Brainconductor: An open science platform for the development of neuroimaging data analysis tools

Kevin Lin[1*], Haixiao Du[2*], Huazhong Yang[2], Yu Wang[2], Han Liu[3]

[1]*Department of Statistics, Carnegie Mellon University, Pittsburgh, PA 15217 USA*
[2]*Department of Electronic Engineering, Tsinghua University, Beijing, China, 100084*
[3]*Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ 08544 USA*
**These authors contributed equally to this work*

Brainconductor project is a project parallel to Bioconductor. Like Bioconductor, Brainconductor is an open platform for sharing neuroimaging data and R-based data analysis software. Brainconductor aims at promoting interdisciplinary collaboration between neuroscientists and statistical scientists, bringing state-of-the-art statistical methodologies and toolboxes into neuroscience, and enhancing the reproducibility of remarkable research results. We describe details of our motivations, goals, methods, and the difference between Brainconductor and related neuroinformatics projects. Finally, we present some working instances to better explain the advantages of Brainconductor.

Modern statistics aims to infer information from high-dimensional data and has progressed significantly over the last decade. However, two obstacles currently hinder the application of these methods to large-scale neuroimaging data in practice. First, domain-specific expertise and computational power is needed to handle the various data file types and to manage the storage of these large-scale datasets. Second, the pipeline of a typical neuroimaging analysis can span multiple software platforms such as FSL, AFNI, Python, Matlab and R. This decentralized workflow hinders a reproducible analysis and discourages new researchers from learning all the necessary tools across different platforms. The Brainconductor project is an open-neuroscience initiative to address these obstacles. As the neuroscience community expands upon open-access datasets such as the International Neuroimaging Data-Sharing Initiative Project (INDI), comparable advances in open-neuroscience platforms are required for high throughput, computationally efficient statistical models to handle high-dimensional data in this big data era.

Our goal is to build a knowledge hub and interdisciplinary community by connecting neuroscientists and statisticians. Neuroscientists drive the scientific process by collecting new data and constructing focused experiments. Non-invasive imaging techniques have gained popularity to investigate the influence of phenotypic variation on brain connectome[1,2], the coordination among different brain regions when users perform tasks wording and citation, the structural architecture of the brain citation and many more. These include structural and functional MRI, diffusion ten-

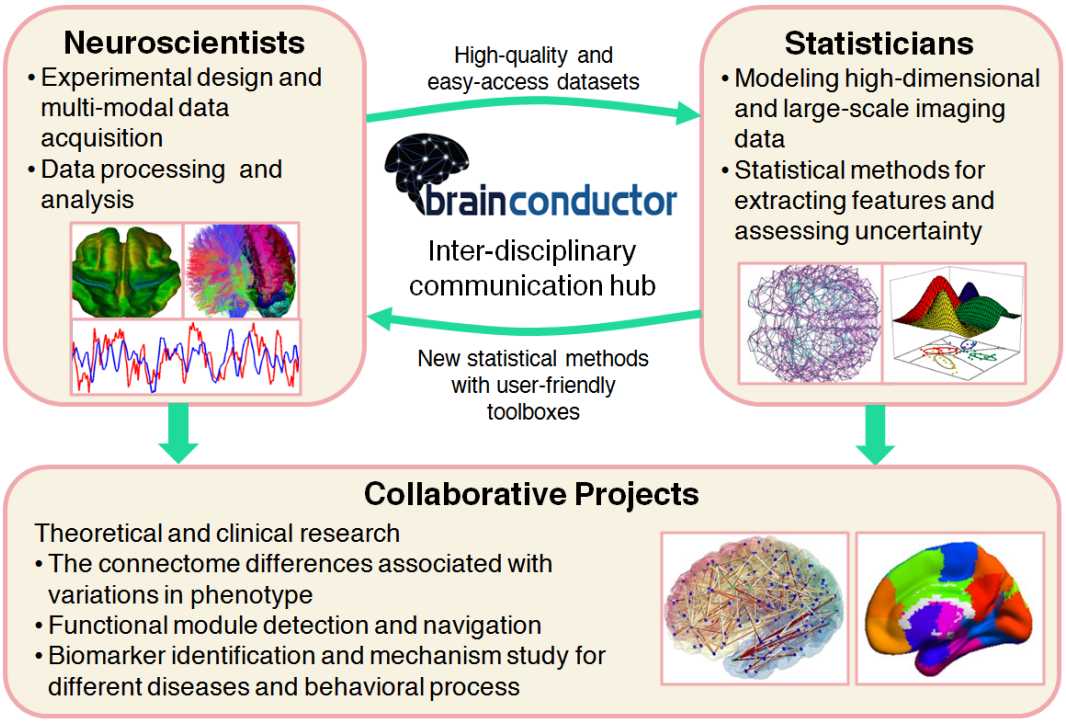

Figure 1: Specialties of neuroscietists and statisticians and the potential topics to explore in collaborative projects.

sor imaging (DTI), and Electroencephalography (EEG). On the other hand, statisticians bring new statistical methods designed to address the high-dimensional, large-scale and complex data. Such techniques can be divided broadly into two categories: feature extraction (i.e., estimating the connectome or structural properties of the brain, summarizing a parcellation of voxels into a single statistic, identifying relevant biomakers associated with changes in the connectome architecture) and uncertainty assessment (i.e., testing for significant difference between the connectomes of two populations, determining if an extracted feature is due to chance). We strive to help enable the ease of collaboration between these two communities. While the neuroscientist will benefit from the modern techniques to analyze these datasets, the statistician will benefit from the lowered barrier-to-entry due to the simplified processes of data acquisition, data management, and data sharing. These are summarized in Figure 1.

Towards this end, we propose the Brainconductor project, an integration of high-quality and easy-access datasets, and user-friendly software for both neuroscientists and statisticians. Brainconductor is a project parallel to Bioconductor. In the past tens of years, Bioconductor[3] made a great contribution to the progress of the human genome research because of its transparency, pursuit of reproducibility, and efficiency of development. Moreover, the open-source developing architecture of Bioconductor based on R language provides facility to take advantage of the mature

statistical packages rather than re-implementing functionality. In future neuroimaging studies, an open-source platform is of necessity to produce collaborative creation of extensible computational tools and enhance the reproducibility of research results.

