Progress Report: Characterizing and Leveraging Numerical Instabilities in Neuroimaging Through Perturbation Analysis

Gregory Kiar
October 18th, 2019

Outline

Thesis Overview

With an explosion in data collection across nearly all scientific disciplines, we are entering the big-data era of scientific exploration. From the Sloan Digital Sky Survey [1] in cosmology, to the UK BioBank [2], Human Connectome Project [3], and the Consortium of Reproducibility and Reliability (CoRR) [4] in Neuroscience, initiatives are being launched to federate this data-tsunami, many with particular emphasis and commitment towards open data-sharing.

As access to data has increased drastically, and the availability of diverse and customizable processing tools has grown similarly, it has become apparent that a lack of reproducibility in data analysis is becoming a plague in many disciplines of science [5]. While on occasion this is the result of p-hacking (i.e. the modification of analyses in search of significant results), it is likely due to much more innocent means such as software bugs [6], operating system selection [7], tool variability [8]–[10], or instability in the face of noise [10], [11]. In the case of growing datasets and increased automation, neuroimaging pipelines are often treated as black boxes used to fit alignments, segmentations, or models of the brain using data with variable signal to noise properties. The nature of these operations leaves them vulnerable to instability when presented with minor perturbations in either the data themselves or their processing implementations. Though the independent evaluation of atomic pipeline components may be feasible in some cases, as was done by Skare et al. [11], this ranges from impractical to impossible in the case of multi-step pipelines.

The objective of this thesis is to characterize instabilities observed in neuroimaging pipelines, identify their significance on downstream analyses and their potential role in a lack of reproducibility in neuroimaging, and attempt to leverage any observed instabilities for creating more stable estimators of brain structure or function. This body of work is thus split into the following five components:

1. Creation of computing infrastructures which enable the deployment, perturbation, and evaluation of software pipelines
2. Comparison of various perturbation models for observing instabilities within structural connectome estimation pipelines
3. Evaluation of the analytical impact of stabilities using domain-tested approaches
4. Aggregation of unstable derivatives into more stable estimators
5. Broad comparison of algorithms, tools, and both their raw and aggregated derivatives to provide recommendations for processing choices within the neuroimaging community

The successful completion of my proposed project has the potential to simultaneously shed light on an important issue that has had an unquantified impact on results in neuroimaging, and mitigate this issue through the statistical evaluation and aggregation of unstable derivatives. The recommendations provided will allow scientists to make principled decisions on tool selection for their studies, and inform further work quantifying the effect of instabilities across other modalities of neuroimaging or disciplines of computational science.

Degree Progress to Date

At our last committee meeting for my comprehensive examination at the beginning of degree-year 2, I had been nearing the completion of my first degree chapter. My involvement in the growing Boutiques and BIDS application standards, as well as development of tooling around them, has resulted in two publications [12], [13]. Building upon these standards, I developed a tool, Clowdr, which enabled the scalable deployment and perturbation of pipelines expressed in these standards across HPC environments, which has since been published in Frontiers in Neuroinformatics [14]. While papers surrounding these infrastructure efforts have been published, their development is ongoing and they continue to be extended to increase their useability both by tool developers and consumers.

With infrastructure in place, I completed the following chapter which captures the exploration of pipeline stability and the leveraging of instabilities to generate stable derivatives. I have begun the evaluation of various perturbation methods on pipeline performance. In collaboration with Dr.s Pablo Oliviera and Eric Petit, I demonstrated the effect of- and identified potential use cases for both iterative (Monte Carlo Arithmetic [15], as implemented by MCALIB [16] and Verificarlo [17]) and one-shot (1-voxel noise injection, demonstrated by Lewis et al. [10]) perturbation methods. This work also begins to explore the aggregation of simulated derivatives, and shows basic properties of naive pooling strategies, such as taking the mean or given percentiles across all observations. I have both shared this work as a preprint [18] and had it accepted for a talk at the Computational Reproducibility at Exascale workshop at the Super Computing conference in Fall 2019. This work is appended at the end of this document and will be submitted to a special issue of the International Journal of High Performance Computing Applications which is dedicated to the study of computational reproducibility.

Using Chapter 2 as a springboard, I have begun work on the following two chapters exploring the analytical impact of instability and more complex aggregation strategies, respectively. I will evaluate analytical impact of instabilities through both graph statistics (in collaboration with Dr. Bratislav Misic) and the use of measures to evaluate the biological feasibility of fibers (in collaboration with Dr. Ariel Rokem). To explore more complex methods of derivative aggregation, I secured an NSERC Michael Smith Forgein Study Supplement (MS-FSS) and Mitacs Globalink Research award to study at Inria (Bretagne-Atlantique) with Dr.s Camille Maumet and Elisa Fromont, where I explored the efficacy of 3D Deep Convolutional Neural Networks (3D CNNs) for minimizing the differences between statistical maps across observations. This work took place over the months of April – June 2019 and will be adapted for structural connectome aggregation for my thesis. Alongside these efforts, I have collaborated with Dr. Gaël Varoquaux on integrating optimizations for image resampling in the Nilearn library to minimize the number of floating point operations which are performed in the (common) special case when no rescaling of the field of view is needed. I plan to continue collaboration with Dr. Varoquaux on the development of simulation-aware aggregation methods, and generate an estimate of the reliability for pooled results.

Proposed Chapters

As stated above, the proposed thesis will consist of work spread across five chapters, spanning the creation of infrastructures for perturbation analysis, an evaluation of the impact of numerical instabilities within neuroimaging, and recommendations for how researchers can navigate these issues.

Chapter 1: Facilitating FAIR Tool Creation, Consumption, and Deployment

Building upon the Boutiques [12] descriptive command-line framework for describing and evaluating tools, I built Clowdr [14] to manage, distribute, and log the execution of Boutiques-described tasks across cloud and cluster environments. This tool facilitates the perturbation and permutation of pipeline executions at large scales, and enables the study of stability that follows in this thesis. In addition to the development of Clowdr, the Boutiques Python Package facilitates the publication and access of tool descriptions, execution parameters, and experimental outcomes, ensuring that all performed analyses are FAIR [19].

Chapter 2: Comparing Methods for Identifying Instabilities in Pipelines

Prior to evaluating the impact of instabilities or whether aggregation is needed, it was necessary to first evaluate methods for observing instabilities within pipelines. Two key types of perturbations were tested: one-shot and iterative. The one-shot perturbation method refers to a disturbance occurring at one stage of processing, such as the 1-voxel noise injection method used by Lewis et al. in [10]. Iterative perturbations refer to much smaller disturbances being introduced throughout tool execution and are here based on Monte Carlo Arithmetic (MCA) [15]. Through tool instrumentation with MCA using mcalib [16] and Verificarlo [17], each floating point operation may be instrumented such that the operation is simulated with a random zero-biased bit-flip at the target precision (in our case, the least significant bit). MCA allows for the detection of instabilities which emerge due to the rounding errors and numerical imprecision that are present in all floating point operations. My work evaluating these perturbation methods for a deterministic tractography pipeline in Dipy has been accepted for a talk at the Computational Reproducibility at Exascale workshop at Super Computing in November 2019, and the pre-print [18] will be submitted to the International Journal of High Performance Computing Applications.

Chapter 3: Evaluating Analytical Impact of Instabilities

Given the presence of instabilities in the deterministic pipeline tested, we can extend this evaluation to look at the impact of these differences in a typical analytical or biological context. Through collaboration with Dr.s Bratislav Misic and Ariel Rokem, I will evaluate simulated connectomes and graphs, respectively, on these axes. Additionally, I will evaluate more complex pipelines in the Dipy ecosystem, such as those using higher dimensional tensor models or probabilistic tractography. This will allow us to identify which algorithms (as implemented with approximately equivalent quality) are inherently more stable and allow for the selection or improvement of methods within this open source tool.

