METHODS TO ENSURE THE REPRODUCIBILITY OF BIOMEDICAL RESEARCH

KONRAD J. KARCZEWSKI
Massachusetts General Hospital, Boston, MA; Broad Institute, Cambridge, MA
Email: konradjkarczewski@gmail.com

NICHOLAS P. TATONETTI
Columbia University, New York, NY
Email: nick.tatonetti@columbia.edu

ARJUN K. MANRAI
Harvard Medical School, Boston, MA
Email: manrai@post.harvard.edu

CHIRAG J. PATEL
Harvard Medical School, Boston, MA
Email: chirag_patel@hms.harvard.edu

C. TITUS BROWN
University of California , Davis, CA
Email: ctbrown@ucdavis.edu

JOHN P. A. IOANNIDIS
Stanford University, Stanford, CA
Email: jioannid@stanford.edu

Science is not done in a vacuum – across fields of biomedicine, scientists have built on previous research and used data published in previous papers. A mainstay of scientific inquiry is the publication of one's research and recognition for this work is given in the form of citations and notoriety -- ideally given in proportion to the quality of the work. Academic incentives, however, may encourage individual researchers to prioritize career ambitions over scientific truth. Recently, the *New England Journal of Medicine* published a commentary calling scientists who repurpose data "research parasites" who *mis*use data generated by others to demonstrate alternative hypotheses[1]. In our opinion, the concept of data hoarding not only runs contrary to the spirit of, but also hinders scientific progress. Scientific research is meant to seek objective truth, rather than promote a personal agenda, and the only way to do so is through maximum transparency and reproducibility, no matter who is using the data.

To maintain the integrity of the scientific process, it is necessary to cultivate practices that ensure reproducibility, especially as large and public heterogeneous databases proliferate. Many of these paradigms can be likened to open-source practices already adopted by much of the computer

science community. These include, but are not limited to, version control, code review, and containerization. There are many benefits to improving reproducibility: aside from the general benefit to science through increased transparency, releasing code enables additional peer review and is educational and efficient as it reduces duplications of efforts. Of course, these approaches require additional time for investigators to document and clean up code and data for release, which is the top reason for not sharing data and code[2] (in addition to managing the intricacies of tools for version control, for example). Various incentive structures have been proposed to improve reproducibility rates across scientific fields, including creation of requirements by funding agencies or establishment of reward systems[3]. Additionally, like many computational skills, these require some initial effort, but have long-term benefits and will eventually become ingrained. Finally, public release of code can enable public code review, which improves programming habits: efforts such as Software Carpentry have been established to teach these skills and have met with recent success[4].

Reproducibility can take a number of forms and the desired extent of reproducibility has been debated in other fora: whatever the ideal solution, there is room for improvement in ensuring that research is reproducible. A growing number of researchers have begun to share their code and processed data, where possible. For instance, the ENCODE project released a virtual machine image that contained the code and data to reproduce the figures in their manuscript[5] [http://encodeproject.org/ENCODE/integrativeAnalysis/VM]. Similarly, the ExAC consortium deposited the figure generating code for their recent papers[6,7] on Github [https://github.com/macarthur-lab/exac_papers; https://github.com/ericminikel/prnp_penetrance]. Some have gone even further as to publicly release a full manuscript under version control[8,9] and document the process for others to do so [http://ivory.idyll.org/blog/2014-our-paper-process.html].

In this session, we feature five papers that explore research on the topic of reproducibility. This year, we required submissions to strive for reproducibility by depositing data and code on public repositories. The authors have stepped up to the challenge and are practicing what they preach: where possible, they have released applicable code and/or data to make their own research as reproducible as possible.

Session Contributions

Cohain, Divaraniya, and colleagues[10] address an important challenge for reproducibility of Bayesian networks. While frequentist approaches can rely on p-values to predict replication, the construction of a Bayesian network is a data-dependent and heuristic process, and consistency between multiple analyses has not been rigorously performed. This paper explores the replication of Bayesian networks, particularly in relation to key driver nodes and hubs, as well as edge reproducibility.

Hundreds of studies have used publicly available data to predict adverse drug reactions and drug indications and have reported seemingly exceptional predictive accuracy: Guney[11] investigates the

issue of performance overestimation for drug side effect and indication, and finds that major assumption of these methods (independence) is violated, which overestimates their performance. Haynes et al[12] present a pipeline for expression meta-analysis, which fills an unmet need for systematic processing and visualization of results from such analyses. Kaushik and colleagues[13] describe a workflow engine that uses graph theory approaches to optimize and ensure reproducible data analyses. Finally, Yang et al[14] provide a detailed look on the reproducibility of clinical genetics data: concordance across variant classifications is reasonably high, but more work will be required to resolve differences and accurately classify all variants as pathogenic or benign. In summary, these exemplar papers demonstrate how to enhance research reproducibility across a variety of biomedical domains critical in this era of "big data" and precision medicine.

References

1. Longo, D. L. & Drazen, J. M. Data Sharing. *http://dx.doi.org.ezp-prod1.hul.harvard.edu/10.1056/NEJMe1516564* **374,** 276–277 (2016).
2. Stodden, V. The Scientific Method in Practice: Reproducibility in the Computational Sciences. *SSRN Journal* (2010). doi:10.2139/ssrn.1550193
3. Ioannidis, J. P. A. How to Make More Published Research True. *PLoS Med* **11,** e1001747 (2014).
4. Wilson, G. Software Carpentry: lessons learned. *F1000Research* **3,** (2014).
5. ENCODE Project Consortium *et al.* An integrated encyclopedia of DNA elements in the human genome. *Nature* **489,** 57–74 (2012).
6. Lek, M. *et al.* Analysis of protein-coding genetic variation in 60,706 humans. *Nature* **536,** 285–291 (2016).
7. Minikel, E. V. *et al.* Quantifying prion disease penetrance using large population control cohorts. *Science Translational Medicine* **8,** 322ra9–322ra9 (2016).
8. Pell, J. *et al.* Scaling metagenome sequence assembly with probabilistic de Bruijn graphs. *Proc Natl Acad Sci USA* **109,** 13272–13277 (2012).
9. Zhang, Q., Pell, J., Canino-Koning, R., Howe, A. C. & Brown, C. T. These Are Not the K-mers You Are Looking For: Efficient Online K-mer Counting Using a Probabilistic Data Structure. *PLoS ONE* **9,** e101271 (2014).
10. Cohain A, Divaraniya AA, Zhu K, Zhu J, Chang R, Dudley JT, Schadt EE. "Exploring the reproducibility of probabilistic causal molecular network models" *Pac. Symp Biocomput* (2017).
11. Guney E. "Reproducible Drug Repurposing: When Similarity Does Not Suffice" *Pac. Symp Biocomput* (2017).
12. Haynes WA, Vallania F, Liu C, Bongen E, Tomczak A, Andres-Terrè M, Lofgren S, Tam A, Deisseroth CA, Li MD, Sweeney TE, Khatri P. "Empowering Multi-Cohort Gene Expression Analysis to Increase Reproducibility" *Pac. Symp Biocomput* (2017).
13. Kaushik G, Ivkovic S, Simonovic J, Tijanic N, Davis-Dusenbery B, Kural D. "Graph Theory Approaches For Optimizing Biomedical Data Analysis Using Reproducible Workflows" *Pac. Symp Biocomput* (2017).
14. Yang S, Cline M, Zhang C, Paten B, Lincoln SE. "Data Sharing and reproducible Clinical genetic testing: successes and challenges" *Pac. Symp Biocomput* (2017)

EXPLORING THE REPRODUCIBILITY OF PROBABILISTIC CAUSAL MOLECULAR NETWORK MODELS

ARIELLA COHAIN[*], APARNA A. DIVARANIYA[*], KUIXI ZHU, JOSEPH R. SCARPA, ANDREW KASARSKIS, JUN ZHU, RUI CHANG, JOEL T. DUDLEY, ERIC E. SCHADT[†]

Icahn Institute and Department of Genetics and Genomics, Icahn School of Medicine at Mount Sinai, One Gustave L. Levy Place, Box 1498,
New York, NY, 10029, USA
Email: eric.schadt@mssm.edu

Network reconstruction algorithms are increasingly being employed in biomedical and life sciences research to integrate large-scale, high-dimensional data informing on living systems. One particular class of probabilistic causal networks being applied to model the complexity and causal structure of biological data is Bayesian networks (BNs). BNs provide an elegant mathematical framework for not only inferring causal relationships among many different molecular and higher order phenotypes, but also for incorporating highly diverse priors that provide an efficient path for incorporating existing knowledge. While significant methodological developments have broadly enabled the application of BNs to generate and validate meaningful biological hypotheses, the reproducibility of BNs in this context has not been systematically explored. In this study, we aim to determine the criteria for generating reproducible BNs in the context of transcription-based regulatory networks. We utilize two unique tissues from independent datasets, whole blood from the GTEx Consortium and liver from the Stockholm-Tartu Atherosclerosis Reverse Network Engineering Team (STARNET) study. We evaluated the reproducibility of the BNs by creating networks on data subsampled at different levels from each cohort and comparing these networks to the BNs constructed using the complete data. To help validate our results, we used simulated networks at varying sample sizes. Our study indicates that reproducibility of BNs in biological research is an issue worthy of further consideration, especially in light of the many publications that now employ findings from such constructs without appropriate attention paid to reproducibility. We find that while edge-to-edge reproducibility is strongly dependent on sample size, identification of more highly connected key driver nodes in BNs can be carried out with high confidence across a range of sample sizes.

1. Introduction

Biological networks provide a graphical framework for organizing complex relationships among many thousands of variables in ways that can reveal coherent structures. These structures reveal knowledge and improve the understanding of molecular processes linked to higher order functioning of living systems. Vast arrays of data are being generated in numerous areas of biomedical research such as large-scale multi-'omic' studies across many cell types, comprehensive characterizations of microbiota living in and around us, advanced imaging data, and deep clinical characterizations of populations to name a few. This upsurge of big data has forced the life and biomedical sciences to rapidly turn to the use of network constructs. One such organizing framework for integrating data comes in the form of probabilistic network models that seek to capture the regulatory states of a system and their association to complex phenotypes such as disease. A particular class of probabilistic causal networks being applied to model the complexity and causal structure of biological data is Bayesian networks (BNs).

[*] Co-first Authors
[†] Corresponding Author

BNs are increasingly used in the field of genetics to describe and predict gene, metabolite, and protein level interactions. These networks are able to infer causal relationships among variables by employing mutual information or conditional independence measures based on Bayes Theorem. Since 2000 when this method was first applied to understand gene regulation[1], numerous studies have showcased the advantage of using such methods to uncover biological insights that are not easily captured through descriptive methods such as hierarchical clustering or coexpression network analysis. Whether predicting regulatory genetic drivers of complex phenotypes such as human diseases or enabling identification of novel drug target interactions and adverse side effects, BNs have helped uncover the individual genes and biological processes involved in a broad range of human conditions, including cancer, diabetes and obesity, asthma and COPD, cardiovascular disease, and Alzheimer's disease[2–9]. For example, BNs generated from ileal pediatric samples identified a causal gene resulting in a predictor for adult-onset inflammatory bowel disease[10]. As sample sizes increase, it can be envisioned that more groups will use BNs to predict individual response to treatment and it will enable fine-tuning for precision medicine[11].

Constructing a BN structure from data is an NP-hard problem with the complexity equaling $O(n^n)$, where n is the number of nodes in the structure. Many heuristic approaches are applied in searching for an optimal structure from the given data. However, these heuristic methods may find many local sub-optimal structures with no guarantee of finding a global optimal structure. To achieve high accuracy BNs, especially with respect to edge direction, large sample sizes or "big data" are required[12,13]. With the number of large datasets for which BN reconstruction algorithms could be applied growing at an exponential rate, the application of BN algorithms face a similar trend regarding the number of networks being constructed to derive data-driven hypotheses. However, assessing the reproducibility of BNs in the context of gene regulatory networks has not kept pace, with there being no studies to our knowledge systematically exploring this issue. Thus, we thought it crucial to test the conservation and reproducibility of BN constructions as a way to gain confidence in the methods currently used in the field. While significant work has been carried out to assess the construction methods that perform best across different types of biological data[14–16], these types of comparisons do not explicitly address the reproducibility of any given BN.

Perhaps among the gravest concerns in the field of biomedical research today is the lack of reproducibility. It is estimated that over $28 billion of research money, or roughly 50% of life-science research, is not reproducible[17]. The scientific method is rooted with principles of reproducibility giving credence to hypotheses only if they can withstand the scrutiny of many groups trying to reproduce them. In the current era of big data biology, the number of hypotheses generated in even a single publication can number in the hundreds (e.g., GWAS study on a complex trait). These hypotheses are difficult to validate across multiple groups, as the number of groups to rigorously pursue every hypothesis generated is limited. While intuition may argue that the large sample sizes and the robustness of the models may inherently address issues relating to reproducibility compared to traditional biological studies, recent claims indicate that about one quarter (25.5%) of studies not reproduced are due to data-analysis and reporting issues[17]. We therefore focused our study on the reproducibility of individual directed edges and key driver nodes of BNs, as these are generally considered targets for biological validation studies.

2. Study design

Two different gene expression datasets and a simulated dataset were used in this reproducibility study. The first gene expression dataset was obtained from the GTEx Consortium where RNA was

extracted from multiple tissues from deceased, healthy individuals. Here, we used data from whole blood, which had a large sample size (N = 379)[18,19]. The second gene expression dataset was comprised of atherosclerosis patients undergoing Coronary Artery Bypass Grafting (CABG) surgery, at which time multiple tissues were extracted and RNA sequenced from the Stockholm-Tartu Atherosclerosis Reverse Network Engineering Team (STARNET)[20] We chose to utilize the liver tissue (N=545), which contained the strongest eQTL signal[20], a prior in the BN reconstruction algorithm we employed that helps reduce the search space and resolve true causal relationships. By leveraging these real-world datasets, we are able to capture the complex correlation structures that derive from gene expression data measured in populations. RNA levels are high fidelity sensors of the state of the system and of technical noise, where the many different variance components (technical, genetic, micro- and macro-environment) form a complex covariance structure that is difficult to reproduce in simulated datasets. In addition, these two biological datasets represent not only two distinct tissues, but also reflect different states of disease and wellness (Table 1).

Table 1. Overview of datasets used. This table provides details on the two datasets used in this study.

	GTEx	STARNET
Tissue	Whole Blood	Liver
Patient Status	Deceased - Healthy	Living - Undergoing CABG
# Samples	379	545
# Genes Used	455	385
Priors	cis eQTLs	cis eQTLs + Causal Inference Priors

To assess and compare networks in a thorough manner, we restricted attention to a subset of genes (N=465) that have been previously identified as highly informative for inflammatory diseases and associated with immune and inflammation response[2,5,8,21–24]. By selecting this set of genes to use in the analysis, we reduced the computational time and cost required to generate each network.

In order to assess the reproducibility of BNs, we subsampled from the complete datasets to generate datasets reflecting different sample sizes under identical conditions. Towards this end, we subsampled the data in three ways: 1) a subsampling of 50% of the samples (referred to as the subsampled-50 networks), 2) a subsampling of 80% of the samples (referred to as the subsampled-80 networks), and 3) a subsampling of 90% of the samples (referred to as the subsampled-90 networks) (Fig 1). All subsampling divisions were replicated five times. The first scenario was intended to mimic the situation in which an initial study producing a BN is followed by an equivalent replication study producing a confirmatory BN, while the second and third scenarios represent incremental data releases, as happens in the context of large studies where data freezes are employed. The same process was used with the simulated dataset, however, here we were able to control the power and increased our sample size (N=1000) to the point of reaching near perfect reproducibility. For the simulated datasets, we subsampled at 50%, 80%, and 90%, with five replicates generated at each level. We also generated the simulated data at a subsampling of 10% to represent how data with limited noise is reproduced at a small sample size (N=100).

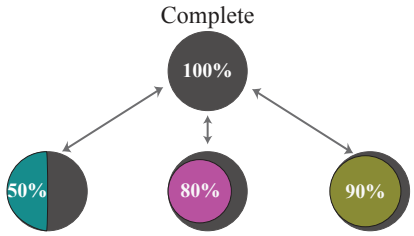


Figure 1. Schematic of the study design.

For all datasets, networks were generated using the Reconstructing Integrative Molecular Bayesian Networks (RIMBANet) algorithm[25,26] as the output has been validated extensively (see methods). When available, eQTL data as well as previous information regarding the causal

association between several genes (nodes) in the network were used as structural priors[5,9,20,25,26]. With BNs, the predominant method for assessing confidence of an edge is based on the posterior probability associated with that edge. This is computed either directly from the network model or is empirically estimated by generating a distribution of models and computing summary statistics across the networks comprising the distribution. We utilized the latter scenario where the posterior probability is approximated by computing the number of networks that contain a particular edge and dividing this number by the total number of networks generated. In this study, we considered nine different posterior probability thresholds (0.1 to 0.9 in 0.1 increments) to explore the reproducibility of edges across different confidence levels. Thus, for each dataset, we generated nine networks for the complete and each of the subsampled datasets.

3. Results

3.1. *Exploring edge-to-edge reproducibility*

Comparing BN's is a multifaceted task in itself as they are complex representations of high-dimensional data. To provide a more intuitive comparison consistent with how BNs are used in practice in the life sciences and biomedical research spaces, we compared networks in two ways: 1) by evaluating the confidence levels of individual edges and 2) by evaluating the higher-level topology of the network.

Table 2: Overlap of five replications of complete BN. For each posterior probability, all combinations of replicates were looked at to calculate the percentage overlap divided by the total edges of each replicate. Here we report the mean percentage and standard deviation.

Posterior Probability	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
GTEx	99% (± 0.008)	99% (± 0.008)	99% (± 0.007)	99% (± 0.008)	99% (± 0.008)	99% (± 0.006)	99% (± 0.004)	98% (± 0.01)	97% (± 0.02)
STARNET	99% (± 0.01)	99% (± 0.01)	99% (± 0.01)	99% (± 0.01)	98% (± 0.01)	99% (± 0.02)	99% (± 0.01)	98% (± 0.02)	96% (± 0.02)

Given the stochastic search employed in the BN construction process, we first compared five networks generated on the complete dataset (includes all samples) for each cohort to characterize the degree of variability. As depicted in Table 2, at a posterior probability of 0.1, both datasets have a mean edge overlap of 99%. While the edges with high confidence (at a posterior probability >0.9) are found on average 97% in other replicates in GTEx and 96% in STARNET, we observe that 100% of these edges are present in other replicates when the posterior probability is > 0.5.

As the stochasticity of the BN reconstruction process does not seem to affect the reproducibility of the BNs, we next calculated the Jaccard index with respect to all network pairs within a given subsampled set (Table 3). The Jaccard index is a measure commonly used when comparing sets, and ranges from 0, for completely unrelated sets, to 1, for highly similar sets. In our case, the edge counts between replicates are comparable when the number of samples and posterior probability are the same (see standard deviations in Table 4), thus the maximum Jaccard index should be close to 1 (complete reproducibility). The Jaccard index had a mean of 0.27 when comparing edges from the subsampled-50 networks across the different posterior probability thresholds within the replicates or to the complete network within each cohort (Table 3).

Interestingly, the Jaccard index achieved values close to 0.5 for edges from the subsampled-90 networks (Table 3), which is very different from the values we saw when comparing the replicates of the complete networks (mean >0.95 in both at a posterior probability >0.1). These results suggest that even with 90% overlap of samples, the edge-set overlap can still be different, highlighting significant reproducibility issues even among highly comparable sample sets. The data suggests that statistical power in resolving network relationships may be primarily responsible for the lower than expected reproducibility, an issue that can be experimentally addressed by increasing the sample size.

Table 3. Jaccard index values. We calculated the Jaccard index (intersection divided by union) for the edges found in the networks at each posterior probability threshold. We compared the subsampling networks to their respective replicates and to the complete BN at the same posterior probability threshold. Standard deviation ranges from 0.01-0.04 in all cases.

		GTEx		STARNET	
Sub-sampling	**Posterior Probability**	**To Other Replicate**	**To Complete**	**To Other Replicate**	**To Complete**
50%	0.1	0.23	0.26	0.22	0.27
	0.5	0.23	0.26	0.21	0.27
	0.9	0.20	0.20	0.16	0.19
80%	0.1	0.34	0.40	0.37	0.43
	0.5	0.34	0.40	0.36	0.43
	0.9	0.29	0.33	0.26	0.35
90%	0.1	0.44	0.51	0.43	0.52
	0.5	0.43	0.53	0.43	0.52
	0.9	0.33	0.39	0.36	0.42
Complete	0.1	0.98		0.97	
	0.5	0.98	---	0.97	---
	0.9	0.95		0.93	

The number of edges in a BN is at least partially a function of power, given that as sample size increases, an increase in the number of edges in the BN is observed (Table 4). Thus, a more applicable measure for assessing reproducibility among networks is by looking at the number of overlapping edges between a subsampled network and the complete network, divided by the number of edges in the subsampled network. This measure relates to precision or positive predictive value, given here we accepted as truth the complete network (in the context of the simulated data, true and false positives are known with certainty). The flip side of precision is recall, or sensitivity, defined by dividing the overlap number of edges by the total number of edges in the complete network (Fig 2A).

For both GTEx and STARNET, when comparing the subsampled and the complete network at the same posterior probability cutoff, we found that on average 44% of GTEx and 38%

Table 4. Number of edges in each network. We calculated the number of edges present in each subsampled network. Displayed are the mean and standard deviation for number of edges at select posterior probabilities.