This desire has already inspired one of the most prominent projects, Neuroimaging Informatics Tools and Resources Clearinghouse (NITRC). NITRC is a resourceful repository collecting popular neuroimaging tools designed for the preprocessing, analysis, and display of neuroimaging data, such as SPM[4], FSL[5], FreeSurfer[6] and AFNI[7]. NITRC also is an open repository for many data sources such as Alzhelmer's Disease Neuroimaging Initiative (ADNI) and the 1000 Functional Connectome Project (FCP). However, as successful as NITRC is, there are fundamental drawbacks from both the developer and user perspectives. From the developer perspective, NITRC provides little guidance on effective software development. This results in two consequences. First, a large amount of effort is spent on implementing functions that have previously been developed by other independent teams. Second, without establishing an universal file format, developers on NITRC have to either write analysis functions for specific file types or spend tremendous efforts to handle all file types. For example, most data sources proffer raw data in the standard industrial DICOM format, but most of the software packages work on NIfTI or ANALYZE formats. From the user perspective, there are also key obstacles. First, software on NITRC are based on different programming models and processing pipelines and have different interfaces. Hence, investigators always need lots of training sessions to learn how to install and use these tools. Second, many statistical analysis software require a 2D matrix where each covariate represents a column and each sample represents a row. Since neuroimaging data is typically represented as a 4D object, new users often struggle to convert their data into a representation suitable for most statistical methods. These concerns are summarized in Figure 2.

The Brainconductor project serves two fundamental goals. In connectome-wide

association studies, one fundamental goal is to relate differences between the macro- and microarchitecture of the human connectome to phenotypic variation among individuals[8]. Towards this end, in our initial release of Brainconductor, we focus on an open-neuroscience solution to manage resting-state fMRI data and study its subsequent brain connectivity patterns. Do we want to explicitly say this? However, there are many scientific goals to a field as broad as neuroscience, and we plan to support all types of neuroimaging analyses within Brainconductor with the help of the community. For example, in imaging genetics, the ultimate goal for Brainconductor project is to identify biomarkers and help early diagnosis for a variety of neuropsychiatric disorders. This will involve an integration of genomics and neuroimaging data and expansion of Brainconductor in the future. Our work paves the foundation for this long-term community project.

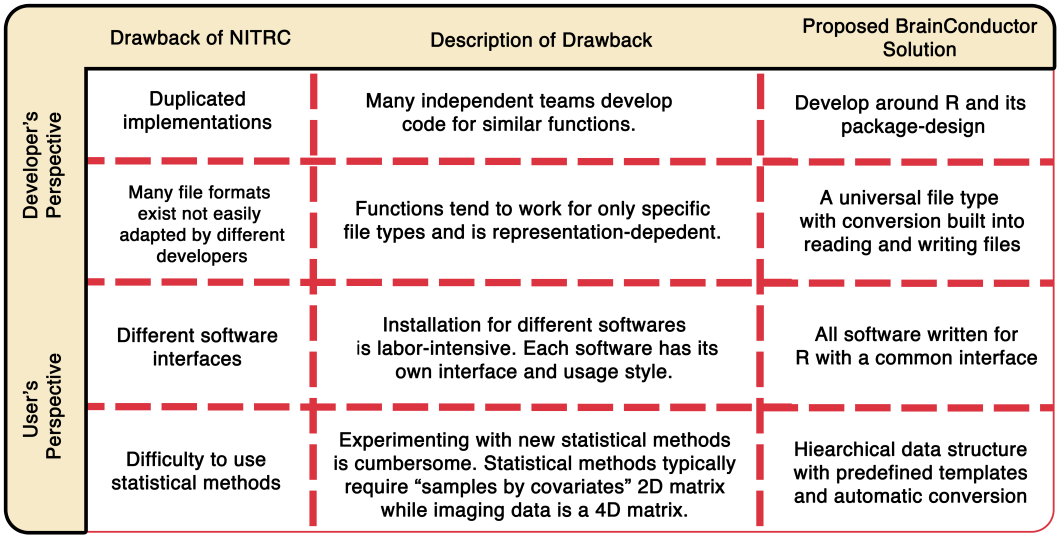

Figure 2: Drawbacks of NITRC framework, description of the drawbacks and proposed solution in Brainconductor. Need to fix alignment

1 Merits of Computational Strategy

Functionality of R. The existing features and communities around a language dictate its usage and accessibility. As stated before, we are interested in problems related to data management, mining and analysis associated with neuroimaging technologies. This orientation requires a programming environment which is able to foster efficiency for software development, allow streamlined and reproducible research and simulate interdisciplinary communication and cooperation.

To meet these requirements, we chose R as the basic programming modal for Brainconductor project. Our rationale closely follows the reasons adopted in Bioconductor which we summarize here[3]. First, R is an accessible high-level language with good numerical capabilities allowing quick prototyping of new statistical computational methods. This allows for a low learning curve for incoming researchers, rapid experimentation of new statistical ideas, and a shorter development cycle. Second, R has a packaging protocol for different software modules that can be developed individually and distributed easily. The packaging system of R enables a worldwide collaborative construction of the Comprehensive R Archive Network (CRAN) with a wide range of high-quality and well-documented statistical and visualization software packages. These hundreds of packages are independently developed for specific objectives, but the objective-oriented programming style of R guarantees the robust interoperability of packages for more complicated applications. All of these characteristics of R would decrease development efforts and release time for reliable software for neuroimaging data analysis. Third, R also provides support for high-performance[9] and parallel computing[10], and interfaces to communicate with specialized code written in lower-level

languages. Finally, the community of R is consisted of thousands of active users and developers including biologists, mathematicians, and engineers. The community provides a natural bridge to connect the neuroscientists and statisticians. This is much in line with the intention of the Brainconductor project and is the most important motivation for our selection of the R language.

Hierarchical Data Structure. The Brainconductor project began with significant investment in the infrastructure construction for software development by formulating the standard for general data structures of neuroimaging data. The data structures are assembled to form our hierarchical data structure called NIdata. Our general data structure provides designers with the convenience of designing code around one interface while enabling users to have the flexibility to store only the data they are interested in their study.