Chapter 4: Improving Pipeline Stability Through Statistical Aggregation

Following the evaluation of the analytical impact of unstable derivatives, I will explore methods for aggregating these derivatives. I will explore the distributions of observed simulated graphs and identify the distribution of values being observed. In addition to the naive application of mean and median values, I will identify the expected value of the observed distribution and perform aggregation accordingly. The accuracy of aggregates will be evaluated using the methods shown in Chapter 3. The variability of aggregates will also be evaluated, allowing us to make informed decisions regarding the number of simulations necessary to obtain a stable aggregate. In addition to the aggregation methods above, the Convolutional Neural Network Autoencoder developed with Dr.s Camille Maumet and Elisa Fromont for reconstruction of functional statistical maps will be modified for structural connectomes, allowing for the evaluation of aggregations spanning a wide range of complexity.

Chapter 5: Improving the Stability of Pipelines Through Instability

Combining the work shown in all preceding chapters, I will demonstrate the application of these methods for the comparison and ranking of tools, algorithms, and tool-dataset combinations. Pipelines from MRtrix and FSL will also be evaluated, and multiple datasets will be used for the comparison, highlighting the importance of tool selection in the context of one's dataset. It will provide recommendations for researchers in terms of how they may be able to perform their analyses with these libraries with respect to obtaining the most stable estimators of structural connectivity, and thus, the most trustworthy analytical findings.

First Author Journal Publication Summary

#	Paper Name	Working Title *(Current Status)* Brief Description
1	Clowdr	A Serverless Tool for Platform Agnostic Computational Experiment Management *(Published: Frontiers in Neuroinformatics [14])* Introduces Clowdr, a tool for launching Boutiques-described tools on HPC environments, iterating over parameters and perturbations, and recording rich provenance logs.
2	Perturbation Methods	Comparing Perturbation Models for Evaluating Stability of Neuroimaging Pipelines *(Talk: CRE19; Preprint [18]; Target: International Journal of High Performance Computing Applications)* Evaluates the impact of MCA and 1-voxel noise on a diffusion connectome estimation workflow. Identifies potential use cases for each noise mode, and introduces the concept of aggregating unstable derivatives.
3	Analytical Impact	Identifying the Analytical Impact of Numerical Instabilities in Neuroimaging Pipelines *(Target: Nature Communications)* Extends the Perturbation Methods paper above to a) evaluate multiple algorithms within Dipy, b) include evaluation of the analytic impact of instability by proxy of various commonly used network statistics and test-retest reliability, and c) attempts normalization as a method for difference mitigation.
4	Aggregation	Estimating Expected Value From Empirical Derivative Distributions Across Simulated Pipeline Executions *(Target: IEEE Transactions in Medical Imaging)* Explores the structure of deviations observed in the Analytical Impact paper, and fits a distribution to them so that the expected value of the derivatives may be estimated. Also continues 3D CNN work for structural connectomes.
5	Tool Comparison	Uncovering the Relative Impact of Numerical Instabilities Across Popular Neuroimaging Libraries *(Target: eLife)* Combines the Analytical Impact and Aggregation papers with the addition of a) more pipelines across other libraries, such as FSL and MRtrix, b) provides a score for each pipeline tested, and c) makes recommendations for researchers to minimize the vulnerabilities of their analyses to instabilities.

External Collaborators

Clowdr: N/A
Perturbation Methods: Dr. Pablo Oliviera, Dr. Eric Petit
Analytical Impact: Dr. Bratislav Misic, Dr. Ariel Rokem
Aggregation: Dr. Camille Maumet, Dr. Elisa Fromont, Dr. Gaël Varoquaux
Tool Comparison: All of the above

Timeline

Degree years	Milestone
0 – 0.5	Ph.D: Initial Meeting Ch. 1: Clowdr started
0.5 – 1	Ch. 1: Boutiques published Ch. 1: Clowdr submitted
1 – 1.5	Ph.D: Comprehensive Exam Ch. 1: Clowdr published Ch. 2: Perturbation Methods started
1.5 – 2	Ch. 2: Perturbation Methods awarded talk and submitted Ch. 3: Analytical Impact started Ch. 4: Aggregation started
current date	all following milestones and dates are "planned"
2 – 2.5	Ch. 2: Perturbation Methods published Ch. 3: Analytical Impact submitted Ch. 4: Aggregation submitted Ch. 5: Tool Comparison started
2.5 – 3	Ch. 3: Analytical Impact published Ch. 4: Aggregation published Ch. 5: Tool Comparison submitted Ph.D: Thesis Pre-Submission
3.5 – 4	Ph.D: Write Thesis Ch. 5: Tool Comparison published Ph.D: Defend Thesis

References

[1]   D. G. York *et al.*, "The sloan digital sky survey: Technical summary," *Astron. J.*, vol. 120, no. 3, p. 1579, 2000.

[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]   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.

[5]   M. Baker, "1,500 scientists lift the lid on reproducibility," *Nature*, vol. 533, no. 7604, pp. 452–454, May 2016.

[6]   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.

[7]   T. Glatard *et al.*, "Reproducibility of neuroimaging analyses across operating systems," *Front. Neuroinform.*, vol. 9, p. 12, Apr. 2015.

[8]   A. Klein *et al.*, "Evaluation of 14 nonlinear deformation algorithms applied to human brain MRI registration," *Neuroimage*, vol. 46, no. 3, pp. 786–802, Jul. 2009.

[9]   A. Bowring, C. Maumet, and T. E. Nichols, "Exploring the Impact of Analysis Software on Task fMRI Results," *biorXiv*, Mar. 2018.

[10] L. B. Lewis *et al.*, "Robustness and reliability of cortical surface reconstruction in CIVET and FreeSurfer," *Annual Meeting of the Organization for Human Brain Mapping*, 2017.

[11] 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.

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

[13] 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, Mar. 2017.

[14] G. Kiar, S. T. Brown, T. Glatard, and A. C. Evans, "A Serverless Tool for Platform Agnostic Computational Experiment Management," *Front. Neuroinform.*, vol. 13, p. 12, Mar. 2019.

[15] D. S. Parker, *Monte Carlo Arithmetic: exploiting randomness in floating-point arithmetic*. University of California (Los Angeles). Computer Science Department, 1997.

[16] M. Frechtling and P. H. W. Leong, "MCALIB: Measuring Sensitivity to Rounding Error with Monte Carlo Programming," *ACM Trans. Program. Lang. Syst.*, vol. 37, no. 2, pp. 5:1–5:25, Apr. 2015.

[17] C. Denis, P. de Oliveira Castro, and E. Petit, "Verificarlo: Checking Floating Point Accuracy through Monte Carlo Arithmetic," *2016 IEEE 23nd Symposium on Computer Arithmetic (ARITH)*. 2016.

[18] G. Kiar *et al.*, "Comparing Perturbation Models for Evaluating Stability of Post-Processing Pipelines in Neuroimaging," *arXiv [q-bio.NC]*, 28-Aug-2019.

[19] M. D. Wilkinson *et al.*, "The FAIR Guiding Principles for scientific data management and stewardship," *Sci Data*, vol. 3, p. 160018, Mar. 2016.

Comparing Perturbation Models for Evaluating Stability of Neuroimaging Pipelines

Gregory Kiar[1], Pablo de Oliveira Castro[2], Pierre Rioux[1], Eric Petit[3],
Shawn T. Brown[1], Alan C. Evans[1], Tristan Glatard[4]
[1]McGill University, Montreal, Canada; [2]University of Versailles, Versailles, France;
[3]Exascale Computing Lab, Intel, Paris, France; [4]Concordia University, Montreal, Canada.