	GTEx			STARNET			Simulation		
Sub-sampling	**0.1**	**0.5**	**0.9**	**0.1**	**0.5**	**0.9**	**0.1**	**0.5**	**0.9**
10%	---	---	---	---	---	---	209 (± 4.637)	192 (± 3.391)	47.4 (± 5.030)
50%	297.8 (± 8.349)	257.2 (± 3.271)	89 (± 9.055)	291.4 (± 5.683)	262.4 (± 4.336)	113.6 (± 11.393)	345.2 (± 3.493)	329 (± 2.550)	149.8 (± 7.396)
80%	390.8 (± 5.586)	343.6 (± 5.459)	136.6 (± 5.459)	373.2 (± 8.349)	329.8 (± 2.775)	135 (± 8.337)	380.6 (± 4.722)	368.6 (± 1.342)	149 (± 4.000)
90%	414.4 (± 6.465)	365 (± 5.099)	135 (± 4.950)	395 (± 6.205)	350.6 (± 3.647)	144.6 (± 3.782)	385.2 (± 1.095)	373.8 (± 3.493)	185.4 (± 8.081)
Complete	441.6 (± 0.894)	388.4 (± 1.517)	138.2 (± 2.280)	396.8 (± 4.382)	364.2 (± 4.025)	151 (± 1.225)	393.2 (± 0.447)	379.8 (± 0.447)	189.8 (± 0.837)

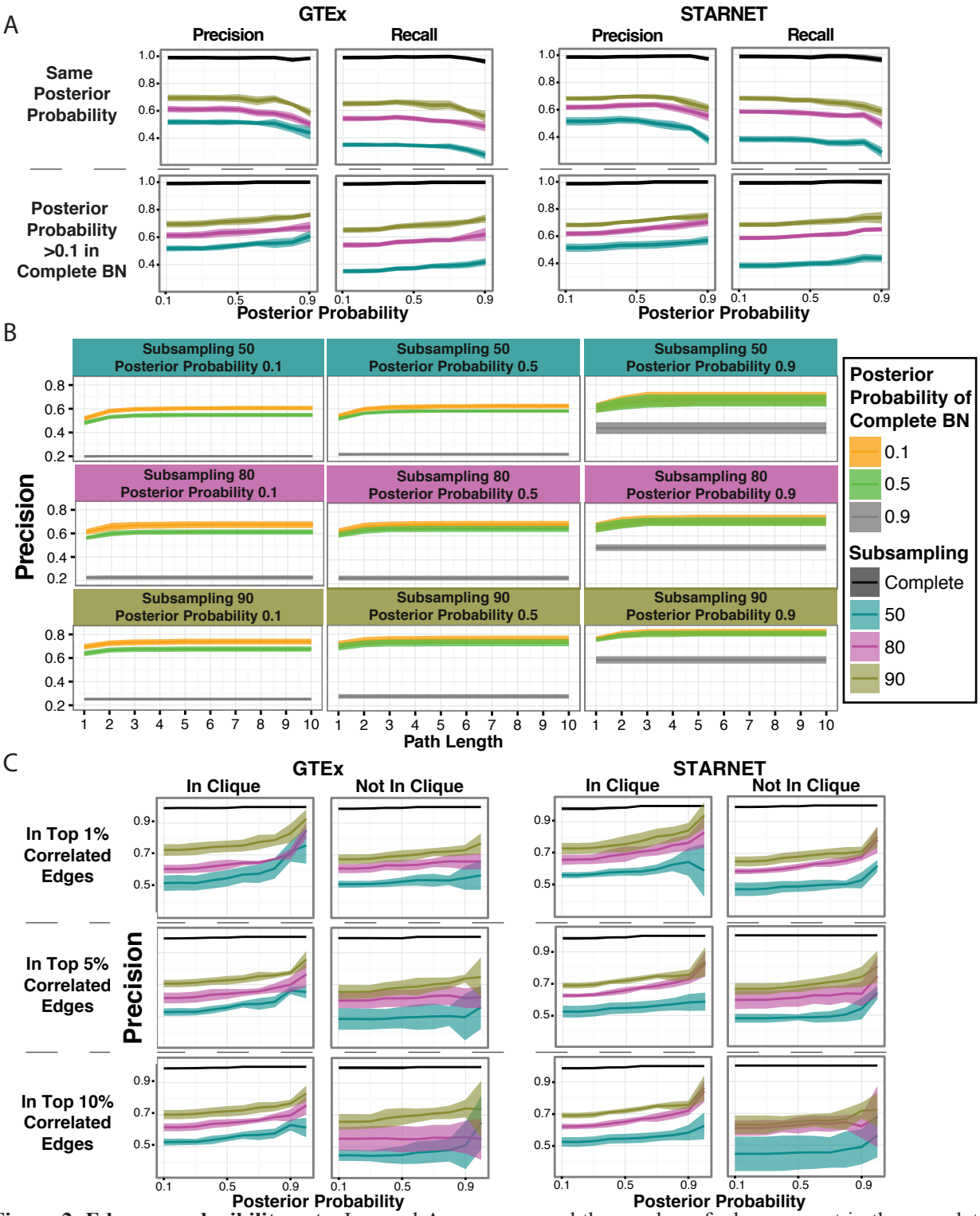


Figure 2. Edge reproducibility rate. In panel A, we compared the number of edges present in the complete BN to the subsampling network at the same posterior probability (top half) and by fixing the threshold for the subsampling networks but allowing any edge for the complete BN (posterior probability >0.1) as seen in the bottom half. In panel B, we show the results from the GTEx data as we allow for edges to be considered reproduced if there is a connection in the complete BN between those two nodes at a path length up to 10. In panel C, we illustrate the percision of edges depending on if the nodes are in the same correlation clique or not. For all panels coloring depicts the subsampling networks and the complete BN.

STARNETs' most confident edges (posterior probability >0.9) in the subsampled-50 networks were reproduced, and this increases to 58% in GTEx and 61% in STARNET for the subsampled-90 networks (Fig 2A). We observed a trend of the precision increasing as the posterior probability increased to 0.4-0.5, but then observed a decrease as the confidence in the edges increased (Fig 2A). This is most likely due to a decrease in the number of edges in the BNs as the posterior probability increases (Table 4). We further evaluated the precision by relaxing the posterior probability for edges in the complete network to >0.1 (Fig 2A). In this case, on average 61% in GTEx and 57% in STARNET of the most confident edges (posterior probability >0.9) were reproduced in the subsampled-50 networks whereas for the subsampled-90 networks 76% in GTEx and 75% in STARNET were reproduced (Fig 2A).

The above definitions of precision at the edge level require the presence of the exact same edge, whereas causal relationships in one network may also be reflected in a different network via intermediary nodes. For example, in one network an edge might be present from A → B (path length=1) and in a second network it may appear as A→C→B, where there is a path from A to B, but via C (path length=2). We hypothesized that this may explain some portion of the edges that failed to reproduce. To test this, we further evaluated if two connected nodes from the subsampled networks were connected in the complete network within a path length of ten. For the GTEx BNs, we saw that in the subsampled-50 networks, the precision increased to an average rate of 67% (up from 61%) at a path length of five for the most confident edges (posterior probability >0.9), while in the subsampled-90 networks, the precision increased to an average rate of 81% (up from 76%) at a path length of three (similar results were seen for STARNET as well). The precision increased with both the path length and sample size (Fig 2B). It should be noted that after a path length of 3, the precision plateaus, providing confidence that increasing the path length further would not have added any new information in the context of our networks.

BNs reflect complex correlation structures or rich substructures in which the expectancy of certain nodes to be more or less connected may be contained within the network. Higher-order correlation structures have been informative for the underlying biology from large datasets[13,27]. To explore whether the correlation structure of the data affected edge reproducibility, we examined whether genes in clique structures (groups of highly interconnected genes) were more or less likely to be reproduced, compared to the average precision of the network. For each data set, we computed the correlation matrix and took the top 1%, 5% and 10% most correlated values to build an undirected, correlation-based network. We focused on the most stringent correlation criteria to define edges, which was the top 1%. From these networks we were able to call all clique communities using the program COS (https://sourceforge.net/projects/cosparallel/). This enabled us to determine if both nodes of an edge were included in the same clique. We found that the precision was further improved in edges whose nodes were found in the same clique (Fig 2C). In the STARNET subsampled-90 networks, the most confident edges (posterior probability >0.9) present in a clique obtained using the top 1% correlated values had a mean precision measure of 85% compared to 71% for edges in which both nodes were not found in the same clique (whereas all edges had a mean precision of 75%). In the subsampled-50 networks, the edges in a clique had a precision rate of 65% versus 53% for edges comprised of nodes that did not both fall within the same clique (whereas all edges had a mean precision of 57%). The GTEx dataset provided similar results, showing that we were able to improve the precision of edges by incorporating correlation clique information.

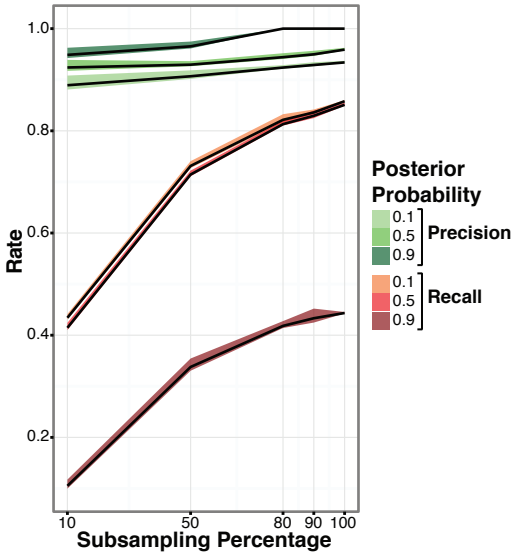


Figure 3. Simulation data precision and recall. We simulated a BN for 300 nodes, 1000 samples with discrete data and looked at the precision and recall for the subsampling at 10%, 50%, 80%, 90%, and 100%. The color scale represents the posterior probability threshold. We show the mean and standard deviation for the five replicates.

Precision and recall trends with the simulated datasets were similar to those observed in the biological datasets. This confirmed not only that our simulated data was reflective of the biological datasets, but also that by increasing sample size we could address the edge-level precision and improve recall (Fig 3). Thus, as larger datasets are generated, the issue of reproducibility of networks should be addressed.

3.2. *On the reproducibility of key driver nodes*

Another important aspect of BNs is their higher order topology. Not all nodes in a BN are equivalent, but rather some are more connected having a substantial causal impact on many more nodes in the network (referred to here as key driver nodes, or KD nodes). One way to assess reproducibility of these types of important topological features is by examining the reproducibility of KDs. KD nodes are important and commonly inferred from networks as they help elucidate the regulatory states of complex systems, and are crucial from a diagnostic and drug discovery standpoint[2,5,28]. Thus, we decided to assess the reproducibility of the detection of these types of nodes.

We calculated the KDs for each network built at each posterior probability threshold and assessed the precision of the KDs in the same manner applied to the edges (see methods). First, we evaluated the overlap of KDs between the complete and subsampled networks when they were built at the same fixed posterior probability. To see if a difference between the ranking of KDs and their precision could be measured, we defined the top KDs as being in the 97.5 – 100 percentile and bottom KDs as being in the 95 – 97.5 percentile. When evaluating the KDs of the network built from the most confident edges (posterior probability >0.9), we found that the top KDs from the subsampled-90 networks were reproduced at an average rate of only 49% while the bottom KDs were reproduced at an average rate of 54% in GTEx. In STARNET, the top KDs were reproduced at an average rate of 85% while the bottom KDs were reproduced at an average rate of 43% (Fig 4)**.** To see if we could improve the reproducibility rate, we relaxed the threshold for the complete BN and allowed for the KD to be present at any posterior probability (similar to what was done with the edges). This drastically improved the reproducibility of the KDs. In GTEx, the top KDs from the subsampled-90 networks built on the most confident edges (posterior probability >0.9) were reproduced at an average rate of 87% while the bottom KDs were reproduced at an average rate of 77%. A similar evaluation of the STARNET results showed the top KDs were reproduced on average 93%, while the bottom KDs were reproduced at 60%. We saw in the subsampled-50 networks, at a posterior probability >0.5 that while the edge-overlap was on average 54% in GTEx and 53% in STARNET, the KD overlap was 58% in GTEx and 66% in STARNET. In the subsampled-90 networks, where the edge-overlap was on average 72% in GTEx and 71% in STARNET, the KD overlap increased to 76% in GTEx and 87% in STARNET. The KDs performed as well if not better than the edges, indicating that the KDs of BNs are more

conserved than edges. Since the networks with fewer samples have fewer edges present, it could help explain why we see such low precision in the subsampled-50 networks. These results further support that a larger sample size, or increased power, will lead to more reproducible KDs.

As the KDs take into account the shortest path to reach all nodes, we thought to additionally assess nodes with the highest number of first-degree downstream targets, hub nodes. These nodes have the most local and direct impact on other nodes. Here we took the top 10% of nodes based on their total number of out edges and applied the same analysis pipeline defined above for KDs. We found that when the posterior probability >0.1 for the complete network, the hub nodes were more reproduced in the subsampled networks, as can be seen by the subsampled-90 networks reaching an average rate of 78% in GTEx and 83% in STARNET at a posterior probability threshold of 0.5 (Fig 5). However, if we hold the posterior probabilities constant in both the complete and subsampled networks, the precision fluctuates in the GTEx dataset but appears to perform better in the STARNET dataset. This could be explained by the larger sample size of the STARNET dataset.

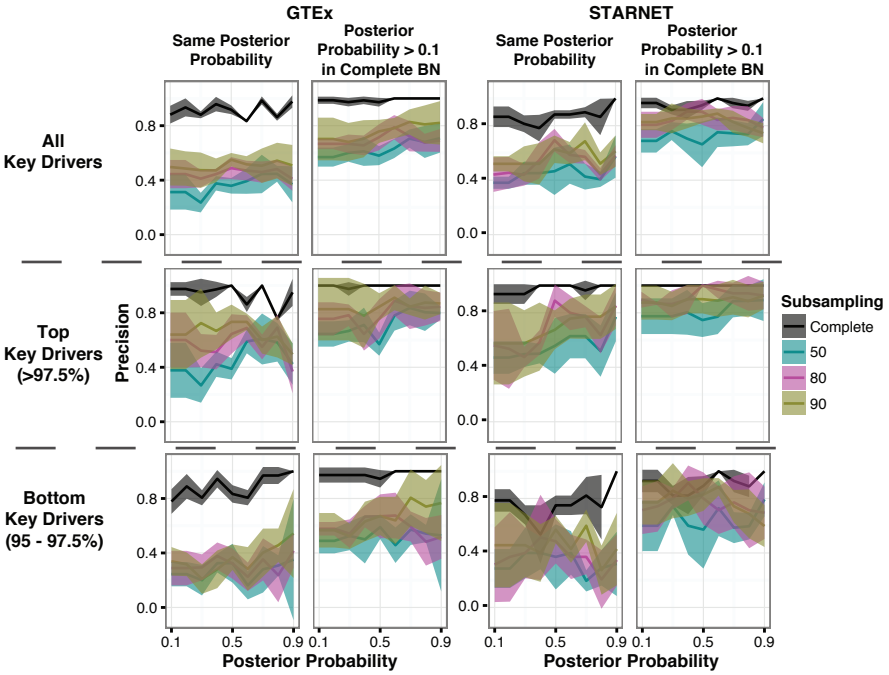


Figure 4. Precision of key driver (KDs). Precision is the % KDs of the subsampling network present in the complete BN (at either the same posterior probability threshold or at any). Left panel shows all KDs; Middle panel shows Top KDs (top 97.5% based on the weighted number of connections, see methods); Right panel, shows bottom KDs (95 – 97.5%). Mean and standard deviation for the five replicates are displayed, and color depicts subsampling.

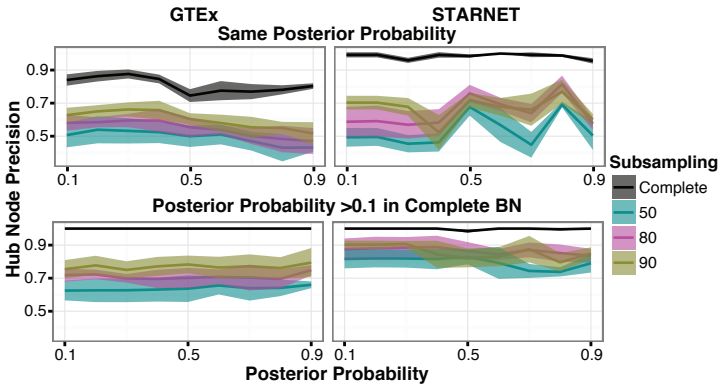


Figure 5. Hub nodes precision. We define hub nodes as nodes in the 90th percentile based on the number of first degree out edges. The top half illustrates the precision when the posterior probability is the same in both the subsampling and the complete BN. The bottom half illustrates the precision when the posterior probability in the subsampling network is fixed but the hub node in the complete BN can be at any posterior probability. The mean and standard deviation for the five replicates is displayed and color depicts subsampling.

4. Discussion

In this study on the reproducibility of BNs in the context of regulatory gene

networks, while we found a high degree of reproducibility at the edge and key driver node levels, we also noted that a large proportion of edges and key driver nodes were not reproduced. Given the rate at which edges and key driver nodes did not reproduce in networks constructed from a moderate number of samples, caution should be exercised when interpreting specific features of a network. Validating hypotheses generated from networks is critical to ensure the accuracy of network predictions. However, we also observed that the lack of reproducibility might be attributed to power issues, which can be straightforwardly addressed by increasing sample sizes for network reconstructions. As obtaining large sample size is difficult and expensive, our results stress the need to assess the reproducibility of methods being deployed in the field. We must be aware of limitations so we can strive to improve them.

While we restricted attention to a coherent subset of several hundred genes to contain computational costs, we have observed similar trends in BNs built on 10,000 or more genes using the GTEx whole blood samples, suggesting that the subset of genes used was a good proxy for how larger networks of genes would behave. Ideally, we would have run our analysis on a completely validated BN from a biological dataset. However, at the time of this study, such a validated network was not available. Instead, we complemented our study of networks constructed from gene expression datasets with examination of simulated datasets containing discretized data for a comparable number of genes.

We used structural priors to generate the BNs, which could bias the structure of the resulting networks. However, we saw a decrease in precision and recall when priors were not used, further demonstrating the importance of high-confidence priors. We chose to include priors as this is typically done in practice today and their use has shown to increase accuracy of networks based on smaller sample sizes[26].

The reproducibility of KDs was of particular interest, given the role they play in current biological investigations of complex systems. KDs represent central information flow points in the network that are identified in disease studies as potential targets of therapeutic intervention or as features that may be critical as biomarkers of disease. We observed that KDs were more reproduced than edges. This suggests that while the edges may be less conserved due to nonlinear interactions or stochasticity, the overall structure of the network may still be well conserved, explaining the increased confidence in key driver node predictions. In particular, the top KDs, which are most connected and predicted to significantly impact network states, were reproduced at exceptionally high rates.

As biomedical and life sciences research gravitates toward network-based constructs, issues of reproducibility will come front and center. It is critical to characterize network reconstruction methods from the standpoint of what is required to lead to reproducible structures that in turn, lead to high-confidence hypotheses. Our analysis shows that well-powered Bayesian networks are highly reproducible. Since high power is not always possible to achieve because samples are scarce and assays are expensive, our results provide guidance on interpreting and using Bayesian networks. In cases of diminished power, it is critical to realize that key drivers, in particular the strongest key drivers, and hub nodes are more robustly reproduced than individual edges.

5. Methods

Bayesian Network Construction: RIMBANet was used to construct all Bayesian Networks[9,12,26]. Continuous data was used for calculating partial priors, which are then used as priors in the network construction. Additional priors included genes that are *cis* eQTLs and the results from the

causal inference test of *cis gene* → *trans gene* (for STARNET only)[20,29]. For the eQTL priors, if a gene also has a strong eQTL associated with it in *cis*, such a gene can be considered as a parent node, given the genotype cannot be the effect of a gene expression change. The data was discretized into 3 states for each gene: high expression levels, low expression levels and unexpressed. This is done by first normalizing the values for each gene to ensure a normal distribution. Then, k-means clustering (k=3) is used with the option of dropping groups should there not be enough members to fill it to assign the values for each sample. In a case where there are only two clusters they would be classified as high and low[30]. For the sake of quicker run times, when looking for the parents of each gene, the other genes were sorted by their mutual information and only the top 80% were considered as candidates. Also, the maximum number of parent nodes that were allowed for any given node was set to 3. After running successfully 1,000 reconstructions, the networks were pooled together. Finally, because a BN is a directed acyclic graph (DAG) by definition, the consensus network was obtained by searching for the shortest cycle and then the edge with the weakest weight (the smallest number of times it occurs in 1,000 reconstructions) was removed. This process was repeated until no cycles were present and the resulting network was a DAG.

Generation of Simulated Dataset: To generate the synthetic true network, we used the SynTRen software v1.2[31]. We extracted a subnetwork with 300 nodes from the background source network "DAG1_clean.sif" with default settings. We limited the node selection to 300 nodes to reduce the computational time required to generate all of the networks and to mimic the size of the biological datasets used in this study. Next, to generate the synthetic discretized data from the known network structure, we utilized Bayes Net Toolbox (BNT) for Matlab [https://code.google.com/archive/p/bnt/]. The conditional probability was customized so that we could discretize the data into three bins, similar to RIMBAnet. Given the configuration of parent node, the child nodes were skewed towards one of the three discretized states with a probability between 0.8 and 0.9, therefore, ensuring assignment to a given bin with high confidence.

Key Driver Node Detection: Key driver nodes (KDs) were detected by calculating the shortest downstream path length between each pair of nodes in the network. For each candidate key driver node, we took the inverse of path length between the candidate key driver node and every other node in the network. We then summed the inverse path lengths to obtain a final score per node. Based on this calculation, we defined nodes in the 95th percentile as KDs[5]. We define top KDs as nodes in the 97.5 - 100 percentile and bottom KDs as nodes in the 95 - 97.5 percentile.

Code and data can be found at https://github.com/divara01/PSB2017_ReproducibilityOfBNs/ and http://research.mssm.edu/integrative-network-biology/Software.html

Acknowledgements

Funding for this project was provided by National Institute of Health (NIH) grants U54CA189201, R01DK098242, 5U01AG046170, and 1R01MH109897 and Leducq Foundation grant 12CVD02.

References

1. Friedman, N., Linial, M., Nachman, I. & Pe'er, D. Using Bayesian Networks to Analyze Expression Data. *J. Comput. Biol.* **7,** 601–620 (2000).
2. Jostins, L. *et al.* Host-microbe interactions have shaped the genetic architecture of inflammatory bowel disease. *Nature* **491,** 119–24 (2012).
3. Korucuoglu, M., Isci, S., Ozgur, A. & Otu, H. H. Bayesian Pathway Analysis of Cancer Microarray Data. *PLoS One* **9,** e102803 (2014).