Currently, the most common file format is NIfTI. NIfTI is a product of the Data Format Working Group (DFWG) from the Neuroimaging Informatics Technology Initiative project. It contains both the header information about the data acquisition parameters and neuroimaging data as a 4D matrix. Currently, the oro.nifti R package defines the 'nifti' S4 class in R to manage NIfTI files. However, there are two drawbacks of this file format that could be encountered by analysts. First, the NIfTI file format is not aligned with most statistical analyses. Typically, phenotype information is stored separately in a different file. Hence, users must use a third-party software to determine the subjects in each cohort before performing the population-level case-control study. Likewise, many statistical methods (i.e., regression and classification) require as input a 2D matrix organized by samples and covariates. This results in users often writing their own code to translate the 4D brain data into a compatible 2D format. Second, neuroimaging data files often contain a large amount of redundant information not relevant for a particular analysis. For example, the analysis of fMRI data primarily focuses on the gray matter, while the processing of diffusion weighted MRI focuses on the white matter region. The Human Connectome Project (HCP) faced a similar problem which they overcome by defining the CIFTI file format and grayordinates, a combined cortical surface and subcortical volume coordinate system[11].

We define an S4 'NIdata' class based on the NIfTI format in our Brainbase package as the general data structure to resolve these problems. We designed the data structure hierarchy to enable convenient integration with downstream statistical analysis as well as to optionally store only portions of the data that the user is interested in. At its core, the NIdata class is a generic file type split into six slots to store the following information: 1) the neuroimaging data, 2) phenotype information, 3) specifics of scan sessions, 4) an optional subject ID, 5) a "wildcard" extra slot for any other information that user wants to store such as statistical results, and 6) text-based comments or notes associated with the data that investigators might document. While the S4 class provides an initial structure for storing phenotype and scan information, users have the flexibility to disregard

or modify these templates before reading the data into R.

Most of the engineering of the NIdata class revolved around a more suitable representation of the neuroimaging data to overcome the potential drawbacks of the NIfTI format. Like in the CIFTI file format[11], NIdata can automatically convert neuroimaging data (a 4D matrix) into a 2D (potentially sparse) matrix where each column represents a different voxel in the brain and each row represents a different sample from the series (i.e., time series in fMRI, directional series in DTI). To ensure that the column indices are compatible across different NIdata objects, we define generic templates in the Brainbase packages that dictate which voxel positions are assigned to which columns. These templates are associated with brain masks and the corresponding tissue priors. For instance, one of the templates we provide is the MNI 152 brain masks and tissue priors typically found in FSL. If users want to keep only voxels corresponding to gray matter, all the column indices associated with non-gray matter will be 'zero'-ed out to save storage. Moreover, if users want to perform a region-of-interest or parcellation-based analysis, we provide additional functions to change the template and perform the data extraction. For example, using the Automated Anatomic Labeling (AAL) parcellation[12], users can reduce the voxel-level data into a parcel-level data where the fMRI signal of each parcel is determined the mean or the first principal component, the two most common data-extraction methods.

We discuss the hierarchical nature of the NIdata class. To implement the strategy mentioned above, we chose to create a collection of various S4 data classes, most of which are combined and used interchangeably in the NIdata class. The S4 objects in R use an object-oriented approach to class design which enforce consistency within the superclasses and subclasses. Our NIdata data structure consists of various nested subclasses, giving rise to the hierarchy. The schematic displaying the input of data to the end NIdata representation is shown in Figure 3. We have also designed multiple functions to extract desired components of this hierarchy so users are not required to learn the hierarchy in order to take advantage of the NIdata class.

We lastly mention that users have a wealth of flexibility on how to manipulate the NIdata objects with aid of our other functions. First, to ensure that our 2D representation of neuroimaging data retains the spatial information of each voxel, the Brainbase package provides an R S4 object containing the mapping of voxel location to column index and a list enumerating the column indices of neighboring voxels for each voxel location. Hence, we can convert our 2D representation back to its original 4D representation to accommodate past software requirements if needed. Thanks to this flexibility, software developers can implement algorithms operating directly on the 2D representation in NIdata without worrying about representation-conversion. This also allows users to convert between NIdata and NIfTI classes easily. Second, users can use any templates to process and store NIdata objects. While we have mentioned several templates provided by

the Brainconductor package, users can create customizeable templates. This can be done via the reslicing functions we provide based on the fslr package.

When designing our NIdata file format in R, we kept the tradition of using S4's pass-by-value system rather than using an pointer-based system to pass-by-reference. A pass-by-reference system would be beneficial as R would not need to create local copies of massive neuroimaging data every time it is passed into a function. While there are R packages to help developers code using pointers[?], there would be a considerable burden on future developers to these new coding methods. Instead, we chose to keep the simplistic S4 system and design the hierarchical data structure to reduce the memory requirements for storing each NIdata object.

Some concrete numbers of how much memory is saved

File Conversion. For developers to write functions using our NIdata class, we need to provide suitable functions to read different input file types into objects of NIdata class as described in the previous section. Here, We discuss three popular file types in current use, DICOM, NIfTI and ANALYZE.

DICOM is the standard industrial format for raw data directly collected from an imaging device. The DICOM format is very broad and very sophisticated. In brief, each .dcm suffix file contains a number of attributes, including not only the image pixel data but also large amounts of meta-data information about the subject, imaging devices and settings during data acquisition.

However, DICOM datasets are redundant, ascribed to the storage of massive numbers of small files. This is due to each image slice stored as a separate file. ANALYZE and NIfTI-1 formats are more widely employed in the neuroimaging community. An ANALYZE format document is composed of one "hdr" file and one "img" file. The former contains information about the acquisition settings, while the "img" file contains the image data. NIfTI was released as an extension of the ANALYZE format. The NIfTI data format merges the header and image information of ANALYZE document into one file (.nii) and enables extending of the header information. The NIfTI format has alleviated problems with data storage and sharing across diverse centers, and became one of the most popular neuroimaging format recently.

Brainconductor offers functions to read all these data format based on existing software in the oro.nifti package. We provide a generic function that handles all of these file types and read data into the desired NIdata object. We noticed that the oro.dicom package was difficult to use for new users, so we altered the implementation of how DICOM files are converted into NIfTI files. An example of the conversion using oro.dicom compared to a conversion based on specialized

software such as MRIcron[?] is shown in Figure 4.

Software Distribution. All software distributed by Brainconductor is in the form of R packages abiding to our NIdata structure. This simplifies software delivery, usage and maintenance but puts a burden on the developers to learn how to write R packages, documentation and test cases.