Abstract—With an increase in awareness regarding a troubling lack of reproducibility in analytical software tools, the degree of validity in scientific derivatives and their downstream results has become unclear. The nature of reproducibility issues may vary across domains, tools, datasets, and computational infrastructures, but numerical instabilities are thought to be a core contributor. In neuroimaging, unexpected deviations have been observed when varying operating systems, software implementations, or adding negligible quantities of noise. In the field of numerical analysis these issues have recently been explored through Monte Carlo Arithmetic, a method involving the instrumentation of floating point operations with probabilistic noise injections at a target precision. Exploring multiple simulations in this context allow the characterization of the result space for a given tool or operation. In this paper we compare various perturbation models to introduce instabilities within a typical neuroimaging pipeline, including i) near-epsilon noise, ii) Monte Carlo Arithmetic, and iii) operating system variation, to identify the significance and quality of their impact on the resulting derivatives. We demonstrate that even low-order models in neuroimaging such as the structural connectome estimation pipeline evaluated here are sensitive to numerical instabilities, suggesting that stability is a relevant axis upon which tools are compared, alongside more traditional criteria such as biological feasibility, computational efficiency, or, when possible, accuracy. Heterogeneity was observed across participants which clearly illustrates a strong interaction between the tool and dataset being processed, requiring that the stability of a given tool be evaluated with respect to a given cohort. We identify use cases for each perturbation method tested, including quality assurance, pipeline error detection, and local sensitivity analysis, and make recommendations for the evaluation of stability in a practical and analytically-focused setting. Identifying how these relationships and recommendations scale to higher-order computational tools, distinct datasets, and their implication on biological feasibility remain exciting avenues for future work.

Index Terms—neuroimaging, diffusion MRI, stability, Monte Carlo Arithmetic

I. INTRODUCTION

A lack of computational reproducibility [1] has become increasingly apparent in the last several years, calling into question the validity of scientific findings affected by published tools. Reproducibility issues may have numerous sources of error, including undocumented system or parametrization differences and the underlying numerical stability of algorithms and implementations employed. While containerization

can mitigate the extent of machine-introduced variability, understanding the effect that these sources of error have on the encapsulated numerical algorithms remains difficult to explore. In simple cases where algorithms are differentiable or invertible, it is possible to obtain closed-form solutions for their stability. However, as software pipelines grow, containing multiple complex steps, using non-linear optimizations and non-differentiable functions, the stability of these algorithms must be explored empirically.

As neuroscience has evolved into an increasingly computational field, it has suffered from the same questions of numerical reproducibility as many other domains [2]. In particular, neuroimaging often attempts to fit alignments, segmentations, or models of the brain using few samples with variable signal to noise properties. The nature of these operations leaves them potentially vulnerable to instability when presented with minor perturbations in either the data themselves or their processing implementations. The independent evaluation of atomic pipeline components may be feasible in some cases, as was done by Skare et al. in [3]. Here, the authors computed the theoretical conditioning of various tensor models used in diffusion modeling, and compared these values to the observed variances in tensor features when fit on simulated data. While approaches like the above provide valuable insights to algorithms and their implementations independently, the impact of these stepwise instabilities within composite pipelines remains unknown. Even if one were able to evaluate each step within a pipeline, identifying the impact these instabilities may have on a result when composed together, both structurally and analytically, remains practically difficult to evaluate.

Various forms of instability have been observed in structural and functional magnetic resonance (MR) imaging, including across operating system versions [4], minor noise injections [5], as well as dataset or implementation of theoretically equivalent algorithms [6], [7]. These approaches may have practical applications in decision making, such as deciding which tool/implementation should be used for an experiment. However, they are relatively far removed from the underlying numerical instabilities being observed. Recent advances in numerical analysis allow for the replacement of floating point operations with Monte Carlo Arithmetic simulations [8] which inject a random zero-bias rounding error to operations for a target floating-point precision [8], [9]. This method can be used for evaluating the numerical stability of tools by wrapping

existing analyses [9] and providing a foothold for scientists wishing to explore the space of their pipeline's compound instabilities [10].

In this paper we explore the effect of various perturbations on a typical diffusion MR image processing pipeline through the use of i) near-epsilon noise injections, ii) Monte Carlo Arithmetic, and iii) varying operating systems to identify the quality and severity of their impact on derived data. This evaluation will inform future work exploring the stability of these pipelines and downstream analyses dependent upon them. The processing pipeline selected for exploration is DiPy [11], popular tool that generates structural connectivity maps (connectomes) for each participant. The pipeline accepts de-noised and co-registered images as inputs, and then performs two key processing steps: tensor fitting and tractography. We demonstrate the relative impact that each of the tested perturbation methods has on the resulting connectomes and explore the nature of where these differences emerge.

II. Methods

All processing described below was run using servers provided by Compute Canada. Software pipelines were encapsulated and run using Singularity [12] version 2.6.1. Tasks were submitted, monitored, and provenance captured using Clowdr [13] version 0.1.2-1. All code for performing the experiments and creating associated figures are available on GitHub at https://github.com/gkiar/stability and https://github.com/gkiar/stability-mca, respectively.

A. Dataset and pre-processing

The dataset used for processing is a 10-session subset of the Nathan Kline Institute Rockland Sample dataset (NKI-RS) [14]. This dataset contains high fidelity structural, functional, and diffusion MR data and is openly available for research consumption. The 10 sessions used were chosen by randomly selecting 10 participants and selecting their alphabetically-first session of data. This data was preprocessed prior to the modelling evaluated here using a standard de-noising and image alignment pipeline [15] built upon the FSL toolbox [16]. The steps in this pipeline include eddy current correction, brain extraction, tissue segmentation, and image registration. The boundary between white and gray matter was obtained by computing the difference between a dilated version of the white matter mask and the original. Data volumes at this stage of processing are four-dimensional and variable in spatial extent (first three dimensions) with a fixed number of diffusion directions (fourth dimension), totalling approximately $100^3 \times 137$ voxels in each case.

B. Modeling

After pre-processing the raw diffusion data using FSL, structural connectomes were generated for an 83-region cortical and sub-cortical parcellation [17] using Dipy [11]. A six-component tensor model was fit to the diffusion data residing within white matter. Seeds were generated in a $2 \times 2 \times 2$
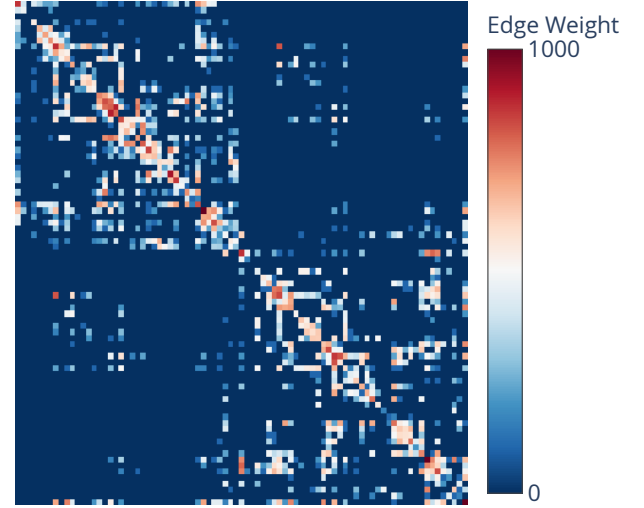

Fig. 1. **Example connectome**. Each row and column corresponds to a region within the brain, and the intersection a connection between them. If no connection is found between regions, the edge strength is zero. If a streamline is found to connect two regions, the weight is incremented by 1. The resulting weights are the sum of all observed connections for every streamline traced within a brain image.

arrangement for each voxel within the boundary mask, resulting in 8 seeds per boundary voxel. Deterministic tracing was then performed using a half-voxel step size, and streamlines shorter than 3-points in length were discarded as spurious. Once streamlines were generated they were traced through the parcellation. Edges were added to the graph corresponding to the end-points of each fiber, and were weighted by the streamline count. This pipeline was implemented in Python, including a few components in Cython, and relies on the Numpy library for a large proportion of operations. Each resulting network is a square connectivity matrix of 83×83 edges, as shown in Fig. 1. This pipeline was chosen as it is both common and simple relative to many alternatives.