4. Schwartz, S. M., Schwartz, H. T., Horvath, S., Schadt, E. & Lee, S.-I. A systematic approach to multifactorial cardiovascular disease: causal analysis. *Arterioscler. Thromb. Vasc. Biol.* **32,** 2821–35 (2012).
5. Zhang, B. *et al.* Integrated systems approach identifies genetic nodes and networks in late-onset Alzheimer's disease. *Cell* **153,** 707–20 (2013).
6. Kidd, B. a, Peters, L. a, Schadt, E. E. & Dudley, J. T. Unifying immunology with informatics and multiscale biology. *Nat. Immunol.* **15,** 118–27 (2014).
7. Schadt, E. E. Molecular networks as sensors and drivers of common human diseases. *Nature* **461,** 218–23 (2009).
8. Greenawalt, D. M. *et al.* A survey of the genetics of stomach, liver, and adipose gene expression from a morbidly obese cohort. *Genome Res.* **21,** 1008–16 (2011).
9. Zhu, J. *et al.* Integrating large-scale functional genomic data to dissect the complexity of yeast regulatory networks. **40,** 854–861 (2008).
10. Li, Q. *et al.* Variants in TRIM22 That Affect NOD2 Signaling Are Associated With Very-Early-Onset Inflammatory Bowel Disease. *Gastroenterology* **150,** 1196–207 (2016).
11. Uzilov, A. V. *et al.* Development and clinical application of an integrative genomic approach to personalized cancer therapy. *Genome Med.* **8,** 62 (2016).
12. Zhu, J. *et al.* An integrative genomics approach to the reconstruction of gene networks in segregating populations. *Cytogenet. Genome Res.* **105,** 363–74 (2004).
13. Langfelder, P. & Horvath, S. WGCNA: an R package for weighted correlation network analysis. *BMC Bioinformatics* **9,** 559 (2008).
14. Marbach, D., Schaffter, T., Mattiussi, C. & Floreano, D. Generating realistic in silico gene networks for performance assessment of reverse engineering methods. *J. Comput. Biol.* **16,** 229–39 (2009).
15. Saez-Rodriguez, J. *et al.* Crowdsourcing biomedical research: leveraging communities as innovation engines. *Nat. Rev. Genet.* **17,** 470–486 (2016).
16. Hill, S. M. *et al.* Inferring causal molecular networks: empirical assessment through a community-based effort. *Nat. Methods* **13,** 310–318 (2016).
17. Freedman, L. P., Cockburn, I. M. & Simcoe, T. S. The Economics of Reproducibility in Preclinical Research. *PLoS Biol.* **13,** e1002165 (2015).
18. Lonsdale, J., Thomas, J., Salvatore, M. & Phillips, R. The genotype-tissue expression (GTEx) project. *Nat. ...* **45,** 580–5 (2013).
19. Ardlie, K. G. *et al.* The Genotype-Tissue Expression (GTEx) pilot analysis: Multitissue gene regulation in humans. *Science (80-.).* **348,** 648–660 (2015).
20. Franzén, O. *et al.* Cardiometabolic risk loci share downstream cis- and trans-gene regulation across tissues and diseases. *Science* **353,** 827–30 (2016).
21. Chen, Y. *et al.* Variations in DNA elucidate molecular networks that cause disease. *Nature* **452,** 429–35 (2008).
22. Emilsson, V. *et al.* Genetics of gene expression and its effect on disease. *Nature* **452,** 423–8 (2008).
23. Wang, I.-M. *et al.* Systems analysis of eleven rodent disease models reveals an inflammatome signature and key drivers. *Mol. Syst. Biol.* **8,** 594 (2012).
24. Yang, X. *et al.* Validation of candidate causal genes for obesity that affect shared metabolic pathways and networks. *Nat. Genet.* **41,** 415–23 (2009).
25. Zhu, J. *et al.* Stitching together multiple data dimensions reveals interacting metabolomic and transcriptomic networks that modulate cell regulation. *PLoS Biol.* **10,** e1001301 (2012).
26. Zhu, J. *et al.* Increasing the Power to Detect Causal Associations by Combining Genotypic and Expression Data in Segregating Populations. **3,** (2007).
27. Song, W.-M. *et al.* Multiscale Embedded Gene Co-expression Network Analysis. *PLOS Comput. Biol.* **11,** e1004574 (2015).
28. Dudley, J. T. *et al.* Computational repositioning of the anticonvulsant topiramate for inflammatory bowel disease. *Sci. Transl. Med.* **3,** 96ra76 (2011).
29. Millstein, J., Zhang, B., Zhu, J. & Schadt, E. E. Disentangling molecular relationships with a causal inference test. *BMC Genet.* **10,** 23 (2009).
30. Zhu, J. *et al.* Complexity of Yeast Regulatory Networks. **40,** 854–861 (2009).
31. Van den Bulcke, T. *et al.* SynTReN: a generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC Bioinformatics* **7,** 43 (2006).

REPRODUCIBLE DRUG REPURPOSING: WHEN SIMILARITY DOES NOT SUFFICE

EMRE GUNEY*

Joint IRB-BSC-CRG Program in Computational Biology, Institute for Research in Biomedicine
c/ Baldiri Reixac 10-12, Barcelona, 08028, Spain
**E-mail: emre.guney@irbbarcelona.org*

Repurposing existing drugs for new uses has attracted considerable attention over the past years. To identify potential candidates that could be repositioned for a new indication, many studies make use of chemical, target, and side effect similarity between drugs to train classifiers. Despite promising prediction accuracies of these supervised computational models, their use in practice, such as for rare diseases, is hindered by the assumption that there are already known and similar drugs for a given condition of interest. In this study, using publicly available data sets, we question the prediction accuracies of supervised approaches based on drug similarity when the drugs in the training and the test set are completely disjoint. We first build a Python platform to generate reproducible similarity-based drug repurposing models. Next, we show that, while a simple chemical, target, and side effect similarity based machine learning method can achieve good performance on the benchmark data set, the prediction performance drops sharply when the drugs in the folds of the cross validation are not overlapping and the similarity information within the training and test sets are used independently. These intriguing results suggest revisiting the assumptions underlying the validation scenarios of similarity-based methods and underline the need for unsupervised approaches to identify novel drug uses inside the unexplored pharmacological space. We make the digital notebook containing the Python code to replicate our analysis that involves the drug repurposing platform based on machine learning models and the proposed disjoint cross fold generation method freely available at `github.com/emreg00/repurpose`.

Keywords: Drug repurposing; Machine learning; Drug similarity; Stratified disjoint cross validation.

1. Introduction

Computational drug repurposing has gained popularity over the past decade, offering a possibility to counteract the increasing costs associated with the conventional drug development pipelines. Several studies have focused on training similarity-based predictors (also known as knowledge-based or guilt-by-association-based methods) using drug chemical, target and side effect similarity between drugs (see Refs. 1–3 for recent reviews). These studies often combine various features including but not limited to chemical 2D fingerprint similarity, overlap or interaction network closeness of drug targets and correlation between drug side effects and build a machine learning model based on different algorithms, such as support vector machines, random forests and logistic regression classifiers.[4–11] The proposed models are then compared in a cross validation setting, in which a portion of the known drug-disease associations are hidden during training and used for the validation afterwards. The areas under reciver operating characteristic (ROC) curves in the cross validation analysis reported for these models range between 75-95%, suggesting that some of these models can accurately identify novel drug-disease associations. Nevertheless, in reality, the applicability of these methods for discovery of novel drug-disease associations has been limited due to "the reliance on data existing nearby in pharmacological space" as highlighted by Hodos et *al.*[2] Moreover, Vilar and colleagues alert

the community about the potential "upstream bias introduced with the information provided in the construction of the similarity measurement" in similarity-based predictors.[12] Yet, since many studies do not provide the data and code used to build the models for repurposing, it is often cumbersome to validate, reproduce and reuse the underlying methodology.

In this study, first, we provide a Python-based platform for reproducible similarity-based drug repurposing and then seek to quantify the effect of the assumptions on the existing data nearby in pharmacological space. Following similar works evaluating various cross validation approaches for drug-target and protein-protein interaction prediction,[13,14] we adopt a stratified disjoint cross validation strategy for splitting drug-disease pairs, where none of the drugs in the training set appear in the test set. We show that, although a simple logistic regression classifier can achieve good performance on the data set under a conventional cross validation setting, it performs poorly when it faces with drugs it has never seen before.

Overall, our results suggest that the prediction accuracies reported by existing supervised methods are optimistic, failing to represent what one would expect in a real-world setting. We believe that the platform provided in this study could be useful for prospective studies to perform benchmarking in a unified manner.

2. Results

2.1. *A Python platform for reproducible similarity-based drug repurposing*

To incentivize reproducibility in computational drug repurposing research, we provide a Python-based platform[a] encapsulating several machine learning algorithms available in Python Scikit-learn package[15] available both as stand alone code and Jupyter notebook. The platform consists of methods to (*i*) parse a publicly available data set containing drug chemical substrucuture, target, side effect information, (*ii*) calculate drug similarity using a combination of the three features provided in the data set, (*iii*) balance data such that the drug-disease pairs have the same proportion of positive and negative instances, (*iv*) apply cross validation, and (*v*) build classifiers (Fig. 1).

The platform facilitates access to several machine learning algorithms and cross validation utilities available in Scikit-learn. By changing the configuration values, the user can build a classifier using default parameters based on logistic regression, k-nearest neighbor classifier, support vector machine, random forest, and gradient boosting classifier. We note, however, these methods are provided as is and the user still has to take the necessary steps for parameter optimization for these methods. The user can also adjust the proportion of the positive and negative pairs within each fold by changing the parameter file. Furthermore, the platform is easily customizable, allowing the user to define her own data balancing, cross validation and classifier building methods.

2.2. *Evaluating similarity-based drug repurposing via cross validation*

Next, we show the utility of the platform by building a logistic regression based drug repurposing classifier that incorporates drug chemical, target, and side effect similarity, a simplified

[a]Available at `github.com/emreg00/repurpose`

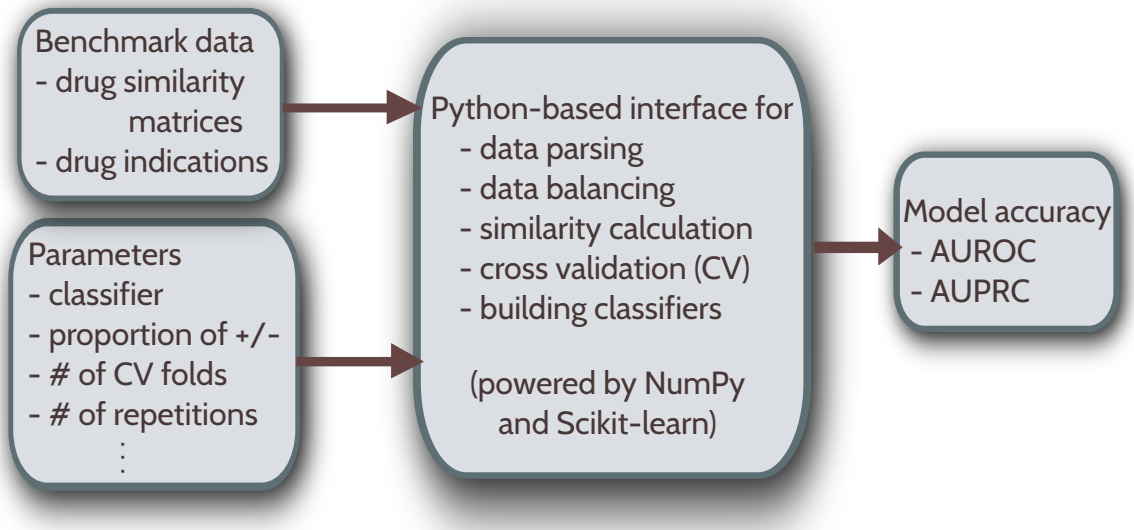


Fig. 1. Overview of the reproducible similarity-based repurposing platform.

version of the classifier suggested in a seminal paper by Gottlieb and colleagues.[4] Our model uses three drug-drug similarity based features compared to the combination of five drug-drug similarity (similarity of targets in terms of gene ontology functions and protein interaction network closeness in addition to the drug chemical, target, and side effect similarity) and two disease-disease similarity-based features (ten in total) proposed by Gottlieb and colleagues. We also incorporate the k-nearest-neighbor approach used by Zhang and coworkers,[7] who recently, built a classifier based on similarity to the 20 most similar drugs and compared it to Gottlieb and colleagues. We build our model on the same data set[b] used by Zhang and coworkers. We calculate the Pearson correlation between drugs using each of the three features mentioned above. For each feature, we assign a score corresponding to the likelihood of a given drug to be indicated for a disease based on the similarity scores and labels of the most similar 20 drugs. These scores are then combined in a logistic regression model and coefficients of the model is derived using a cross validation scheme (see Methods).

We test the prediction accuracy of the classifier under a ten fold cross validation scheme, where we split the available data set into ten groups, leave one group for testing the accuracy of the classifier and use the remaining groups to train the classifier. We repeat the cross validation analysis ten times to get estimates on the mean and standard deviation of the areas under ROC curves (AUC) and report these values in Table 1. We find that the AUC of the classifier is 84%, comparable to 87% reported by Zhang and coworkers. The slight discrepancy between the values can be explained by *(i)* the original study using imputation on the feature set and/or *(ii)* the authors reporting the AUC value from a single run instead of the mean

[b]Made publicly available by the Zhang *et al.* at `http://astro.temple.edu/~tua87106/drugreposition.html`

over multiple cross validation runs (due to the random subsampling of the data, the AUC values in consequent runs might vary slightly).

Table 1. Areas under ROC and Precision-Recall curves (AUC and AUPRC) under various validation schemes averaged over ten runs of ten-fold cross validation (S.d.: Standard deviation).

Disjoint folds	Mean AUC (%)	S.d. AUC (%)	Mean AUPRC (%)	S.d. AUPRC (%)
No	84.1	0.3	83.7	0.3
Yes	65.6	0.5	62.8	0.5

2.3. *Revisiting cross validation using disjoint folds*

Existing studies often assume that the drugs that are in the test set will also appear in the training set, a rather counter-intuitive assumption as, in practice, one is often interested in predicting truly novel drug-disease associations (i.e. for drugs that have no known indications previously). We challenge this assumption by evaluating the effect of having training and test sets in which none of the drugs in one overlaps with the drugs in the other. Accordingly, we implement a disjoint cross validation fold generation method that ensures that the drug-disease pairs are split such that none of the drugs in the training set appear in the test set (Fig. 2, see Methods for details).

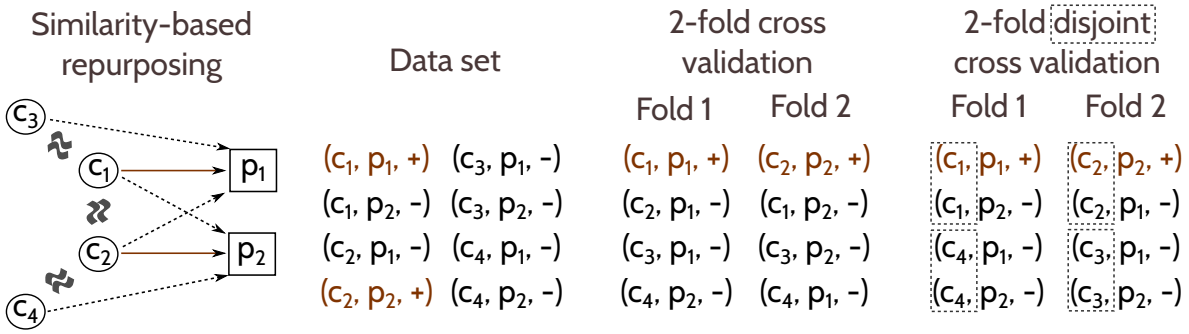


Fig. 2. Schematic representation of similarity-based repurposing and cross validation strategy. On a toy data set consisting of four compounds c_1, c_2, c_3, c_4 and two phenotypes p_1, p_2, the similarity-based drug repurposing approach is illustrated. c_1 and c_2 are indicated for p_1 and p_2, respectively. For instance, c_3 can be inferred to be useful for p_1 due to its similarity to c_1. Conventionally, k-fold cross validation randomly splits the data into k groups preserving the overall proportion of the labels in the data. We propose a disjoint cross validation scheme for paired data, such as drug-disease pairs in drug repurposing studies, that does not only preserve the proportion of the labels but also ensures that none of the drugs from the pairs in one fold are in the other folds. We demonstrate this on the toy data for k = 2 (two-fold cross validation).

In fact, several studies aim to investigate the prediction performance when the drugs in the test set are dissimilar to those in the training data set. Nonetheless, they usually do not guarantee that the trained models are unbiased with respect to unseen data. For instance,

Luo *et al.*[11] use an independent set of drug-disease associations, yet, 95% of the drugs in the independent set are also in the original data set (109 out of 115). On the other hand, Gottlieb *et al.*[4] create the folds such that 10% of the drugs are hidden instead of 10% of the drug-disease pairs, but they do not ensure that the drugs used to train the model are disjoint from the drugs in the test set.

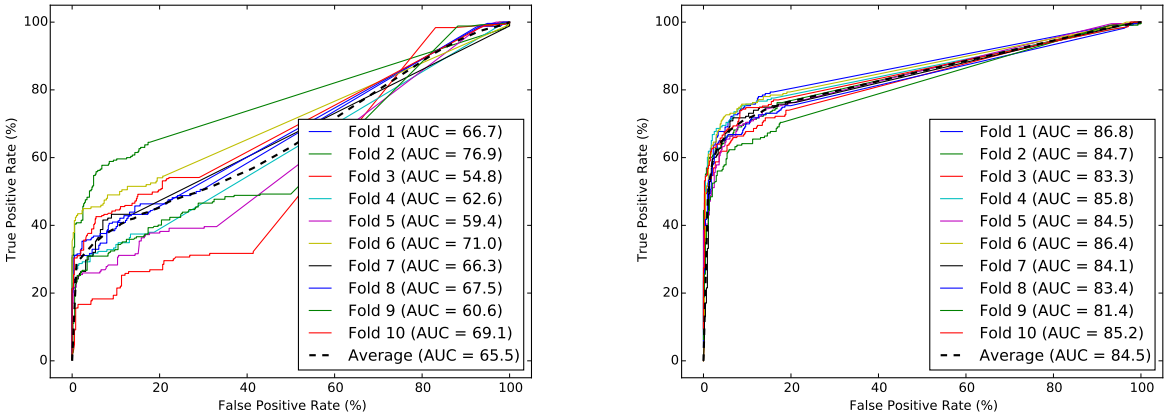


Fig. 3. ROC curves for each fold with and without disjoint cross validation (in a single run).

2.4. *Effect of the cross validation strategy on classifier performance*

We use the drug-wise disjoint cross validation strategy to study its effect on the classifier performance. We observe that the AUC drops significantly from 84% to 66% ($P = 6.9 \times 10^{-23}$, assessed by two-sided t-test) when the classifier is trained with drug-disease associations coming from the drugs that do not exist in the test data set (Table 1).

We suspect that this is due to the limited information within the test set from which the similarity-based drug-disease associations are calculated (using 20 most similar drugs) before they are fed to the classifier. To verify this, we repeat the analysis using two-fold, five-fold and 20-fold cross validation and show that the number of folds does indeed have an effect on the classifier performance (Table 2). In the two-fold disjoint cross validation scheme, the classifier accuracy is almost as good as the ten-fold cross validation accuracy without using disjoint folds, probably due to the number of drug-disease pairs within the test fold being large enough to capture the similarity relationships between drugs. Conversely, in the 20-fold disjoint cross validation scheme, the AUC drops to 59%, emphasizing the effect of the test set size due to the increased number of folds.

We next turn to the ROC curve of each cross validation fold under the two different strategies to examine the consistency among different folds (Fig. 3). We recognize that the variance between the ROC curves is higher when the folds are drug-wise disjoint compared to when drugs are shared among folds. As a result, the standard deviation over the corresponding AUC values is larger in the drug-wise disjoint case (6.0% in disjoint vs 1.5% in non-disjoint),

Table 2. Areas under ROC and Precision-Recall curves under disjoint k-fold cross validation scheme for $k = 2$, 5, 10 and 20 averaged over ten runs.

Number of folds	Mean AUC (%)	S.d. AUC (%)	Mean AUPRC (%)	S.d. AUPRC (%)
2	80.7	0.3	79.3	0.3
5	73.6	0.7	71.9	0.7
10	65.6	0.5	62.8	0.5
20	59.1	0.6	57.0	0.3

suggesting that the predictions are less robust against the partitioning of the drugs in disjoint cross validation.

Compiled mainly via text mining, the drug side effect information in SIDER is prone to a high number of false positives. Given the reduced number of drugs with high similarity, the effect of false positive associations might be more pronounced in the disjoint cross validation than the non-disjoint scenario. Thus, to inspect whether the observed decline in the AUC can be attributed to one of the features used in the classifier –such as side effect based similarity–, we check the contribution of each feature under the disjoint cross validation scheme (Fig. 4). We confirm that this is not the case. In fact, the feature based on side effect similarity is slightly more predictive than the rest (AUC=65% for side effect similarity vs 62% and 61% for chemical and target similarity, respectively), corroborating the promise of side effect profiles to describe similarities between drugs,[4,7,16,17] despite potential noise in the annotations.

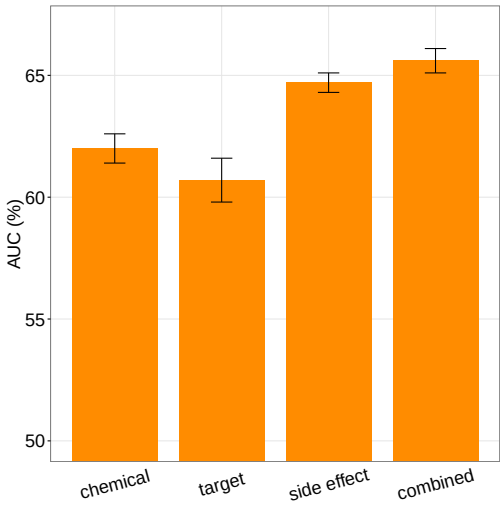


Fig. 4. Prediction accuracy (AUC) when each similarity feature used individually in disjoint cross validation. Error bars show standard deviation of AUC over ten runs of ten-fold cross validation.