As seen by the success of CRAN and Bioconductor, the packaging system lies at the heart of why R has been a successful language. We follow their examples. Changes are tracked using a central, publicly readable Subversion software repository so the details of all changes are fully accessible via Subversion. Simultaneously, since R itself is continually changing to improve performance and functionality, all packages in Brainconductor undergo testing monthly. These required tests are designed by the developers themselves and is performed in an automated fashion to ensure all code examples and unit tests run without errors.

For developers to submit their software to Brainconductor for others to use, we have designed a specific "Developers" section of our Brainconductor website to guide the process. The package guidelines revolve around usage-oriented documentation and understanding the input and outputs of each parameter. Once a package is submitted, feedback is given to revise the code or the package is accepted. From then on, the package will be available on the development branch of the project and be part of in the next release of Brainconductor. We expect releases to be made every 6 months. Each package has a designated maintainer responsive by email address and reacts to errors, bugs, and user questions. Packages without an active maintainer will be orphaned and no longer be part of the Brainconductor release.

TODO: Change wording, currently copied from Bioconductor paper.

2 User Perspective

Package Ecosystem and Reference Datasets. Brainconductor provides data packages with well-preprocessed neuroimaging data from popular data sources. The user experience of Brainconductor starts with the website. Here, users can find out currently available software, documentation, and reference datasets. These packages include functions to process fMRI data, perform statistical analysis and visualize the results. Once Brainconductor framework has been set up in R, users can then seamlessly install any of the Brainconductor packages through our dedicated installation fuction 'BCoInstall' or through CRAN itself. We have further wrapped popular functions in existing R packages within the CRAN Medical Imaging task view to handle our new NIdata format.

The base installation of Brainconductor also comes with standard imaging resources used by the neuroscience community. These include datasets such as the Montreal Neurological Institute (MNI) brain atlases, the Automated Anatomical Labeling (AAL) parcellation[12], tissue prior From where?, and the various lists of region of interests. While most of these resources are available when installing FSL, we have taken efforts to integrate these datasets into the analysis in R itself. As brain shapes vary across individuals and fMRI machines can scan at different resolutions, we have dedicated effort to ensure streamlined compatibility of these imaging resources with our NIdata format.

Reading Data and Interface. As neuroimaging data spans formats such as DICOM, NIfTI and ANALYZE, we have spent tremendous efforts in our generic read-function 'BCoRead' to hide all the complexities from the user. With an input NIfTI file name or directory of DICOM files, 'BCoRead' outputs the desired NIdata file. One important argument to 'BCoRead' is the input of the subject-scan specific ID number if available. Typically, users might want to update their NIdata files to incorporate the phenotype information. In that case, Brainconductor has a 'BCoLink.phenotype' function which can appropriately link phenotype information in a Comma-Separated Value (CSV) file to NIdata objects with the corresponding ID. BCoRead has an extension to automatically generate the variable names to store the NIdata objects. This can be useful when users have hundreds of NIfTI objects to load into R and also ensures reproducible research.

One of the fundamental obstacles of neuroimaging data analysis is bringing a sample-level analysis into a population-level analysis. As mentioned in the prequel, users can store their fMRI data using NIdata classes where each NIdata object stores the data of subject's scan session. Due to the heterogeneity across each individual's brain, neuroscientists cannot naively aggregate all the NIdata objects prior to performing their analysis.

To facilitate this problem, Brainconductor provides a general function called 'BCoPopulation.analysis' which uses five distinct functions: 1) 'BCoSubjectFinder' to find the relevant NIdata objects in the R Workspace corresponding to fMRI data a user wishes to analyze, 2) an optional 'BCoClassifier' object which separates the NIdata objects into different cohorts based on the loaded phenotype information such as 'case' and 'control', 3) a 'BCoSubject.analysis' subject-level statistical function to be performed on each NIdata object in the analysis, and 4) a 'BCoPopuation.aggregate' population-level statistical function that post-processes each of the results in 'BCoSubject.analysis' to form a population estimate, and 5) an optional 'BCoDifference' function to compare the difference among the different phenotypes based on the results of 'BCoPopulation.aggregate'. Each of these five functions come with defaults, but users can customize these functions to their desire. We believe this general function will alleviate many coding difficulties when researchers experiment with new statistical ideas. A flowchart illustrating this process is

shown in Figure 5.

We note that our "BCoPopulation.analysis" can reverse the steps "BCoPopulation.aggregate" and "BCoDifference" to handle paired case-control studies.

Visualization. Visualization plays a tremendously vital role in neuroimaging analysis. It serves as both an exploratory tool to understand the dataset or ensure correctness of postprocessing. It is also used to interpret high-dimensional statistical results. We provide appropriate functions for both built ontop of existing R packages such as 'brainR' and 'fmri'.

In the spirit of popular visualization softwares such as AFNI, FSLviewer, and MRIcron, we have developed 'BCoView', an interactive plotter based in R that allows users to scroll around and zoom-in, zoom-out of the neuroimage using keyboard commands. With other keyboard commands, users can also show the corresponding series where the current viewing cursor is. This is especially useful when viewing time-series data. We expect this viewer to be a comparable alternative to 'fslview' in the FSL software.

To understand how well a given preprocessed fMRI data matches a template (i.e., MNI brain atlas), we also develop 'BCoView.registration' which takes in two NIdata objects. It plots one neuroimaging data in slices and overlays the segmentation of the second neuroimaging data. Users can then visually see how well the two neuroimaging data aligns with one another. We also develop a 'BCoView.parcellation' which takes a brain atlas and a parcellation assignment and produces either 2D slices or 3D plot of where the parcellations are located in the brain. Lastly, to visualize the connectome, we develop 'BCoView.connectome' which visualizes the 3D brain and the corresponding ROI-analysis or parcellation-analysis.

Reproducible Research. It can be surprisingly difficult to retrace the computational steps performed in neuroimaging analysis. One of the goals of Brainconductor is to help scientists report their analysis in a way that allows exact recreation by a third party when given the input data. This should include all figures, tables and numeric results. Brainconductor achieves this in two ways. First, as described in the methods above, Brainconductor focuses on automating standard functions to manipulate, convert, and extract data such that users can focus on only the core elements of the statistical analysis. Since all the required functions are run within R, the potential of human-caused errors is dramatically reduced compared to other cross-platform analyses.