C. Stability Evaluation

Near-epsilon and Monte Carlo perturbation modes were tested 100x per image. Noise was represented by percent deviation of the Frobenius norm of a resulting connectome from the corresponding reference (no noise injection). A deviation of 50% indicates that the norm of the difference between the noisy and reference networks is 50% the size of the norm of the reference graph. This is formalized below in Eq. (1):

$$\%Dev(A, B) = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij} - b_{ij}|^2} \Big/ \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2}, \quad (1)$$

where A is the reference graph, B is the perturbed graph, and $\square_{ij}$ is an element therein at row i and column j.

2

The perturbation methods evaluated, presented below, are summarized in Table I.

D. Subject-Level Variation

Comparison between subjects will be used as a reference error. If the differences observed by other methods are similar in magnitude to the subject-level difference, then the validity of the processed networks for use in downstream phenotypic analysis becomes questionable as subjects cannot be reliably distinguished from one another. This error is computed as the pairwise distance between all 10 subjects included in this cohort.

E. Near-Epsilon Noise

The goal of near-epsilon noise was to inject data perturbations sufficiently small that the resulting images would be indistinguishable from the original. This is meant to test the lower-bound of noise sensitivity for processing pipelines. The type of near-epsilon noise used here will be referred to as 1-voxel noise and is similar to the method employed in [5]. In our case, the intensity of a single voxel in the defined range will be scaled based on a scaling factor. The voxels modified in this case were randomly generated within the mask of brain regions being modeled by the pipeline.

The two modes of 1-voxel noise injection tested here were: a) a single voxel per entire image of size (X, Y, Z, D) (approximately $100^3 \times 137$ for all images), or b) a single voxel per 3D volume of size (X, Y, Z) (approximately 100^3 for all images), and are referred to as "single" and "independent" modes, respectively. The intensity of the scaling was consistent as in both cases the original intensity was doubled.

F. Monte Carlo Arithmetic

Verificarlo [10] is an extension of the LLVM compiler which automatically instruments floating point operations at build-time for software written in C, C++, and Fortran. Once compiled with Verificarlo, the Monte Carlo emulation method and target precision can be set as environment variables. For all simulations a rounding error on the least significant floating point bit in the mantissa (bit 53) was introduced. The simulations were computed using the custom QUAD backend which is optimized to reduce computation time over the traditional mcalib MPFR backend leveraging GNUs multiple precision library [9]. Noise through Verificarlo can be injected as "Precision Bounded", simulating floating point cancellations, "Random Rounding", simulating only rounding errors on computation, and "MCA", which includes both of these modes. A particularity of the Random Rounding mode is that it only injects rounding noise on inexact floating-point operations (i.e. operations that have a rounding error in IEEE-754 at the target precision). Therefore, RR mode preserves the original exact operations, it is a more conservative noise simulation. We used both the RR and MCA modes of simulation.

Verificarlo was used to instrument tools in two modes we will refer to as "Python" and "Full Stack". In the Python instrumentation, the core Python libraries were recompiled

with Verificarlo as well as any subsequently installed Cython libraries. In the Full Stack instrumentation, BLAS and LA-PACK were also recompiled, meaning that Numpy, a dominant Python library for linear algebra, was also instrumented. The Full Stack implementation did not run successfully using the MCA mode. We suspect that some libraries require exact floating-point operations or are sensitive to cancellation errors, so only the Random Rounding (RR) mode was able to be evaluated for the Full Stack. These images are available on DockerHub at gkiar/fuzzy-python.

G. Operating System Variation

Operating system noise was evaluated across Alpine Linux 3.7.1 and Ubuntu 16.04. Alpine is a lightweight distribution which comes with minimal packages or libraries, and Ubuntu is a popular Linux distribution with a large user and development community. Alpine was chosen as its lightweight nature makes it an efficient choice for the packaging and distribution of libraries in scientific computing, reducing the overhead of shipping code towards data sources. Ubuntu was chosen due to is high adoption and community support by major libraries. While Alpine comes with a minimal set of libraries, a core difference between these systems as noted by DistroWatch (https://distrowatch.com/) is their dependence on a different version of the Linux kernel.

Ubuntu was used as the base operating system for all simulations other than this comparison.

H. Aggregation of Simulated Graphs

To structurally evaluate each simulation setting, connectomes were aggregated within setting and subject combinations. Several aggregation methods were explored to preserve various sensitivity and stability properties across the aggregated graphs. In each case, the operations are performed edge-wise, so the aggregated graph is not guaranteed to be single graph in the set of perturbed graphs. The aggregation operations are the edge-wise mean and the 0th (min), 10th, 50th (median), 90th, and 100th (max) %-iles. The mean aggregate will include a non-zero weight for every edge which appears in at least one simulation, and the 0th and 100th %-iles will include the lowest and highest observed weight for every edge, respectively. The 90th, 50th, and 10th %-iles increasingly aggressively filter edges based on their

TABLE I
DESCRIPTION OF PERTURBATION MODES

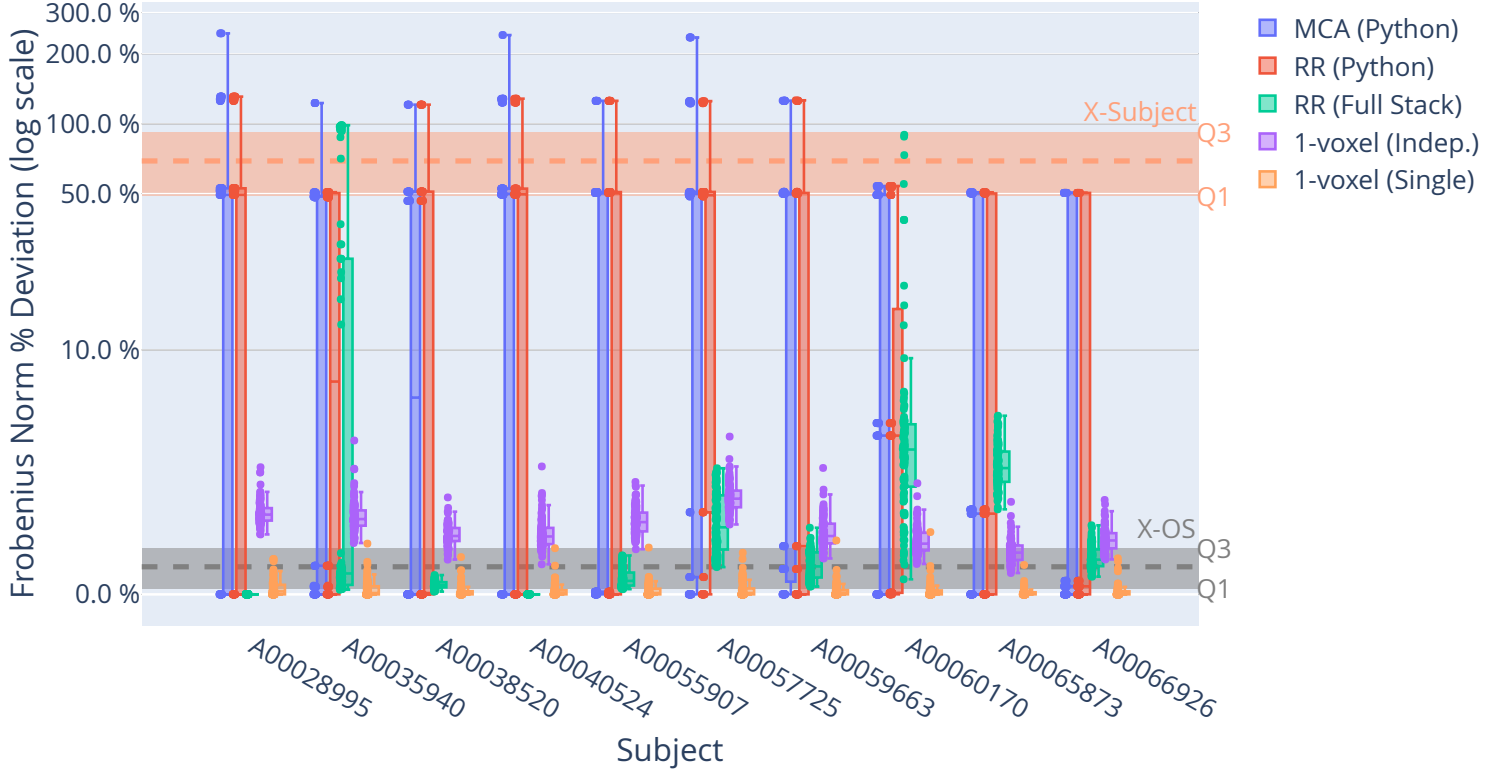| Permutation	Description
1-voxel	Intensity value doubled for either **Single** (one voxel in entire 4D volume) or **Independent** (one voxel per 3D sub-volume) voxels.
MCA	Simulation of all floating point operations in **Python** (Python and Cython-compiled libraries).
RR	Simulation of all rounding operations in **Python** or the **Full Stack** (BLAS, and LAPACK, Python and Cython-compiled libraries).
X-OS	One of **Ubuntu 16.04** or **Alpine 3.7.1**.
X-Subject	Pairwise comparison of sessions based on **Subject ID**.

