

comprehensive analyses. Further, it means that results can be reproduced using the original computation's set of tools and parameters. If we had run the original TCGA best-practices RNA-seq pipeline with one sample per node, it would have cost ~\$800,000. Through the use of efficient algorithms (STAR and Kallisto) and Toil, we were able to reduce the final cost to \$26,071 (**Supplementary Note 9**).

We have demonstrated the utility of Toil by creating one of the single largest, consistently analyzed, public human RNA-seq expression repositories, which we hope the community will find useful.

Editor's note: This article has been peer-reviewed.

Note: Any Supplementary Information and Source Data files are available in the online version of the paper.

ACKNOWLEDGMENTS

This work was supported by (BD2K) the National Human Genome Research Institute of the National Institutes of Health award no. 5U54HG007990 and (Cloud Pilot) the National Cancer Institute of the National Institutes of Health under the Broad Institute subaward no. 5417071-5500000716. The UCSC Genome Browser work was supported by the NHGRI award 5U41HG002371 (Corporate Sponsors). The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health or our corporate sponsors.

AUTHOR CONTRIBUTIONS

J.V., A.A.R. and B.P. wrote the manuscript. J.V., A.A.R., A.N., J.A., C.K., J.N., H.S., P.A., J.P., A.D.D., B.O. and B.P. contributed to Toil development. F.A.N. and A.M. contributed to Toil-Spark integration. J.V. wrote the RNA-seq pipeline and automation software. M.H. and C.B. contributed WDL and cloud support. P.A. and S.Z. contributed CWL support. J.Z., B.C. and M.G. hosted quantification results on UCSC Xena. K.R. hosted GTEx results in UCSC Genome Browser. W.J.K., J.Z., S.Z., G.G., D.A.P., A.D.J., M.C., D.H. and B.P. provided scientific leadership and project oversight.

Data availability. Data are available from this project at the Toil xena hub (<https://genome-cancer.ucsc.edu/proj/site/xena/datapages/?host=https://toil.xenahubs.net>).

COMPETING FINANCIAL INTERESTS

The authors declare competing financial interests: details are available in the [online version of the paper](#).

John Vivian¹, Arjun Arkal Rao¹, Frank Austin Nothafi^{2,3}, Christopher Ketchum¹, Joel Armstrong¹, Adam Novak¹, Jacob Pfeil¹, Jake Narkizian¹, Alden D Deran¹, Audrey Musselman-Brown¹, Hannes Schmidt¹, Peter Amstutz⁴, Brian Craft¹, Mary Goldman¹, Kate Rosenbloom¹, Melissa Cline¹, Brian O'Connor¹, Megan Hanna⁵, Chet Birger⁵, W James Kent¹, David A Patterson^{2,3}, Anthony D Joseph^{2,3}, Jingchun Zhu¹, Sasha Zaranek⁴, Gad Getz⁵, David Haussler¹ & Benedict Paten¹

¹Computational Genomics Lab, UC Santa Cruz Genomics Institute, University of California

Santa Cruz, Santa Cruz, California, USA. ²AMP Lab, University of California Berkeley, Berkeley, California, USA. ³UC Berkeley ASPIRE Lab, Berkeley, California, USA. ⁴Curoverse, Somerville, Massachusetts, USA. ⁵Broad Institute of Harvard and Massachusetts Institute of Technology (MIT), Cambridge, Massachusetts, USA.
e-mail: benedict@soe.ucsc.edu