Secondly, we support and advocate using Jupyter Notebooks for this task. While initially supporting Python, Jupyter Notebooks now support the usage of R. The advantage of using Jupyter Notebooks over other competitors such as Sweave and knitr lies in the live coding environment and

integration of code, figures and text into only one file, the iPython file.

On the Brainconductor website, we advocate the usage of Jupyter Notebooks in two ways. First, the majority of the base packages and reference datasets in Brainconductor are accompanied with a Jupyter "datasheet." This is a Jupyter file that cleanly lists attributes of the dataset, plots the dataset and prints simple statistics. Using Jupyter Notebooks makes this process easy to maintain while enabling users to get a clear picture of the data without having to open R. Second, users can upload their analysis pipeline as a Jupyter Notebook onto the Brainconductor website for others to download and view. This directly support reproducible research.

Integration with Statistical Analysis Software. Brainconductor provides analytical tools dedicated to neuroimaging from FMRIB Software Library (FSL) and the abundant of statistical functions within various packages on CRAN. First, a recent R package called FSLR citation is an R-wrapper to invoke FSL commands from the R console. We link this package to Brainconductor by providing an additional wrapper to save and load NIdata objects into the working directory for FSL to access. Second, thanks to the 2D representation and the explicit storage of phenotypes of NIdata, most classification, graphical model, or regression techniques found in CRAN packages can be directly used on NIdata. The main function to facilitate in this process is 'BCoReduction', a function which brings a voxel-level analysis into a ROI-level or parcel-level analysis. This is a customizable function that can be used in conjugation when reading in the data which aggregates the series in different voxels or eliminates voxels that are not relevant to the analysis.

Undirected graphs provide a powerful tool for understanding the interrelationships among different covariates. For example, in fMRI analysis, each covariate represents a different voxel/-parcel/RoI while an edge connects to covariates if there is statistical dependency between the two covariates. High-dimensional graphical models is a large area of statistical literature, and we have converted a multitude of graphical model estimators to accommodate our NIdata format.

Might want to expand on the graphical model stuff? Maybe to Lindquist's paper for other analysis with generalized linear regresssion?

A brief Conclusion at last.

3 Case Study

Autism spectrum disorders (ASD) are characterized by qualitative impairment in reciprocal social communication, as well as by repetitive, restricted, and stereotyped behaviors. Previously con-

sidered rare, ASD are now recognized to occur in more than 1% of children, causing immense suffering to individuals and their families. We provide a short demonstration of the functionalities of Brainconductor to estimate the differences in the connectome between autistic subject and controls based on the 57 resting-state fMRI (R-fMRI) scans (30 autistic subjects, 27 controls) from the Autism Brain Imaging Data Exchange (ABIDE) collected at the University of Pittsburgh, School of Medicine[13]. The subjects are 7 to 35 years of age, where the autistic subjects were diagnosed with the Autism Diagnostic Interview-Revised and the Autism Diagnostic Observation Schedule-General. The controls are selected to be healthy individuals to match the autistic subjects in age (within 3.5 years), IQ (within 12 points on the full-scale) and gender.

The 57 scans were preprocessed with the Configurable Pipeline for the Analysis of Connectomes (C-PAC) alpha version 0.3.9. This preprocessing happened outside of R. The image preprocessing steps included slice-timing and motion correction based on the Friston Model, nuisance signal regression (including 5 CompCorr signals, the cerebrospinal fluid (CSF), motion and the global, linear, and quadratic signals) and temporal filtering (0.001-0.08Hz). The derived R-fMRI measures were normalized to Montreal Neurological Institute (MNI152) stereostatic space ($2mm^3$ isotropic) with linear regressions and spatially smoothed (applied FWHM = 6mm).

mention the memory size savings

After the data has been preprocessed, we use various tools developed in BCoViewer to visually explore the data. In Figure 6, we see two different instances of our BCoViewer function to either interactively explore a subject's imaging data (both functional and anatomical simultaneously). In Figure 7, we show our plotting capabilities to visualize parcellations. On the left, we visualize the AAL parcellation in 3D shapes, and on the right, BCoViewer plots the AAL parcellation in 2-dimensional cross-sections. But another question investigators typically have is how to determine the success of their preprocessing pipeline. We developed another feature in BCoViewer to address this, shown in Figure 8 to mimic currently-used methods in C-PAC.

In order to find the presence of abnormal functional connectivity, we carried out the whole-brain analysis using the Automated Anatomical Labeling[12] parcellation. This divides the brain into 116 anatomical connected regions. We extracted the mean time series from each of the 116 regions for each subject. We then performed a subject-level connectome analysis using neighborhood selection[14]. For each subject, we chose the tuning parameter of this method to estimate a connectome with 150 edges (roughly sparsity of 0.025%). Based on case-control pairings matching the age, gender and IQ as closely as possible, we then took the graph-difference between each pair of case and control. Here, the graph difference is represented as signed edges in one connectome but not the other. We then computed the median graph[?] with 50 edges to generate a

population-level graph difference between the cases and controls. The entire analysis was done using our BCoPopulation.analysis framework.

Our analysis determined that several connections within the cerebellum were present in controls while severed in cases. In particular, over half the case-control pairings showed that in the control's estimated brain connectome, there was statistical dependency between Right lobule IV, V of the cerebellar hemisphere (Cerebelum_4_5_R) and Lobule VII of vermis (Vermis_7) while in the case's estimated brain connectome, this connection was missing. A plot of these two regions is shown in Figure 9. While the cerebellum is responsible for motor learning and coordination, there is new evidence that the cerebellum supports cognitive functions, including language and executive functions, as well as affective regulation[?,?]. There is supporting evidence from current autism research that the cerebellum plays a substantial role in the development of autism[?,?,?].

Methods

A Using Brainconductor.

The current release of Brainconductor is a test version 1.0; we require R version to be above 3.1.1. Users of older R versions must update their installation to start with Brainconductor. Download the latest release of R, then download and install basic packages of Brainconductor by starting R and entering the commands

```
> source("http://10.8.7.219/packages/BrainCo/BCoinstall.R")
> BCoInstall()
```

The BCoinstall.R script installs BrainCoSetup package. BCoInstall is a function of BrainCoSetup package to install core packages if called by default arguments. To install specific packages, e.g., "fmri" and "AnalyzeFMRI", call the BCoInstall function with

```
> BCoInstall(c("fmri","AnalyzeFMRI"))
```

BCoInstall acquiescently installs the core packages in the MedicalImaging task view on CRAN. Users can suppress the default installation with

```
> BCoInstall(installmedicalimgTV = FALSE)
```

For details of installing a CRAN task view, please see the help document of R package "ctv".

