fixed tool)
	  ❏ Tool (fixed datasets)
	❏ Compare each setting, and identify axes and regions of instability		

Notes	❏ *Based on collaboration with Dr. Camille Maumet, this work may be extended to cover functional MRI applications. This will leverage her experience with fMRI evaluation, and will require the development of theory similar to that presented in [27] on algorithms used in fMRI.		

Leveraging the approach taken here I plan to partially replicate and extend [27] towards characterizing several tensor estimation algorithms, and developing a space-invariant statistic for empirically estimating the conditioning of algorithms for which closed-form solutions cannot practically be obtained, and use this to evaluate the stability of complete pipelines. Table 1 summarizes my experimental plan for accomplishing this work.

The proposed empirical conditioning statistic should share several properties with true condition values, such as having a lower bound of 1 and monotonically increasing along with the instability of the operation. One definition of a condition number is:

$$\kappa(A) \geq max_x \frac{\| \delta f \| / \| f \|}{\| \delta x \| / \| x \|}, \quad (1)$$

where $\kappa(A)$, the conditioning of analysis A, is greater than or equal to the maximum ratio between the relative change in output, $\| \delta f \| / \| f \|$, to the relative change of inputs, $\| \delta x \| / \| x \|$. In this definition, the values of x and f considered to be "true" values, and the relative deviation from true is what is being compared.

There are several key limitations preventing the use of (1) in practice. First, this representation doesn't allow for the direct comparison of analyses. By including a term summarizing the experimental design - including properties such as dataset or noise - this method could be extended to allow for direct comparison of the stability of experimental designs with respect to one another.

Second, this form assumes that the input and output data are in the same space. This raises two important additional limitations: the norms used for the input and output must be the same, and the variance between inputs and outputs is expected to be equal. In the case of domains such as neuroimaging, it is not uncommon for outputs to live in a unique space from the inputs, such as in the case of derived connectomes and raw MR images. It is also not uncommon for perturbations on input data to be small relative to the original images, and trigger minimal or no change on the output (in the case of thresholding, for example, a minor change in voxel intensity is unlikely result in a different binarization). In both of these cases, the fractions in (1) would tend to unity, leading to a potentially uninformative statistic. By substituting the unperturbed term in both the numerator and denominator with the standard deviation of that quantity over all observed x and f, we gain the ability to appropriately weight minor changes in stable data, and are able to jointly evaluate input and output data in different spaces.

Modifying the notation slightly, we will now represent f as a function of x, $f(x)$. We can collapse x and δx into $x_d = x + \delta x$, where $x_d \in D \subset A$, where D is the tested design in analysis A, and $x \in A - D$. We will denote an arbitrary domain-appropriate norm for inputs and results as $\|\square\|_I$ and $\|\square\|_R$, respectively. I will also rename this quantity as the estimated

condition, $\hat{C}$. We will use σ to denote the standard deviation of each series of values. As a result, we have the following:

$$\hat{C}(A,\ D)\ \geq max_{x_d}\ \frac{\|\ f(x_d) - f(x)\ \|_R\ /\ \sigma_{f(x)}}{\|\ x_d - x\ \|_I\ /\ \sigma_x}\ ,\ \forall\ \ x_d \in D,\ x\ \in A - D\ . \tag{2}$$

Looking closely at the above definition, we can see that in each the numerator and denominator we are computing something similar to the z-score of the f and x, respectively. While this form is also closely related to the definition of theoretical conditioning shown in (1), input and output quantities have now be normalized with respect to both their magnitude and their variability.

The distinction in (2) of specifying the outputs as explicitly conditional on design inputs, x_d, provides added flexibility such that the relative impact of changes to analyses can be explored independently. For instance, if D describes an experiment using a specific subset of datasets processed in A, then the condition would evaluate the stability of results made on this dataset with respect to the remaining datasets. This formulation allows for the comparison and evaluation of any data-based perturbation to analysis A. However, extending this to compare across tools requires further modification.

In the case of evaluating multiple processing tools $f(x)$ with no data perturbation, the resulting condition estimate in (2) would infinite. To obtain a real estimate of conditioning in this setting we can return to a logical interpretation of (2), where we observe the relative normalized change in output of a function with respect to the relative normalized change in input. In the case where we wish to compare pipelines rather than the impact of input perturbations, we can remove the dependence on input, and directly compare one pipeline, $f_d(x)$, to others, $f(x)$, simplifying (2) for relative tool conditioning, $\hat{C}_t$, as follows:

$$\hat{C}_t\,(A,\ D)\ \geq max_x\ \|\ f_d(x)\ -\ f(x)\ \|_R\ /\ \sigma_{f_d(x)}\ ,\ \forall\ \ x\ \in A\ . \tag{3}$$

In summary of the above expressions, using (2) we are able to evaluate the effect of both variation in dataset selection and noise, allowing exploration of numerical stability and generalizability of results. The addition of (3) allows for the comparison of operating system, tools, and hyper parameters. Together, these two estimators for stability allow for a rich characterization of the effect of various independent variables on computational analyses.

Once I have demonstrated the use of estimated conditioning on tensor estimation, I will apply it to evaluating the three dominant tools in diffusion MRI connectome estimation: FSL [9], MRtrix [10], and Dipy [11]. I will adopt the default pipelines for each, and use them to generate structural connectomes on the subset of the Consortium for Reliability and Reproducibility (CoRR) [8] collection containing diffusion and structural MR images. The CoRR datasets were selected due to their relatively large sample sizes, public availability, and the similarity of data collection procedures used across individual datasets.

In addition to processing the raw CoRR datasets, I will also inject noise of three types: Rician, Gaussian, and 1-voxel. Rician noise was selected because it is commonly present during MRI

acquisition [28]. Gaussian noise was selected as it is a simple and commonly observed form of noise found in many domains. 1-voxel noise refers to the targeted perturbation of a single voxel within an image and has been shown to have a significant effect on some processing applications in MRI [7]. The specific details regarding how many variations of each noise settings will be tested will be determined once the original processing of the CoRR datasets has completed, as the processing duration of the selected pipelines will inform this decision.

Once the CoRR datasets have been processed using each pipeline on each of the four noise conditions (with the fourth being noise-free), I will compute each of the following:

1. $\hat{C}$ with respect to noise, conditional on a fixed pipeline and dataset. This most closely resembles the work presented in [27] and traditional condition analyses;
2. $\hat{C}$ with respect to the dataset in the no noise setting, conditional on a fixed tool. This is equivalent to evaluating the generalizability of derivatives across datasets; and
3. $\hat{C}_t$ with respect to the tool in the no noise setting, conditional on a fixed dataset. This allows us to quantify the impact that tool selection has on results.

In each of the above, I will also compute the Test-Retest reliability in the form of Discriminability [18] as a proxy for the consistency of meaningful variation in the outputs. I will produce plots that summarize the instability across each of the moving variables in each setting.