- Weinstein, J.N. *et al.* *Nat. Genet.* **45**, 1113–1120 (2013).
- Zhang, J. *et al.* *Database*. <http://dx.doi.org/10.1093/database/bar026> (2011)
- Siva, N. *Lancet* **385**, 103–104 (2015).
- McKenna, A. *et al.* *Genome Res.* **20**, 1297–1303 (2010).
- UNC Bioinformatics. TCGA mRNA-seq pipeline for UNC data. https://webshare.bioinf.unc.edu/public/mRNAseq_TCGA/UNC_mRNAseq_summary.pdf (2013).
- Albrecht, M., Michael, A., Patrick, D., Peter, B. & Douglas, T. in *Proceedings of the 1st ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies (SWEET '12)* 1. ACM (Association of Computing Machinery. <http://dx.doi.org/10.1145/2443416.2443417> (2012).
- Bernhardsson, E. & Frieder, E. Luigi. *GitHub* <https://github.com/spotify/luigi> (2016).
- Goecks, J., Nekrutenko, A. & Taylor, J. *Genome Biol.* **11**, R86 (2010).
- UCSC. Xena <http://xena.ucsc.edu> (2016).
- Amstutz, P. Common workflow language. *GitHub* <https://github.com/common-workflow-language/common-workflow-language> (2016).
- Frazer, S. Workflow description language. *GitHub* <https://github.com/broadinstitute/wdl> (2014).
- Vivian, J. Toil scripts. *GitHub* https://github.com/BD2KGenomics/toil-scripts/tree/master/src/toil_scripts (2016).
- Apache Software Foundation. Apache Spark <http://spark.apache.org/> (2017).
- Massie, M. *et al.* ADAM: genomics formats and processing patterns for cloud scale computing. University of California, Berkeley, Technical Report No. UCB/ECS-2013-207 (2013).
- Gentzsch, W. in *Proceedings First IEEE/ACM International Symposium on Cluster Computing and the Grid* 35–36 <http://dx.doi.org/10.1109/ccgrid.2001.923173> (IEEE, 2001).
- Yoo, A.B., Jette, M.A. & Mark, G. in *Lecture Notes in Computer Science* 44–60 (2003) Springer, Berlin, Heidelberg.
- Apache Software Foundation. Apache Mesos <http://mesos.apache.org/>
- GTEx Consortium. *Science* **348**, 648–660 (2015).
- Dobin, A. *et al.* *Bioinformatics* **29**, 15–21 (2013).
- Li, B. & Dewey, C.N. *BMC Bioinformatics* **12**, 323 (2011).
- Bray, N.L., Pimentel, H., Melsted, P. & Pachter, L. *Nat. Biotechnol.* **34**, 525–527 (2016).
- Barker, A.D. *et al.* *Clin. Pharmacol. Ther.* **86**, 97–100 (2009).
- Kent, W.J. *et al.* *Genome Res.* **12**, 996–1006 (2002).

Nextflow enables reproducible computational workflows

To the Editor:

The increasing complexity of readouts for omics analyses goes hand-in-hand with concerns about the reproducibility of experiments that analyze 'big data'^{1–3}. When analyzing very large data sets, the main source of computational irreproducibility arises from a lack of good practice pertaining to software and database usage^{4–6}. Small variations across computational platforms also contribute to computational irreproducibility by producing numerical instability⁷, which is especially relevant to high-performance computational (HPC) environments that are routinely used for omics analyses⁸. We present a solution to this instability named Nextflow, a workflow management system that uses Docker technology for the multi-scale handling of containerized computation.

In silico workflow management systems are an integral part of large-scale biological analyses. These systems enable the rapid prototyping and deployment of pipelines that combine complementary software packages. In genomics the simplest pipelines, such as Kallisto and Sleuth⁹, combine an RNA-seq quantification method with a differential expression module (**Supplementary Fig. 1**). Complexity rapidly increases when all aspects of a given analysis are included. For example,

the Sanger Companion pipeline¹⁰ bundles 39 independent software tools and libraries into a genome annotation suite. Handling such a large number of software packages, some of which may be incompatible, is a challenge. The conflicting requirements of frequent software updates and maintaining the reproducibility of original results provide another unwelcome wrinkle. Together with these problems, high-throughput usage of complex pipelines can also be burdened by the hundreds of intermediate files often produced by individual tools. Hardware fluctuations in these types of pipelines, combined with poor error handling, could result in considerable readout instability.

Nextflow (<http://nextflow.io>; **Supplementary Methods, Supplementary Note and Supplementary Code 1**) is designed to address numerical instability, efficient parallel execution, error tolerance, execution provenance and traceability. It is a domain-specific language that enables rapid pipeline development through the adaptation of existing pipelines written in any scripting language.

We present a qualitative comparison between Nextflow and other similar tools in **Table 1** (ref. 11). We found that multi-scale containerization, which makes it possible to

bundle entire pipelines, subcomponents and individual tools into their own containers, is essential for numerical stability. Containers can be produced *ad hoc* by users or by following recently proposed standards (BioBoxes¹², Bioshaddock¹³ and AlgoRun¹⁴). Another key specification of Nextflow is its integration with software repositories (including GitHub and BitBucket) and its native support for cloud systems. Some of the practical implications of this integration are relevant to computational reproducibility; the impact of GitHub has been recently highlighted as a driving force behind data sharing¹⁵.