Fig. 2. **Comparison of perturbation modes**. As evaluated by the percent deviation from reference in the Frobenius Norm of a resulting connectome, each of the 10 processed subjects were re-processed 100 times for each perturbation method. We see that the MCA and RR (Python) methods resulted in distinct modes for the outputs in all cases reaching extreme deviations equivalent to cross-subject variation. The RR (Full Stack) method shows high variability across subjects, and only reaching cross-subject variation in the case of 2 subjects. The 1-voxel methods result in considerably less deviation from reference, and are more consistent across subjects than the RR (Full Stack) method.

prominence across simulations. The combination of percentile aggregates also enable isolation of the most spurious edges, such as by taking the difference of maximum and minimum aggregates. A volatile aggregate was created to this effect which consists of edges which are found in the maximum aggregate but not the minimum aggregate. Note that in this case, the weight for these edges is not implied and can be defined as an alternative function of the graph collection, such as mean, but as the weight does not appear when comparing binary edges, no recommendation for this weighting is made here.

III. RESULTS

All perturbation modes were applied to either the input data or post-processing pipeline described in the Section II-B, and were evaluated according to Eq. (1).

A. Perturbation Induced Differences

Fig. 2 shows the percentage deviation for each simulation mode on 10 subjects. Introduced perturbations show highly-variable changes in resulting connectomes across both the perturbation model and subject, ranging from no change to deviations equivalent to difference typically observed across

subjects. For the 10 subjects tested, we see that the Python-instrumented MCA and RR pipelines resulted in the largest deviation from the reference connectome. In these cases we also see that the results are modal, where each subject has discrete states that may be settled in, some of which result in deviations comparable to subject-level noise. This modality is likely due to minor differences introduced at crucial branch-points which then cascaded throughout the pipeline. This hypothesis is supported by observing that the Full Stack implementation with RR perturbations shows a continuous distribution of differences that are highly variable in intensity, ranging from no deviation to subject-level in some cases for some subjects.

The 1-voxel independent mode unsurprisingly produces larger changes than the 1-voxel single mode. These changes are larger than or comparable to operating system variability, respectively, resulting in small deviations from the reference, and are relatively minor in comparison to the extremes observed with Monte Carlo Arithmetic. Operating system deviations are very low or even zero in some cases. In all perturbation settings we can see that there is large variability both across simulations on the same data and across subjects.
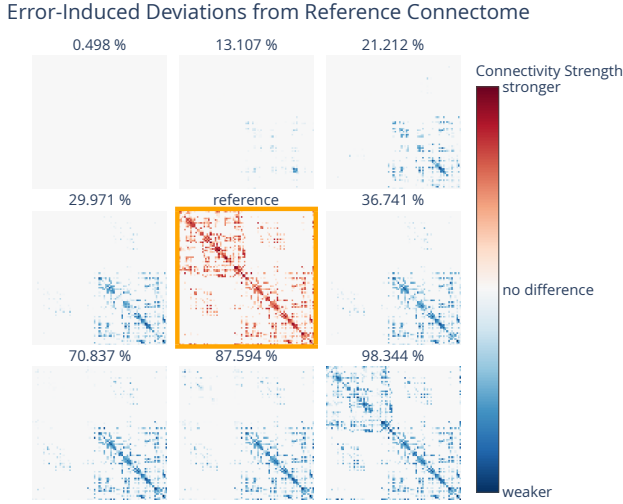
Fig. 3. **Structure of Deviations**. Shown in increasing deviation from left–right and top–bottom, with the reference in the centre, are the difference connectomes observed for the RR (Full Stack) perturbations of subject A00035940. In this case, the left hemisphere (bottom-right portion of the graph) begins to degrade quickly, eventually reaching an almost complete loss in signal.

B. Progression of Deviations in a Continuous Setting

In the case of subject A00035940, the Full Stack RR perturbations led to a continuous distribution of outputs, ranging in difference from none to subject-level from the reference. Fig. 3 explores the progression of these deviations by visualizing the difference-connectome for samples along various points of this distribution. In the center we show the reference connectome, and surrounding it the difference graph for a simulated sample with labelled $\%Dev$ from this reference. In this case, we can see a progression of structurally consistent deviations. In particular, edges corresponding to regions in the left hemisphere become increasingly distorted (bottom-right portion of the connectome), whereas the within-hemisphere connectivity for the right hemisphere (top-left portion) remains largely intact in all cases except the extreme difference case. We notice in all cases that the connectivity between regions is decreasing until the edges disappear entirely. While this behaviour is not consistent across all subjects, this observation suggests a peculiarity in the quality of data in this region for the subject in question. This could be due to artifacts caused by motion or other factors, ultimately reducing the stability of modeling connectivity in this region.

C. Structural Properties of Introduced Perturbation

While the case investigated above notably showed a significant degradation of regional signal quality for Full Stack RR noise in a single subject, Fig. 4 explores the relative change in connectivity from the reference for each perturbation mode and subject. Edges in the presented graphs are weighted by their standard deviation across all simulations for that

participant, and coloured as positive or negative deviations based on whether the mean weight for all simulations was greater or lower than the reference weight, respectively. All edges with a standard deviation of 0 across all simulations were blacked out for clarity.

For the Python instrumented MCA and RR implementations, edge weight was generally inflated non-specifically for existing edges in the reference connectome for all subjects. The Full Stack RR implementation shows significant variability across subjects, where the number of affected edges ranges from none to all. In each case where there exists some deviation, intensities appear to be spatially linked, suggesting the differences may be due to variable quality in the underlying data. In this case, Monte Carlo Arithmetic may have served to shed light on poor signal-to-noise properties present within regions of the images being modelled.

For 1-voxel noise, the differences introduced across independent injections impacted a larger portion of edges than single injections, unsurprisingly. By design (i.e. injection at random locations for each simulation), the deviations appear non-specifically spatially distributed. However, 1-voxel noise could be modified to spatially constrain the location for noise injection regionally, allowing the evaluation of modelling for particular sub-structures within the images.

D. Aggregation Across Simulations

For each simulation method there existed a graph nearly identical to the reference, but the variability introduced by these simulation was highly variable both in terms the method of perturbation used and the dataset being processed. The aggregation of the simulated graphs into a consensus graph allows features of this variation to be encoded implicitly in connectomes which may be used for downstream analyses. Fig. 5 shows the relative percentage of added and missing edges for each setting across all subjects using a variety of such aggregation methods.