2.5. *When similarity does not suffice*

The drop in the AUC confirms that many drug-disease associations are missed when the drugs in the test set have not been seen while training the classifier. For instance, the gold standard

data contains several lipid lowering agents indicated for hypercholesterolemia: cholesterol absorption inhibitors (ezetimibe); fibrates (clofibrate, fenofibrate, gemfibrozil); and statins (atorvastatin, fluvastatin, lovastatin, pravastatin, simvastatin). We observe that most of these drugs can be predicted for hypercholesterolemia due to their chemical, target, and side effect based similarity to the other drugs within the same family when drugs are allowed to overlap across cross validation folds. However, when the classifier is trained using disjoint cross validation, most of these drug-disease associations can not be predicted correctly. Likewise, the drugs used for juvenile rheumatoid arthritis (diclofenac, ibuprofen, methotrexate, naproxen, oxaprozin, sulfasalazine, toletin) fail to manifest similarity to other drugs in the cross validation fold, hence missed by the classifier. We also note a similar trend for acute myeloid leukemia drugs (cyclophosphamide, daunorubicin, etoposide, idarubicin, mitoxantrone). In Table 3, we highlight the similarity-based scores of the drug to the other drugs and the probability calculated by the logistic regression classifier in a cross validation fold for several of these drug-disease associations.

Table 3. Similarity scores and logistic regression based probabilities for several known drug-disease associations missed using disjoint cross validation.

	Non-disjoint cross validation				Disjoint cross validation			
Drug	Chemical score	Target score	Side effect score	Probability	Chemical score	Target score	Side effect score	Probability
Hypercholesterolemia drugs								
fenofibrate	0.76	0.71	1.10	0.82	0.57	0	0.46	0.36
lovastatin	1.93	1.97	2.92	0.99	0	0	0	0.14
Juvenile rheumatoid arthritis drugs								
ibuprofen	0.82	3.50	1.08	1.00	0	0.50	0.43	0.43
sulfasalazine	1.39	1.99	0.43	0.96	0	0.50	0.43	0.43
Acute myeloid leukemia drugs								
daunorubicin	1.77	1.50	0	0.87	0	0	0	0.15
idarubicin	0.78	2.00	0.81	0.97	0	0	0	0.14

3. Methods

3.1. *Data sets*

We have retrieved the data set Zhang *et al.* curated for the analysis of the drug repurposing classifier they proposed.[7] They collected 1,007 approved drugs and their targets from DrugBank,[18] the chemical structure information of these drugs from PubChem[19] and the side effect information from SIDER.[20] The drugs were represented by a combination of 775 targets extracted from DrugBank and 881 substructures in PubChem. They were able to map side effects of 888 out of 1,007 drugs using SIDER, covering 61,102 drug-side effect associations coming from 1,385 side effects. The known drug-disease indications span 3,250 associations between 799 drugs and 719 diseases and were extracted from the National Drug File - Reference Ter-

minology (NDF-RT) as suggested in a previous study by Li and Lu.[21] The data set is publicly available online at `http://astro.temple.edu/~tua87106/drugreposition.html`. We used the 536 drugs that were common among chemical, target, side effect, and indication data, corresponding to 2,229 drug-disease associations covering 578 diseases and 40,455 drug-side effect associations covering 1,252 side effects.

3.2. *Drug similarity definitions*

We used the data sets described above to build a drug-drug similarity matrix for each one of the three feature types: chemical substructures, targets, side effects. For each feature type, the drug i was defined by a binary vector $X_i = [x_1, x_2, \ldots, x_n]^T$, corresponding to the existence of the feature for that drug (1 if exists, 0 otherwise). The Pearson product-moment correlation coefficient between two drugs i and j was then calculated using $\rho_{ij} = \frac{C_{ij}}{\sqrt{C_{ii} * C_{jj}}}$, where C_{ij} given by

$$C_{ij} = \mathrm{cov}(X_i, X_j) = \mathrm{E}[(X_i - \mathrm{E}(X_i))(X_j - \mathrm{E}(X_j))]$$

The *corrcoef* function implemented in NumPy Python package was used to calculate correlation coefficients for each drug-drug pair.

3.3. *Similarity-based logistic regression classifier*

We trained a logistic regression model to predict the drug-disease associations based on the drug-drug similarities defined by the targets, chemical substructures, and side effects combined for the 20 most similar drugs to the drug in concern. Therefore, the probability of observing an association between the drug i and the disease j is

$$P(Y_{ij} = 1 | s_{ij}^{\text{chemical}}, s_{ij}^{\text{target}}, s_{ij}^{\text{side effect}}) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 * s_{ij}^{\text{chemical}} + \beta_2 * s_{ij}^{\text{target}} + \beta_3 * s_{ij}^{\text{side effect}})}}$$

where for each feature $f \in \{$ chemical, target, and side effect $\}$, the similarity-based drug-disease score s_{ij}^f is defined as

$$s_{ij}^f = \sum_{k \in \mathrm{NN}(i)} \mathrm{sim}^f(i,k) * X(k,j)$$

with $sim^f(i,k)$ being the similarity between two drugs i and k (calculated via Pearson product-moment correlation coefficient as explained above), $\mathrm{NN}(i)$ is the set of 20 most similar drugs to drug i (nearest neighbors in the similarity space), and $X(k,j)$ being an indicator function with values 1 if drug k is a known indication for disease j, and 0 otherwise.

We used the *LogisticRegression* function in Scikit-learn Python package with the L2 regularization option and the default values (inverse regularization strength of 1 and stopping tolerance of 0.0001).

3.4. *Prediction accuracy evaluation*

To assess the prediction performance of the logistic regression classifier, we calculated the area under ROC curve (AUC) using k-fold cross validation scheme (e.g., $k = 2, 5, 10, 20$). We used 2,229 known drug-disease associations as the positive instances and marked all remaining possible associations between 536 drugs and 578 diseases ($536 \times 578 - 2,229 = 307,579$ associations) as negative instances. Following the previous studies, we balanced the data set such that it contained twice as many negative instances as positives.[4,7] Thus, in a k-fold cross validation run, we created k groups containing $2,229/k$ positive instances and $2 \times 2,229/k$ negative instances that were randomly chosen among all negative instances. Each fold was used as the test set once, in which all the remaining folds were used to train the classifier. In order to get robust estimates of the AUC, we repeated the cross validation procedure ten times and recorded the mean and the standard deviation of the AUC values over these runs. Note that, the classifier we built relies on both the similarity and the labels of the training drug-disease associations, as we calculate a drug-disease association score using the most similar 20 drugs and their indication information. We made sure not to use the training information in the test phase and calculated the drug-disease association scores within the training and test folds separately. We used the *roc_curve* and *auc* functions in Scikit-learn Python package to first get the true and false positive rates at various cutoffs and then to calculate the AUC using the trapezoidal rule.

3.5. *Stratified disjoint cross validation for defining non-overlapping drug groups*

To investigate the robustness of the drug-disease association classifier in the case of unseen data, we used a disjoint cross validation scheme, in which none of the drugs in one fold appear in another fold. We created cross validation folds such that all the drugs with the same name were in the same fold by first converting the drug's name into an integer value and then taking the modulo (k) of this value (for k-fold cross validation). To produce different groupings at each run, we added a random integer to the integer value of the drug calculated based on its name. The details of the algorithm are as follows:

D: data set containing drug-disease pairs, c: drug, p: disease,
l: label (1 if c is known to be indicated for p, 0 otherwise), k: number of cross validation folds,
$fold$: dictionary containing the fold index of each drug-disease pair
$i :=$ random($[0, 100]$)
$fold := \{\}$
for each $(c, p, l) \in D$ **do**
  $sum := 0$
  for each $x \in$ characters(c) **do**
    $sum := sum +$ to_integer(x)
  $fold(c, p) :=$ modulo($sum + i, k$)
return $fold$

To preserve the balance between positive and negative instances (stratified cross validation), we first grouped the data set into positive ($D^{l=1}$) and negative ($D^{l=0}$) pairs and applied

the proposed disjoint fold generation algorithm above to each group.

4. Conclusions

Many recent similarity-based drug repurposing studies reported stunningly high prediction performances, suggesting that drugs can be predicted for novel uses almost with perfect accuracy. Yet, there has not been an observable improvement in the drug discovery in the pharma industry over the past years. We suspect this could be *(i)* because similarity-based methods do not provide insights on the mechanism of action of drugs, failing to explain clinical failures due to the lack of efficacy and safety and/or *(ii)* the reported accuracies being unrealistic due to the underlying validation scheme.

To look into various validation schemes and toward increasing the reproducibility in computational drug repurposing research, we provide a Python-based platform encapsulating machine learning algorithms available in Python Scikit-learn package and propose a disjoint cross fold generation method. This platform allows us to easily evaluate the prediction performance of a logistic regression classifier built using drug chemical, target, and side effect similarity under various experimental settings. Using this platform, we investigate the role of the experimental settings in similarity-based drug repurposing studies in producing optimistic prediction accuracies. In particular, we seek to validate the drug repurposing model when it has never seen the drug beforehand. To test this idea, we use a cross validation approach in which the data is split such that none of the drugs in the test set are in the training set. We show that the high success rate of the model drops sharply under such cross validation setting.

Indeed, in many computational biology problems dealing with paired data, such as predicting drug targets, side effects, drug-drug interactions, protein-protein interactions, functional annotations, and disease-genes, researches aim to leverage machine learning methods using similarity between biomolecules. Our findings suggest that failure to take into account the parity in such data sets can produce optimistic prediction accuracies, supporting earlier studies on drug-target and protein-protein interaction prediction.[13,14] We particularly point out the effect of the training set size when the drugs in the training and test sets do not overlap. Hence, we argue that, though useful in highlighting potential unknown drug-disease pairs, similarity-based methods are likely to be ineffective to explore drugs that are not in the nearby pharmacological space, i.e. the drugs with low chemical similarity or for which target and side effect data are not abundant.

Alternatively, systems-level drug discovery approaches can offer insights on the mechanism of action of the drugs by matching gene expression signatures upon drug treatment to compensate the genomic changes caused by the disease[22,23] or exploiting the paths from drug targets to the genes perturbed in the diseases to explain the efficacy of treatments given the interaction network.[24] Nonetheless, these approaches are still at their infancy and their accuracies remain modest,[24,25] leaving room for improvement.

Acknowledgments

The author is grateful to Dr. Patrick Aloy for hosting EG who is supported by EU-cofunded Beatriu de Pinós incoming fellowship from the Agency for Management of University and

Research Grants (AGAUR) of Government of Catalunya. I also would like to thank Zhang and colleagues for making the data used in their analysis online.

References

1. G. Jin and S. T. C. Wong, Toward better drug repositioning: prioritizing and integrating existing methods into efficient pipelines, *Drug Discovery Today* **19**, 637 (May 2014).
2. R. A. Hodos, B. A. Kidd, K. Shameer, B. P. Readhead and J. T. Dudley, In silico methods for drug repurposing and pharmacology, *Wiley Interdisciplinary Reviews: Systems Biology and Medicine* **8**, 186 (May 2016).
3. S. Vilar and G. Hripcsak, The role of drug profiles as similarity metrics: applications to repurposing, adverse effects detection and drug-drug interactions, *Briefings in Bioinformatics* , p. bbw048 (June 2016).
4. A. Gottlieb, G. Y. Stein, E. Ruppin and R. Sharan, PREDICT: a method for inferring novel drug indications with application to personalized medicine, *Mol Syst Biol* **7**, p. 496 (June 2011).
5. J. Li and Z. Lu, A new method for computational drug repositioning using drug pairwise similarity, in *2012 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, October 2012.
6. F. Napolitano, Y. Zhao, V. M. Moreira, R. Tagliaferri, J. Kere, M. D'Amato and D. Greco, Drug Repositioning: A Machine-Learning Approach through Data Integration, *Journal of Cheminformatics* **5**, p. 30 (2013).
7. P. Zhang, P. Agarwal and Z. Obradovic, Computational Drug Repositioning by Ranking and Integrating Multiple Data Sources, in *Machine Learning and Knowledge Discovery in Databases*, eds. H. Blockeel, K. Kersting, S. Nijssen and F. ZeleznyLecture Notes in Computer Science (Springer Berlin Heidelberg, September 2013) pp. 579–594. DOI: 10.1007/978-3-642-40994-3_37.
8. M. Oh, J. Ahn and Y. Yoon, A Network-Based Classification Model for Deriving Novel Drug-Disease Associations and Assessing Their Molecular Actions, *PLOS ONE* **9**, p. e111668 (October 2014).
9. Z. Liu, F. Guo, J. Gu, Y. Wang, Y. Li, D. Wang, L. Lu, D. Li and F. He, Similarity-based prediction for Anatomical Therapeutic Chemical classification of drugs by integrating multiple data sources, *Bioinformatics* **31**, 1788 (June 2015).
10. W. Dai, X. Liu, Y. Gao, L. Chen, J. Song, D. Chen, K. Gao, Y. Jiang, Y. Yang, J. Chen and P. Lu, Matrix Factorization-Based Prediction of Novel Drug Indications by Integrating Genomic Space, *Computational and Mathematical Methods in Medicine* **2015**, p. e275045 (May 2015).
11. H. Luo, J. Wang, M. Li, J. Luo, X. Peng, F.-X. Wu and Y. Pan, Drug repositioning based on comprehensive similarity measures and Bi-Random walk algorithm, *Bioinformatics* , p. btw228 (May 2016).
12. S. Vilar, E. Uriarte, L. Santana, T. Lorberbaum, G. Hripcsak, C. Friedman and N. P. Tatonetti, Similarity-based modeling in large-scale prediction of drug-drug interactions, *Nature Protocols* **9**, 2147 (September 2014).
13. T. Pahikkala, A. Airola, S. Pietil, S. Shakyawar, A. Szwajda, J. Tang and T. Aittokallio, Toward more realistic drugtarget interaction predictions, *Briefings in Bioinformatics* **16**, 325 (March 2015).
14. Y. Park and E. M. Marcotte, A flaw in the typical evaluation scheme for pair-input computational predictions, *Nature methods* **9**, 1134 (December 2012).
15. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, Scikit-learn: Machine Learning in Python, *Journal of Machine Learning Research* **12**, p. 28252830 (October 2011).

16. M. Campillos, M. Kuhn, A.-C. Gavin, L. J. Jensen and P. Bork, Drug Target Identification Using Side-Effect Similarity, *Science* **321**, 263 (July 2008).
17. L. Yang and P. Agarwal, Systematic Drug Repositioning Based on Clinical Side-Effects, *PLOS ONE* **6**, p. e28025 (December 2011).
18. D. S. Wishart, C. Knox, A. C. Guo, D. Cheng, S. Shrivastava, D. Tzur, B. Gautam and M. Hassanali, DrugBank: a knowledgebase for drugs, drug actions and drug targets, *Nucleic Acids Research* **36**, D901 (January 2008).
19. Y. Wang, J. Xiao, T. O. Suzek, J. Zhang, J. Wang and S. H. Bryant, PubChem: a public information system for analyzing bioactivities of small molecules, *Nucleic Acids Research* **37**, W623 (July 2009).
20. M. Kuhn, M. Campillos, I. Letunic, L. J. Jensen and P. Bork, A side effect resource to capture phenotypic effects of drugs, *Molecular Systems Biology* **6**, p. 343 (2010).
21. J. Li, X. Zhu and J. Y. Chen, Building disease-specific drug-protein connectivity maps from molecular interaction networks and PubMed abstracts, *PLoS computational biology* **5**, p. e1000450 (July 2009).
22. J. Lamb, E. D. Crawford, D. Peck, J. W. Modell, I. C. Blat, M. J. Wrobel, J. Lerner, J.-P. Brunet, A. Subramanian, K. N. Ross, M. Reich, H. Hieronymus, G. Wei, S. A. Armstrong, S. J. Haggarty, P. A. Clemons, R. Wei, S. A. Carr, E. S. Lander and T. R. Golub, The Connectivity Map: Using Gene-Expression Signatures to Connect Small Molecules, Genes, and Disease, *Science* **313**, 1929 (September 2006).
23. M. Sirota, J. T. Dudley, J. Kim, A. P. Chiang, A. A. Morgan, A. Sweet-Cordero, J. Sage and A. J. Butte, Discovery and preclinical validation of drug indications using compendia of public gene expression data, *Science Translational Medicine* **3**, p. 96ra77 (August 2011).
24. E. Guney, J. Menche, M. Vidal and A.-L. Barabási, Network-based in silico drug efficacy screening, *Nature Communications* **7**, p. 10331 (February 2016).
25. J. Cheng, L. Yang, V. Kumar and P. Agarwal, Systematic evaluation of connectivity map for disease indications, *Genome Medicine* **6** (December 2014).

EMPOWERING MULTI-COHORT GENE EXPRESSION ANALYSIS TO INCREASE REPRODUCIBILITY

WINSTON A HAYNES[1,2,3], FRANCESCO VALLANIA[1], CHARLES LIU[1,4], ERIKA BONGEN[1], AURELIE TOMCZAK[1,3], MARTA ANDRES-TERRÈ[1], SHANE LOFGREN[1], ANDREW TAM[1], COLE A DEISSEROTH[1,4], MATTHEW D LI[1], TIMOTHY E SWEENEY[1,3], and PURVESH KHATRI[1,3,*]

[1]*Stanford Institute for Immunity, Transplantation, and Infection, Stanford University*
[2]*Biomedical Informatics Training Program, Stanford University*
[3]*Stanford Center for Biomedical Informatics Research, Stanford University*
[4]*Stanford Institutes of Medicine Research Program, Stanford University*
Stanford, CA 94305 USA
[*]*E-mail: pkhatri@stanford.edu*

A major contributor to the scientific reproducibility crisis has been that the results from homogeneous, single-center studies do not generalize to heterogeneous, real world populations. Multi-cohort gene expression analysis has helped to increase reproducibility by aggregating data from diverse populations into a single analysis. To make the multi-cohort analysis process more feasible, we have assembled an analysis pipeline which implements rigorously studied meta-analysis best practices. We have compiled and made publicly available the results of our own multi-cohort gene expression analysis of 103 diseases, spanning 615 studies and 36,915 samples, through a novel and interactive web application. As a result, we have made both the process of and the results from multi-cohort gene expression analysis more approachable for non-technical users.

Keywords: Multi-cohort Analysis; Meta-Analysis; Gene Expression; Reproducibility; Web Application; Software

1. Introduction

Prior to translation of the results of a biological experiment into clinical practice, they must be replicated and validated in multiple independent cohorts. However, the majority of findings fail to validate, leading to a 'reproducibility crisis' in science.[1,2] One of the factors in this lack of reproducibility is that traditional, single cohort studies do not represent the heterogeneity observed in the real world patient population.[3] As a result, observed and reported effects are often specific to a population subset instead of generalizable across the population.

More than two million publicly available gene expression microarrays present novel opportunities to incorporate the real-world heterogeneity observed in patient populations into analysis.[4,5] However, the biological (tissue, treatment, demographics) and technical (experimental protocol, microarray) heterogeneity present in such data poses a daunting challenge for their integration and reuse. Consequently, many tools, which allow reuse of these data, are unable to combine evidence across multiple data sets and place that burden on the end user, leading to under-utilization of these datasets.[6,7]

Previously, we have described a novel multi-cohort analysis framework for integrating multiple heterogeneous datasets to identify robust and reproducible signatures by leveraging the biological and technical heterogeneity in these datasets. We have repeatedly demonstrated the utility of our framework for identifying novel diagnostic and prognostic biomarkers, drug targets, and repurposing FDA-approved drugs in diverse diseases, including organ transplan-

tation, cancer, infection, and neurodegenerative diseases.[8–16] In each of these analyses, we analyzed more than a thousand human samples from more than 10 independent cohorts to generate and validate data-driven hypotheses. Many of these results also been further validated in experimental settings.[8,11,16] These results have further demonstrated the ability of our framework to create "Big Data" by combining multiple smaller studies that are collectively representative of the real word patient population heterogeneity.

We recently published a systematic comparison of gene expression meta-analysis to evaluate existing tools, including GeneMeta, MAMA, MetaDE, ExAtlas, rmeta, and metafor,[17–21] and described best practice guidelines for gene expression meta-analysis.[22] While these existing packages perform both generic and gene expression meta-analysis, none provide coverage of the entire gene expression meta-analysis workflow: downloading data from public repositories, rigorously implementing gene expression meta-analysis best practices, and providing visualizations of the final results.

2. Multi-Cohort Gene Expression Analysis with MetaIntegrator

Despite its demonstrated utility in identifying robust, reproducible, and biologically as well as clinically relevant disease signatures, our multi-cohort analysis framework has previously required manual dataset download, pipeline set up, and visualization generation. To lower this barrier to entry, we have developed MetaIntegrator, an R package that automates most of the multi-cohort analysis framework. Our package guides the user from data download to execution of statistical analysis to evaluation of the results [Figure 1].

2.1. *Data Processing*

The first step in the multi-cohort analysis is downloading the requisite experimental information, notably the class labels (case or control), the gene expression data, and any interesting phenotypic information about the samples. Since we have found that most users will download data from the NCBI's Gene Expression Omnibus (GEO), we have integrated an automatic downloading and processing of GEO data into our analysis pipeline. MetaIntegrator will automatically download the expression data and all available annotations, perform sanity checks that the data have been appropriately normalized, and compile the data into the MetaIntegrator object format.