Nextflow enables users to run any current or previous version of a pipeline for any published and properly deposited analyses. The GitHub integration allows consistent tracking of software changes and versions, the containerization ensures numerical stability, and the cloud support provides rapid computation and effective scaling. When using this process, any set of results, including a table, a graph or any kind of numerical quantification can be associated with a single command line and referenced, updated, reproduced or improved on demand. Nextflow uses a functional reactive programming model in which each operation (typically a workflow task) is isolated in its own execution context. Outputs from one operation in Nextflow are streamed to other operations by dedicated channels in a process similar to UNIX pipes. Parallelization is an implicit consequence of the way inputs and outputs of each process are channeled into other processes. This approach spares users the need to implement an explicit parallelization strategy. Another advantage of Nextflow is its reliance on the dataflow programming paradigm, in which tasks are automatically started once data are received through input channels.

The dataflow model is superior to alternative solutions based on a Make-like approach, such as Snakemake¹⁶, in which computation involves the pre-estimation of all computational dependencies, starting from the expected results up until the input raw data (Table 1). A Make-like procedure requires a directed acyclic graph (DAG), whose storage requirement is a limiting factor for very large computations. In contrast, as the top to bottom processing model used by Nextflow follows the natural flow of data analysis, it does not require a DAG. Instead, the graph it traverses is merely incidental and does not need to be pre-computed or even stored, thereby ensuring high scalability¹⁷.

The use of communication channels between tasks also contributes to

Table 1 Comparison of Nextflow with other workflow management systems

Workflow	Nextflow	Galaxy	Toil	Snakemake	Bpipe
Platform ^a	Groovy/JVM	Python	Python	Python	Groovy/JVM
Native task support ^b	Yes (any)	No	No	Yes (BASH only)	Yes (BASH only)
Common workflow language ^c	No	Yes	Yes	No	No
Streaming processing ^d	Yes	No	No	No	No
Dynamic branch evaluation	Yes	?	Yes	Yes	Undocumented
Code sharing integration ^e	Yes	No	No	No	No
Workflow modules ^f	No	Yes	Yes	Yes	Yes
Workflow versioning ^g	Yes	Yes	No	No	No
Automatic error failover ^h	Yes	No	Yes	No	No
Graphical user interface ⁱ	No	Yes	No	No	No
DAG rendering ^j	Yes	Yes	Yes	Yes	Yes
Container management					
Docker support ^k	Yes	Yes	Yes	No	No
Singularity support ^l	Yes	No	No	No	No
Multi-scale containers ^m	Yes	Yes	Yes	No	No
Built-in batch schedulersⁿ					
Univa Grid Engine	Yes	Yes	Yes	Partial	Yes
PBS/Torque	Yes	Yes	No	Partial	Yes
LSF	Yes	Yes	No	Partial	Yes
SLURM	Yes	Yes	Yes	Partial	No
HTCondor	Yes	Yes	No	Partial	No
Built-in distributed cluster^o					
Apache Ignite	Yes	No	No	No	No
Apache Spark	No	No	Yes	No	No
Kubernetes	Yes	No	No	No	No
Apache Mesos	No	No	Yes	No	No
Built-in cloud^p					
AWS (Amazon Web Services)	Yes	Yes	Yes	No	No

Among the several workflow management tools similar to Nextflow, Bpipe¹¹ is probably the most closely related to Nextflow. However, one problem with Bpipe is a lack of support for multi-scale containerization, which is present in Nextflow and other workflow managers such as Galaxy and Toil. ^aThe technology and the programming language in which each framework is implemented. ^bThe ability of the framework to support the execution of native commands and scripts without re-implementation of the original processes. ^cSupport for the CWL specification. ^dAbility to process tasks inputs/outputs as a stream of data. ^eSupport for code management and sharing platforms, such as GitHub, BitBucket and GitLab. ^fSupport for module, sub-workflows or workflow compositions. ^gAbility to track pipeline changes and to execute different versions at any point in time. ^hSupport for automatic error handling and resume execution mechanism. ⁱImplementation of a graphical user interface to interact with the pipeline. ^jAbility to visualize the graph of the task dependencies and executions. ^kIntegrated support for Docker containers technology. ^lIntegrated support for Singularity containers technology. ^mAbility to manage the execution of multiple container instances in a distributed/HPC cluster or cloud. ⁿAbility to spawn the executions of pipeline tasks through a cluster batch scheduler without the need of custom scripts or commands. It must be noted that although this support is not built-in for Snakemake, it merely requires the user to provide the cluster-specific job control commands. ^oAbility to spawn the executions of pipeline tasks through a distributed cluster. ^pAbility to deploy and run distributed workflows taking advantage of distinctive cloud infrastructure features such as elastic resource allocations and spot-instances. Boldface indicates list of similar supported components or applications.