By aggregating the simulated connectomes in a variety of methods, the resulting edges would be a product of applying some filter to the set of observed edges, and succinctly represented in a single graph. While minor deviations in one edge may reduce the strength of connectivity between two strongly linked regions, the addition of a connection between two regions which were previously unconnected may be significant in one aggregation method but ignored in another. In the case of the above example, despite the strength of connectivity remaining low between the newly connected nodes many graph theoretic measures rely on binarized graphs and may be considerably affected, such as the degree.

We notice that the 1-voxel independent (i.e. single voxel per 3D volume) method shows the most variability across each aggregation method. Where all of the MCA-derived methods perturb the pipeline non-locally, both epsilon-level methods add local noise at arbitrary locations. This distinction seems to manifest in more widely added or knocked-out edges for the 1-voxel cases, as the location of noise may have considerable impact on a multitude of nearby fibers, where MCA methods
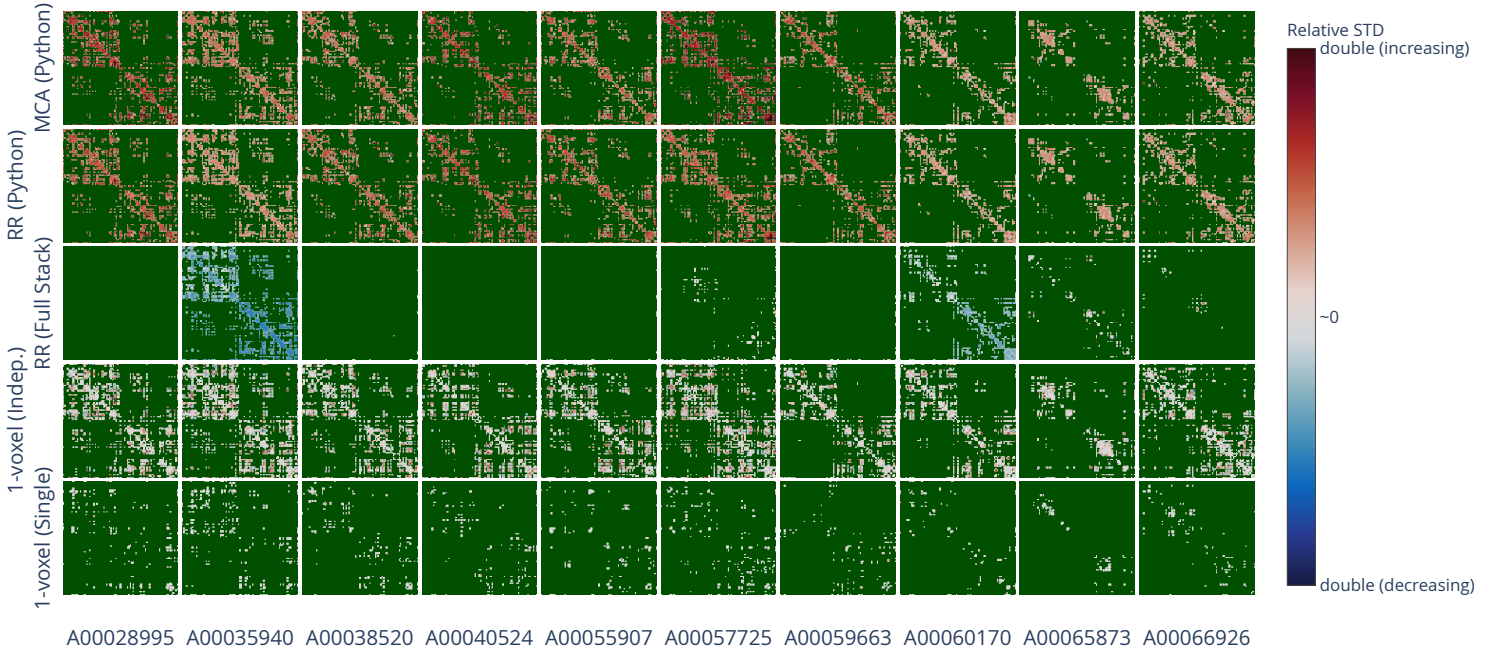
Structural Differences Across Perturbation Modes and Subjects

Fig. 4. **Perturbation introduced structural differences**. The variance of each edge is shown relative to the reference edge strength, and coloured either red or blue based on the mean perturbed weight was higher or lower than that of the reference, respectively. Edges which experienced no variation were coloured as green to be distinct from all edges which experience any variation.

have a zero-bias noise globally, meaning all deviations from the reference are spurious and due to numerical error rather than the introduction of a systemic change that sheds light on an underlying cascading instability.

Unsurprisingly, the only aggregation method which shows considerable amount of both new and missing edges is the volatile technique, which takes edges that exist in the binary difference of 100th and 0th percentile graphs, eliminating all extremely stable edges from the graph (i.e. those which exist for the reference and all simulations). While the mean sparsity of the reference graphs is 0.30, meaning 30% of possible connections have non-zero weight on average, the sparsity of the volatile aggregates ranges from 0.005 to 0.130, or, the aggregates contain between 2.5% and 43.0% the number of edges as the reference graphs.

E. Comparison of Simulation Performance

While the application of each perturbation model tested sheds light on different properties of pipeline stability, the resource consumption of these methods has significant bearing when processing data in the context of a real experiment often consisting of dozens to hundreds of subjects worth of data. Fig. 6 shows the Time-on-CPU for a single simulation of each method tested, relative to the reference task with no instrumentation. For Monte Carlo Arithmetic instrumented executions, we expect to see a considerable increase in computation time as additional overhead is added to each floating

point operation. In the case of 1-voxel noise it is expected to see a minor increase in computation time as the perturbed data volumes were generated at runtime, reducing the data redundancy on disk.

The Python MCA and RR modes show a slight increase in computation time to the reference task, whereas the Full Stack version approaches a nearly $7\times$ slowdown, on average. This discrepancy further supports the hypothesis stated above that floating point logic implemented directly in Python, without the use of Numpy or external libraries, account for a minor portion of the total floating point operations. In the case of 1-voxel perturbations, we see a slowdown approximately equivalent to that of the Python instrumentation, not exceeding a $2\times$ increase.

IV. DISCUSSION

We have demonstrated through the application of multiple perturbation methods how noise can be effectively injected into neuroimaging pipelines enabling the exploration and evaluation of the stability of resulting derivatives. These methods operate by either perturbing the datasets or tools used in processing, resulting in a range of structurally distinct noise profiles and distributions which may each provide value when exploring the stability of analyses. While 1-voxel noise is injected directly into the datasets prior to analysis, MCA and RR methods iteratively add significantly smaller amounts of noise to each operation performed.
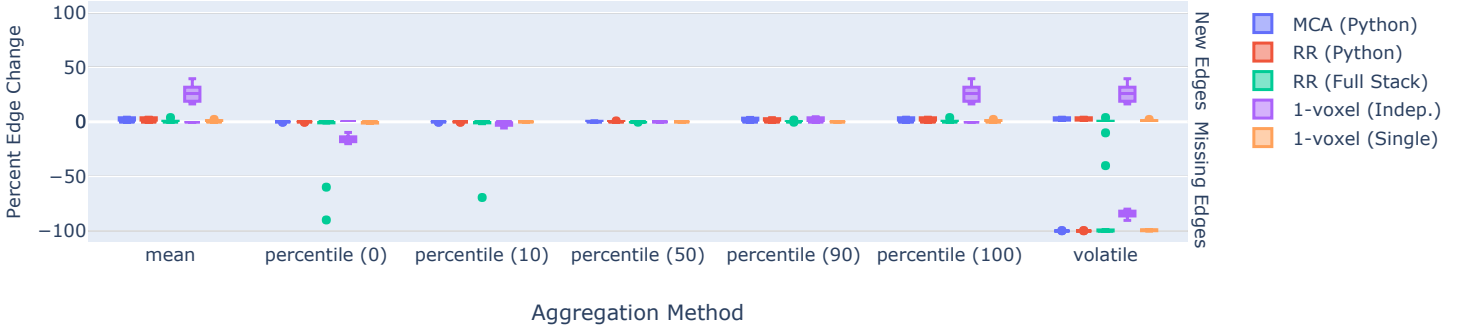
Deviations in Aggregated Edge Count from Reference

