Packaging Biocomputing Software to Maximize Distribution and Reuse

William S. Bush, Nicholas Wheeler
Cleveland Institute for Computational Biology, Department of Population and Quantitative Health Sciences, Case Western Reserve University
Cleveland, OH, 44106, USA
Email: wsb36@case.edu; nrw16@case.edu

Brett Beaulieu-Jones
Department of Biomedical Informatics, Harvard Medical School
Boston, MA, 02115, USA
Email: Brett_Beaulieu-Jones@hms.harvard.edu

Christian Darabos
Research, Teaching and Learning at IT&C, Dartmouth College
Hanover, NH, 03755, USA
Email: Christian.Darabos@dartmouth.edu

The majority of accepted papers in computational biology and biocomputing describe new software approaches to relevant biological problems. While journals and conferences often require the availability of software and source code, there are limited resources available to maximize the distribution and use of developed software within the scientific community. The accepted standard is to make source code available for new approaches in published work, the growing problem of system configuration issues, language, library version conflicts, and other implementation issues often impede the broad distribution, availability of software tools, and reproducibility of research. There are a variety of solutions to these implementation issues, but the learning curve for applying these solutions is steep. This tutorial demonstrates tools and approaches for packaging and distribution of published code, and provides methodological practices for the broad and open sharing of new biocomputing software.

1. Rationale for Tutorial

A cornerstone of biocomputing and computational biology is the release of new algorithms for data analysis, often in the form of an author-developed software implementation. With the ever increasing need for algorithmic processing of experimental data in scientific studies, the reproducibility of individual studies has declined (Baker and Penny 2016; Monya and Dan 2016). The lack of reproducibility and open sharing of methods has had downstream impacts into more expensive clinical research, leading to an estimated $200 billion of wasted research funds (Chalmers and Glasziou 2009). The subsequent resulting calls to optimize the research and discovery pipeline to minimize reporting of false discoveries and to reduce research waste (Macleod et al. 2014) have led to proposals for "best practices" in computational research. In their *Ten simple rules for reproducible computational research,* Sandve and colleagues enumerate the need for archiving exact versions of external programs, version controlling all custom scripts, storing intermediate data and raw output, and providing public access to scripts, runs and results (Sandve et al. 2013).

The traditionally accepted approach for standardization and version control of software is the use of package repositories. The Comprehensive R Archive Network (CRAN) is an extensively mirrored repository of distributions, extensions, and documentation for the R statistical package (Hornik 2018). Similarly, Bioconductor serves as an extension of the R environment for computational biology and bioinformatics packages (Gentleman et al. 2004). Analogs of these repositories have also been developed for the Python language (Dale et al. 2018), and custom software and version control is now routinely stored and managed using Git and GitHub (Chacon and Straub 2014).

While package management systems have dramatically improved version control and accessibility of software, duplicating the precise software environment used to process experimental data in a publication has long remained a major challenge. Within the last five years, the dramatic rise of containerization technologies like Docker (Merkel 2014) have for the first time allowed seamless distribution of data, software, and its native processing environment together as a single entity. Containerization technology has been adapted for bioinformatics tasks (Belmann et al. 2015), deployed into custom bioinformatics registries (Moreews et al. 2015), and specifically adapted to high-performance computing environments (Kurtzer, Sochat, and Bauer 2017). Containers have been especially useful in the distribution of complex workflows with dependencies on multiple software tools, such as the processing of next-generation sequencing data (Kim et al. 2017; Schulz et al. 2016). The BioContainers Community has produced a list of recommendations for standardizing bioinformatics packages and containers (Gruening et al. 2019).

Even with software version control and entire software environments available for download, specific analysis steps of a given publication may not be well-documented. While package management systems have dramatically improved version control and accessibility of software, and containerization allows duplication of the precise software environment, the exact process for analyzing experimental data may still prove difficult to reproduce without detailed documentation. To address these challenges, Jupyter notebooks have emerged as a composite digital document that seamlessly blends code (from a variety of languages), documentation, and data visualization in an easy-to-follow format (Kluyver et al. 2016). Jupyter notebooks have gained popularity in other computation-heavy fields like astronomy (Wofford et al. 2019), however their stability and accessibility is not always persistent after publication. While there are also repositories for storing Jupyter notebooks, specific practices are needed to ensure long-term availability of accessed documents (Bouquin et al. 2018).