BCoInstall also updates outdated R packages with a prompt. Users can suppress the prompt easily using the argument ask = FALSE.

In some cases, underlying alterations in the operating system, especially in Linux system, require recompiling all installed packages. Users can start a new R session and enter

```
> source("http://Domain/BCoinstall.R")
> pkgs <- rownames(installed.packages())
> BCoInstall(pkgs, type="source")
```

Users can check packages that are either outdated or too new for their Brainconductor version with

```
> library(BrainCoSetup)
> BCoValid()
```

The output provides possible solutions to identified problems, and the help page ?BCoValid shows detailed arguments and behaviours of the function.

1. Sporns, O., Tononi, G. & Kötter, R. The human connectome: a structural description of the human brain. *PLoS Comput Biol* **1**, e42 (2005).

2. Sporns, O. The human connectome: a complex network. *Annals of the New York Academy of Sciences* **1224**, 109–125 (2011).

3. Gentleman, R. C. *et al.* Bioconductor: open software development for computational biology and bioinformatics. *Genome biology* **5**, R80 (2004).

4. Penny, W. D., Friston, K. J., Ashburner, J. T., Kiebel, S. J. & Nichols, T. E. *Statistical parametric mapping: the analysis of functional brain images: the analysis of functional brain images* (Academic press, 2011).

5. Jenkinson, M., Beckmann, C. F., Behrens, T. E., Woolrich, M. W. & Smith, S. M. Fsl. *Neuroimage* **62**, 782–790 (2012).

6. Fischl, B. Freesurfer. *Neuroimage* **62**, 774–781 (2012).

7. Cox, R. W. Afni: software for analysis and visualization of functional magnetic resonance neuroimages. *Computers and Biomedical research* **29**, 162–173 (1996).

8. Milham, M. P. Open neuroscience solutions for the connectome-wide association era. *Neuron* **73**, 214–218 (2012).

9. Buckner, J. *et al.* The gputools package enables gpu computing in r. *Bioinformatics* **26**, 134–135 (2010).

10. Schmidberger, M. *et al.* State of the art in parallel computing with r. *Journal of Statistical Software* **31**, 1–27 (2009).

11. Glasser, M. F. *et al.* The minimal preprocessing pipelines for the human connectome project. *Neuroimage* **80**, 105–124 (2013).

12. Tzourio-Mazoyer, N. *et al.* Automated anatomical labeling of activations in spm using a macroscopic anatomical parcellation of the mni mri single-subject brain. *Neuroimage* **15**, 273–289 (2002).

13. Di Martino, A. *et al.* The autism brain imaging data exchange: towards a large-scale evaluation of the intrinsic brain architecture in autism. *Molecular psychiatry* **19**, 659–667 (2014).

14. Meinshausen, N. & Bühlmann, P. High-dimensional graphs and variable selection with the Lasso. *The Annals of Statistics* 1436–1462 (2006).

Competing Interests The authors declare that they have no competing financial interests.

Correspondence Correspondence and requests for materials should be addressed to Han Liu, Ph.D., Sherred Hall 224, Princeton University, Princeton, NJ 08544 USA (email: hanliu@princeton.edu), and Yu Wang, Ph.D., Room 4-303, Rohm Building, E.E. Dept., Tsinghua University, Beijing, China 100084 (email: yu-wang@tsinghua.edu.cn)

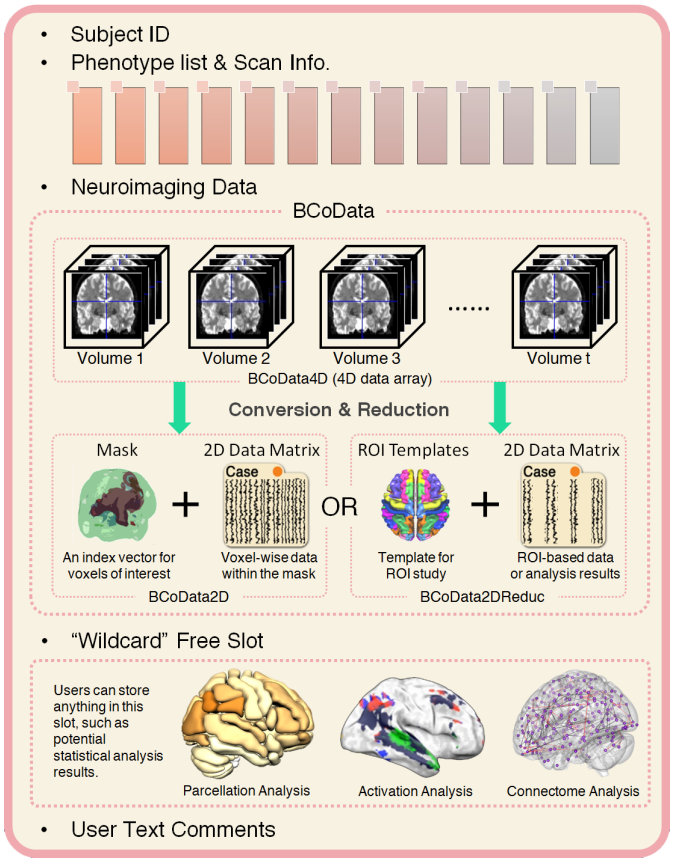

Figure 3: Schematic of the hierarchical data format. The top-most bullet represents the slot for the Subject ID. The next bullet represents the slots for the user-defined phenotype information and scan info (i.e. from the NIfTI header). Next, the neuroimaging slot represents a hierarchical data structure where data can be stored as 4-dimensional matrix (BCoData4D), as a 2-dimensional matrix where each column represents a voxel in the mask (BCoData2D) or as a 2-dimensional matrix where each column represents a summary of each region of interest (BCoData2DReduc). The next slot represents a wildcard where users can store anything of relevance such as analytical results. The last slot is for users to write any notes for future reference.