2.2. *Multi-cohort Analysis*

2.2.1. *Combining effect sizes*

Our meta-analysis approach computes an Hedges g effect size for each gene in each dataset defined as:

$$g = J\frac{\bar{X}_1 - \bar{X}_0}{\sqrt{\frac{(n_1-1)S_1^2+(n_0-1)S_0^2}{n_1+n_0-2}}} \quad (1)$$

where $\bar{X}_1$ and $\bar{X}_0$ are the average expression for cases and controls, S_1 and S_0 are the standard deviations for cases and controls, and n_1 and n_0 are the number of cases and controls.[8,23] J is

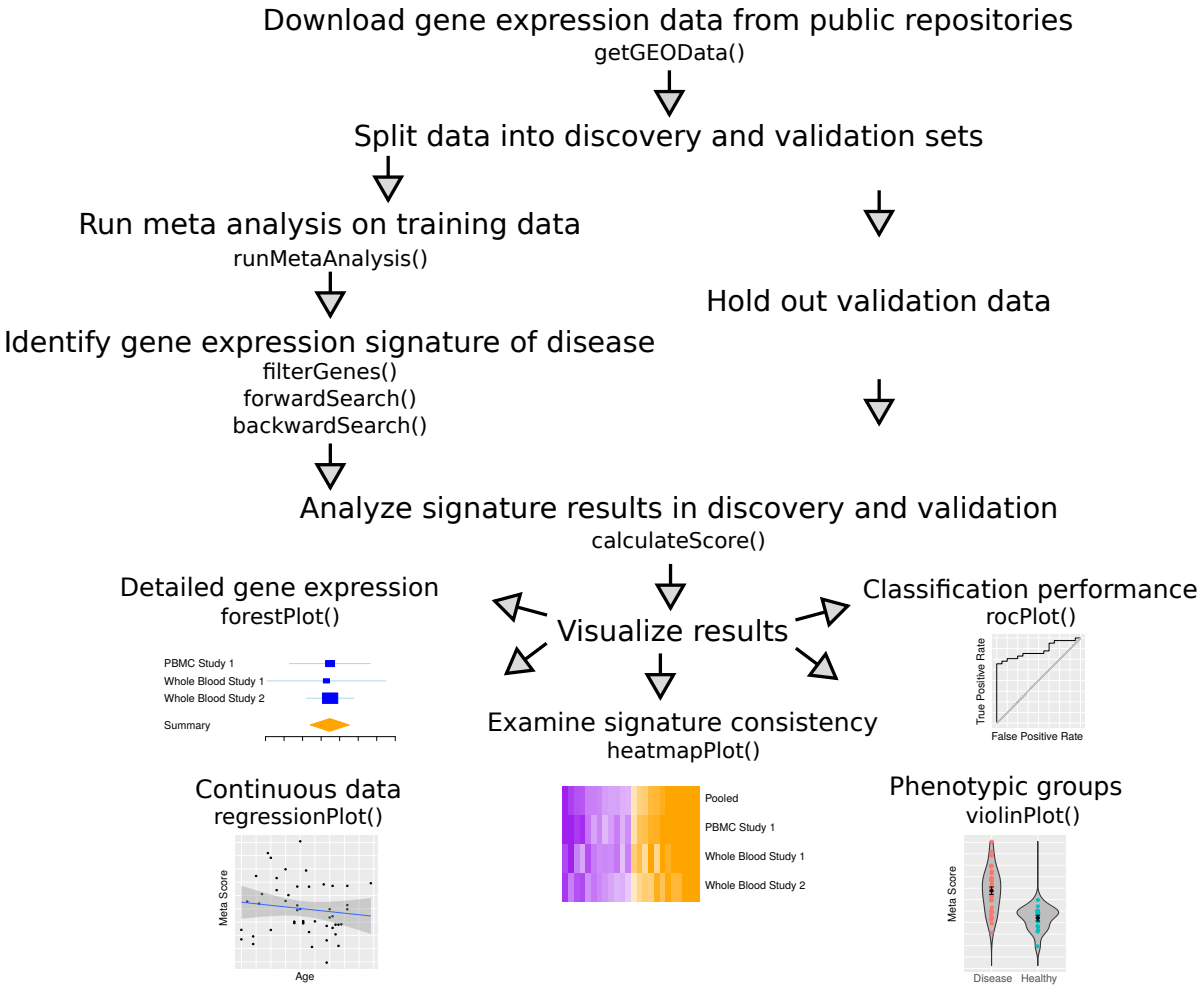


Fig. 1. Gene expression meta-analysis workflow with MetaIntegrator utility functions.

the Hedges' g correction factor, which is computed as:

$$J = 1 - \frac{3}{4df - 1} \tag{2}$$

where df are the degrees of freedom.

To pool these effect sizes across datasets, the summary effect size g_s is computed using a random effect model as:

$$g_s = \frac{\sum_i^n W_i g_i}{\sum_i^n W_i} \tag{3}$$

where n is the number of studies, g_i is the Hedges' g of that gene within dataset i, W_i is a weight equal to $1/(V_i + T^2)$, V_i is the variance of that gene within a given dataset i, and T^2 is the inter-dataset variation as estimated by the DerSimonian-Laird method.[23,24] The standard error for the summary effect size is $SE_{g_s} = \sqrt{\frac{1}{\sum_i^n W_i}}$. Given g_s and SE_{g_s}, we calculate a p-value

based on a standard normal distribution and perform a Benjamini-Hochberg FDR correction for multiple hypothesis testing across all genes.[25]

2.2.2. *Heterogeneity of effect size*

We calculate Cochrane's Q value for evaluating heterogeneity of effect size estimates between studies:

$$Q = \sum_{i=1}^{n} W_i \left(g_i - g_s\right)^2 \tag{4}$$

where W_i, g_i, and g_s are the same as above.[23] The p-value of Cochrane's Q is calculated against a chi-squared distribution and adjusted for multiple hypothesis testing using the Benjamini-Hochberg FDR method across all genes.[25] A statistically significant Cochrane's Q indicates significant heterogeneity of effect sizes between studies.

2.2.3. *Combining p-values*

We use Fisher's method for combining p-values across studies.[26] We calculate the log sum of p-values that each gene is up-regulated as:

$$F_{\text{up}} = -2 \sum_{i=1}^{n} \log(p_i) \tag{5}$$

where n is the number of studies and p_i is the t-test p-value that the gene of interest is up-regulated in study i. Similarly, we calculate F_{down} as the log-sum of p-values that each gene is down-regulated.

For each gene, we calculate the p-value of F_{up} and F_{down} under a chi-squared distribution and perform a Benjamini-Hochberg FDR correction across all genes.[25]

2.3. *Signature Selection*

Once meta-analysis is performed, a subset of genes must be identified as the disease signature. MetaIntegrator allows the user to identify these genes by varying the filtering parameters based on gene effect size, effect size false discovery rate, Fisher's method false discovery rate, heterogeneity of effect size, and the number of studies in which the gene was present. In order to avoid disproportionate influence of a single study, MetaIntegrator allows the user only include genes which were similarly significant across all leave-one-dataset-out analyses. By varying these criterion, the user may control whether they identify a larger set of genes, which may be ideal for understanding molecular pathogenesis and identifying drug targets, or a smaller set of genes, which may be optimal developing a parsimonious clinical diagnostic.

For users that are particularly interested in developing a powerful diagnostic, we have integrated forward and backward search, which reduce gene set size to optimize the area under the receiver operating characteristic curve on the training data.[10]

2.4. *Score Calculation*

For a set of signature genes, a signature score can be computed for every sample, i, as:

$$S_i = \left(\prod_{\text{gene} \in \text{pos}} x_i(\text{gene}) \right)^{\frac{1}{\|\text{pos}\|}} - \left(\prod_{\text{gene} \in \text{neg}} x_i(\text{gene}) \right)^{\frac{1}{\|\text{neg}\|}} \quad (6)$$

where pos and neg are the sets of positive and negative genes, respectively, and x_i(gene) is the expression of any particular gene in sample i (a positive score indicates an association with cases and a negative score with controls). This score S_i is normalized to a z-score to center the samples for each study around zero.

2.5. *Visualization*

With scores calculated for each sample, we are able to visualize comparisons of cases vs. controls, regression of continuous variables against the score, and consistency of gene expression across datasets. Some of the built in visualizations, in counter-clockwise order from Figure 1:

- **Forest plots.** Examine the effect sizes and standard errors for a single gene across studies, including the summary effect size.
- **Regression plots.** Evaluate the relationship of the signature score with continuous variables like clinical severity and time.
- **Heatmap plots.** Observe consistency of differential expression for all signature genes across studies.
- **Violin plots.** Compare signature scores across categorical variables like disease subtypes, treatment protocols, and demographic groups.
- **ROC plots.** Evaluate classification performance for signature score on a single dataset in terms of specificity and sensitivity.

3. Data-Driven Biological Hypotheses with MetaSignature

We have created MetaSignature (`http://metasignature.stanford.edu`), a web application which empowers researchers to generate data-driven hypotheses by enabling access to the results of our multi-cohort gene expression analysis framework. We focused on enabling intuitive data access for researchers with specific interest in either a disease, a gene, or several genes, while requiring little or no analytic background.

3.1. *Data*

Thus far, we have aggregated 615 gene expression studies composed of more than 35,000 human samples with approximately 1.5 billion data points from 103 diseases, a number which we will continue to grow. For each disease, we applied our multi-cohort analysis approach to compute the gene expression differences between the manually curated cases and controls. To perform these multi-cohort analyses, we searched for relevant studies in GEO, identified cases and controls in every study, and calculated disease effect sizes using the MetaIntegrator R

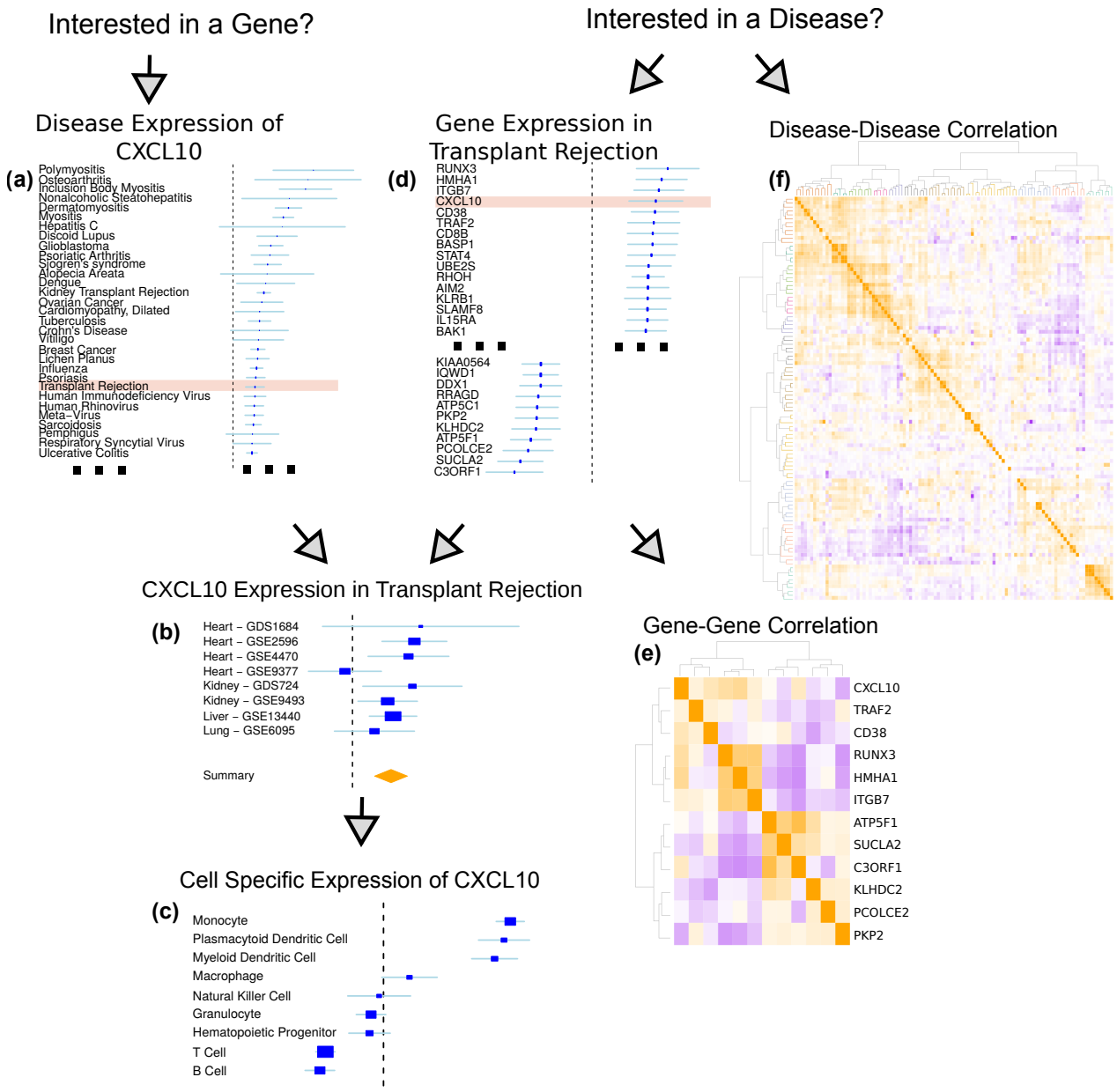


Fig. 2. Diagram of the MetaSignature web application.

package. We stored the multi-cohort analysis results in a MySQL database for rapid retrieval. As more studies are incorporated into our database, we recalculate the disease summary effect sizes.

3.2. *Gene-centric Analysis*

For researchers that are interested in the expression of a particular gene, MetaSignature provides visualizations that allow researchers to quickly identify the diseases in which specified gene is most differentially expressed [Figure 2a], study-level data of the gene expression in

particular diseases [Figure 2b], and cell type-specific gene expression patterns [Figure 2c].

For instance, consider a researcher who has developed a drug, such as atorvastatin, that effectively reduces plasma levels of *CXCL10*, and seeks to identify the most promising clinical applications. Using MetaSignature, she determines *CXCL10* is significantly up-regulated in transplant rejection [Figure 2a]. A drilldown further identifies eight separate studies that have measured *CXCL10* in transplant rejection, indicating a highly positive effect size in all except one of these studies [Figure 2b]. The researcher further observes that *CXCL10* is up-regulated in monocytes, compared to other immune cell types. [Figure 2c]. Taken together, these findings would motivate a clinical investigation of the use of a *CXCL10* inhibitor, such as atorvastatin, in monocytes of patients at risk for transplant rejection. We have already verified this data-driven hypothesis in mouse models and patient electronic health records, where, in both cases, atorvastatin increases graft survival.[8]

Beyond single gene analysis, MetaSignature empowers users to examine gene sets in terms of correlation of those genes based on their disease effect sizes [Figure 2e] and correlation of diseases based on expression of that set of genes [similar to Figure 2f]. These visualizations enable dissection of positively- and negatively-correlated members of gene families.

3.3. *Disease-centric Analysis*

If a researcher is more interested in a particular disease, then MetaSignature enables identification of genes that are most up- or down-regulated in that disease [Figure 2d] and exploration of that disease's relationship to other diseases based on gene expression [Figure 2f]. When we compute disease-disease correlation based on gene expression data, we observe clustering patterns that map to established disease categories.

To follow our example from the gene-centric analysis, consider a researcher who is interested in improving transplant rejection outcomes. To gain a global understanding of transplant rejection, the researcher observes that transplant rejection falls into a cluster of inflammatory diseases, including discoid lupus, Crohn's disease, and interstitial cystitis [Figure 2f]. By examining the transplant rejection expression data in MetaSignature, he or she would recognize that *CXCL10*, a chemokine important in inflammatory response, is one of the most up-regulated genes in transplant rejection [Figure 2d].[27] After verifying that this observation is consistent across studies [Figure 2b], the researcher identifies that *CXCL10* is a reasonable target for therapeutic inhibition in transplant rejection. Looking at other genes which are up- and down-regulated in transplant rejection, he or she recognizes that CXCL10 expression is in a positively correlated with several other genes, including *TRAF2* and *CD38* [Figure 2e]. Collectively from these observations, the researcher has learned that transplant rejection is related to inflammatory diseases, which is consistent with the observed up-regulation of CXCL10, an inflammatory chemokine. As noted in the gene-centric analysis above, we have observed increased graft survival through administration of atorvastatin.[8]

4. Discussion

The reproducibility crisis in biomedical research has led to erroneous conclusions and wasted resources. Here, we present a vertically integrated platform that can both assist with gene

expression multi-cohort analysis (MetaIntegrator) and provide aggregated results for users who wish to rapidly test hypotheses (MetaSignature). By leveraging the growing public data available for study, this new resource can drastically reduce the time and effort for biological hypothesis testing across numerous studies and diseases. While many software packages exist for similar analyses,[17–21] ours offers simple, custom software for plotting and analysis, automated downloading of data from GEO, and integration to the MetaSignature database.

Our package is complementary to the recently published OMiCC platform, which enables curation and meta-analysis of GEO studies.[28] OMiCC relies on the RankProd package for performing meta-analysis using rank-based statistics for identifying differentially expressed genes.[29] While others have provided thorough comparisons of the different meta-analysis methods, the most notable difference between RankProd and MetaIntegrator is that rank-based statistics fail to produce a summary effect size across multiple studies.[30,31] By leveraging our MetaIntegrator package, OMiCC could produce differential gene expression profiles across multiple studies instead of internal to single studies.

Although MetaIntegrator is currently applicable to microarray gene expression data, we plan to expand the MetaIntegrator package to handle the count data which is generated by RNAseq experiments. Additionally, we intend to enable download from additional data repositories including ArrayExpress and, once RNAseq processing is implemented, Sequence Read Archive.[?,5]

Our work promises to increase reproducibility of research for both data analysts and wet lab researchers. For data analysts, we have made multi-cohort gene expression analysis publicly available through a straightforward R package. By performing integrative, multi-cohort analyses, these analysts will generate more reproducible results. For wet lab researchers, we are empowering data-driven hypotheses prior to experimentation. Rather than performing broad assays to identify disease related genes, researchers can focus on performing targeted experiments on genes which are reproducible across cohorts.

5. Package and Source Code Distribution

The MetaIntegrator R package, including an introductory vignette, may be installed using the following command in R:

```
install.packages("MetaIntegrator")
```

The source code for MetaIntegrator is available at:
`https://cran.rstudio.com/web/packages/MetaIntegrator/`
MetaSignature was developed using R and Shiny and is hosted at:
`http://metasignature.stanford.edu/`

6. Acknowledgements

We thank Alex Schrenchuk for computer support. WAH is funded by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-114747. FV is funded by the National Institute of Health K12 Career Award 5K12HL120001-02. MAT is funded by La Caixa Foundation. EB is funded by Gabilan Fellowship. PK is funded by the National

Institute of Allergy and Infectious Diseases grants 1U19AI109662, U19AI057229, U54I117925, and U01AI089859.

References

1. J. P. A. Ioannidis, *PLoS medicine* **2**, p. e124 (August 2005).
2. M. Baker, *Nature* , 452 (2016).
3. J. P. Ioannidis, E. E. Ntzani, T. A. Trikalinos and D. G. Contopoulos-Ioannidis, *Nature Genetics* **29**, 306 (November 2001).
4. R. Edgar, *Nucleic Acids Research* **30**, 207 (January 2002).
5. A. Brazma, H. Parkinson, U. Sarkans, M. Shojatalab, J. Vilo, N. Abeygunawardena, E. Holloway, M. Kapushesky, P. Kemmeren, G. G. Lara, A. Oezcimen, P. Rocca-Serra and S.-A. Sansone, *Nucleic Acids Research* **31**, 68 (January 2003).
6. J. M. Engreitz, R. Chen, A. A. Morgan, J. T. Dudley, R. Mallelwar and A. J. Butte, *Bioinformatics* **27**, 3317 (December 2011).
7. R. Petryszak, T. Burdett, B. Fiorelli, N. A. Fonseca, M. Gonzalez-Porta, E. Hastings, W. Huber, S. Jupp, M. Keays, N. Kryvych, J. McMurry, J. C. Marioni, J. Malone, K. Megy, G. Rustici, A. Y. Tang, J. Taubert, E. Williams, O. Mannion, H. E. Parkinson and A. Brazma, *Nucleic Acids Research* **42**, D926 (January 2014).
8. P. Khatri, S. Roedder, N. Kimura, K. De Vusser, A. A. Morgan, Y. Gong, M. P. Fischbein, R. C. Robbins, M. Naesens, A. J. Butte and M. M. Sarwal, *The Journal of experimental medicine* **210**, 2205 (October 2013).
9. R. Chen, P. Khatri, P. K. Mazur, M. Polin, Y. Zheng, D. Vaka, C. D. Hoang, J. Shrager, Y. Xu, S. Vicent, A. J. Butte and E. A. Sweet-Cordero, *Cancer Research* **74**, 2892 (May 2014).
10. T. E. Sweeney, A. Shidham, H. R. Wong and P. Khatri, *Science Translational Medicine* **7**, p. 287ra71 (May 2015).
11. M. Andres-Terre, H. M. McGuire, Y. Pouliot, E. Bongen, T. E. Sweeney, C. M. Tato and P. Khatri, *Immunity* **43**, 1199 (December 2015).
12. M. D. Li, T. C. Burns, A. A. Morgan and P. Khatri, *Acta neuropathologica communications* **2**, p. 93 (January 2014).
13. P. K. Mazur, N. Reynoird, P. Khatri, P. W. T. C. Jansen, A. W. Wilkinson, S. Liu, O. Barbash, G. S. Van Aller, M. Huddleston, D. Dhanak, P. J. Tummino, R. G. Kruger, B. A. Garcia, A. J. Butte, M. Vermeulen, J. Sage and O. Gozani, *Nature* **advance on** (May 2014).
14. P. K. Mazur, A. Herner, S. S. Mello, M. Wirth, S. Hausmann, F. J. Sánchez-Rivera, S. M. Lofgren, T. Kuschma, S. A. Hahn, D. Vangala, M. Trajkovic-Arsic, A. Gupta, I. Heid, P. B. Noël, R. Braren, M. Erkan, J. Kleeff, B. Sipos, L. C. Sayles, M. Heikenwalder, E. Heß mann, V. Ellenrieder, I. Esposito, T. Jacks, J. E. Bradner, P. Khatri, E. A. Sweet-Cordero, L. D. Attardi, R. M. Schmid, G. Schneider, J. Sage and J. T. Siveke, *Nature Medicine* **21**, 1163 (September 2015).
15. T. E. Sweeney, L. Braviak, C. M. Tato and P. Khatri, *The Lancet Respiratory Medicine* **4**, 213 (2016).
16. T. E. Sweeney, H. R. Wong and P. Khatri, *Science translational medicine* **8**, p. 346ra91 (July 2016).
17. L. Lusa, R. Gentleman and M. Ruschhaupt, *GeneMeta: MetaAnalysis for High Throughput Experiments*.
18. I. Ihnatova., *MAMA: Meta-Analysis of MicroArray*, (2013).
19. T. Lumley, *rmeta: Meta-analysis*, (2012).
20. X. Wang, D. D. Kang, K. Shen, C. Song, S. Lu, L. C. Chang, S. G. Liao, Z. Huo, S. Tang, Y. Ding, N. Kaminski, E. Sibille, Y. Lin, J. Li and G. C. Tseng, *Bioinformatics* **28**, 2534 (2012).