In this tutorial, we outline a technology stack that ensures high availability and easy distribution of software, encapsulated data, software environment, and analysis approaches. Docker containers are proposed as a foundational layer, providing a stable, version-controlled operating system along with its associated programming languages and packages, and data files that can be cached within the environment. R and Python packages are the distribution method for custom software implementations, and are accessible within distributable containers. And Jupyter notebooks provide detailed documentation of all analysis steps in an interactive fashion.

2. Tutorial Speakers

William S. Bush, Ph.D. is an Associate Professor in the Department of Epidemiology and Biostatistics and Assistant Director for Computational Methods in the Cleveland Institute for Computational Biology at Case Western Reserve University. Dr. Bush received his Ph.D. at Vanderbilt University in Human Genetics in 2008 and then continued as a post-doctoral fellow in the Neurogenomics Training Program at Vanderbilt. Dr. Bush was recently named a Mt. Sinai Health Care Foundation Scholar. As a human geneticist and bioinformatician, Dr. Bush's research interests include understanding the functional impact of genetic variation, developing statistical and bioinformatics approaches for integrating functional genomics knowledge into genetic analysis, and the use of electronic medical records for translational research.

Nicholas Wheeler, Ph.D. is a Research Associate in the Cleveland Institute for Computational Biology at Case Western Reserve University. Dr. Wheeler is a macromolecular scientist and engineer by training with extensive expertise in the use of "big data" technologies for large scale data aggregation and analysis. Dr. Wheeler manages genomic datasets and their associated meta-data within a Spark/Hadoop cluster, with extensions to the open-source HAIL platform for genomic analysis, which ensures standardization and reproducibility of experimental analyses. Over the course of his career, Dr. Wheeler has created, validated, and submitted multiple R and Python packages into public repositories.

Brett Beaulieu-Jones, Ph.D. is a Post-doctoral Research Fellow in Biomedical Informatics in the Kohane lab at Harvard University. He received his PhD from the Perelman School of Medicine at the University of Pennsylvania under the supervision of Dr. Jason Moore and Dr. Casey Greene. Dr. Beaulieu-Jones' doctoral research focused on using machine learning-based methods to more precisely define phenotypes from large-scale biomedical data repositories, e.g. those contained in clinical records. He is currently performing large-scale data integration (genomic, therapeutic, imaging) to both better understand disease etiology as well as provide precise therapeutic recommendations. Initially, he is working to develop targeted models of drug selection for patients with refractory epilepsy and to further develop machine learning methods that model the way patients progress over time using longitudinal data.

Christian Darabos, Ph.D. is the Assistant Director for Research Informatics at Dartmouth College. He graduated with a double Ph.D. degree in Computer Science and Computational. During his Post-doctoral work with Dr. Jason Moore, he developed on computational genetics analytics pipelines of large datasets using network-based approaches. At Dartmouth, Dr. Darabos conducts a series of workshops and tutorials on computational tool and Reproducible Research.

3. Acknowledgements

This work is partially supported by the National Institute on Aging of the National Institutes of Health (NIH) Grants U01 AG058654 and U54 AG052427.

4. References

Baker, Monya, and Dan Penny. 2016. “Is There a Reproducibility Crisis?” *Nature* 533(7604): 452–54.

Belmann, Peter et al. 2015. “Bioboxes: Standardised Containers for Interchangeable Bioinformatics Software.” *GigaScience* 4(1).

Bouquin, Daina, Sophie Hou, Matthew Benzing, and Lee Wilson. 2018. *Jupyter Notebooks: A Primer for Data Curators Link w/ Release Notes*. http://datacurationnetwork.org.

Chacon, Scott, and Ben Straub. 2014. *Pro Git*. 2nd ed. Berkely, CA, USA: Apress.

Chalmers, Iain, and Paul Glasziou. 2009. “Avoidable Waste in the Production and Reporting of Research Evidence.” *Lancet (London, England)* 374(9683): 86–89. http://www.ncbi.nlm.nih.gov/pubmed/19525005 (October 7, 2019).

Dale, Ryan et al. 2018. “Bioconda: Sustainable and Comprehensive Software Distribution for the Life Sciences.” *Nature Methods* 15(7): 475–76.

Gentleman, Robert C et al. 2004. “Bioconductor: Open Software Development for Computational Biology and Bioinformatics.” *Genome biology* 5(10): R80. http://www.ncbi.nlm.nih.gov/pubmed/15461798 (October 7, 2019).