computational flexibility and robustness in Nextflow. For example, in Snakemake, the task execution sequence is defined by rules and patterns that depend on input/output file names. These dependencies make it difficult to handle multiple dynamically generated output files, and often require the implementation of low-level output management procedures to deal with a pipeline's individual stages. Nextflow can use any data structure and outputs are not limited to files but can also include in-memory values and data objects.

Nextflow is designed specifically for bioinformaticians familiar with programming. This sets it apart from Galaxy¹⁸, which addresses the numerical stability issue with a custom package manager called Tool Shed¹⁹. Although the graphical

user interface (GUI) in Galaxy offers powerful support for *de novo* pipeline implementation by non-specialists, it also imposes a heavy development burden because any existing and validated third-party pipeline must be re-implemented and re-parameterized using the GUI. This can be very demanding in the case of elaborate pipelines, such as the Sanger Companion pipeline, which is made with a complex combination of tools. Similar reimplementations requirements also apply to other tools, including Toil²⁰.

To illustrate how Nextflow can be used, we first determined the size of the numerical stability problem for a sample complex pipeline, in this case the Sanger Companion pipeline¹⁰. We used it to carry out a gene annotation prediction for the *Leishmania*

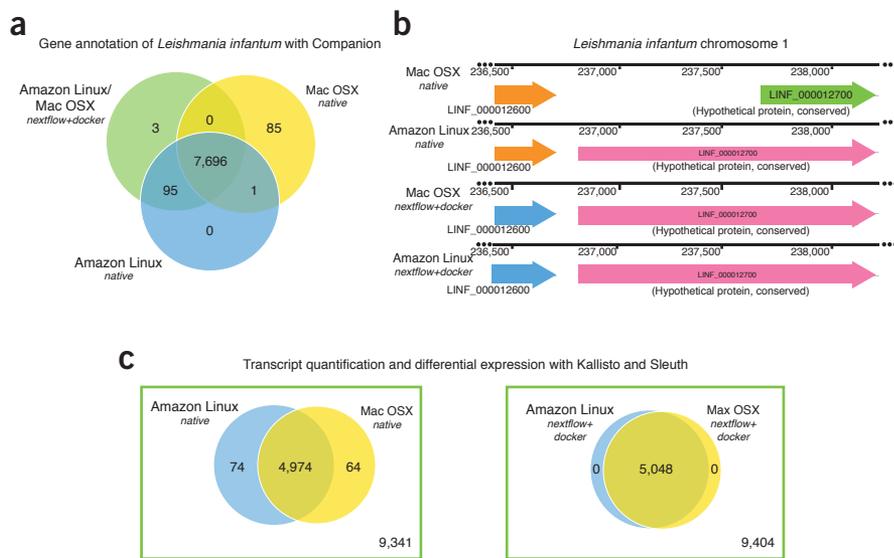


Figure 1 Nextflow enables stable analyses on different platforms. (a) *Leishmania infantum* clone JPCM5 genome annotation was carried out using either a native or a dockerized (Debian Linux) version of the Companion eukaryotic annotation pipeline. The native and dockerized versions were run on both Mac OSX and Amazon Linux platforms. The Venn diagram, plotted using the annotated genes, reveals small discrepancies when comparing the genomic coordinates of predicted coding genes, non-coding RNAs, and pseudogenes. (b) Some disparities included the annotation of the same genes at different genomic coordinates. Completely identical annotations were obtained when using the dockerized version on either the Mac OSX or Amazon Linux platforms. (c) Comparison of the Kallisto and Sleuth pipelines, applied to find differentially expressed genes (q -value < 0.01) in an RNA-seq experiment, using data from human lung fibroblasts, revealed differences when carried out on either the Mac OSX or the Amazon Linux platform. Both platforms produced identical readouts when deploying the dockerized version of the pipeline. All analyses were carried out at least twice and checked for numerical stability.