21. A. A. Sharov, D. Schlessinger and M. S. H. Ko, *Journal of Bioinformatics and Computational Biology* **13**, p. 1550019 (2015).
22. T. E. Sweeney, W. A. Haynes, F. Vallania, J. P. Ioannidis and P. Khatri, *Nucleic acids research* , p. gkw797 (September 2016).
23. M. Borenstein, L. V. Hedges, J. P. T. Higgins and H. R. Rothstein, *Introduction to Meta-Analysis* 2009.
24. R. DerSimonian and R. Kacker, *Contemporary Clinical Trials* **28**, 105 (2007).
25. Y. Benjamini and Y. Hochberg, *Journal of the Royal Statistical Society. Series B (Methodological)* **57**, 289 (1995).
26. R. Fisher, *Statistical methods for research workers*, 1925).
27. L. F. Neville, G. Mathiak and O. Bagasra, *Cytokine & Growth Factor Reviews* **8**, 207 (September 1997).
28. N. Shah, Y. Guo, K. V. Wendelsdorf, Y. Lu, R. Sparks and J. S. Tsang, *Nature Biotechnology* (June 2016).
29. F. Hong, R. Breitling, C. W. McEntee, B. S. Wittner, J. L. Nemhauser and J. Chory, *Bioinformatics (Oxford, England)* **22**, 2825 (November 2006).
30. L.-C. Chang, H.-M. Lin, E. Sibille and G. C. Tseng, *BMC bioinformatics* **14**, p. 368 (2013).
31. A. Ramasamy, A. Mondry, C. C. Holmes and D. G. Altman, *PLoS medicine* **5**, p. e184 (September 2008).

RABIX: AN OPEN-SOURCE WORKFLOW EXECUTOR SUPPORTING RECOMPUTABILITY AND INTEROPERABILITY OF WORKFLOW DESCRIPTIONS

GAURAV KAUSHIK[†]
Seven Bridges Genomics
1 Main Street, Cambridge, MA 02140, USA
Email: gaurav@sevenbridges.com

SINISA IVKOVIC
Seven Bridges Genomics
Omladinskih brigada 90g, Belgrade, Republic of Serbia
Email: sinisa.ivkovic@sevenbridges.com

JANKO SIMONOVIC
Seven Bridges Genomics
Omladinskih brigada 90g, Belgrade, Republic of Serbia
Email: janko.simonovic@sevenbridges.com

NEBOJSA TIJANIC
Seven Bridges Genomics
Omladinskih brigada 90g, Belgrade, Republic of Serbia
Email: boysha@sevenbridges.com

BRANDI DAVIS-DUSENBERY
Seven Bridges Genomics
1 Main Street, Cambridge, MA 02140, USA
Email: brandi@sevenbridges.com

DENIZ KURAL
Seven Bridges Genomics
1 Main Street, Cambridge, MA 02140, USA
Email: deniz.kural@sevenbridges.com

As biomedical data has become increasingly easy to generate in large quantities, the methods used to analyze it have proliferated rapidly. Reproducible and reusable methods are required to learn from large volumes of data reliably. To address this issue, numerous groups have developed workflow specifications or execution engines, which provide a framework with which to perform a sequence of analyses. One such specification is the Common Workflow Language, an emerging standard which provides a robust and flexible framework for describing data analysis tools and workflows. In addition, reproducibility can be furthered by executors or workflow engines which interpret the specification and enable additional features, such as error logging, file organization, optim[1]izations to computation and job scheduling, and allow for easy computing on large volumes of data. To this end, we have developed the Rabix Executor[a], an open-source workflow engine for the purposes of improving reproducibility through reusability and interoperability of workflow descriptions.

[1]This project has been funded in whole or in part with Federal funds from the National Cancer Institute, National Institutes of Health, Department of Health and Human Services, under Contract No. HHSN261201400008C.
[†] Corresponding author
[a]The Rabix Executor is available on GitHub: http://github.com/rabix/bunny

1. Introduction

Reproducible analyses require the sharing of data, methods, and computational resources.[1] The probability of reproducing a computational analysis is increased by methods that support replicating each analysis and the capability to reuse code in multiple environments. In recent years, the practice of organizing data analysis via computational workflow engines or accompanying workflow description languages has surged in popularity as a way to support the reproducible analysis of massive genomics datasets.[2,3] Robust and reliable workflow systems share three key properties: flexibility, portability, and reproducibility. Flexibility can be defined as the ability to gracefully handle large volumes of data with multiple formats. Adopting flexibility as a design principle for workflows ensures that multiple versions of a workflow are not required for different datasets and a single workflow or pipeline can be applied in many use cases. Together, these properties reduce the software engineering burden accompanying large-scale data analysis. Portability, or the ability to execute analyses in multiple environments, grants researchers the ability to access additional computational resources with which to analyze their data. For example, workflows highly customized for a particular infrastructure make it challenging to port analyses to other environments and thus scale or collaborate with other researchers. Well-designed workflow systems must also support reproducibility in science. In the context of workflow execution, computational reproducibility (or recomputability) can be simply defined as the ability to achieve the same results on the same data regardless of the computing environment or when the analysis is performed. Workflows and the languages that describe them must account for the complexity of the information being generated from biological samples and the variation in the computational space in which they are employed. Without flexible, portable, and reproducible workflows, the ability for massive and collaborative genomics projects to arrive at synonymous or agreeable results is limited.[4,5]

Biomedical or genomics workflows may consist of dozens of tools with hundreds of parameters to handle a variety of use cases and data types. Workflows can be made more flexible by allowing for transformations on inputs during execution or incorporating metadata, such as sample type or reference genome, into the execution. They can allow for handling many use cases, such as dynamically generating the appropriate command based on file type or size, without needing to modify the workflow description to adjust for edge cases. Such design approaches are advantageous as they alleviate the software engineering burden and thus the accompanying probability of error associated with executing extremely complex workflows on large volumes of data. In addition, as the complexity of an individual workflow increases to handle a variety of use cases or criteria, it becomes more challenging to optimally compute with it. For example, analyses may incorporate nested workflows, business logic, memoization or the ability to restart failed workflows, or require parsing of metadata -- all of which compound the challenges in optimizing workflow execution.

As a result of the increasing volume of biomedical data, analytical complexity, and the scale of collaborative initiatives focused on data analysis, reliable and reproducible analysis of biomedical data has become a significant concern. Workflow descriptions and the engines that interpret and execute them must be able to support a plethora of computational environments and ensure reproducibility and efficiency while operating across them. It is for this reason that we have developed the Rabix Executor (on GitHub as Project "Bunny")[a], an open-source workflow engine designed to support computational reproducibility/recomputability through the use of standard workflow descriptions, a software model that supports metadata integration, provenance over file organization, the ability to reuse workflows efficiently, and which combines an array of optimizations used separately in existing workflow execution methods.[6–12]

For the 1.0 release of the Rabix Executor (or Rabix), we've focused on supporting the Common Workflow Language (CWL), an open, community-driven specification for describing tools and workflows with a focus on features that support reproducibility.[2] The Common Workflow Language is used to describe individual "processes" or "applications", which can be either a single tool or an entire workflow. Workflows are described as a series of "steps," each of which is a single tool or another, previously-described workflow. Each step in the workflow has a set of "ports" which represent data elements that are either inputs or outputs of the tool. A single port represents a specific data element that is required for execution of the tool or is the result of its execution. For data elements that are passed between applications, there must be an output port from the upstream application and a complementary input port on the downstream application.

CWL is designed to be extensible, so the specification may grow based on the community's needs. However, the software model for Rabix was designed with an abstract workflow execution model to anticipate support for additional workflow languages or syntax used by other workflow engines.

2. Software model used by Rabix to interpret and compute workflows

The Rabix Executor allows users to execute applications described by a workflow description language. First, the workflow description is submitted to the engine. Then, the Rabix engine interprets the workflow description and translates it into discrete computational processes or "jobs." Finally, the jobs are queued to a backend or computational infrastructure, such as a local machine, cluster, or cloud instances, for scheduling and execution. Each component of the executor (frontend, bindings, engine, queue, backend) is abstracted from each other to enable complete modularity; Developers are able to design custom frontends (e.g. command line or graphical user interface), bindings for the engine to parse different workflow languages, use the queuing protocol of their choice, and submit computational jobs to different backends. This flexible software model means that Rabix can be modified to perform data analysis on many different infrastructures as desired by the user or developer and achieve identical results or incorporate tools described by different languages or syntaxes into a single workflow.

3. Abstract representation of data analysis workflows in Rabix

Computational workflows are frequently understood as a directed acyclic graph (DAG)[3,13,14], a kind of finite graph which contains no cycles and which must be traversed in a specific direction. In this representation, each node is either an individual executable command, a "nested" workflow, or a set of commands that can be executed in parallel. The edges in the DAG represent execution variables (data elements such as files or parameters) which pass from upstream nodes to downstream ones.

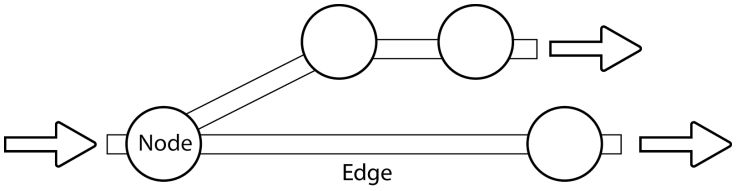


Figure 1. Illustration of a directed acyclic graph (DAG). The DAG may be traversed from left-to-right, moving from node-to-node along the edges that connect them.

Workflows can be described as machine-readable serialized data objects in either a general-purpose programming language (GPL), domain-specific language (DSL), or serialized object models for workflow

description.[2,9,15] For example, an object model-based approach may describe the steps in a workflow in JSON format with a custom syntax. This workflow description can then be parsed by an engine or executor to create the DAG representation of the workflow. The executor may then translate the directions for workflow execution to actionable jobs in which data is analyzed on a computational infrastructure, such as a cloud computing instance, a high-performance computing cluster, or a personal computer.

A primary design constraint of the Rabix executor is to abstract components of a workflow to a data model that is comprehensive enough to allow for mapping the syntax of different workflow systems, whether they are DSLs or serialized data objects. In this way, tools and workflows from different systems can be used together in a single workflow.

3.1. *General structure of a workflow execution*

There are three general steps in preparing a workflow for execution: interpretation of a machine-readable workflow description, generation of the workflow DAG, and finally decomposition into individual jobs that can be scheduled for execution. At the beginning of execution, a workflow engine or interpreter is provided with the workflow description and the required inputs for execution of the workflow, such as parameters and file paths (Fig. 2a). The workflow description object is then parsed and a DAG is created (Fig. 2b), which contains the initial set of nodes and edges required for computation.

In addition to representing the steps in the workflow as a DAG (Fig. 2c), certain workflow ontologies model computational jobs as a composite (tree) pattern in which there are “parent nodes” (workflows), which can contain multiple executables or “leaf nodes” or other "parent" nodes (Fig. 2d).[16–20] The Rabix engine extends this model by generalizing “parent” nodes to include groups of jobs, such as when parallelization is possible at that node. It is important to note that the “parent-child” terminology is also applied to relations between individual workflow nodes by the Toil project, an executor which can also interpret Common Workflow Language.[10] However, Rabix uses these terms to refer to computational "jobs" and "subjobs", e.g. a “nested” workflow node is a child of a workflow and can be decomposed into an array of “subjobs”. The engine handles the "execution" or parsing of these parent jobs, while leaves are queued for scheduling and execution on a backend. This model allows for more efficient resolution of DAG features such as nodes in which steps can be parallelized or are nested. It also maintains a one-to-one mapping between the internal DAG representation and the workflow description supplied by the author.

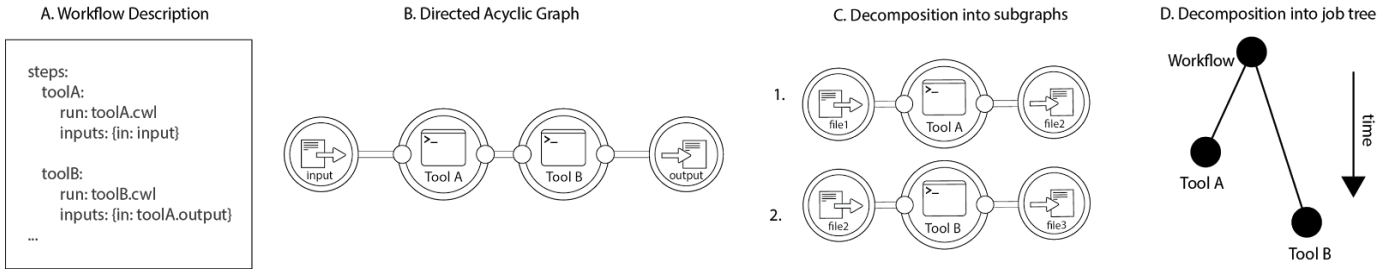


Figure 2. The process of parsing a workflow description. **A.** The machine-readable document is interpreted, from which **B.** a DAG is produced. From the DAG, **C.** subgraphs representing computational jobs that can be sent to backends for scheduling/execution and **D.** a job tree is resolved, which identifies “parent” and “leaf” nodes. Each leaf represents an individual job.

4. Optimization of CWL workflows via DAG transformations

The Rabix Executor began its development by examining how to interpret Common Workflow Language and interoperate on different versions or earlier drafts, in a way that is extensible to future versions and other workflow syntaxes. Rabix currently supports tools and workflows described in CWL Draft 2, Draft 3, and version 1.0, either individually or in combination.

When a CWL workflow is represented as a DAG, applications become nodes and edges indicate the flow of data elements between ports of linked tools. In the case of a simple workflow, there are no possible transformations of the DAG; each node represents a single command line execution and all data elements are simply passed from tool-to-tool as-is (Fig. 3).

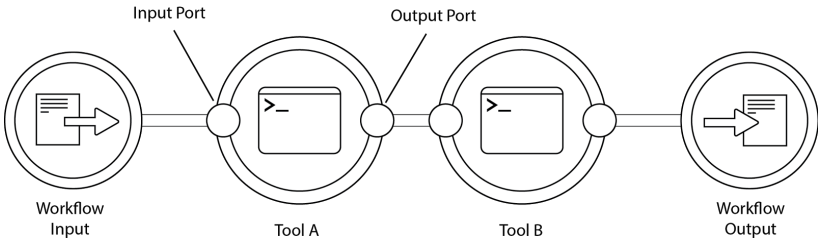


Figure 3. A DAG created from a workflow described by the Common Workflow Language which contains two tools (**A, B**). Tools have input and output ports, which define discrete data elements that are passed downstream along the edges of the DAG.

Additionally, CWL workflows can be designed such that data elements and the execution itself can be transformed during runtime. Developers are given several options for describing workflows which can enhance their utility and flexibility in handling biomedical data analysis:

1. The ability to generate “dynamic expressions” or transformations on data elements, inputs, outputs, and other command line arguments.
2. The ability to perform “scatter/gather I/O (input/output)”, also known as vectored I/O, in which execution of the input data can be parallelized based on specific criteria. A common genomics use case for this is performing an analysis per chromosome, in which the set of chromosomes is delivered to a node as an array (e.g. [1, 2, 3, X]).
3. The ability to nest workflows within workflows, which allows for rapid composition of complex workflows and the ability to quickly reuse existing code.

4.1 *Rabix uses a custom data model and port-level inspection for workflow execution*

Though CWL provides a specification for how to describe the execution of tools and workflows, the exact way in which these features are implemented is left entirely to the execution engine that is interpreting it. Therefore, the Rabix engine has been designed to handle CWL descriptions with two optimizations:

1. Reacting to "port ready" events rather than "job done" events. “Port ready” is a state triggered by the evaluation of data elements produced by a port, whereas “job done” refers to *all* ports of a node being evaluated. In this approach, possible downstream executions are triggered if the edges leading to it are resolved. This allows further dynamic transformations of the DAG to optimize for when all prerequisites for downstream jobs are ready.
2. Reacting to "port ready" events from dynamically created subjobs and rewiring them to their final destinations, possibly creating and running subjobs before their parent fully evaluated (referred to as “look-ahead” method).

These functionalities enable the Rabix engine to create additional edges and nodes as needed, in order to decompose the workflow DAG as early as possible, allowing downstream jobs to be scheduled as soon as actual prerequisites are met.

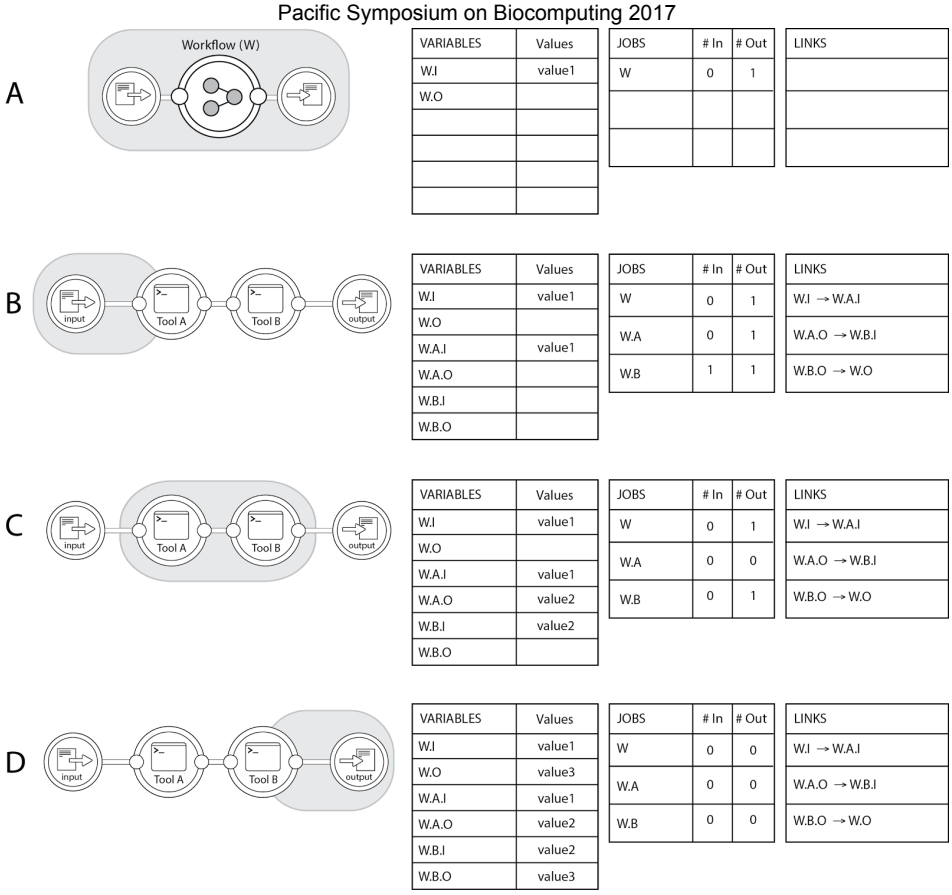
The workflow DAG is stored in three tables, Variables, Jobs, and Links, which are accessed when a port value is updated. The Variables table contains the ports and their explicit values. The Jobs table stores each node of the workflow and a counter for the inputs and outputs that have been evaluated at that node. The Links table stores the edges in the DAG that is traversed.

As compared to other CWL execution models[2,10], computational events are triggered by "port" events instead of "job" events. In other words, when a port is evaluated, this triggers the executor to scan or update these tables in the following order: Variables, Jobs, Links. Any node for which all input ports are now evaluated is then executed.

Suppose for example, Rabix is executing the workflow in Figure 4. The engine will first parse the workflow description as a workflow DAG with two variables (W.I, W.O; Fig. 4a), which are yet to be evaluated. Additionally, there are two ports (#In, #Out), an input and an output. Next, the engine inspects the contents of the workflow (Fig. 4b) and is able to see the following steps: Tool A, Tool B, each of their ports, and the link between each step within scope.

After this, any known values are curried downstream through their links. The input for the workflow (W.I) is curried to Tool A through the link that has been identified between the two (W.I → W.I.A). The input job counter (#In) for Tool A is decremented to 0, thereby triggering an input event where a job (execution of Tool A with value1) is distributed to a backend for computation. The engine now waits for an event in which the output of Tool A (W.A.O) is reported as value.

Once the output for the job is evaluated and reported to the engine (value2), an output event is triggered. The output port for W.A is decremented to 0, the link from W.A.O to W.B.I is traversed, and W.B.I is evaluated as value2. This reduces the #In counter for W.B to 0 in the Jobs table and triggers a job, the execution of Tool B with its input (Fig. 4c). The execution finally concludes until the input port counter for W reaches 0 and W.O is evaluated (Fig. 4d).



VARIABLES	Values
W.I	value1
W.O	

JOBS	# In	# Out
W	0	1

LINKS

VARIABLES	Values
W.I	value1
W.O	
W.A.I	value1
W.A.O	
W.B.I	
W.B.O	

JOBS	# In	# Out
W	0	1
W.A	0	1
W.B	1	1

LINKS
W.I → W.A.I
W.A.O → W.B.I
W.B.O → W.O

VARIABLES	Values
W.I	value1
W.O	
W.A.I	value1
W.A.O	value2
W.B.I	value2
W.B.O	

JOBS	# In	# Out
W	0	1
W.A	0	0
W.B	0	1

LINKS
W.I → W.A.I
W.A.O → W.B.I
W.B.O → W.O

VARIABLES	Values
W.I	value1
W.O	value3
W.A.I	value1
W.A.O	value2
W.B.I	value2
W.B.O	value3

JOBS	# In	# Out
W	0	0
W.A	0	0
W.B	0	0

LINKS
W.I → W.A.I
W.A.O → W.B.I
W.B.O → W.O

Figure 4. The algorithm as it is traversed. **A.** The engine interprets the top-level of the workflow description and **B.** inspects the contents of the workflow node and determines the DAG structure and links between each step (edges). The currying of value1 from the workflow input to the input of Tool A triggers an input event, where a job (analysis of Tool A with its inputs) is sent to a backend node. **C.** The execution continues and the engine traverses the DAG. **D.** The workflow is completed when the output of the final tool (W.B.O., value3) is curried to the overall workflow output (W.O). The port counters allow the engine to track when nodes are ready to be executed even if upstream jobs are only partially completed.

In the case where the engine is traversing a portion of the workflow that maps to a parent node beneath the root parent node, each output update event will generate an additional output update event. This strategy allows the engine to "look-ahead" towards future executions and apply optimizations to dynamic portions of the DAG, as outlined in the following sections.

4.2. *DAG transformations: parallelization with scatter/gather*

By evaluating workflows through this port-counter and trigger system, Rabix is capable of rewiring parallelizable nodes in the DAG when upstream jobs are only partially completed. Suppose we have a workflow where a data file and an array are inputs for a single tool, which then produces an output file (Fig. 5a). In this case, the tool is capable of being scattered over an array of variables (e.g. [1, 2, 3]). Normally, these executions will be performed sequentially on a single core, or on multiple threads if the tool allows it. However, on a workflow level, additional parallelization can be enabled by scattering the data over three separate executions of the tool based on the values in the array (Fig. 5b), thus allowing the jobs to be distributed to separate computational instances as needed.