Gruening, Bjorn et al. 2019. “Recommendations for the Packaging and Containerizing of Bioinformatics Software.” *F1000Research* 7: 742.

Hornik, Kurt. 2018. “R FAQ.” https://cran.r-project.org/doc/FAQ/R-FAQ.html.

Kim, Baekdoo et al. 2017. “Bio-Docklets: Virtualization Containers for Single-Step Execution of NGS Pipelines.” *GigaScience* 6(8).

Kluyver, Thomas et al. 2016. “Jupyter Notebooks-a Publishing Format for Reproducible Computational Workflows.” https://nbviewer.jupyter.org/,.

Kurtzer, Gregory M., Vanessa Sochat, and Michael W. Bauer. 2017. “Singularity: Scientific Containers for Mobility of Compute.” *PLoS ONE* 12(5).

Macleod, Malcolm R et al. 2014. “Biomedical Research: Increasing Value, Reducing Waste.” *Lancet (London, England)* 383(9912): 101–4. http://www.ncbi.nlm.nih.gov/pubmed/24411643 (October 7, 2019).

Merkel, Dirk. 2014. “Docker: Lightweight Linux Containers for Consistent Development and Deployment.” *Linux J.* 2014(239). http://dl.acm.org/citation.cfm?id=2600239.2600241.

Monya, Baker, and Penny Dan. 2016. “Reproducibility Crisis (Nature).” *Nature* 533: 452–54.

Moreews, François et al. 2015. “BioShaDock: A Community Driven Bioinformatics Shared Docker-Based Tools Registry.” *F1000Research* 4.

Sandve, Geir Kjetil, Anton Nekrutenko, James Taylor, and Eivind Hovig. 2013. “Ten Simple Rules for Reproducible Computational Research.” *PLoS computational biology* 9(10): e1003285. http://www.ncbi.nlm.nih.gov/pubmed/24204232 (October 7, 2019).

Schulz, Wade L., Thomas J.S. Durant, Alexa J. Siddon, and Richard Torres. 2016. “Use of Application Containers and Workflows for Genomic Data Analysis.” *Journal of Pathology Informatics* 7(1).

Wofford, Morgan et al. 2019. “Jupyter Notebooks as Discovery Mechanisms for Open Science: Citation Practices in the Astronomy Community.” *Computing in Science & Engineering*: 1–1.

Translational Bioinformatics: Biobanks in the Precision Medicine Era

Marylyn D Ritchie
Department of Genetics and Institute for Biomedical Informatics, The Perelman School of Medicine,
University of Pennsylvania, A301 Richards Building, 3700 Hamilton Walk
Philadelphia, PA 19104, USA
Email: marylyn@pennmedicine.upenn.edu

Jason H Moore
Department of Biostatistics, Epidemiology, & Informatics, Department of Genetics, and Institute for Biomedical Informatics, The Perelman School of Medicine, University of Pennsylvania, D202 Richards Building, 3700 Hamilton Walk
Philadelphia, PA 19104, USA
Email: jhmoore@upenn.edu

Ju Han Kim
Department of Biomedical Sciences, Seoul National University Graduate School, Biomedical Science Building 117,
103 Daehakro, Jongro-gu, Seoul 110-799, Korea
Email: juhan@snu.ac.kr

Translational bioinformatics (TBI) is focused on the integration of biomedical data science and informatics. This combination is extremely powerful for scientific discovery as well as translation into clinical practice. Several topics where TBI research is at the leading edge are 1) the use of large-scale biobanks linked to electronic health records, 2) pharmacogenomics, and 3) artificial intelligence and machine learning. This perspective discusses these three topics and points to the important elements for driving precision medicine into the future.

Keywords: translational bioinformatics, precision medicine, pharmacogenomics, artificial intelligence, machine learning, electronic health records, biobank

1. Introduction

Translational bioinformatics (TBI) is a multi-disciplinary and rapidly emerging field of biomedical data sciences and informatics that includes the development of technologies that efficiently translate basic molecular, genetic, cellular, and clinical data into clinical products or health implications. TBI is a relatively young discipline that spans a wide spectrum from big data to comprehensive analytics to diagnostics and therapeutics. TBI involves applying novel methods to the storage, analysis, and interpretation of a massive volume of genetics, genomics, multi-omics, and clinical data; this includes diagnoses, medications, laboratory measurements, imaging, and