

GigaScience

Streamlining Data-Intensive Biology with Workflow Systems

--Manuscript Draft--

Manuscript Number:	GIGA-D-20-00245	
Full Title:	Streamlining Data-Intensive Biology with Workflow Systems	
Article Type:	Review	
Funding Information:	Gordon and Betty Moore Foundation (GBMF4551)	Dr. C. Titus Brown
	STATE AND FEDERAL CONTRACTORS WATER AGENCY (A19- 1844)	Not applicable
	National Science Foundation (1711984)	Dr. N. Tessa Pierce
Abstract:	<p>As the scale of biological data generation has increased, the bottleneck of research has shifted from data generation to analysis. Researchers commonly need to build computational workflows that include multiple analytic tools and require incremental development as experimental insights demand tool and parameter modifications. These workflows can produce hundreds to thousands of intermediate files and results that must be integrated for biological insight. Data-centric workflow systems that internally manage computational resources, software, and conditional execution of analysis steps are reshaping the landscape of biological data analysis, and empowering researchers to conduct reproducible analyses at scale. Adoption of these tools can facilitate and expedite robust data analysis, but knowledge of these techniques is still lacking. Here, we provide a series of practices and strategies for leveraging workflow systems with structured project, data, and resource management to streamline large-scale biological analysis.</p>	
Corresponding Author:	Nuri Teresa Pierce, Ph.D University of California Davis Davis, CA UNITED STATES	
Corresponding Author Secondary Information:		
Corresponding Author's Institution:	University of California Davis	
Corresponding Author's Secondary Institution:		
First Author:	Taylor Reiter	
First Author Secondary Information:		
Order of Authors:	Taylor Reiter	
	Phillip T. Brooks	
	Luiz Irber	
	Shannon E.K. Joslin	
	Charles M. Reid	
	Camille Scott	
	C. Titus Brown	
	N. Tessa Pierce	
Order of Authors Secondary Information:		
Additional Information:		
Question	Response	
Are you submitting this manuscript to a	No	

<p>special series or article collection?</p>	
<p>Experimental design and statistics</p> <p>Full details of the experimental design and statistical methods used should be given in the Methods section, as detailed in our Minimum Standards Reporting Checklist. Information essential to interpreting the data presented should be made available in the figure legends.</p> <p>Have you included all the information requested in your manuscript?</p>	<p>Yes</p>
<p>Resources</p> <p>A description of all resources used, including antibodies, cell lines, animals and software tools, with enough information to allow them to be uniquely identified, should be included in the Methods section. Authors are strongly encouraged to cite Research Resource Identifiers (RRIDs) for antibodies, model organisms and tools, where possible.</p> <p>Have you included the information requested as detailed in our Minimum Standards Reporting Checklist?</p>	<p>Yes</p>
<p>Availability of data and materials</p> <p>All datasets and code on which the conclusions of the paper rely must be either included in your submission or deposited in publicly available repositories (where available and ethically appropriate), referencing such data using a unique identifier in the references and in the “Availability of Data and Materials” section of your manuscript.</p> <p>Have you have met the above requirement as detailed in our Minimum Standards Reporting Checklist?</p>	<p>Yes</p>

--	--

Streamlining Data-Intensive Biology With Workflow Systems

This manuscript ([permalink](#)) was automatically generated from [dib-lab/2020-workflows-paper@d401e38](#) on August 5, 2020.

Authors

- **Taylor Reiter** [0000-0002-7388-421X](#) · [taylorreiter](#) · [ReiterTaylor](#) Department of Population Health and Reproduction, University of California, Davis · Funded by Moore Foundation GBMF4551
- **Phillip T. Brooks** [0000-0003-3987-244X](#) · [brooksph](#) · [brooksph](#) Department of Population Health and Reproduction, University of California, Davis
- **Luiz Irber** [0000-0003-4371-9659](#) · [luizirber](#) · [luizirber](#) Department of Population Health and Reproduction, University of California, Davis · Funded by Moore Foundation GBMF4551
- **Shannon E.K. Joslin** [0000-0001-5470-1193](#) · [shannonekj](#) · [IntrprtnGnmcs](#) Department of Animal Science, University of California, Davis · Funded by State and Federal Water Contractors A19-1844
- **Charles M. Reid** [0000-0002-7337-8894](#) · [charlesreid1](#) Department of Population Health and Reproduction, University of California, Davis · Funded by Moore Foundation GBMF4551
- **Camille Scott** [0000-0001-8822-8779](#) · [camillescott](#) · [camille_codon](#) Department of Population Health and Reproduction, University of California, Davis · Funded by Moore Foundation GBMF4551
- **C. Titus Brown** [0000-0001-6001-2677](#) · [ctb](#) · [ctitusbrown](#) Department of Population Health and Reproduction, University of California, Davis · Funded by Moore Foundation GBMF4551
- **N. Tessa Pierce** [0000-0002-2942-5331](#) · [bluegenes](#) · [saltyscientist](#) Department of Population Health and Reproduction, University of California, Davis · Funded by NSF 1711984