infantum genome (Supplementary Methods). Our results indicated variations across different UNIX platforms (Fig. 1a,b and Supplementary Table 1). This instability contrasts with the deterministic behavior measured on each individual platform. As the Companion pipeline was implemented in Nextflow we were able to confirm readout stability when using a dockerized version of the same pipeline across the three UNIX-like operating systems (Fig. 1a).

Gene annotation is not the only type of genomic analysis affected. We identified similar problems when using Kallisto (an expression quantification tool) combined with the Sleuth differential expression package⁹. In this case, variations in the identification of differentially expressed genes were observed when running the pipeline on two different systems (Fig. 1c, left panel). However, no such differences were observed when running a Nextflow dockerized version of the same pipeline on those systems (Fig. 1c, right panel, and Supplementary Fig. 2). Finally, a platform-dependent effect was also observed when estimating maximum likelihood trees with RaxML²¹ (Supplementary Fig. 3). Nextflow was able to control effectively those variations when using a Docker-

enabled Nextflow version of the same pipeline (Supplementary Fig. 3). All three computational experiments are available on GitHub (pipelines; <https://github.com/cbcrg/kallisto-nf-reproduce/tree/nbt-v1.0>, <https://github.com/cbcrg/raxml-nf/tree/nbt-v1.0>, <https://github.com/cbcrg/companion>) and Zenodo (data and results; https://zenodo.org/record/159153#.WMgK52_yvcs).

Nextflow is a solution to the numerical instability issue that occurs when data-analytic pipelines are used on different computational platforms. Numerical instability affects most types of *in silico* analyses, and although the overall impact of numerical stability on final readouts may seem modest, the lack of effective solutions to date has left users with a daunting challenge when it comes to choosing which platform to use. Careful monitoring of database and software versions is simply not enough. A lack of numerical stability can have problematic consequences when processing experimental data, such as compromising verification and updating of previous results. For example, in a personalized medicine application, numerical instability might result in treatment variations with unforeseen consequences.

Numerical stability is likely to be a long-standing problem²². We report here that the use of Nextflow confers numerical stability of computational workflows across commonly used clusters of computers and clouds.

Editor's note: This article has been peer-reviewed.

Note: Any Supplementary Information and Source Data files are available in the online version of the paper.

ACKNOWLEDGMENTS

We wish to thank T. Ferrar and J. Cotter for manuscript revision and helpful comments, and R. Guigo for suggesting ENCODE applications. This project was supported by the Center for Genomic Regulation and the Spanish Plan Nacional and the Spanish Ministry of Economy and Competitiveness, 'Centro de Excelencia Severo Ochoa 2013-2017'. M.C. and E.W.F. were supported by the La Caixa Foundation. We acknowledge the support of the CERCA Programme/Generalitat de Catalunya.

AUTHOR CONTRIBUTIONS

C.N. directed the work, contributed ideas, and helped to write the manuscript. P.D.T. contributed ideas, implemented the Nextflow package and helped to write the manuscript. E.P. contributed ideas and helped to implement the Nextflow package. M.C. contributed ideas, helped to implement the Nextflow package, implemented NF pipelines and helped to write the manuscript. P.P.B. contributed ideas, helped in implementing the Nextflow package, implemented NF pipelines and helped in testing the software. E.W.F. contributed ideas, helped in implementing and testing NF pipelines, and writing the manuscript.

COMPETING FINANCIAL INTERESTS

The authors declare no competing financial interests.