Fig. 5. **Gain and loss of edges in aggregation of simulations**. The relative gain and loss of edges is shown for each aggregation method and perturbation method in terms of binary edge count. The volatile aggregation is the difference between percentile (100) and percentile (0) aggregates, and is contains all edges which do not appear in every graph. The volatile set of edges for each of MCA (Python), RR(Python), RR (Full Stack), 1-voxel (independent), and 1-voxel (single) contain 2.5%, 2.5%, 18.5%, 43.0%, and 1.7% of the number of edges found in the reference, respectively. In the worst case, 1-voxel (independent), this means that the existence of nearly half the edges in the graph fail to have consensus across the simulations.

In the case of partial (Python) instrumentation with MCA and RR, distinct and considerably distinct modes emerged in all tested subjects. We believe it is likely that software branching played a role leading to this unexpected result. As the majority of numerical analysis in Python is traditionally performed using the Numpy library, and therefore BLAS and LAPACK, it is possible that the error introduced by Python was allowed to cascade throughout the pipeline without correction, growing to the often subject-level differences observed. These modes would then be the result of a small number of instrumented numerically-sensitive operations, leading to a bounded set of possible outcomes of an otherwise deterministic process. It is possible that these distinct modes could serve as upper-bounds for the deviation due to instabilities within a pipeline, and is an area for further exploration. Future work will also more closely instrument libraries with functionality that will enable the identification of crucial branch points, as this functionality is already present within Verificarlo. The identified crucial branch points could be leveraged for the re-engineering of pipelines with more stable behaviour.

An exciting application of MCA and RR (Python) analyses in cases where pipeline modification is not feasible (i.e. closed source code) is the generation of synthetic datasets. Using each mode or an aggregated collection of modes as samples in the MCA-boosted dataset, this could potentially increase the statistical power of analyses for datasets which may suffer from small samples, or be used to increase the robustness of derivatives by bagging the results using an appropriate averaging technique for the simulated derivatives.

While the Python instrumentation with MCA and RR resulted in derivative modes, the Full Stack instrumentation with RR produced a continuous distribution of derivatives which were often less distinct from the reference results. Extending the hypothesis posited above, this continuous set of results may be due to a law of large numbers effect emerging when

performing a considerable number of small perturbations, leading to a normalized error distribution and effectively a self-correction of deviations. This will be addressed in future work in which tools are instrumented to progressively compare derivatives and their deviation from a reference execution, allowing the emergence and correction of simulation-induced errors to be observed in near real-time.

As the significance of RR (Full Stack) perturbation was highly variable across participants, this technique could also be used for automated quality control, flagging high-variance subjects for further inspection or exclusion from analyses. Inspecting the regional degradation of signal across these perturbations as shown in Fig. 3, researchers could lead a targeted
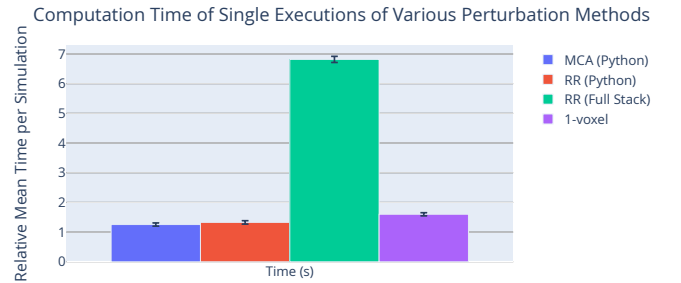

Computation Time of Single Executions of Various Perturbation Methods

Fig. 6. **Computation time for each perturbation method**. Shown in relative time to the reference execution, plotted is the average execution time for the perturbation methods. MCA and RR (Python) have a small increase in computation time per run, as few floating point operations were instrumented in these settings. The RR (Full Stack) method has nearly a 7× slowdown. In this case, all floating point operations were instrumented, but the slowdown of less than the estimated 100× would suggest that the bulk of computation time is not spent on floating point arithmetic. The 1-voxel implementations had a minor slowdown due to the regeneration of data prior to pipeline execution. In every case, the real-world slowdown is $S\times$ larger, where S is the number of simulations, in this case 100.

7

interrogation of their raw datasets to identify underlying causes of signal loss.

The differences observed when performing 1-voxel perturbations were often comparable in magnitude to the variation introduced across Operating Systems. As OS noise is not controlled and may differ greatly among distributions, package updates, etc., it is likely an insufficiently descriptive evaluation method, and should be used as a reference alongside others. The level of control made available through 1-voxel perturbations in terms of both locality and strength of noise makes it a flexible option that could potentially be used to target known areas of key importance for subsequent analyses. The fact that these perturbations represent a near-epsilon change to input images, this method could also be used for estimating global pipeline stability in a classical sense (i.e. conditioning).

While each of the perturbation modes showed distinct differences with respect to the magnitude and continuity of their induced deviations, Fig. 4 illustrates that the structure of these deviations was also highly variable across both perturbation method and data. This suggests different applications and use cases for each perturbation method. While MCA and RR Python implementations impact connectomes globally, these could be applied to generate synthetic datasets. Full Stack RR is highly variable with respect to dataset, suggesting applications in quality control of the derivatives being evaluated. Both 1-voxel methods add noise locally, and can test the sensitivity of specific pipeline components or regions of interest to variation.

In addition to generating unstable derivatives which could be looked at or analyzed independently, this type of perturbation analyses enables the aggregation of derivatives. As is summarized in Fig. 5, the method by which graphs or edges are aggregated can drastically change the construction of resulting graphs. While the mean and max (i.e. 100th percentile) methods both retain all edges that have appeared in even a single graph, the minimum (0th percentile) and other low-percentile aggregations require a stricter consensus of edges for inclusion in the final graph. A benefit of performing multiple aggregations is the composition of graphs with complex edge composition, such as the most volatile edges, as is shown in the final column of Fig. 5. While the binary edge count in the composite graphs varies in each of these methods, it is unclear how derived graph statistics will be affected, and that remains an exciting question for further exploration.

From a resource perspective, each of the perturbation methods evaluated requires multiple iterations to get a sense of the pipeline stability or build aggregates, here taken as 100 iterations. Though the MCA-based methods have the obvious disadvantage of extra computational overhead within each execution cycle of the pipeline, the noise-injection methods do not increase the computation time for a single pipeline execution itself but in this case added computational burden for the generation of synthetic data dynamically, reducing the redundancy of stored images on disk. While Verificarlo has been demonstrated to account for an approximately $100\times$ slowdown in floating point operations [10], the largest slow-

down observed in this pipeline is approximately a factor of 7, as shown in Fig. 6. This suggests that the bulk of time on CPU for this pipeline is not spent on floating point operations, but perhaps other operations such as looping, data access, or manipulation of information belonging to other data types. While this slowdown is observed for the the Full Stack implementation, the Python-only implementation is negligibly slower than the reference execution, suggesting that even fewer of the floating point logic is directly written in Python. The slowdown in the 1-voxel setting is of a similar scale to that of the Python-only implementation, with the slowdown likely caused by the addition of 2 read and 1 write operations to the pipeline's execution (reading of simulation parameters and original image, application of simulation, and subsequent writing of perturbed image to temporary storage). Note that the figures shown in Fig. 6 are for a single simulation, and real relative CPU time in each case would be $100\times$ larger for the experimental application of these methods.