1 **Abstract**

2 As the scale of biological data generation has increased, the bottleneck of research has shifted from data
3 generation to analysis. Researchers commonly need to build computational workflows that include
4 multiple analytic tools and require incremental development as experimental insights demand tool and
5 parameter modifications. These workflows can produce hundreds to thousands of intermediate files and
6 results that must be integrated for biological insight. Data-centric workflow systems that internally
7 manage computational resources, software, and conditional execution of analysis steps are reshaping the
8 landscape of biological data analysis, and empowering researchers to conduct reproducible analyses at
9 scale. Adoption of these tools can facilitate and expedite robust data analysis, but knowledge of these
10 techniques is still lacking. Here, we provide a series of practices and strategies for leveraging workflow
11 systems with structured project, data, and resource management to streamline large-scale biological
12 analysis.

13

14 **Author Summary**

15 We present a guide for workflow-enabled biological sequence data analysis, developed through our own
16 teaching, training and analysis projects. We recognize that this is based on our own use cases and
17 experiences, but we hope that our guide will contribute to a larger discussion within the open source and
18 open science communities and lead to more comprehensive resources. Our main goal is to accelerate the
19 research of scientists conducting sequence analyses by introducing them to organized workflow practices
20 that not only benefit their own research but also facilitate open and reproducible science.

21 **Introduction**

22 Biological research has become increasingly computational. In particular, genomics has experienced a
23 deluge of high-throughput sequencing data that has already reshaped our understanding of the diversity
24 and function of organisms and communities, building basic understanding from ecosystems to human
25 health. The analysis workflows used to produce these insights often integrate hundreds of steps and
26 involve a myriad of decisions ranging from small-scale tool and parameter choices to larger-scale design
27 decisions around data processing and statistical analyses. Each step relies not just on analysis code written
28 by the researcher, but on third-party software, its dependencies, and the compute infrastructure and
29 operating system on which the code is executed. Historically, this has led to the patchwork availability of
30 underlying code for analyses as well as a lack of interoperability of the resulting software and analysis
31 pipelines across compute systems [1]. Combined with unmet training needs in biological data analysis,
32 these conditions undermine the reuse of data and the reproducibility of biological research, vastly limiting
33 the value of our generated data [2].

34 The biological research community is strongly committed to addressing these issues, recently formalizing
35 the FAIR practices: the idea that all life sciences research (including data and analysis workflows) should
36 be Findable, Accessible, Interoperable, and Reusable [3]. For computational analyses, these ideals are
37 readily achievable with current technology, but implementing them in practice has proven difficult,
38 particularly for biologists with little training in computing [3]. However, the recent maturation of data-
39 centric workflow systems designed to automate and facilitate computational workflows is expanding our
40 capacity to conduct end-to-end FAIR analyses [5]. These workflow systems are designed to handle some
41 aspects of computational workflows internally: namely, the interactions with software and computing
42 infrastructure, and the ordered execution of each step of an analysis. By reducing the manual input and
43 monitoring required at each analysis juncture, these integrated systems ensure that analyses are repeatable
44 and can be executed at much larger scales. In concert, the standardized information and syntax required

45 for rule-based workflow specification makes code inherently modular and more easily transferable
46 between projects [5,6]. For these reasons, workflow systems are rapidly becoming the workhorses of
47 modern bioinformatics.

48 Adopting workflow systems requires some level of up-front investment, first to understand the structure
49 of the system, and then to learn the workflow-specific syntax. These challenges can preclude adoption,
50 particularly for researchers without significant computational experience [4]. In our experiences with both
51 research and training, these initial learning costs are similar to those required for learning more traditional
52 analysis strategies, but then provide a myriad of additional benefits that both facilitate and accelerate
53 research. Furthermore, online communities for sharing reusable workflow code have proliferated,
54 meaning the initial cost of encoding a workflow in a system is mitigated via use and re-use of common
55 steps, leading to faster time-to-insight [5,7].