In addition to the above experiment, I have sought funding to begin a collaboration with Dr. Camille Maumet which may allow for the extension of this method to functional MRI applications and tools.

The successful completion of this proposed project has the potential to shed light on the effect that numerical instabilities have on neuroimaging analyses, and identify a significant dependence between scientific claims and the tools used to generate them. I will use the methodology and statistic developed in this chapter in Chapter 3, where I will begin to identify sources of instability or tool-dependence in pipelines.

Chapter 3: Exploring Sources of Instability & Dependence Within Pipelines

In the previous chapter, I developed a method for empirically estimating the estimated condition of analyses with respect to arbitrary fixed variables in the analysis such as algorithms, complete pipelines, dataset, or noise. While in the previous chapter this was explored to evaluate the latter three components, this chapter aims to describe the stability of unique algorithms in the context of larger composite pipelines. By assessing the stability of pipeline components as well as the entire pipeline, we may be able to identify sources of instability within pipelines and then substitute them for more stable tools which perform a similar function, ultimately improving the quality of the pipeline. Table 2 summarizes the plan for experiments proposed in this chapter which aim to address this.

Following closely the paradigm established in Chapter 2, I will process the same datasets and pipelines as above and perturb input data to estimate the stability of each tool. Instead of applying this step solely at the beginning of pipelines and observing the ultimate effect, I will dissect the pipelines into several key components: denoising (i.e. eddy correction and auto-alignment), registration to a template, model fitting (i.e. estimating tensors or orientation direction functions from the diffusion images), and tractography and graph generation.

Table 2: Summarized experiment plan for the work performed in Chapter 3 of the proposed thesis.

Purpose	Explaining sources of instability and variability in tools		
Outcomes	❏ Comparison of the stability of individual pipeline components with the overall tool stability ❏ Identification of sources of instability in pipelines ❏ Principled method for reconstructing pipelines with stable algorithms		
Datasets	**Modality**	**Derivatives**	**Tools**
Consortium of Reproducibility and Reliability	Diffusion MRI, Structural MRI	Structural Connectomes	Dipy, FSL, MRtrix

Experiment	❏ Dissect structural pipelines into independently runnable components
	❏ For each pipeline component, beginning with the first:
	  ❏ Process CoRR datasets (or derivatives of previous step) with:
	  ❏ 1-voxel perturbation
	  ❏ Rician noise*
	  ❏ Gaussian noise
	  ❏ Calculate conditioning for each component across:
	  ❏ Noise (fixed tool and dataset)
	  ❏ Datasets (fixed tool)
	  ❏ Tool (fixed datasets)
	❏ Compare each algorithm for each setting		

Notes	❏ *Rician noise will only be added to either raw MR images or minimally preprocessed MR images, since it is unexpected in other contexts.		

Each of the processing steps will be evaluated independently, and noise injected immediately prior to each step. In each case where the data structure being processed is different, the noise properties introduced will need to vary accordingly. For instance, in the case of diffusion tensors being supplied to tractography, Rician noise is not realistic to appear in this data.

Analyzing each component following the same procedure as above, we will be able to identify sources of instability within pipelines. Extending this, we will be able to consider the relative instability of each pipeline component with respect to the composite pipeline stability, and identify a relationship between them. This may shed important light on the nature of stability in

the processing of neuroimaging data, such that instability in a particular processing step has potentially more significant impact on the total pipeline as compared to another. This chapter will demonstrate a principled method for the construction of highly generalizable software pipelines.

Future Work

The projects proposed throughout my thesis focus primarily on the assessment of the stability of pipelines and analyses in neuroimaging. While I believe this work is imperative for producing truly trustworthy scientific findings, it makes no claim as to the quality of the algorithms tested. In Chapter 2 I briefly mentioned using Test-ReTest reliability in addition to stability scores, and while this can serve as a stand-in for a lower-bar of the quality of derivatives, it is a metric which can be cheated and skewed by non-meaningful similarity or differences in derivatives.

Future work in this area I would like to pursue includes assessing other properties of pipelines including their accuracy, processing performance, robustness, and generalizability. These could each be explored through methods similar to those proposed here, performing a bootstrap on splits in datasets and use a relevant statistic to evaluate the subsets of data processed each iteration. I believe the extension of this paradigm to other summary statistics which address the concerns above would be incredibly valuable for the future of pipeline development, both in neuroimaging and other domains.

Suitability Statement

I believe that I am well suited to complete the proposed thesis due to my strong background in computational science, neuroimaging, and numerical methods. During my Master's degree, I spent several years developing and evaluating a one-click structural neuroimaging pipeline, ndmg [18], which has been optimized for the reliability of its results. I have since worked on projects using distributed computing resources for tackling neuroimaging applications [29], developing standards for representing analysis tools both domain-agnostically [13] and in neuroimaging specifically [16].

Conclusion

The project proposed here aims to explore a crucial axis of computational science, and that may play a significant role in the current reproducibility epidemic. The successful completion of these projects will result in a suite of tools and methods that researchers will be able to easily extend to use on their own applications for developing, deploying, and sharing their experiments, and evaluating the stability of their findings and tools. The outcome of this project has considerable implication on the state of many existing scientific explorations and tools used throughout neuroimaging and can serve as a springboard for many future studies exploring the quality and development of new computational analysis methods.

Expected Contributions to Knowledge

Chapter 1

- ❏ An accessible and portable tool for developing, performing and sharing reproducible and provenance-rich experiments

Chapter 2

- ❏ A statistic for estimating the stability or condition of pipeline and algorithms empirically
- ❏ Understanding the stability of commonly used tools in diffusion MRI processing
- ❏ Understanding the degree of tool-dependence for scientific claims in neuroimaging

Chapter 3

- ❏ Method for identifying sources of instability in pipelines
- ❏ Understanding the relationship between the stability of algorithms and pipelines

Expected First-Author Publications

Chapter 1

1. Clowdr (under review: Front. in Neuroinf.)

Chapter 2

2. An estimator for estimating conditioning of tools (target: IEEE Trans. on Med. Img.)
3. Exploring stability and generalizability in neuroimaging analyses (target: Nature)

Chapter 3

4. Identifying sources of instability within neuroimaging pipelines (target: eLife)

Expected Collaborations

1. Boutiques - Dr. Tristan Glatard
2. Mapping structural and functional connectivity - Estefany Suarez, Dr. Bratislav Misic
3. Network evolution in development - Dr. Budha Khundrakpam
4. Heritability of structural connectomes - Dr. Joshua T. Vogelstein, Dr. Carey E. Priebe
5. Evaluating the stability of functional MRI software - Dr. Camille Maumet
6. Enhancing data discovery and querying - Dr. Jean-Baptiste Poline

References

[1] P. Patil, R. D. Peng, and J. Leek, “A statistical definition for reproducibility and replicability,” *bioRXiv*, Jul. 2016.