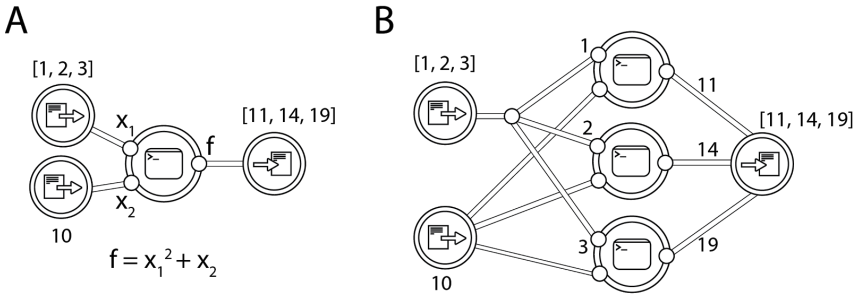


Figure 5. Graph transformations when performing parallelization. In this workflow, a function is performed on two inputs, an *int* and an *array of ints*. **B.** The flattened DAG created by the engine. Each value of the array is scattered as a single process to reduce computation time.

The advantages of the transformation approach is further demonstrated by another use case, in which there are two sequential, parallelizable jobs (Fig. 6a). Rabix employs a "look-ahead" strategy (Fig. 6b) which can mark downstream jobs as ready even though not all sub-jobs (leaves) are done from the upstream parent job.

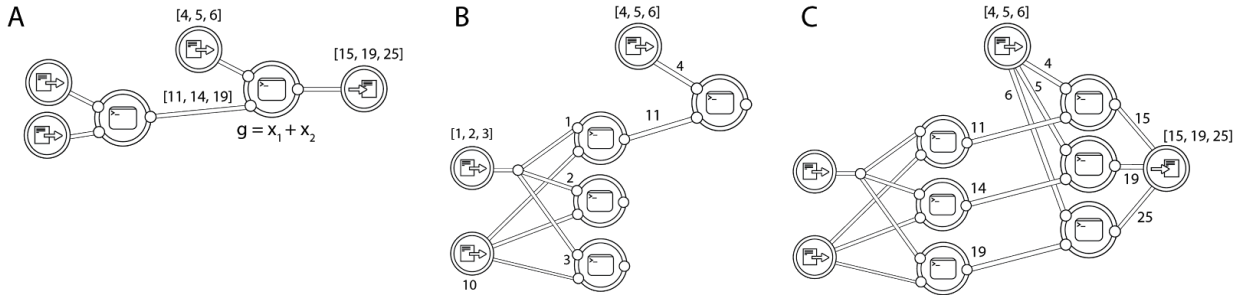


Figure 6. Graph transformations for sequential scattered nodes. **A.** The workflow from Fig. 5 with an additional downstream function with an input that can be scattered. **B.** During execution, the engine is able to look ahead to the next stage in the workflow. If any input is available (e.g. value of 11 returned by a tool), downstream processes which can proceed are started. **C.** The completed workflow.

Each node in the DAG does not need to be scheduled independently. Instead, (sub)jobs that work with same data can be explicitly dispatched to the same backend. (Fig. 7). For example, in the case of executions scattered across chromosome number, jobs processing the same chromosome can be distributed to the same node to optimize cost.

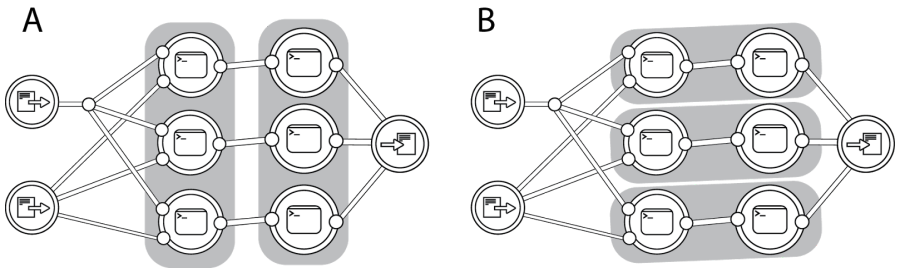


Figure 7. Jobs can be grouped (grey background) for execution on a backend node from criteria set by the workflow or tool author.

Figure 7 demonstrates two possible job group assignments. In the case of Figure 7a, the first tool can be executed simultaneously for each chunk of the data on a single backend node. Once any single job in the first group is finished, the second group of jobs can begin execution on a second node. In the case of Figure 7b, each chunk of data is parallelized across three nodes and the final output is gathered at the end.

The engine is also able to send information to the backends about upcoming jobs, which allows a backend scheduler to pre-allocate resources for them. When executing CWL workflows, both of these optimizations are enabled through the "hints" feature.

Whether these optimizations can be used are sometimes dependent on how the workflow is constructed. For example, a workflow author can make use of optimizations in Fig. 6 by grouping nodes that can be scattered into a nested workflow. This optimization can be especially useful when combined with nested workflow optimizations described in the next section, and allows for reusability of previously made workflows, as encouraged by CWL.

4.3. *Graph transformations: nested workflows*

CWL developers have the ability to reuse existing code and import previously-described workflows into other workflows. This feature means that it is possible to reuse code for additional workflows in lieu of refactoring and potentially introducing errors that break reproducibility. However, the ability to nest workflows presents a challenge to interpretation and optimization by the engine. If no DAG transformations are applied and nested workflows are only executed recursively, this can lead to unnecessarily prolonged execution time and cost.

Suppose a developer has described a workflow that takes two inputs and produces two outputs from two tools (Fig. 8a). In this workflow, one of the outputs is created by the upstream tool and one from the downstream tool. Later, the developer wishes to reuse this workflow description in another workflow, where the output of the upstream tool is passed to another tool for further analysis (Fig. 8b). As with sequentially scattered tools, the engine is capable of passing values from the nested workflow, once they're produced, to steps downstream using the "look-ahead" strategy. Commonly, the tool outside the nested workflow is blocked from execution until all outputs from the nested workflow are produced, leading to increased computation time and cost.

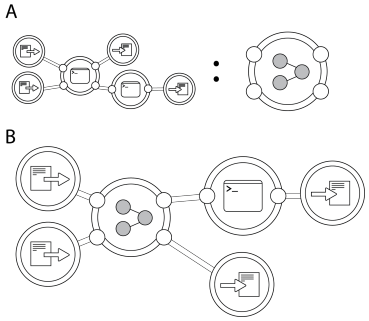


Figure 8. Graph transformations of nested workflows to optimize total execution time. **A.** Workflow consisting of two tools. **B.** Workflow in Fig. 8a. extended with third tool. The engine allows the downstream tool to start executing once the necessary inputs are ready, even if the upstream workflow has yet to produce all of its outputs. No code refactoring from the workflow in 8a is required.

4.4. Benefits to Logging, Orchestration, and Computation

The model used by the Rabix engine allows for improved optimization of data analysis at the workflow level. Further, it provides the ability to implement additional optimizations or features to enhance orchestration of jobs and computation, regardless of whether such features are supported by a workflow description language or specification.

Rabix keeps track of all jobs executed from the workflow and caches results. In addition, each parameter of a job is recorded and automatically logged for the researcher. These include the explicit command line

arguments used, the files/paths, attributes of the data, metadata attributes, and any logs associated with the execution. In addition, a snapshot of the application is stored, along with the explicit values used in the execution. All of this is done at the job level, allowing for granular replication of subsets of a workflow. If the workflow contains a job that has previously been executed and the outputs are still available, the engine can reuse them even if the job was part of a different workflow run. Importantly, even if cached results are not available, the engine will look ahead in the DAG and may encounter cached downstream jobs which do have these files available, and so can resume failed or modified workflow jobs. This makes the caching mechanism comparable to declarative workflow description such as GNU Make.[8]

Additional business logic outside of a workflow specification can also be implemented. For example, CWL does not yet allow for conditional workflows, in which the entirety of DAG is not necessarily traversed but only paths based on checkpoints during the execution. Additionally, though a DAG is acyclic, Rabix could in principle enable loops for a tool or workflow which use iterative operations.

4.5. *Caveats to Graph Transformations and Possible Solutions*

An important caveat for these optimizations are external transformations in which the structure of data elements is modified before execution and thus cannot be anticipated by the engine. For example, CWL and other workflow description languages allow for modifications of input types before tool execution. In certain cases, such as for a tool which can be scattered, the data type may change or the length of the array that is being scattered cannot be known ahead of time. If the engine is unable to anticipate the length of an array that must be scattered upon execution, it is impossible for it to re-wire the DAG before evaluation. However, such hurdles can be overcome by allowing users to either define a mapping for individual array items or declaratively specifying the method of combining multiple ports before scattering (cross-product or dot-product). In these cases, the engine can still maintain its look-ahead optimizations.

4.6. *Furthering reproducibility by extending CWL to execution descriptions*

Workflows described using the Common Workflow Language require two objects for execution: the description of an application and an input object specifying the explicit values of the required inputs. Recording a task that has been previously executed is not, however, within the scope of CWL. However, an analyst may want to reinspect a prior analysis, reuse a workflow with a specific set of parameters on new data, or reanalyze the same data with a different workflow version. It is for these reasons that we have enabled an additional layer of task description and annotation within Rabix, alleviating the burden of logging the workflow execution.

Following the execution of a workflow, additional outputs and logs are produced by Rabix as a matter of course. The explicit command line execution, an object describing the output of the execution, and a description of the workflow execution are all recorded. From these objects, it is directly possible to reproduce a prior analysis or reanalyze additional data with the exact same parameters as previous. Rabix allows for replication of a previous execution or reproduction an exact workflow on new data with a single command line call. In this way, it is possible for an analyst to not only publish a workflow but also the explicit tasks as plain text files. These functionalities can be extended with new modules or plugins to enable a variety of use cases centered on reproducibility.

5. Rabix in the context of existing workflow models and engines

The primary design guideline for Rabix was to support Common Workflow Language in a way which will allow for supporting additional workflow languages, whether they are domain-specific languages or object-based. Further, "tools" or workflows described in different syntaxes should be interoperable such that a single workflow may be comprised of tools and workflows from a variety of syntaxes. In effect, certain optimizations described in Rabix above have been implemented in other systems, but not yet in a single executor capable of supporting emerging standards.

Most of the focus in this paper was on port-level inspection, an abstract data model for tools and workflows, and how they can enable additional optimizations when used in conjunction. However, certain features described here are also used by existing workflow systems,[6,7,10–12] most notably the support for multiple infrastructures. Additionally, there are certain features not yet implemented in Rabix but which are seen in other systems, such as conditional steps in a workflow, as seen in Toil. Though the Rabix model allows for conditional operations (e.g. for, if, while), we chose to focus on features supporting reusability and interoperability and computational optimizations for this manuscript.

6. Conclusions

The Rabix Executor is an open-source project designed to enable scalable and reproducible analysis of portable workflows, which is available on GitHub (http://github.com/rabix/bunny). Computational reproducibility, the ability to replicate a prior analysis or reuse prior workflows on new data, is required for accurately judging scientific claims or enabling large-scale data analysis initiatives in which synonymous results can be compared.[4,5,21] The Rabix engine additionally aims to optimize workflow executions by intelligently interpreting and handling complex workflows. This is achieved through a composite model in which workflows can be more fully decomposed. Finally, additional logic can be applied to optimize for user-defined variables, such as cost or execution time, regardless of the workflow description language being interpreted.

References

1. Peng, R. D. Reproducible Research in Computational Science. *Science* **334,** 1226–1227 (2011).
2. Amstutz, P. *et al.* Common Workflow Language v1.0. *FigShare* (2016). doi:10.6084/m9.figshare.3115156.v2
3. Leipzig, J. A review of bioinformatic pipeline frameworks. *Brief. Bioinform.* (2016). doi:10.1093/bib/bbw020
4. Kanwal, S., Lonie, A., Sinnott, R. O. & Anderson, C. Challenges of Large-Scale Biomedical Workflows on the Cloud -- A Case Study on the Need for Reproducibility of Results. in *2015 IEEE 28th International Symposium on Computer-Based Medical Systems* 220–225 (IEEE). doi:10.1109/CBMS.2015.28
5. Alioto, T. S. *et al.* A comprehensive assessment of somatic mutation detection in cancer using whole-genome sequencing. *Nat. Commun.* **6,** 10001 (2015).
6. Jason P Kurs, Manuele Simi, Fabien Campagne. NextflowWorkbench: Reproducible and Reusable Workflows for Beginners and Experts. (2016). doi:10.1101/041236
7. Cromwell: Workflow Execution Engine using WDL. Available at: https://github.com/broadinstitute/cromwell. (Accessed: 2016)
8. *GNU Make: A Program for Directing Recompilation : GNU Make Version 3.79.1*. (Free Software Foundation, 2002).
9. Köster, J. & Rahmann, S. Snakemake--a scalable bioinformatics workflow engine. *Bioinformatics* **28,** 2520–2522 (2012).
10. John Vivian, Arjun Rao, Frank Austin Nothaft, Christopher Ketchum, Joel Armstrong, Adam Novak, Jacob Pfeil, Jake Narkizian, Alden D. Deran, Audrey Musselman-Brown, Hannes Schmidt, Peter Amstutz, Brian Craft, Mary Goldman, Kate Rosenbloom, Melissa Cline, Brian O'Connor, Megan Hanna, Chet Birger, W. James Kent, David A. Patterson, Anthony D. Joseph, Jingchun Zhu, Sasha Zaranek, Gad Getz, David Haussler, Benedict Paten. Rapid and efficient analysis of 20,000 RNA-seq samples with Toil. *bioRxiv* (2016). doi:10.1101/062497
11. Wolstencroft, K. *et al.* The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud. *Nucleic Acids Res.* **41,** W557–61 (2013).
12. Goecks, J., Nekrutenko, A., Taylor, J. & Galaxy Team. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biol.* **11,** R86 (2010).
13. Deelman, E. *et al.* Pegasus, a workflow management system for science automation. *Future Gener. Comput. Syst.* **46,** 17–35 (2015/5).
14. Guo, F., Yu, L., Tian, S. & Yu, J. A workflow task scheduling algorithm based on the resources' fuzzy clustering in cloud computing environment. *Int. J. Commun. Syst.* **28,** 1053–1067 (2015).
15. Workflow Description Language - Specification and Implementations. Available at: https://github.com/broadinstitute/wdl. (Accessed: 2016)
16. Belhajjame, K. *et al.* Using a suite of ontologies for preserving workflow-centric research objects. *Web Semantics: Science, Services and Agents on the World Wide Web* **32,** 16–42 (2015/5).
17. Terstyanszky, G. *et al.* Enabling scientific workflow sharing through coarse-grained interoperability. *Future Gener. Comput. Syst.* **37,** 46–59 (2014/7).
18. Gamma, E., Helm, R., Johnson, R. & Vlissides, J. *Design Patterns: Elements of Reusable Object-Oriented Software with Applying Uml and Patterns:An Introduction to Object-Oriented Analysis and Design and the Unified Process*. (Addison Wesley, 2003).
19. PROVO-O: The PROV Ontology. (2013). Available at: https://www.w3.org/TR/prov-o. (Accessed: 2016)
20. Hettne, K. M. *et al.* Structuring research methods and data with the research object model: genomics workflows as a case study. *J. Biomed. Semantics* **5,** 41 (2014).
21. Sandve, G. K., Nekrutenko, A., Taylor, J. & Hovig, E. Ten simple rules for reproducible computational research. *PLoS Comput. Biol.* **9,** e1003285 (2013).

DATA SHARING AND REPRODUCIBLE CLINICAL GENETIC TESTING: SUCCESSES AND CHALLENGES

SHAN YANG
Invitae
San Francisco, California, USA
Email: shan.yang@invitae.com

MELISSA CLINE
University of California Santa Cruz
Santa Cruz, California, USA
Email: cline@soe.ucsc.edu

CAN ZHANG
University of California Santa Cruz
Santa Cruz, California, USA
Email: mollyzhang@soe.ucsc.edu

BENEDICT PATEN
University of California Santa Cruz
Santa Cruz, California, USA
Email: benedict@soe.ucsc.edu

STEPHEN E. LINCOLN
Invitae
San Francisco, California, USA
Email: steve.lincoln@me.com

Open sharing of clinical genetic data promises to both monitor and eventually improve the reproducibility of variant interpretation among clinical testing laboratories. A significant public data resource has been developed by the NIH ClinVar initiative, which includes submissions from hundreds of laboratories and clinics worldwide. We analyzed a subset of ClinVar data focused on specific clinical areas and we find high reproducibility (>90% concordance) among labs, although challenges for the community are clearly identified in this dataset. We further review results for the commonly tested *BRCA1* and *BRCA2* genes, which show even higher concordance, although the significant fragmentation of data into different silos presents an ongoing challenge now being addressed by the BRCA Exchange. We encourage all laboratories and clinics to contribute to these important resources.

1. Background

1.1. *Clinical genetic testing*

Clinical genetic tests of germline DNA are routinely used to direct patient care in oncology, cardiology, neurology, pediatrics, obstetrics, and other clinical specialties. Excitement surrounds the future of medical genetics, which will likely involve routine and proactive sequencing of patient genomes or exomes. However, even today genetics is used pervasively: over one million clinical genetic tests will be performed in 2016 to inform various pressing medical decisions facing doctors and patients. This number is considerably larger if tests for infectious disease and tumors (somatic testing) are included. Such testing is regulated, often paid for by private insurance and public health systems, and written into many current clinical care guidelines established by payers and medical professional societies.

It is not glib to say that many of these tests are ordered in life-or-death situations. One example is *BRCA1* and *BRCA2* (collectively, *BRCA1/2*) tests, where erroneous results can have substantial deleterious consequences for patients. With a false positive, a radical preventative procedure such as prophylactic bilateral oophorectomy may be indicated, thereby causing an otherwise healthy woman to enter premature menopause and to face the multiple health risks associated with that procedure and with the hormone replacement therapy that often follows. Prophylactic chemotherapy (specifically, tamoxifen) is another option offered to some healthy *BRCA1/2* carriers, with significant side effects. Conversely a false negative could eliminate the chance to prevent a fatal early-onset carcinoma. Such errors are either analytic (reporting a variant to be present in a patient when it is not, or vice versa) or interpretive (concluding that a variant is pathogenic [disease causing] when it is not, or vice versa). This paper focuses on the latter subject.

1.2. *Clinical variant interpretation*

In response to concerns about reproducibility among laboratories, the American College of Medical Genetics (ACMG) and the Association for Molecular Pathology (AMP) jointly developed revised guidelines for clinical variant interpretation [Richards 2015]. These guidelines require laboratory directors to scrutinize the literature and all other available evidence for each variant observed in a patient. The guidelines provide a structured framework for which evidence is weighed in final interpretations. Under these guidelines, variants are classified as pathogenic (P), likely pathogenic (LP), variants of uncertain significance (VUS), likely benign (LB), or benign (B). Despite the significant improvement in standardization that these new guidelines represent compared with their predecessor, laboratory directors must still use a significant degree of expert judgment, which can result in different classifications from different laboratories for the same variant. Date also matters: classifications that pre-date availability of an important piece of evidence should indeed be different than those that post-date it.

1.3. *Data sharing and clinical genetics*

Of course, the first step toward achieving reproducibility is measuring reproducibility, which requires data to be shared among clinical labs. The sharing of genetic data from research projects has long been accepted and encouraged (despite being incompletely implemented). Unfortunately, the open sharing of de-identified clinical genetic data has been far less common owing to a combination of informed consent issues, the commercial interests of certain healthcare providers, and the lack of a community mechanism for doing so.

Recently, the National Institutes of Health established ClinVar, "a freely available archive for interpretations of clinical significance of variants for reported conditions" [Landrum 2016]. By storing only individual variants and classifications, the re-identification of patients whose genotypes are submitted to ClinVar becomes essentially impossible, at least without an independent test of the same variant in the same patient for comparison (in which case, the patient's genotype is already known). Thus, fully de-identified clinical genetic data can be disclosed publicly under US laws and regulations. The American Medical Association (AMA) and National Society of Genetic Counselors (NSGC), among others, have issued recommendations urging laboratories to share such data.

Some commercial and academic laboratories have, unfortunately, declined to participate. Most famously, Myriad Genetics, the largest *BRCA1/2* testing laboratory in the world, has maintained its large genetic database as a proprietary asset [Cook-Deegan 2013]. Moreover, Myriad claims that by leveraging this database, it can deliver superior variant classifications compared to other labs [Angrist 2014]. This stands in sharp contrast with the American Medical Association and the National Society for Genetic Counselors recommendations. It also is inconsistent with accepted practice in many non-genetics medical fields in which data sharing is common. Thankfully thousands of de-identified Myriad reports have been submitted to ClinVar by ordering clinicians through the Sharing Clinical Reports Project [SCRP website].

2. ClinVar

Since its inception in 2013, ClinVar has grown rapidly, and as of August 2016 contains more than 186,000 records from 560 submitters, most of which are clinical genetic testing laboratories [ClinVar website]. Importantly, three of the top eight submitters to ClinVar are commercial laboratories (GeneDx, Invitae, and Ambry). Another three are large academic laboratories (Harvard Partners Laboratory for Molecular Medicine, Emory Genetics Laboratory, and the University of Chicago Genetic Services Laboratories), and two are academic efforts that aggregate literature-based information (OMIM and GeneReviews) These submitters account for more than half of the data in ClinVar, although the many smaller submitters provide key data as well. This high degree of industry–academic collaboration is encouraging and critical given the degree of privatization in the American healthcare system.

2.1. *Data set used for analysis*

We extracted variant classifications from ClinVar (May 2016 XML download, which remains archived online [ClinVar website]). We included data for genes in six different clinical specialties that our laboratory (Invitae) offered for clinical testing at the time and with which we were thus familiar (Supplemental Data). For simplicity, when one gene may be tested by multiple specialties, we used the most common one. Because variant-phenotype assertions are inconsistently populated in ClinVar these were ignored. We further limited our data set to classifications of germline (not somatic) variants from licensed clinical diagnostic laboratories. Thus data submitted by literature curation efforts (e.g. OMIM), expert panels (e.g., ENIGMA, InSiGHT) and research were also excluded, as these do not reflect actual clinical test reports provided to physicians. Finally, we required that variant classifications be on the 5-class ACMG system and be asserted by at least two submitters. Our data set contained 9875 variants in 409 genes (Table 1, Supplemental Data). We note that many of these classifications pre-date the 2015 ACMG guidelines mentioned above.