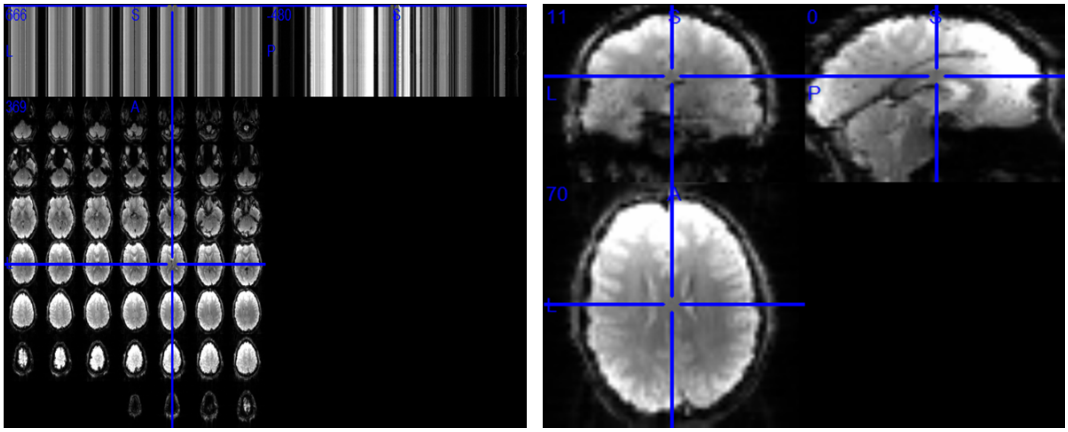

Figure 4: Comparison between conversion functions from oro.dicom package (left) and from our Brainbase package (right). Our function showed better compatibility with widely-used visualization tools such as MRIcron.

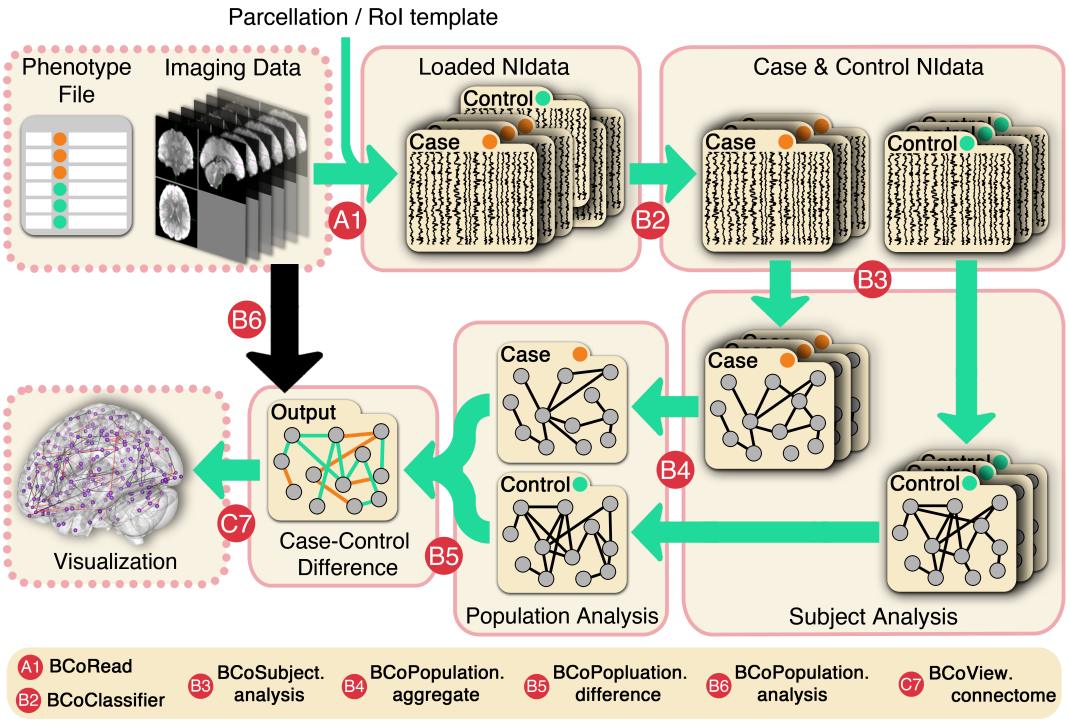

Figure 5: typo in population The workflow using Brainconductor's interface. Our illustration is about estimating the connectome from fMRI data, but functions within can be adapted to estimate any desired quantity. BCoRead (A1) reads the raw neuroimaging data and phenotype file into our NIdata objects. This converts the voxel-level data into the parcellation- or RoI-level data. Steps B2 to B5 are the individual customizable functions that perform the case-control connectome analysis. After B5, an example output of the analysis could be the estimated connectivity in controls not in cases and vice-versa. BCoPopulation.analysis (B6) is the all-in-one function that performs the entire analysis once the functions B2 to B5 are define. The last step is visualization (C6).

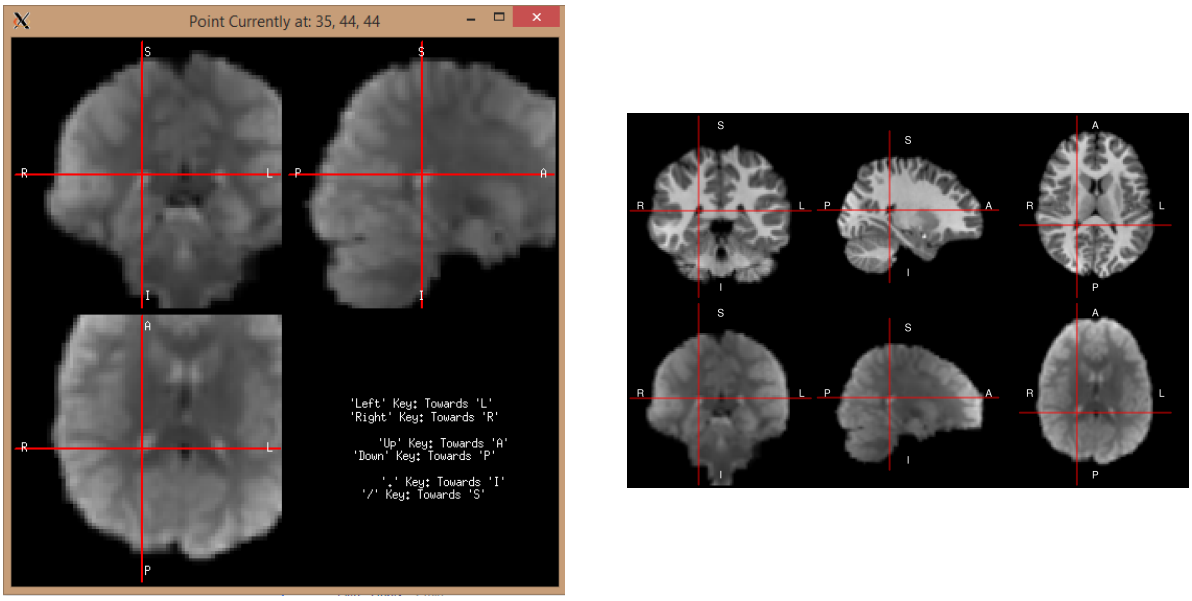

Figure 6: (Left) BCoView, an interactive viewer in R designed to be similar to Fslview. (Right) An alternative setting in BCoView to determine alignment between anatomical and functional images.