[2] Open Science Collaboration, “PSYCHOLOGY. Estimating the reproducibility of psychological science,” *Science*, vol. 349, no. 6251, p. aac4716, Aug. 2015.

[3] K. A. Baggerly and K. R. Coombes, “DERIVING CHEMOSENSITIVITY FROM CELL LINES: FORENSIC BIOINFORMATICS AND REPRODUCIBLE RESEARCH IN HIGH-THROUGHPUT BIOLOGY,” *Ann. Appl. Stat.*, vol. 3, no. 4, pp. 1309–1334, 2009.

[4] J. P. A. Ioannidis, “Why most published research findings are false,” *PLoS Med.*, vol. 2, no. 8, p. e124, Aug. 2005.

[5] A. Bowring, C. Maumet, and T. E. Nichols, “Exploring the Impact of Analysis Software on Task fMRI Results,” *bioRXiv*, Mar. 2018.

[6] A. Salari, L. Scaria, G. Kiar, and T. Glatard, *Numerical error propagation in the HCP structural pre-processing pipelines*. 2018.

[7] L. B. Lewis, “Robustness and reliability of cortical surface reconstruction in CIVET and FreeSurfer,” *Annual Meeting of the Organization for Human Brain Mapping*, 2017.

[8] X.-N. Zuo *et al.*, “An open science resource for establishing reliability and reproducibility in functional connectomics,” *Sci Data*, vol. 1, p. 140049, Dec. 2014.

[9] M. Jenkinson, C. F. Beckmann, T. E. J. Behrens, M. W. Woolrich, and S. M. Smith, “FSL,” *Neuroimage*, vol. 62, no. 2, pp. 782–790, Aug. 2012.

[10] J. D. Tournier and F. Calamante, “MRtrix: diffusion tractography in crossing fiber regions,” *International Journal of*, 2012.

[11] E. Garyfallidis *et al.*, “Dipy, a library for the analysis of diffusion MRI data,” *Front. Neuroinform.*, vol. 8, p. 8, Feb. 2014.

[12] T. Glatard, G. Kiar, T. Aumentado-Armstrong, N. Beck, R. Ferreira da Silva, and M. E. Rousseau, “Boutiques: A descriptive command-line framework,” *Zenodo*. Sep-2017.

[13] T. Glatard *et al.*, “Boutiques: a flexible framework to integrate command-line applications in computing platforms,” *Gigascience*, vol. 7, no. 5, Mar. 2018.

[14] K. Gorgolewski *et al.*, “Nipype: a flexible, lightweight and extensible neuroimaging data processing framework in python,” *Front. Neuroinform.*, vol. 5, p. 13, Aug. 2011.

[15] K. J. Gorgolewski *et al.*, “The brain imaging data structure, a format for organizing and describing outputs of neuroimaging experiments,” *Sci Data*, vol. 3, p. 160044, Jun. 2016.

[16] K. J. Gorgolewski *et al.*, “BIDS apps: Improving ease of use, accessibility, and reproducibility of neuroimaging data analysis methods,” *PLoS Comput. Biol.*, vol. 13, no. 3, p. e1005209, 2017.

[17] G. Kiar, W. R. Gray Roncal, D. Mhembere, E. W. Bridgeford, R. Burns, and J. T. Vogelstein, “ndmg: NeuroData’s MRI Graphs pipeline,” *Zenodo*. Aug-2016.

[18] G. Kiar *et al.*, “A High-Throughput Pipeline Identifies Robust Connectomes But Troublesome Variability,” Cold Spring Harbor Laboratory, 2018.

[19] G. Kiar, “Clowdr: Accessible pipeline deployment and sharing,” *Zenodo*. Mar-2018.

[20] M. Črepinšek, S.-H. Liu, and M. Mernik, “Replication and comparison of computational experiments in applied evolutionary computing: Common pitfalls and guidelines to avoid them,” *Appl. Soft Comput.*, vol. 19, pp. 161–170, Jun. 2014.

[21] T. Sherif *et al.*, “CBRAIN: a web-based, distributed computing platform for collaborative neuroimaging research,” *Front. Neuroinform.*, vol. 8, p. 54, May 2014.

[22] A. L. Vaccarino et *al*., “Brain-CODE: A Secure Neuroinformatics Platform for Management, Federation, Sharing and Analysis of Multi-Dimensional Neuroscience Data,” *Front. Neuroinform.*, vol. 12, p. 892, May 2018.
[23] R. A. Poldrack et *al*., “Toward open sharing of task-based fMRI data: the OpenfMRI project,” *Front. Neuroinform.*, vol. 7, p. 12, Jul. 2013.
[24] D. E. Rex, J. Q. Ma, and A. W. Toga, “The LONI Pipeline Processing Environment,” *Neuroimage*, vol. 19, no. 3, pp. 1033–1048, Jul. 2003.
[25] P. Bellec, S. Lavoie-Courchesne, P. Dickinson, J. P. Lerch, A. P. Zijdenbos, and A. C. Evans, “The pipeline system for Octave and Matlab (PSOM): a lightweight scripting framework and execution engine for scientific workflows,” *Front. Neuroinform.*, vol. 6, p. 7, Apr. 2012.
[26] J. Davidson, “Regression Diagnostics: Identifying Influential Data and Sources of Collinearity.” JSTOR, 1981.
[27] S. Skare, M. Hedehus, M. E. Moseley, and T. Q. Li, “Condition number as a measure of noise performance of diffusion tensor data acquisition schemes with MRI,” *J. Magn. Reson.*, vol. 147, no. 2, pp. 340–352, Dec. 2000.
[28] S. Basu, T. Fletcher, and R. Whitaker, “Rician noise removal in diffusion tensor MRI,” *Med. Image Comput. Comput. Assist. Interv.*, vol. 9, no. Pt 1, pp. 117–125, 2006.
[29] G. Kiar et *al*., “Science in the cloud (SIC): A use case in MRI connectomics,” *Gigascience*, vol. 6, no. 5, pp. 1–10, May 2017.

Appendix 1: Clowdr Paper

Please find the Clowdr paper attached, beginning on the following page.