Paolo Di Tommaso^{1,3}, Maria Chatzou¹⁻³, Evan W Floden¹⁻³, Pablo Prieto Barja^{1,2}, Emilio Palumbo¹ & Cedric Notredame¹

¹Centre for Genomic Regulation (CRG), The Barcelona Institute for Science and Technology, Barcelona, Spain. ²Universitat Pompeu Fabra (UPF), Barcelona, Spain. ³These authors contributed equally to this work. e-mail: cedric.notredame@crg.eu

- Allison, D.B., Brown, A.W., George, B.J. & Kaiser, K.A. *Nature* **530**, 27–29 (2016).
- Editorial. *Nat. Methods* **12**, 1099 (2015).
- Editorial. *Nat. Biotechnol.* **33**, 319 (2015).
- Piccolo, S.R. & Frampton, M.B. *Gigascience* **5**, 30 (2016).
- Masca, N.G.D. *et al.* *eLife* <http://dx.doi.org/10.7554/eLife.05519> (2015).
- LeVeque, R.J., Mitchell, I.M. & Stodden, V. *Comput. Sci. Eng.* **14**, 13–17 (2012).
- Garijo, D. *et al.* *PLoS One* **8**, e80278 (2013).
- Loman, N. & Watson, M. *Nat. Biotechnol.* **31**, 996–998 (2013).
- Bray, N.L., Pimentel, H., Melsted, P. & Pachter, L. *Nat. Biotechnol.* **34**, 525–527 (2016).
- Steinbiss, S. *et al.* *Nucleic Acids Res.* **44**, 1, W29–34 (2016).
- Sadedin, S.P., Pope, B. & Oshlack, A. *Bioinformatics* **28**, 1525–1526 (2012).
- Belmann, P. *et al.* *Gigascience* **4**, 47 (2015).
- Moreews, F. *et al.* *F1000Res* **4**, 1443 (2015).
- Hosny, A., Vera-Licona, P., Laubenbacher, R. & Favre, T. *Bioinformatics* **32**, 2396–2398 (2016).
- Perkel, J. *Nature* **538**, 127–128 (2016).

16. Köster, J. & Rahmann, S. *Bioinformatics* **28**, 2520–2522 (2012).
 17. Di Tommaso, P. *et al. PeerJ* **3**, e1273 (2015).
 18. Goecks, J., Nekrutenko, A. & Taylor, J. *Genome Biol.* **11**, R86 (2010).
 19. Blankenberg, D. *et al. Genome Biol.* **15**, 403 (2014).

20. Vivian, J. *et al.* Preprint at *bioRxiv* <http://biorxiv.org/content/early/2016/07/07/062497> (2016).
 21. Stamatakis, A. *Bioinformatics* **22**, 2688–2690 (2006).
 22. Byron, S.A., Van Keuren-Jensen, K.R., Engelthaler, D.M., Carpten, J.D. & Craig, D.W. *Nat. Rev. Genet.* **17**, 257–271 (2016).

Reproducible RNA-seq analysis using *recount2*

To the Editor:

RNA sequencing (RNA-seq) is used to measure gene expression levels across the transcriptome for a huge variety of samples. For example, RNA-seq has been applied to study gene expression in individuals with rare diseases¹, in hard-to-obtain tissues² or for rare forms of cancer³. Recently, enormous RNA-seq datasets have been produced in the GTEx (Genotype-Tissue Expression) study⁴, which comprises 9,662 samples from 551 individuals and 54 body sites, and in the Cancer Genome Atlas (TCGA) study, which comprises 11,350 samples from 10,340 individuals and 33 cancer types. Public data repositories, such as the Sequence Read Archive (SRA), host >50,000 human RNA-seq samples. It is estimated that these repositories are likely to double in size every 18 months⁵. Deposited data are provided as raw sequencing reads, which are costly for standard academic labs researchers to analyze. Efforts have been made to standardize and publish ready-to-analyze summaries of both DNA sequencing⁶ and exome-sequencing⁷ data. Adopting a similar approach for archived RNA-seq data, we have developed *recount2*, which comprises >4.4 trillion uniformly processed and quantified RNA-seq reads.

Many researchers rely on processed forms of publicly available data, such as gene counts, for statistical methods development and re-analysis of candidate genes. Although these quantified data are sometimes available through the Gene Expression Omnibus⁸, there are no requirements to deposit these data, nor are data always processed with standard or complete pipelines^{9–13}. Five years ago, we began to address this problem by summarizing RNA-seq data into concise gene count tables and making these processed data and metadata available as Bioconductor¹⁴ ExpressionSet objects with one documented processing pipeline. Together this formed an RNA-seq resource named ReCount¹⁵ that contained 8 billion reads from 18 studies. ReCount was used in the development of the DESeq2 (ref. 16), voom¹⁷ and metagenomeSeq¹⁸ methods for differential

expression and normalization, compilation of co-expression networks¹⁹ and to study the effect of ribosomal DNA dosage on gene expression²⁰. The amount of archived RNA-seq data has massively increased over the past five years. To meet the needs of researchers, we have produced *recount2*, which contains >4.4 trillion uniformly processed and quantified RNA-seq reads that are derived from in excess of 70,603 human RNA-seq samples deposited in the SRA, GTEx and TCGA projects aligned with Rail-RNA^{21,22}.