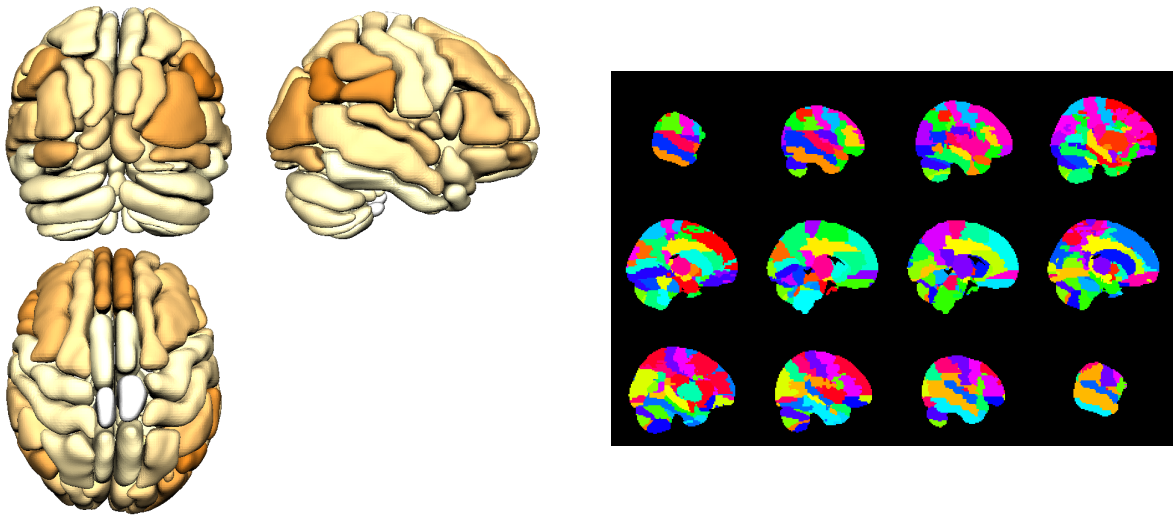

Figure 7: (Left) A three-dimensional visualization of the different parcels in AAL. (Right) A parcellation plotter in BCoViewer. Each color represents a different parcel in AAL.

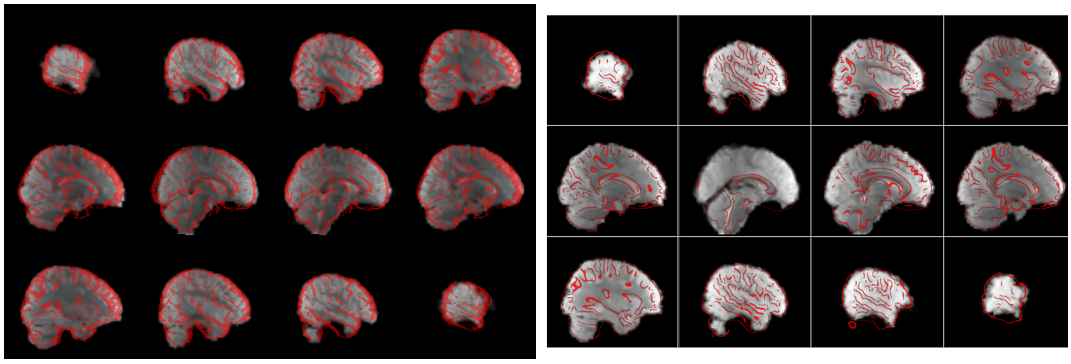

Figure 8: A comparison between our plotter in BCoViewer to visually determine how well a fMRI scan has been registered to the standard MNI template. Our BCoViewer method in R is shown on the left, while the method used in C-PAC is shown on the right. In both plots, the subject's functional data is shown in grayscale while the red points/lines trace the outline of the MNI 2mm template.

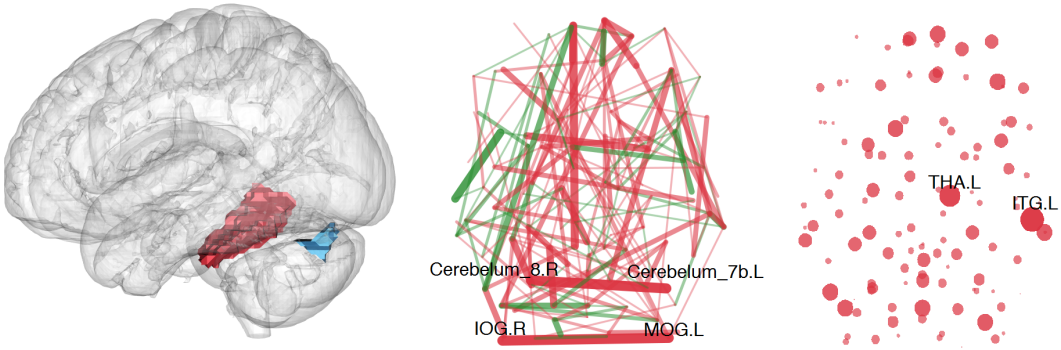

Figure 9: (Left): Perspective plot showing right lobule IV, V of cerebellar hemisphere (red) and Lobule VII of vermis (blue). include pictures of entire graph (Middle): A plot of the population connectome differences plotted from the axial view. Red edges are ones in controls but not in autistic subjects, while green is the opposite. The thickness of the line denotes how many subject-pairs displayed this connectome difference. (Right): A plot of all 116 parcels represented by a circle where the size of the circle denotes the degree of that parcel in the population connectome difference.