A Serverless Tool for Platform Agnostic Computational Experiment Management

Gregory Kiar[a,b], Shawn T. Brown[a], Tristan Glatard[*,c], Alan C. Evans[*,a,b,d]

a: Montreal Neurological Institute, McGill University, Montreal, Canada
b: Department of Biomedical Engineering, McGill University, Montreal, Canada
c: Department of Computer Science, Concordia University, Montreal, Canada
d: Department of Neurology and Neurosurgery, McGill University, Montreal, Canada
*: Authors contributed equally
Corresponding author email: <greg.kiar@mcgill.ca>

Abstract

Neuroscience has been carried into the domain of big data and high performance computing (HPC) on the backs of initiatives in data collection and an increasingly compute-intensive tools. While managing HPC experiments requires considerable technical acumen, platforms and standards have been developed to ease this burden on scientists. While web-portals make resources widely accessible, data organizations such as the Brain Imaging Data Structure and tool description languages such as Boutiques provide researchers with a foothold to tackle these problems using their own datasets, pipelines, and environments. While these standards lower the barrier to adoption of HPC and cloud systems for neuroscience applications, they still require the consolidation of disparate domain-specific knowledge. We present Clowdr, a lightweight tool to launch experiments on HPC systems and clouds, record rich execution records, and enable the accessible sharing of experimental summaries and results. Clowdr uniquely sits between web platforms and bare-metal applications for experiment management by preserving the flexibility of do-it-yourself solutions while providing a low barrier for developing, deploying and disseminating neuroscientific analysis.

The increasing adoption of distributed computing, including cloud and high-performance computing (HPC), has played a crucial role in the expansive growth of neuroscience. With an emphasis on big-data analytics, collecting large datasets such as the Consortium for Reliability and Reproducibility [1], UK-Biobank [2], and Human Connectome Project [3] is becoming increasingly popular and necessary. While these datasets provide the opportunity for unprecedented insight into human brain function, their size makes non-automated analysis impractical.

At the backbone of science is the necessity that claims are reproducible. The reproducibility of findings has entered the spotlight as a key question of interest, and has been explored extensively in psychology [4], neuroimaging [5], [6], and other domains [7]. Computational

experiments must be re-executable as a critical condition for reproducibility, and this bare minimum requirement becomes increasingly challenging with larger datasets and more complex analyses. When re-executable applications fail to reproduce findings, there is a gray area where the source of errors are often unknown and may be linked to misinterpretation of data, computing resources or undocumented execution details, rather than scientific meaning.

As a result, new tools and standards have emerged to aid in producing more reusable datasets and tools, and thereby more reproducible science. The Brain Imaging Data Structure (BIDS) [8] and the associated BIDS apps [9] prescribe a standard for sharing and accessing datasets, and therefore, increasing the accessibility and impact of both datasets and tools. The Boutiques framework [10] allows for software documentation in a machine-interpretable way, allowing the automation of tool execution and evaluation, and software containerization initiatives such as Docker [11] and Singularity [12] allow this to happen consistently across arbitrary computing environments with minimal burden on the user.

Web-platforms such as OpenNeuro [13], LONI Pipeline [14], and CBRAIN [15] simplify the analysis process further by providing an accessible way to construct neuroscience experiments on commonly-used tools and uploaded-datasets. These systems deploy tools on HPC environments and record detailed execution information so that scientists can keep accurate records and debug their workflows. Tools such as LONI's provenance manager [16], Reprozip [17], and ReCAP [18] capture system-level properties such as system resources consumed and files accessed, where tools supporting the Neuroimaging Data Model (NIDM) [19], a neuroimaging-specific provenance model based on W3C-PROV [20], capture information about the domain-specific transformations applied to the data of interest.

The initiatives enumerated above have synergistic relationships, where each solves a small but significant piece of the larger puzzle that is computational and scientific reproducibility and replicability. However, the learning curve associated with adopting each of these technologies is considerable and leveraging them in an impactful way is difficult. We present Clowdr, a lightweight tool that lowers the barrier on researchers to develop, perform, and disseminate reproducible, provenance-rich neuroscience computation.

Emergent Technologies in Reproducible Neuroscience

Conducting reproducible analyses in neuroscience requires many complementary facets, building on technologies which are commonly adopted as *de facto* standards.

(i) Data and code interoperability Due in part to its simplicity and active public development community, BIDS [8] has become an increasingly prominent data organization format in neuroimaging. This standard makes use of the Nifti file format [21] and human-readable JSON files to capture both imaging and subject-specific data. An important benefit of this data organization is the ability to launch data processing pipelines in the form of BIDS applications [9], which expose a consistent set of instructions compatible with the data organization. Together, these complementary standards are suitable for performing a large variety of

neuroimaging experiments. In contexts where tools have heterogeneous interfaces, or data organizations are custom-built for a particular context, the Boutiques [10], [22] framework allows the rich description of a pipeline such that tool execution, validation, and exploration can be automated.

(ii) Software virtualization While virtual machines have long been used for deploying analysis pipelines with complex dependencies in heterogeneous environments, software containers have recently emerged as lighter-weight alternatives suitable for transient data processing applications. Docker [11] provides this functionality across all major host operating systems, but is often not supported by HPC centers due to security vulnerabilities [23], [24]. Singularity [12] addresses the security risks of Docker, but currently only supports Linux operating systems, filling the niche of containerization on shared computing resources. A detailed comparison in the context of medical imaging can be found in [25].

(iii) Workflow engines For researchers composing custom pipelines, Nipype [26] (Python) and PSOM [27] (GNU Octave or MATLAB) enable the construction of dependency graphs between pipeline components, and allow the deployment either to cluster scheduling software or using multiple threads. These tools contain interfaces for many popular tools and simplify their adoption. They also embed provenance capture, fault-tolerance features, and data tracking to avoid recomputations across similar executions.

(iv) Provenance Building on the W3-PROV [20] standard for data provenance metadata put forth by the World Wide Web Consortium, NIDM [19] is a standard which represents processing and data provenance specific to neuroimaging analyses. While this standard is machine-interpretable and interoperable-by-design, supporting this standard currently requires tight integration with analysis pipelines. In LONI pipeline, a provenance model exists which includes detailed records of data use and file lifecycle [16], which is designed to inform data consumers what types of analyses can be and have been performed with the data in question; this tool is tightly coupled with the LONI pipeline ecosystem. The ReCAP [18] project has been developed to evaluate the resource consumption of arbitrary pipelines on the cloud and can aid in cloud-instance optimization. While this tool has potential for a large impact in designing both cost effective and scalable analyses, there is considerable overhead as it manages executions through a persistent server and workflow engine. While various other libraries exist to monitor some piece of data or compute provenance, Reprozip [17] is perhaps the most exciting as it uniquely captures records of all files accessed and created throughout an execution, which allows for the creation of rich file dependency graphs. The limitation of this technique is that it requires data of interest to be written to disk, as opposed to managed in memory.

(v) Web Platforms Increasing the portability and accessibility of launching large scale analyses, web platforms such as CBRAIN [15], LONI pipeline [14], and OpenNeuro [13] provide science-as-a-service where users can upload and process their data on distant computing resources. Additionally, these platforms provide an accessible and immediate way to share the results produced from experiments with the public. These tools provide incredible value to the

community and allow the deployment of production-level pipelines from the web, but they are not suitable for prototyping analyses or developing pipelines, and it is cumbersome to run these services on a lab's own resources. In addition, monolithic Web interfaces are only suitable for a certain type of use-cases and high-level users, while developers or computer-savvy users prefer to rely on modular command-line tools and libraries.