	Variants	*Genes*	*Classifications*	*Variants/Gene*	*Classifications/Variant*
Cancer	4802	55	12,703	87.3	2.7
Cardiology	3289	163	7611	20.2	2.3
Epilepsy	739	58	1659	12.7	2.2
Metabolic	383	56	850	6.8	2.2
Neurology	662	77	1376	8.6	2.1
Total	**9875**	**409**	**24,199**	**24.1**	**2.5**

Table 1. ClinVar-based data set used in this analysis.

Overall, variants considered benign (B or LB) by most or all submitters composed the largest group (44.5%). Pathogenic variants (P or LP) made up 17.9% of the data set. Many variants (26.9%) were considered VUS, and 10.7% had no consensus (as defined below) for any category. This distribution varied significantly by clinical area (Figure 1).

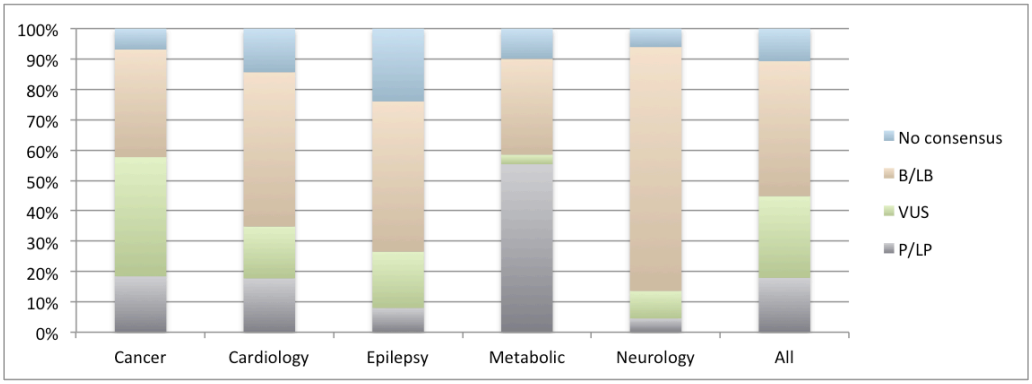


Figure 1. Fraction of variants in ClinVar for each clinical area by consensus pathogenicity.

2.2. *Rarity of clinically observed variants*

Because our data set was limited to variants from two or more submitters, it was naturally biased away from the rarest of variants. Nevertheless, this data set was predominantly composed of rare variants (Figure 2). Most (62%) of the ClinVar variants that also appear in ExAC [Lek, 2016] had population allele frequencies less than 0.001, and for 36%, that frequency was less than 0.0001. Another 22.8% of the ClinVar variants were not in ExAC at all, either because they are very rare or because they lie outside of ExAC's well-covered regions. This rarity also manifests itself in the number of submitters who have classified each variant: Most variants had been classified by only two or three of the 23 submitters in this data set (Table 1). Even in the case of *BRCA1/2*, one of the most common clinically tested genes, the average was only 2.9 classifications per variant. Rare variants comprise an even larger fraction of ClinVar overall, particularly variants with only a single submitter which were excluded from this data set.

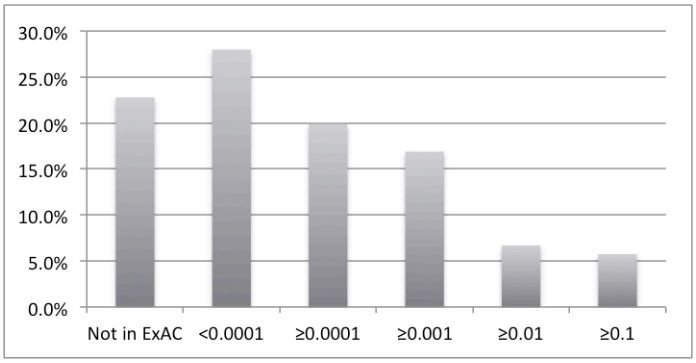


Figure 2. Histogram of allele frequency in ExAC for all ClinVar variants in our analysis regardless of pathogenicity. Note that the vast majority of ClinVar variants are in or near exons.

2.3. *Concordance of variant classifications*

We compared variant classifications in ClinVar to assess the degree of agreement among clinical testing laboratories (Figure 3). We first focused on differences between positive (P or LP) classifications, which are potentially clinically actionable, as opposed to findings that are not actionable (VUS, B, or LB). We refer to this analysis as the P-NP (positive versus not positive) comparison. Counting each of the 9875 variants as a data point, concordance among laboratories was high: 96.1% of variants agreed across all (two or more) submitters. For an additional 0.9% of variants, there was a consensus among a majority of the submitters. We defined consensus as agreement in two-thirds of the submissions (i.e., consensus required two of two submissions to agree, or 2/3, 3/4, 4/5, 4/6, etc.). In 3% of variants, there were only two submitters who disagreed, and only one variant had four submitters with a 2–2 tie. Clinical care guidelines generally state that patients with only VUS should be managed according to their personal and family histories and not their genetic test results [e.g. NCCN 2016]. Thus the P-NP comparisons correlate most with the impact of interpretation discordance on patient care decisions.

When the comparison was performed on a different basis—not combining VUS with B/LB classifications—concordance was, of course, lower. We refer to this analysis as the P-V-B (pathogenic versus VUS versus benign) comparison. In this evaluation, only 83% of variants agreed among all submitters. A further 6% achieved consensus but with some submitter(s) in dissent. This much lower rate indicates that the criteria for discriminating between VUS and B/LB variants varies among laboratories, more so than criteria for establishing pathogenicity.

Concordance varied considerably among clinical areas. On a P-NP basis, variants in cardiology and metabolic genes had concordances lower that those in the other areas, although in all cases concordance was greater than 90%. On a P-V-B basis, epilepsy genes fared the worst, followed by cardiology. The gap between P-NP and P-V-B is particularly large in epilepsy genes, suggesting that evidence against pathogenicity is used quite inconsistently by labs. Cursory analysis suggests that classification date, as expected, plays a significant role in discordance (Supplemental Data). A detailed analysis of the basis for discordance is important future work.

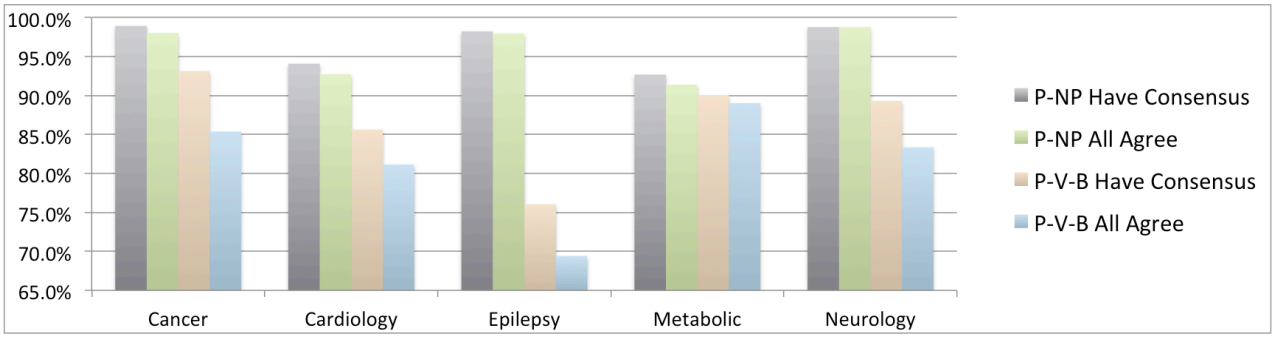


Figure 3. Concordance among labs measured in different ways. See text.

3. *BRCA1/2*

BRCA1/2 had the largest number of variants of any gene(s) in our ClinVar data set (1771 combined) for several reasons: *BRCA1* and *BRCA2* are not only among the most commonly tested genes in clinical practice today, but also have been clinically tested for more than 20 years. Moreover, a significant international effort has focused on adding *BRCA1/2* variants to ClinVar, whereas data sharing efforts for some other commonly tested genes center on previously established databases (e.g. the CFTR2 database for cystic fibrosis). Finally, compared with most human genes, *BRCA1/2* have relatively large coding sequences and thus can harbor an atypically large number of variants. Thus, the "long tail" of *BRCA1/2* variants is particularly long, and new variants needing classification are continually uncovered, as shown by our own data (Figure 4). This conclusion is consistent with unpublished reports from Myriad Genetics, which claims to encounter >50 new variants per week despite offering testing for 20 years [Myriad 2015]

3.1. *Concordance among BRCA1/2 variant classifications*

In a separate study, we performed a much more detailed comparison of ClinVar data for *BRCA1/2* using a ClinVar data set of more than 2000 comparable variants [Lincoln 2016]. This

analysis considered only classifications from clinical labs with significant experience (as evidenced by submitting 200 or more variants to ClinVar) and excluded submitters where most classifications were >5 years old. On a P-NP basis, 98.5% of variants showed no disagreement among submitters—a concordance higher than that observed in ClinVar overall. This previous study also showed that variants with classification discordance were rare (allele frequencies were always less than 0.0005 and usually were immeasurably low). Although they are numerous, rare variants by definition appear in very few patients: less than 15% of the 30,000 patients studied carried any rare variants in *BRCA1* or *BRCA2*, and most of those were concordantly classified. In this prior study, concordance per patient (not per variant) was thus estimated to be 99.8%.

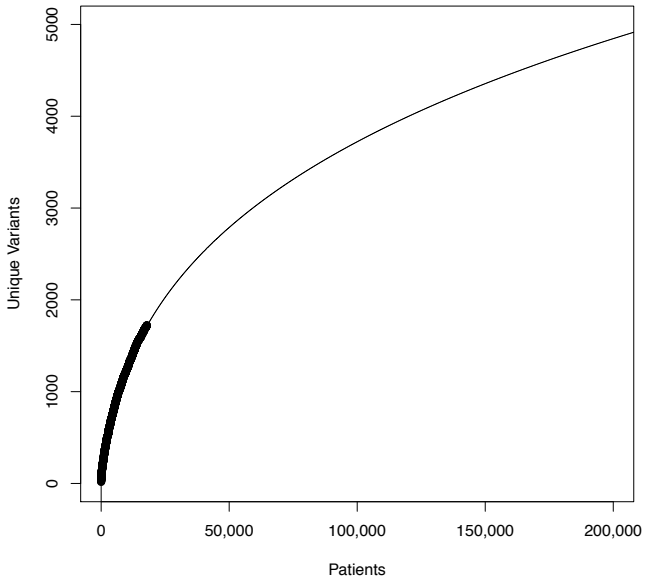


Figure 4. The relationship of number of unique BRCA1/2 variants to number of patients tested at Invitae (dark curve). The extrapolation (light curve) was fit in R using the formula poly(log(Patients), 3). We chose the polynomial degree empirically by minimizing the Akaike Information Criteria [Sakamoto 1986].

3.2. *Variants of Uncertain Significance (VUS) in BRCA1/2*

VUS can present a challenge in day-to-day clinical decision-making, and the most prevalent type of VUS are rare missense changes. VUS rates are traditionally defined as the fraction of patients with one or more VUS and no positive findings. Major U.S. laboratories report VUS rates in the range of 3–5% for *BRCA1/2*, although this rate varies considerably with ethnic mix and with the fraction of cancer-affected versus unaffected patients [Lincoln 2015]. On a per-variant (rather than per-patient) basis, the VUS rate is much higher: 31.4% of *BRCA1/2* variants in our data set (Table 1) were VUS, although most are very rare and thus appear in very few patients.

The evidence suggests that the majority of VUS are actually benign variants that have inadequate evidence to demonstrate that fact. This is supported by our own experience that most

VUS, when reclassified, are "downgraded" to LB or B. We also observed this in a sequential analysis of ClinVar releases from the past 2 years (available at [ClinVar website]) in which roughly 95% of *BRCA1/2* VUS reclassifications were downgrades. Others have also observed this in Myriad data [Murray 2011]. In terms of clinical impact, a rough approximation is that if 4% of patients have a VUS, and if 5% of those findings are truly pathogenic variants lacking evidence of pathogenicity, then 1/500 *BRCA1/2*-positive patients may currently be missed.

BRCA1/2 tests are increasingly being replaced by multi-gene panels that assay additional genes that significantly increase the risk of various cancers. By virtue of testing more genes, the VUS rate in these panels is substantially larger. For example, VUS rates of roughly 40% have been reported by 25-29 gene panels [Lincoln 2015; Desmond 2015; Tung 2015], although again, experience suggests that the majority of these VUS will ultimately be classified as benign.

3.3. *The BRCA Exchange*

As of August 2016, ClinVar contains more than 9000 variants in *BRCA1/2*, many of which are either unclassified or are considered VUS. Most of these variants have been reported by only a single submitter. These data still represent only a fraction of the known human variation in *BRCA1/2,* much of which is either not submitted to ClimVar or is not appropriate for ClinVar (yet is useful to have linked). In an effort to collect a more comprehensive view of *BRCA1/2* variation, the BRCA Exchange project has been initiated under the auspices of the Global Alliance for Genomics and Health's BRCA Challenge. European laboratory data, coordinated by the Leiden Open Variation Database (LOVD), population databases, and other data sources are being combined with ClinVar in this *BRCA1/2*-specific public database. In its current preliminary form, the BRCA Exchange describes more than 13,000 variants, many of which originate from only a single source database (Figure 5). Not only is the BRCA exchange database open, but the code that populates it is open source. Future analyses of the type described in this paper could and should leverage this code in order to further improve reproducibility of such research.

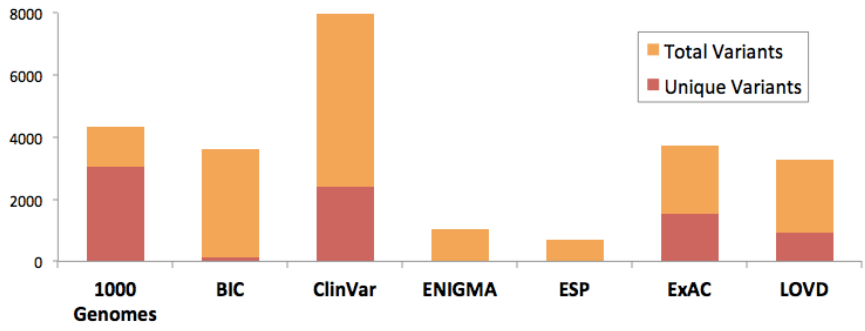


Figure 5. Sources of data in the pre-release BRCA exchange. Many variants not in ClinVar and indeed many are unique to a single database. For details and references see brcaexchange.org.

4. Discussion

4.1. *Summary: Most variant classifications agree, but . . .*

In the analysis described above, we examined nearly 10,000 variants from ClinVar in more than 400 genes across six clinical areas and found generally high (>90%) variant classification concordance among clinical laboratories in terms of potential effect on clinical management (our P-NP comparison). In separate prior studies, we examined *BRCA1/2* in particular detail found higher concordance on both a per-variant (99.0%) and a per-patient (99.8%) basis than is seen in the broader gene list. It is reassuring, at least for geneticists, to note that this level of concordance is higher than that observed among pathologists reading breast biopsies or radiologists reading mammograms [Elmore 2015(a,b); Elmore 2016; Sprague 2016]. Nevertheless, resolving differences in variant classification is critical to doctors and patients. Moreover, many variants (30.1%) have classifications that are concordantly VUS and much more work is required to classify these variants definitively even though laboratories agree.

Public databases such as ClinVar play critical roles in the identification of both disagreements and uncertainties, and these databases can facilitate collaborative interactions that will resolve many such issues. The value of such collaboration in improving variant classifications has recently been demonstrated by multiple groups [Amendola 2016]. Efforts are now organized into disease-specific working groups by the ClinGen consortium [Rehm 2015; Pfimister 2015] and support pre-existing efforts such as ENIGMA and InSiGHT. Those with interest and expertise in these areas should certainly consider joining and contributing.

Public databases can also play a critical role in laboratory quality control by allowing detailed independent peer scrutiny of all variant classifications by the global community. In our opinion, no laboratory could (or probably would) mount such an effort alone, and publication peer review processes can not provide this type of ongoing quality assessment. In our opinion, laboratory directors who are both confident in their quality yet continually working to improve should have no reservations about unrestricted public data submission of their data.

4.2. *Considerations when using public clinical databases*

Our analysis highlights important considerations users must keep in mind when accessing public databases such as ClinVar. Foremost is that it is a fallacy to say, for example, “ClinVar says that variant X is pathogenic.” ClinVar itself generates no assertions; it only collects them from submitters. Database users must pay careful attention to the original source of each classification, which may be a reputable clinical laboratory rigorously following accepted classification guidelines, or it may not be. Dates are important, as submissions to these databases can become outdated, which results in false discrepancies.

It is important that users understand the biology and medical practice considerations for each gene they examine in a public database. Consider three examples of the rates of variant pathogenicity (Figure 1) which we find unsurprising: (a) In some genes (e.g., most hereditary

cancer genes) loss-of-function variants are pathogenic, and nature provides many means of disabling genes or their proteins. In other cases (e.g., some neurology and cardiology genes), gain-of-function mutations are clinically more important, and these, by their very nature, are less numerous, reducing the fraction of pathogenic variants in ClinVar. (b) The large fraction of pathogenic variants and small fraction of VUS in metabolic genes reflect the fact that experimental confirmation of pathogenicity (e.g., through blood chemistry and urinalysis) is relatively straightforward and standard clinical practice. However, the relatively low concordance in metabolic genes (Figure 3) suggests that these procedures are imperfect. (c) In cardiology, complexities in both phenotyping and penetrance are well known to increase the complexity of variant classification [Van Driest].

Deliberate (and not nefarious) submission biases also affect ClinVar. Notably, laboratory policies vary as to whether and when B/LB variants are reported to patients/physicians or to ClinVar (even though benign polymorphisms are frequently observed). Similarly, practices for the detection and reporting of non-coding variants vary. Although many routine tests detect copy number variants, these variants are less commonly reported to ClinVar for logistical reasons (a situation we hope will change). Furthermore, a test may or may not be sensitive to complex alterations such as copy-neutral inversions, Alu insertions, or variants in low complexity or highly conserved regions. Although ClinVar can record the observed prevalence of any variant, this field is rarely filled in. Finally, ClinVar submissions generally represent laboratory patient series, which are subject to many undocumented ascertainment biases. For these reasons, ClinVar cannot be used to evaluate the spectrum of disease-causing or benign variation in any gene.

4.3. *Whither data sharing*

Although sharing of clinical genetic data has been successful, and clearly impactful, challenges remain. For example, during our various analyses of ClinVar, we uncovered a number of out of date and erroneous submissions, which are an obvious concern. A bigger problem is the multiple laboratories who do not contribute. In addition to not contributing, Myriad Genetics has updated its terms of service to, in theory, prohibit ordering clinicians from sharing data with ClinVar [Robinson 2016]. A further challenge is the fragmentation of data into multiple silos. Although the BRCA Exchange aims to address this problem for *BRCA1/2*, this is a considerable effort and only applies to these two genes, not the many others of clinical relevance.

In environmental policy, the term “greenwashing” has emerged to describe the characterization of various activities as environmentally friendly when in fact they are not. Activities can occur in our field that one might perhaps call “sharewashing”. For example two large commercial labs (Labcorp and Quest) currently contribute variants only to BRCAShare, a database whose terms effectively prohibit either incorporation of the data into a common repository (like the BRCA Exchange) or its use in comparisons such as those described here. We hope this changes, but at present these data are not available in unrestricted form. The BRCAShare terms also prohibit use of the data by other commercial labs without paying a significant fee (unlike ClinVar). Separately, Myriad has tried to argue that its participation in the PROMPT patient registry comprises data sharing. PROMPT is indeed valuable, but serves a very different purpose than

ClinVar. We encourage all groups to support and contribute to open, unrestricted, public databases, particularly ClinVar.

5. References

Amendola LM, et al. Am J Hum Genet 2016; 10.1016/j.ajhg.2016.03.024.
Angrist M and Cook-Deegan R. Appl Transl Genom 2014;3(4):124-127.
ClinVar website: www.clinvar.com
Cook-Deegan R, et al. Eur J Hum Genet 2013;21(6):585-8.
Desmond A, et al. JAMA Oncol 2015;1(7):943-51.
ENIGMA website: enigmaconsortium.org
Elmore 2015(a): Elmore JG, et al. JAMA 2015;313(11):1122-32.
Elmore 2015(b): Elmore JG, et al.. JAMA 2015;314(1):83-4.
Elmore 2015(c): Elmore JG, et al. Ann Intern Med 2016;164(10):649-55.
Genereviews website: www.ncbi.nlm.nih.gov/books/NBK1116/
InSiGHT website: insight-group.org
Landrum MJ, et al. Nucleic Acids Res 2016;44(D1):D862-8.
Lek M, et al. Nature 2016; 536:285–291
Lincoln 2016: Lincoln SE, ACMG 2016 platform presentation, copy available at www.invitae.com; manuscript in review.
Lincoln 2015: Lincoln SE, et al. J Mol Diagn 2015;17(5):533-44.
Murray ML, et al. Genet Med 2011;13(12):998-1005.
Myriad Analyst Day Presentation, September 2015 from www.myriad.com
NCCN (National Comprehensive Cancer Network). NCCN Practice Guidelines in Oncology. Genetic/Familial High Risk Assessment: Breast and Ovarian, Version 2.2016. www.nccn.org
OMIM website: omim.org
Phimister EG. New Engl J Med 2015;372(23):2227-8.
Richards S, et al. Genet Med 2015;17(5):405-24.
Rehm HL, et al. New Engl J Med 2015;372(23):2235-42.
Robinson 2016: L. Robinson, genetic counselor, UT Southwestern; personal communication.
Sakamoto Y, et al. 1986. Akaike Information Criterion Statistics. (D. Reidel Publishing)
SCRP website: www.clinicalgenome.org/data-sharing/sharing-clinical-reports-project-scrp/
Sprague BL, et al. Ann Intern Med 2016; 10.7326/M15-2934.
Tung N, et al. Cancer 2015, 121:25e33
Van Driest SL, et al. JAMA 2016;315(1):47-57.

6. Supplement

The dataset upon which this analysis is based is available at:
https://drive.google.com/drive/folders/0B79LNgCdve9BSWN0VHhodFFsMmM