56 Building upon the rich literature of “best” and “good enough” practices for computational biology
57 [8,9,10], we present a series of strategies and practices for adopting workflow systems to streamline data-
58 intensive biology research. This manuscript is designed to help guide biologists towards project, data, and
59 resource management strategies that facilitate and expedite reproducible data analysis in their research.
60 We present these strategies in the context of our own experiences working with high-throughput
61 sequencing data, but many are broadly applicable to biologists working beyond this field.

62 **Workflows facilitate data-intensive biology**

63 Data-intensive biology typically requires that researchers execute computational workflows using
64 multiple analytic tools and apply them to many experimental samples in a systematic manner. These
65 workflows commonly produce hundreds to thousands of intermediate files and require incremental
66 changes as experimental insights demand tool and parameter modifications. Many intermediate steps are
67 central to the biological analysis, but others, such as converting between file formats, are rote

68 computational tasks required to passage data from one tool to the next. Some of these steps can fail
69 silently, producing incomplete intermediate files that imperceptively invalidate downstream results and
70 biological inferences. Properly managing and executing all of these steps is vital, but can be both time-
71 consuming and error-prone, even when automated with scripting languages such as bash.

72 The emergence and maturation of workflow systems designed with bioinformatic challenges in mind has
73 revolutionized computing in data intensive biology [11]. Workflow systems contain powerful
74 infrastructure for workflow management that can coordinate runtime behavior, self-monitor progress and
75 resource usage, and compile reports documenting the results of a workflow (**Figure 1**). These features
76 ensure that the steps for data analysis are minimally documented and repeatable from start to finish. When
77 paired with proper software management, fully-contained workflows are scalable, robust to software
78 updates, and executable across platforms, meaning they will likely still execute the same set of commands
79 with little investment by the user after weeks, months, or years.

80

81 *Figure 1: **Workflow Systems:** Bioinformatic workflow systems have built-in functionality that facilitates*
82 *and simplifies running analysis pipelines. **A. Samples:** Workflow systems enable you to use the same code*
83 *to run each step on each sample. Samples can be easily added if the analysis expands. **B. Software***
84 ***Management:** Integration with software management tools (e.g. conda, singularity, docker) can automate*
85 *software installation for each step. **C. Branching, D. Parallelization, and E. Ordering:** Workflow*
86 *systems handle conditional execution, ensuring that tasks are executed in the correct order for each*
87 *sample file, including executing independent steps in parallel if possible given the resources provided. **F.***
88 ***Standard Steps:** Many steps are now considered “standard” (e.g. quality control). Workflow languages*
89 *keep all information for a step together and can be written to enable you to remix and reuse individual*
90 *steps across pipelines. **G. Rerun as necessary:** Workflow systems keep track of which steps executed*
91 *properly and on which samples, and allow you to rerun failed steps (or additional steps) rather than re-*

92 *executing the entire workflow. **H. Reporting:** Workflow languages enable comprehensive reporting on*
93 *workflow execution and resource utilization by each tool. **I. Portability:** Analyses written in workflow*
94 *languages (with integrated software management) can be run across computing systems without changes*
95 *to code.*

96 To properly direct an analysis, workflow systems need to encode information about the relationships
97 between every workflow step. In practice, this means that each analysis step must specify the input (or
98 types of inputs) needed for that step, and the output (or types of outputs) being produced. This structure
99 provides several additional benefits. First, workflows become minimally self-documented, as the directed
100 graph produced by workflow systems can be exported and visualized, producing a graphical
101 representation of the relationships between all steps in a pipeline (see **Figure 5**). Next, workflows are
102 more likely to be fully enclosed without undocumented steps that are executed by hand, meaning analyses
103 are more likely to be reproducible. Finally, each step becomes a self-contained unit that can be used and
104 re-used across multiple analysis workflows, so scientists can spend less time implementing standard steps,
105 and more time on their specific research questions. In sum, the internal scaffolding provided by workflow
106 systems helps build analyses that are generally better documented, repeatable, transferable, and scalable.