The Clowdr Microtool

While the technologies enumerated above are essential pieces towards reproducible neuroscience, they are largely isolated from one another and place a large burden on researchers who wish to adopt all of these best practices. Clowdr leverages these tools to increase the deployability, provenance capture, and shareability of experiments. In summary, Clowdr:

(i) is tightly based on Boutiques and is BIDS-aware, supporting both arbitrary pipelines and providing an accessible entrypoint for neuroimaging;

(ii) executes both bare-metal workflows and Docker or Singularity virtualized pipelines through Boutiques on local, HPC, and cloud resources;

(iii) supports the parallelized batch deployment of pipelines constructed with workflow-engines, while being agnostic to programming language and construct;

(iv) captures system-level provenance information (i.e. CPU and RAM usage), supports Reprozip, and internal provenance captured by arbitrary pipelines such as NIDM; and

(v) supports the deployment of both development- and production-level tools without an active server, and provides a web-report for exploring and sharing experiments.

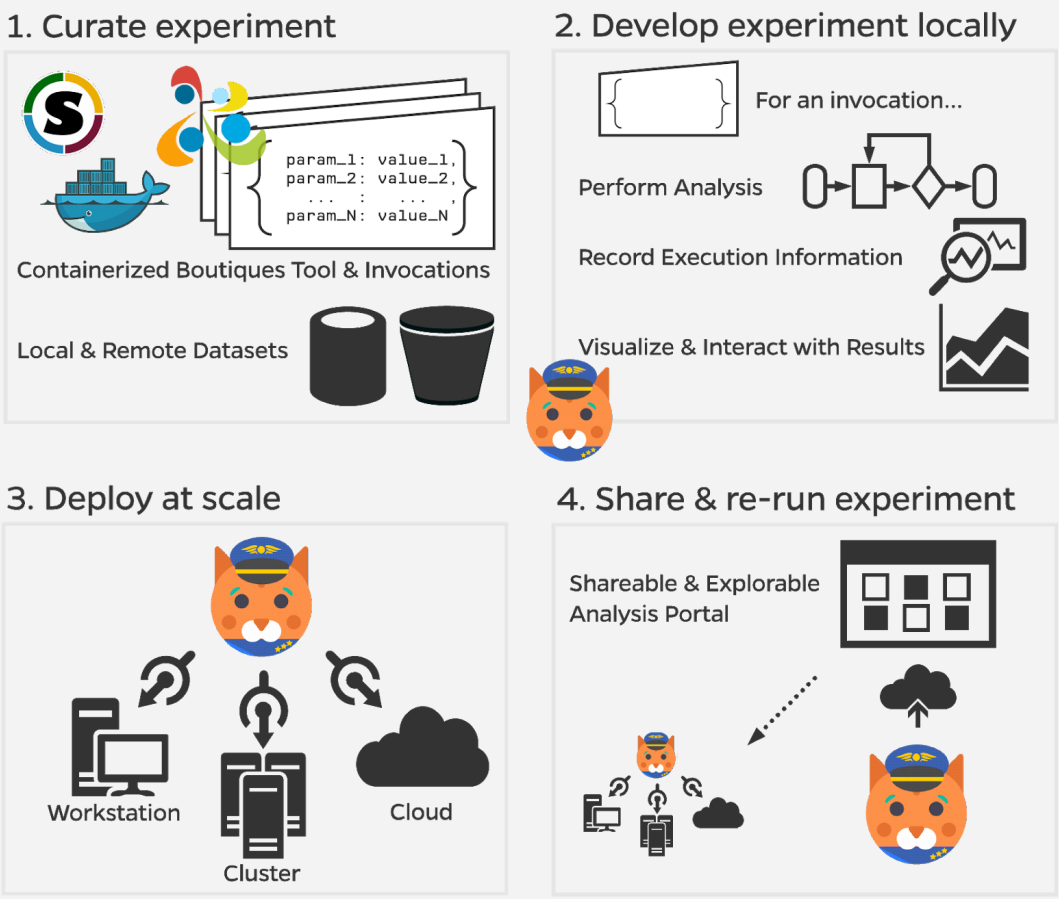

Figure 1: Clowdr Workflow. (1) Prior to launching an analysis with Clowdr, users must curate the dataset(s) of interest and analysis tools, like in any analysis. For the sake of portability, Clowdr supports containerized applications described with Boutiques, and both local and Amazon S3-hosted datasets. (2) Clowdr then analyze a subset of the dataset through local-deployments while collecting records of each execution for review, enabling the scientist to document the tuning of their analysis. (3) Clowdr can then be used to deploy experiments on either a local workstation, HPC environment, or an available cloud resource. (4) The metadata collected during these executions can be explored and shared. Together, Clowdr encapsulates an entire experiment reproducibly and with low overhead to the scientist.

A typical workflow using Clowdr is summarized in Figure 1. A Clowdr experiment follows the same workflow as traditional experiments, beginning with tool and data curation through prototyping, deployment, and exploration. Clowdr enhances this experimental process by providing an easy-to-use and lightweight method for launching analyses in distributed, consistent, and richly recorded experimental environments. Clowdr provides graphical reports which can be explored and published alongside the experimental derivatives.

Figure 2 shows the execution lifecycle within Clowdr. Starting from user-provided Boutiques descriptor and invocation(s), and access to any required datasets, Clowdr begins by identifying a list of tasks to launch. For a new experiment, tasks are identified in one of three main ways: 1) a one:one mapping from a list of invocations, 2) a one:many mapping from a single invocation in which parameter(s) have been specified for sweeping during execution, or a BIDS-specific 3) one:many mapping from a single invocation for a BIDS app, which will iterate upon the participant- and session-label fields, and described in the BIDS app specification [9]. Experiments can be re-run and determine the task-list based on whether a full, failure-only, or incomplete-only re-execution is desired. Once the task-list is determined, Clowdr creates an independent invocation which explicitly defines the arguments used in each task.