The work presented here demonstrates that even low order computational models such as a 6-component tensor used in diffusion modelling are susceptible to noise. This suggests that stability is a relevant axis upon which tools should be compared, developed, or improved, alongside more commonly considered axes such as accuracy/biological feasibility or performance. The heterogeneity observed across participants clearly illustrates that stability is a property of not just the data or tools independently, but their interaction. Characterization of stability should therefore be evaluated for specific analyses and performed on a representative set of subjects for consideration in subsequent statistical testing. Additionally, identifying how this relationship scales to higher-order models is an exciting next step which will be explored. Finally, the joint application of perturbation methods with more complex post-processing bagging or signal normalization techniques may lead to the development of more numerically stable analyses while maintaining sensitivity that would be lost in traditional approaches such as smoothing.

V. Conclusion

All pipeline perturbation methods showed unique non-zero output noise patterns in low-order diffusion modeling, demonstrating their viability for exploring numerical stability of pipelines in neuroimaging. MCA and RR (Python) instrumented pipelines resulted in a wide range of variability, sometimes equivalent to subject-level differences, and are recommended as possible methods to estimate the lower-bound of stability of analyses, generation of synthetic datasets, and possible identification of Python-introduced critical branch points. RR (Full Stack) perturbations resulted in continuously distributed connectomes that were highly variable across datasets, ranging from negligible deviations to complete regional signal degradation. We recommend the use of RR (Full Stack) noise for automated quality control and identifying global pipeline stability. While 1-voxel methods result in considerably smaller maximum deviations than the MCA-based methods, they are far more flexible and enable evaluating the sensitivity of

pipelines to minor local data perturbations. While the MCA-based methods are more computationally expensive than direct 1-voxel noise injections, the slowdown was found to be less significant in practice than the $100\times$ scaling factor estimated per floating point operation, presumably due to a significant portion of the pipeline computation time being spent on data management or string and integer processing rather than the constant use of floating point arithmetic.

In all cases, while tool instrumentation enables the parallelized simulation of a particular set of instructions, the aggregation of the simulated graphs is an essential component of the downstream analyses both when exploring the nature of instabilities or developing inferences upon the pipeline's derivatives. We recommend a percentile approach to aggregation, where the threshold can be adjusted based on the desired robustness of the resulting graphs. An advantage of percentile approaches is also that composite aggregates can be formed, isolating edges based on their prevalence across simulations. Further exploration of the distribution of perturbed results should be performed to conclude on the relevance of the aggregation used, as the desired aggregate should be close to the expected value of the distribution.

While both MCA and random-injection simulations are computationally expensive in that they require the evaluation of many simulations, they provide an opportunity to characterize processing modes that may emerge when analyzing either noisy datasets or unstable tools. This work also highlighted an important relationship between the noise properties of an incoming dataset and the tool, validating the need to jointly evaluate the stability of tool–dataset combinations.

Where this work demonstrates a range of numerical variation across minor changes in the quality of data or computation, it does not address the analytic impact of these deviations on downstream statistical approaches. This open question, as well as the relative impact of normalization techniques on this process, present avenues for research which will more clearly place these results in a biologically relevant context, allowing characterization of the functional impact of the observed instabilities.

REFERENCES

[1] R. D. Peng, "Reproducible research in computational science," Science, vol. 334, no. 6060, pp. 1226–1227, Dec. 2011.

[2] M. Baker, "1,500 scientists lift the lid on reproducibility," Nature, vol. 533, no. 7604, pp. 452–454, May 2016.

[3] 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," Journal of Magnetic Resonance, vol. 147, no. 2, pp. 340–352, Dec. 2000.

[4] T. Glatard, L. B. Lewis, R. Ferreira da Silva, R. Adalat, N. Beck, C. Lepage, P. Rioux, M.-E. Rousseau, T. Sherif, E. Deelman, N. Khalili-Mahani, and A. C. Evans, "Reproducibility of neuroimaging analyses across operating systems," Frontiers in Neuroinformatics, vol. 9, p. 12, Apr. 2015.

[5] L. B. Lewis, C. Y. Lepage, N. Khalili-Mahani, M. Omidyeganeh, S. Jeon, P. Bermudez, A. Zijdenbos, R. Vincent, R. Adalat, and A. C. Evans, "Robustness and reliability of cortical surface reconstruction in CIVET and FreeSurfer," Annual Meeting of the Organization for Human Brain Mapping, 2017.

[6] A. Bowring, C. Maumet, and T. E. Nichols, "Exploring the impact of analysis software on task fMRI results," biorXiv, Mar. 2018.

[7] A. Klein, J. Andersson, B. A. Ardekani, J. Ashburner, B. Avants, M.-C. Chiang, G. E. Christensen, D. L. Collins, J. Gee, P. Hellier, J. H. Song, M. Jenkinson, C. Lepage, D. Rueckert, P. Thompson, T. Vercauteren, R. P. Woods, J. J. Mann, and R. V. Parsey, "Evaluation of 14 nonlinear deformation algorithms applied to human brain MRI registration," Neuroimage, vol. 46, no. 3, pp. 786–802, Jul. 2009.

[8] D. S. Parker, Monte Carlo Arithmetic: exploiting randomness in floating-point arithmetic. University of California (Los Angeles). Computer Science Department, 1997.

[9] M. Frechtling and P. H. W. Leong, "MCALIB: Measuring sensitivity to rounding error with monte carlo programming," ACM Transactions in Programming Language Systems, vol. 37, no. 2, pp. 5:1–5:25, Apr. 2015.

[10] C. Denis, P. de Oliveira Castro, and E. Petit, "Verificarlo: Checking floating point accuracy through monte carlo arithmetic," 2016.

[11] E. Garyfallidis, M. Brett, B. Amirbekian, A. Rokem, S. van der Walt, M. Descoteaux, I. Nimmo-Smith, and Dipy Contributors, "Dipy, a library for the analysis of diffusion MRI data," Frontiers in Neuroinformatics, vol. 8, p. 8, Feb. 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] G. Kiar, S. T. Brown, T. Glatard, and A. C. Evans, "A serverless tool for platform agnostic computational experiment management," Frontiers in Neuroinformatics, vol. 13, p. 12, Mar. 2019.

[14] K. B. Nooner, S. J. Colcombe, R. H. Tobe, M. Mennes, M. M. Benedict, A. L. Moreno, L. J. Panek, S. Brown, S. T. Zavitz, Q. Li, S. Sikka, D. Gutman, S. Bangaru, R. T. Schlachter, S. M. Kamiel, A. R. Anwar, C. M. Hinz, M. S. Kaplan, A. B. Rachlin, S. Adelsberg, B. Cheung, R. Khanuja, C. Yan, C. C. Craddock, V. Calhoun, W. Courtney, M. King, D. Wood, C. L. Cox, A. M. C. Kelly, A. Di Martino, E. Petkova, P. T. Reiss, N. Duan, D. Thomsen, B. Biswal, B. Coffey, M. J. Hoptman, D. C. Javitt, N. Pomara, J. J. Sidtis, H. S. Koplewicz, F. X. Castellanos, B. L. Leventhal, and M. P. Milham, "The NKI-Rockland sample: A model for accelerating the pace of discovery science in psychiatry," Frontiers in Neuroscience, vol. 6, p. 152, Oct. 2012.

[15] G. Kiar, "BIDS app - FSL diffusion preprocessing," Feb. 2019.

[16] 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.

[17] L. Cammoun, X. Gigandet, D. Meskaldji, J. P. Thiran, O. Sporns, K. Q. Do, P. Maeder, R. Meuli, and P. Hagmann, "Mapping the human connectome at multiple scales with diffusion spectrum MRI," Journal of Neuroscience Methods, vol. 203, no. 2, pp. 386–397, Jan. 2012.