The *recount2* resource summarizes expression data for genes, exons, exon–exon splice junctions and base-level coverage (**Supplementary Methods**), which enables multiple downstream analyses, including testing for differential expression of potentially unannotated transcribed sequence²³. A searchable interface is available at this site (<https://jhubiostatistics.shinyapps.io/recount/>) and via the accompanying Bioconductor package (<http://bioconductor.org/packages/recount>).

We first compared *recount2*-processed data with the publicly available data from the GTEx project, which comprises 9,662 samples from >250 individuals²⁴ to demonstrate that our processing pipeline produced gene counts similar to the published counts (**Supplementary Methods**). We downloaded the official release of the gene counts from the GTEx portal and compared them with the *recount2* gene counts (**Supplementary Note 1**, Section 4). For protein coding genes, the gene expression levels that we estimated using the *recount2* pipeline had a median (IQR) correlation of 0.987 (0.971, 0.993) with the v6 release from GTEx (**Fig. 1a** and **Supplementary Note 1**, Section 4). A differential expression analysis comparing colon and whole blood samples using the gene expression measurements from *recount2* matched the results obtained using the v6 release from the GTEx portal ($r^2 = 0.92$ between fold changes for *recount2* and GTEx v6 counts for protein coding genes; **Fig. 1b** and **Supplementary Note 1**, Section 5). These results suggest that *recount2* produces directly

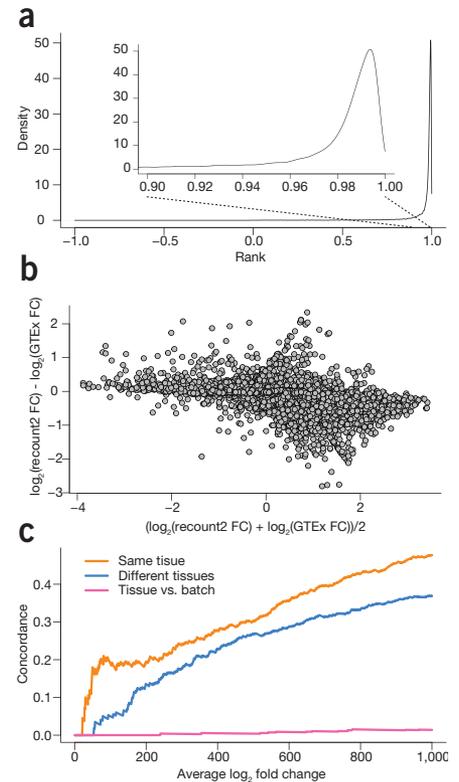


Figure 1 Meta-analysis and study comparison facilitated by *recount2*. (a) The distribution of correlations between gene expression estimates for GTEx v6 from the GTEx portal and the counts calculated in *recount2* for protein coding genes. The gene expression counts are highly correlated between both quantifications for almost all genes. (b) A comparison of the fold changes for differential expression between colon and whole blood using the quantifications from GTEx and from *recount2* for protein coding genes. The majority of genes have a similar fold-change between the two analyses. (c) A concordance versus rank (i.e., ‘concordance at the top’, CAT) plot showing comparisons between a meta-analysis tissue comparison of whole blood and colorectal tissue in data from the sequence read archive and the GTEx project. When comparing the same tissues, there is a strong concordance between differential expression results on public data and GTEx (orange), less concordance when different tissues are compared (blue) and almost none when comparing different analyses (pink).

comparable gene counts to one of the largest published studies.

The advantage of using the *recount2* version of GTEx data is that all data are identically processed, therefore enabling integrated analyses of multiple datasets. To illustrate how *recount2* can be used to investigate or validate cross-tissue differences using publicly available data, we computed expression differences comparing samples from healthy colon tissue and whole blood from healthy individuals (**Supplementary Methods, Supplementary Note 2**, Section 1.8 and **Supplementary**