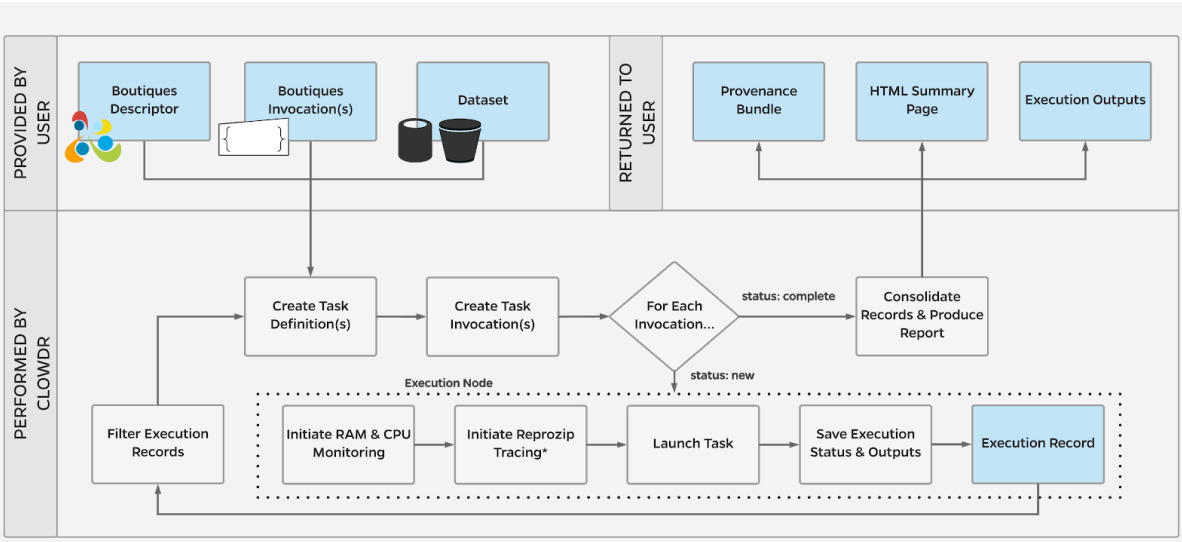

Figure 2: Clowdr Data Flow. Beginning with user-supplied tool and data descriptions, Clowdr identifies unique tasks to launch and wraps each with usage and log monitoring tools, to ultimately provide a rich record of execution to the user alongside the expected output products of the experiment. Clowdr ultimately produces an HTML summary for users to explore, update, filter, and share the record of their experiment. In the above schematic, blue boxes indicate data, where gray indicate processing steps. *External reprozip tracing is supported on limited infrastructures, as running virtualized environments within a trace capture requires elevated privileges which may be a security risk on some systems.

At this stage, Clowdr distributes tasks to the Cloud system or local cluster scheduler being used for deployment. Each task is launched through a Clowdr-wrapper, which initializes CPU and RAM monitoring and triggers Reprozip tracing prior to launching the analysis itself. Reprozip tracing has limited support in conjunction with containerized analyses on HPC systems due to potential security issues. Reprozip is built upon the Linux command “ptrace,” which traces processes to monitor or control them. To eliminate the potential risk of using this tool, it is common for systems to disallow the tracing of administrator-level processes. The requirement of limited administrator privileges by Singularity (during the creation of multiple user

namespaces) and Docker (for interacting with the daemon) makes encapsulating these tools within the restricted ptrace scope not possible on many shared systems. For more information on the specific conditions in which these technologies can be made to interoperate please view the Github repository for this manuscript, linked below.

Upon completion of the analysis, Clowdr bundles the system monitored records, standard output and error, exit status, and any other information collected by either the tool itself or the Boutiques runtime engine, and concludes its execution. Once the experiment has begun, Clowdr provides the user with the Clowdr provenance directory which will be updated automatically as executions progress.

The researcher can monitor the provenance directory using the Clowdr share portal (Figure 3), which provides a web interface summarizing the task executions. Once the analysis concludes, the figures on this web page and the associated metadata can be saved and serve as a record of the experiment either for evaluation or dissemination alongside published results.

The Clowdr package is open-source on Github [28], and installable through the Python Package Index.

Performing Experiments with Clowdr

Here we explore an experiment in which we used Clowdr to process the Human Connectome Project (HCP) [3] dataset with a structural and functional connectome estimation pipeline, ndmg [29], [30]. The records of this experiment, and materials and instructions that can be used to reproduce a similar analysis with Clowdr using the publicly available DS114 BIDS dataset [13] an the example BIDS application [9] can be found on Github at: https://github.com/clowdr/clowdr-paper. Specific packages and their versions for both experiments can be found at the end of this manuscript.

As summarized above, performing an analysis with Clowdr requires the creation of a Boutiques descriptor summarizing the pipeline of interest, an invocation containing the parameters to supply to this pipeline on execution, and curation of the data to be processed.

Clowdr experiments can be launched locally, on cluster, or submitted to cloud resources. In each case, invocations and task definitions are created locally, and then the jobs are run serially, submitted to a cluster queue, or pushed to cloud storage and called remotely, respectively. Upon completion of each tasks, summary files created by Clowdr can be either inspected manually or consolidated and visualized in the web with the Clowdr share tool (Figure 3).

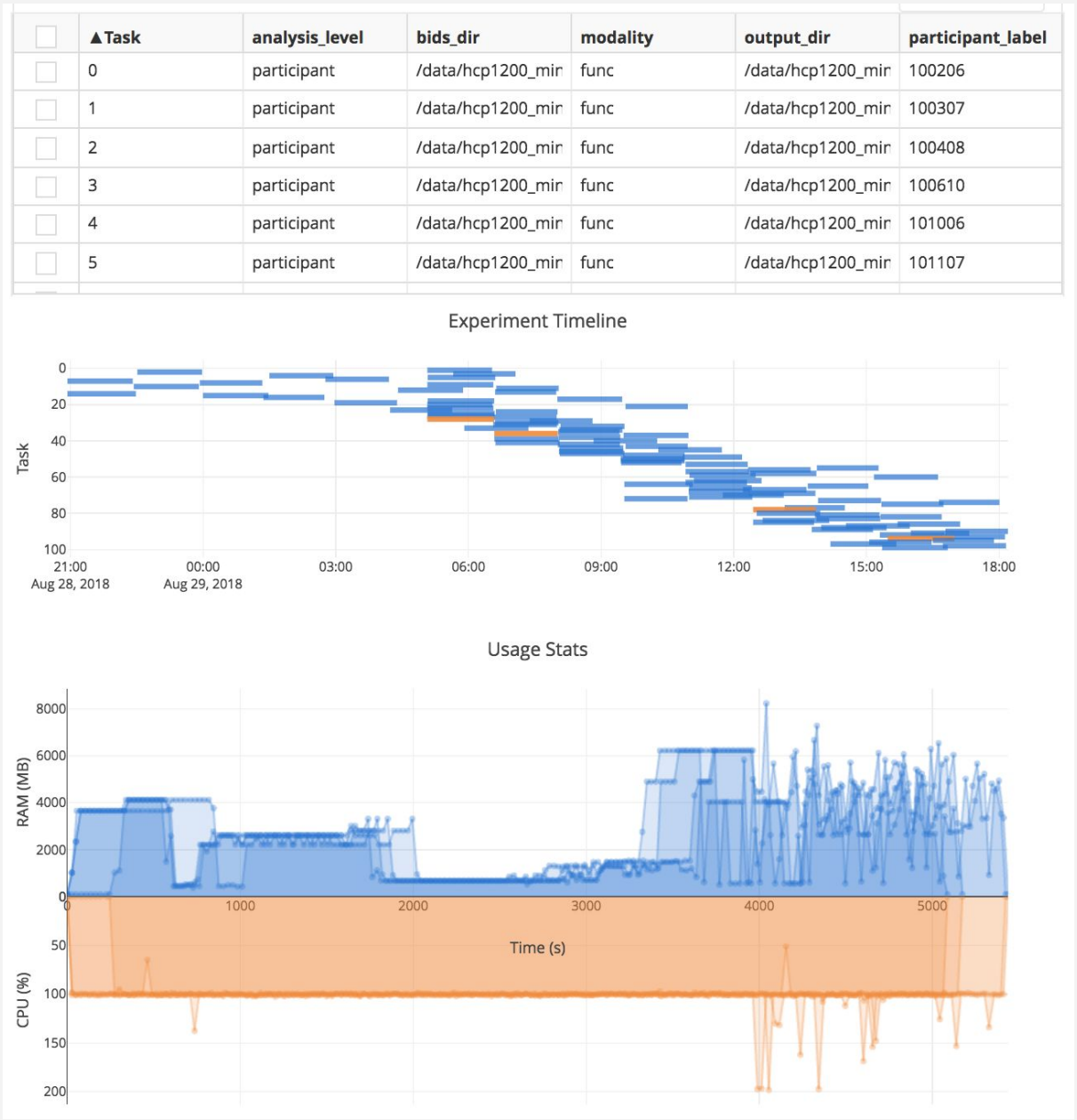

Figure 3: Clowdr Experiment Viewer. Once a Clowdr experiment has been created a lightweight service can be launched to monitor experiment progress and outcomes. The page is produced using Plotly Dash to produce highly interactive plots and tables, enabling rich filtering and exploration of executions. The table can be toggled to present summary statistics about experiment execution or invocation parameters identifying parameters used for each task in the experiment. The subsequent Gantt plot shows the timeline of executed tasks in the experiment, where those selected for visualization in the usage plot below are highlighted. The final plot in this view shows the memory and processing footprint throughout all selected tasks. Selection and filtering may be done by value in the tables or selection in the task timeline.

The share tool, launchable on any computer with access to the experiment, creates a lightweight web service displaying summary statistics and invocation information from the experiment, including memory usage, task duration, launch order, and log information. The visualizations provided are filterable and sortable, enabling users to interrogate and identify outliers in their experiment, explore potential sources of failure, and effectively profile the analysis pipeline in use. The modified figures can be downloaded from this interface, serving as accessible records of execution.

In the example above, the HCP dataset has been used to create estimates of structural and functional connectivity. The plot has been filtered to show the usage for the five most RAM-intensive tasks when performing functional connectome estimation. Using these executions as a reference, we are able to identify resource requirements and an experimental trajectory for the ndmg pipeline, and identify outliers in our execution by either filtering explicitly on processing statistics, or comparing across individual trajectories and output or error logs. Clowdr provides a layer of quality control on executions, in addition to that which is regularly performed by researchers on their datasets. This enables automated outlier extraction from an analysis, regardless of the application.

While the share tool currently requires maintaining an active server, the plots can be exported statically and it is in the development roadmap to enable exporting the entire web page as static files, as discussed here: https://github.com/plotly/dash/issues/266. Since the record created by Clowdr is stored in the machine-readable and JSON format, researchers can easily extract their records and integrate it into other interfaces that suit their application.

Discussion

Clowdr addresses several barriers to performing reproducible neuroscience. Clowdr experiments consist of enclosed computational environments, versioned-controlled Boutiques-described tools with explicit usage parameters, rich execution history, and can be re-executed or distributed with minimal effort. Clowdr provides an accessible interface for initially running analyses locally, and translating them seamlessly to HPC environments. The rich record keeping provided with Clowdr is system-agnostic resulting in uniformly interpretable summaries of execution. As a Python library, Clowdr can be used as a module in a larger platform, or directly as a command-line tool.

Clowdr uniquely packages an executable tool summary, parameters, and results together, in a language- and tool-agnostic way, and therefore, greatly increasing the transparency and shareability of experiments. Importantly, this adds clarity to experimental failures and documents the hyper-parameter tuning process of experiments, which has been historically largely undocumented in literature [31].

Where workflow systems such as Nipye and PSOM provide rich provenance records and enable complex construction of processing pipelines, Clowdr maintains the ability to wrap these

as “black box” tools. This permits users to use these tools and benefit from their architecture without requiring knowledge of the underlying language constructs being used in these systems. Platforms like CBRAIN provide a similar type of abstraction, where tools are treated as single objects, but these systems come with the added overhead of maintaining complex database architectures, and primarily accessing resources through a web-based interface. An additional limitation of large platforms is that they are often designed for consumers of widely-adopted tools consumers rather than tool developers. Clowdr fills the void between these two extremes of pipeline deployment systems by providing a programmatic tool-independent method for managing job submission and collecting provenance across multiple architectures and enabling the rapid prototyping of analyses.

Several immediate applications of the provenance information captured by Clowdr include the benchmarking of tools and resource optimization during the selection of cloud resources, as was done in [18]. While the value of comparing provenance records has not been demonstrated here, other studies such as [32] have demonstrated the efficacy of leveraging provenance information to identify sources of variability or instability within pipelines.

Future work includes adopting a W3C-PROV compatible format for Clowdr provenance records, increasing the machine-readability and interoperability of these records with other standards such as NIDM. Integrating the reports produced by Clowdr with a system such as Datalad would allow for record versioning and more strictly enforce the complete reporting of experiments. Clowdr will also continually be extended to support more HPC schedulers, clouds, and provenance capture models.

Acknowledgements

Funding for this work was provided by The Canada First Research Excellence Fund, Healthy Brains for Health Lives, and The Natural Sciences and Engineering Research Council of Canada, and the Canadian Institutes of Health Research. The authors would also like to thank Pierre Rioux for his insight and many helpful discussions. The authors declare no conflicts of interest.

Tools and Versions

The following is a list of tools and data used in this manuscript, and their respective versions. The architecture and analysis presented for the Clowdr package corresponds to version 0.1.0. The key Python packages and specific versions tested are: boutiques (version 0.5.12), boto3 (1.7.81), botocore (1.10.81), slurmpy (0.0.7), psutil (5.4.7), pandas (0.23.4), plotly (3.1.1), and plotly-dash (0.24.1), including dash-core-components (0.27.1), dash-html-components (0.11.0), dash-renderer (0.13.0), dash-table-experiments (0.6.0), and flask (0.12.2). Executions were tested locally using Docker (17.12.0-ce), and on Compute Canada’s Cedar high performance cluster using Singularity (2.5.1-dist). The Docker container used for ndmg can be found on Dockerhub as neurodata/m3r-release (0.0.5), which contains ndmg (0.1.0-f). The Singularity container used was pulled and dynamically created from this Dockerhub endpoint. The dataset use was a subset of the HCP 1200 collection [3].

References

[1] X.-N. Zuo et al., “An open science resource for establishing reliability and reproducibility in functional connectomics,” *Sci Data*, vol. 1, p. 140049, Dec. 2014.
[2] C. Sudlow et al., “UK biobank: an open access resource for identifying the causes of a wide range of complex diseases of middle and old age,” *PLoS Med.*, vol. 12, no. 3, p. e1001779, Mar. 2015.
[3] D. C. Van Essen et al., “The WU-Minn Human Connectome Project: an overview,” *Neuroimage*, vol. 80, pp. 62–79, Oct. 2013.
[4] Open Science Collaboration, “PSYCHOLOGY. Estimating the reproducibility of psychological science,” *Science*, vol. 349, no. 6251, p. aac4716, Aug. 2015.
[5] A. Eklund, T. E. Nichols, and H. Knutsson, “Cluster failure: Why fMRI inferences for spatial extent have inflated false-positive rates,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 113, no. 28, pp. 7900–7905, Jul. 2016.
[6] Alexander Bowring Camille Maumet, “Exploring the Impact of Analysis Software on Task fMRI Results,” *BioRxiv*.
[7] M. Baker, “1,500 scientists lift the lid on reproducibility,” *Nature*, vol. 533, no. 7604, pp. 452–454, May 2016.
[8] K. J. Gorgolewski et al., “The brain imaging data structure, a format for organizing and describing outputs of neuroimaging experiments,” *Sci Data*, vol. 3, p. 160044, Jun. 2016.
[9] K. J. Gorgolewski et al., “BIDS apps: Improving ease of use, accessibility, and reproducibility of neuroimaging data analysis methods,” *PLoS Comput. Biol.*, vol. 13, no. 3, p. e1005209, 2017.
[10] T. Glatard et al., “Boutiques: a flexible framework to integrate command-line applications in computing platforms,” *Gigascience*, vol. 7, no. 5, May 2018.
[11] D. Merkel, “Docker: Lightweight Linux Containers for Consistent Development and Deployment,” *Linux J.*, vol. 2014, no. 239, Mar. 2014.
[12] G. M. Kurtzer, V. Sochat, and M. W. Bauer, “Singularity: Scientific containers for mobility of compute,” *PLoS One*, vol. 12, no. 5, p. e0177459, May 2017.
[13] R. A. Poldrack et al., “Toward open sharing of task-based fMRI data: the OpenfMRI project,” *Front. Neuroinform.*, vol. 7, p. 12, Jul. 2013.
[14] D. E. Rex, J. Q. Ma, and A. W. Toga, “The LONI Pipeline Processing Environment,” *Neuroimage*, vol. 19, no. 3, pp. 1033–1048, Jul. 2003.
[15] T. Sherif et al., “CBRAIN: a web-based, distributed computing platform for collaborative neuroimaging research,” *Front. Neuroinform.*, vol. 8, p. 54, May 2014.
[16] I. Dinov et al., “Neuroimaging study designs, computational analyses and data provenance using the LONI pipeline,” *PLoS One*, vol. 5, no. 9, Sep. 2010.
[17] F. S. Chirigati, D. E. Shasha, and J. Freire, “ReproZip: Using Provenance to Support Computational Reproducibility,” *Tappi J.*, 2013.
[18] K. Hasham, K. Munir, and R. McClatchey, “Cloud infrastructure provenance collection and management to reproduce scientific workflows execution,” *Future Gener. Comput. Syst.*, vol. 86, pp. 799–820, Sep. 2018.
[19] V. Sochat and B. N. Nichols, “The Neuroimaging Data Model (NIDM) API,” *Gigascience*, vol. 5, no. suppl_1, pp. 23–24, Nov. 2016.
[20] P. Missier, K. Belhajjame, and J. Cheney, “The W3C PROV Family of Specifications for Modelling Provenance Metadata,” in *Proceedings of the 16th International Conference on Extending Database Technology*, Genoa, Italy, 2013, pp. 773–776.
[21] R. W. Cox et al., “A (Sort of) New Image Data Format Standard: NIfTI-1: WE 150,” *Neuroimage*, vol. 22, p. e1440, Jun. 2004.
[22] T. Glatard, G. Kiar, T. Aumentado-Armstrong, N. Beck, R. Ferreira da Silva, and M. E. Rousseau, “Boutiques: A descriptive command-line framework,” *Zenodo*. Sep-2017.
[23] T. Bui, “Analysis of Docker Security,” *arXiv [cs.CR]*, 13-Jan-2015.
[24] T. Combe, A. Martin, and R. D. Pietro, “To Docker or Not to Docker: A Security Perspective,” *IEEE Cloud Computing*, vol. 3, no. 5, pp. 54–62, 2016.
[25] J. Matelsky, G. Kiar, E. Johnson, C. Rivera, M. Toma, and W. Gray-Roncal, “Container-Based Clinical Solutions for Portable and Reproducible Image Analysis,” *J. Digit. Imaging*, vol. 31, no. 3, pp. 315–320, May 2018.
[26] K. Gorgolewski et al., “Nipype: a flexible, lightweight and extensible neuroimaging data processing framework in python,” *Front. Neuroinform.*, vol. 5, p. 13, Aug. 2011.
[27] P. Bellec, S. Lavoie-Courchesne, P. Dickinson, J. P. Lerch, A. P. Zijdenbos, and A. C. Evans, “The pipeline system for Octave and Matlab (PSOM): a lightweight scripting framework and execution engine for scientific workflows,” *Front. Neuroinform.*, vol. 6, p. 7, Apr. 2012.
[28] G. Kiar, “Clowdr: Accessible pipeline deployment and sharing,” *Zenodo*. Mar-2018.
[29] G. Kiar, W. R. Gray Roncal, D. Mhembere, E. W. Bridgeford, R. Burns, and J. T. Vogelstein, “ndmg: NeuroData’s MRI Graphs pipeline,” *Zenodo*. Aug-2016.
[30] G. Kiar et al., “A High-Throughput Pipeline Identifies Robust Connectomes But Troublesome Variability,” Cold

Spring Harbor Laboratory, 2018.
[31] J. Reunanen, “Overfitting in Making Comparisons Between Variable Selection Methods,” *J. Mach. Learn. Res.*, vol. 3, no. Mar, pp. 1371–1382, 2003.
[32] A. Salari, L. Scaria, G. Kiar, and T. Glatard, *Numerical error propagation in the HCP structural pre-processing pipelines*. 2018.