107 **Getting started with workflows**

108 The workflow system you choose will be largely dependent on your analysis needs. Here, we draw a
109 distinction between two types of workflows: “research” workflows that are under iterative development to
110 answer novel scientific questions, and “production” workflows, which have reached maturity and are
111 primarily used to run a standard analysis on new samples. In particular, research workflows require
112 flexibility and assessment at every step: outliers and edge cases may reveal interesting biological
113 differences, rather than sample processing or technical errors. Many workflow systems can be used for
114 either type, but we note cases where their properties facilitate one of these types over the other.

115 **Using workflows without learning management systems** While the benefits of encoding a workflow in
116 a workflow system are immense, the learning curve associated with implementing complete workflows in
117 a new syntax can be daunting. It is possible to obtain the benefits of workflow systems without learning a
118 workflow system. Websites like Galaxy, Cavatica, and EMBL-EBI MGnify offer online portals in which
119 users build workflows around publicly-available or user-uploaded data [12,13,14]. On the command line,
120 many research groups have used workflow systems to build user-friendly pipelines that do not require
121 learning or working with the underlying workflow software. These tools are specified in an underlying
122 workflow language, but are packaged in a user-friendly command-line script that coordinates and
123 executes the workflow. Rather than writing each workflow step, the user can specify data and parameters
124 in a configuration file to customize the run. Some examples include the nf-core RNA-seq pipeline [1,15],
125 the ATLAS metagenome assembly and binning pipeline [16,17], the Sunbeam metagenome analysis
126 pipeline [18,19], and two from our own lab, the dammit eukaryotic transcriptome annotation pipeline [20]
127 and the elvers *de novo* transcriptome pipeline [21]. These tools allow users to take advantage of the
128 benefits of workflow software without needing to invest in curating and writing their own pipeline. The
129 majority of these workflows are production-level workflows designed to execute a series of standard
130 steps, but many provide varying degrees of customizability ranging from tool choice to parameter
131 specification.

132 **Choosing a workflow system** If your use case extends beyond these tools, there are several scriptable
133 workflow systems that offer comparable benefits for carrying out your own data-intensive analyses. Each
134 has its own strengths, meaning each workflow software will meet an individual's computing goals
135 differently (see **Table 1**). Our lab has adopted Snakemake, in part due to its similarity and integration
136 with Python, its flexibility for building and testing new analyses in different languages, and its intuitive
137 integration with software management tools (described below)[22]. Snakemake and Nextflow are
138 commonly used for developing new research pipelines, where flexibility and iterative, branching
139 development is a key feature [23]. Common Workflow Language (CWL) and Workflow Description

140 Language (WDL) are workflow specification formats that are more geared towards scalability, making
 141 them ideal for production-level pipelines with hundreds of thousands of samples [24]. WDL and CWL are
 142 commonly executed on platforms such as Terra [25] or Seven Bridges Platform [26]. Language-specific
 143 workflow systems, such as ROpenSci’s Drake [27], are limited in the scope of tasks they can execute, but
 144 are powerful within their language and easier to integrate for those comfortable with that language.

145 *Table 1: Four of the most widely used bioinformatics workflow systems (2020), with links to*
 146 *documentation, example workflows, and general tutorials. In many cases, there may be tutorials online*
 147 *that are tailored for use cases in your field. All of these systems can interact with tools or tasks written in*
 148 *other languages and can function across cloud computing systems and high-performance computing*
 149 *clusters. Some can also import full workflows from other specification languages.*

Workflow System	Documentation	Example Workflow	Tutorial
Snakemake	snakemake.readthedocs.io	https://github.com/snakemake-e-workflows/chipseq	https://snakemake.readthedocs.io/en/stable/tutorial/tutorial.html
Nextflow	www.nextflow.io	https://github.com/nf-core/sarek	https://www.nextflow.io/docs/latest/getstarted.html
Common workflow language	www.commonwl.org	https://github.com/EBI-Metagenomics/pipeline-v5	https://www.commonwl.org/user_guide/02-1st-example/index.html
Workflow description language	openwdl.org	https://github.com/gatk-workflows/gatk4-data-processing	https://support.terra.bio/hc/en-us/articles/360037127992-1-howto-Write-your-first-WDL-script-running-GATK-HaplotypeCaller

150 The best workflow system to choose may be the one with a strong and accessible local or online
 151 community in your field, somewhat independent of your computational needs. The availability of field-
 152 specific data analysis code for reuse and modification can facilitate the adoption process, as can
 153 community support for new users. Fortunately, the standardized syntax required by workflow systems,
 154 combined with widespread adoption in the open science community, has resulted in a proliferation of
 155 open access workflow-system code for routine analysis steps [28,29]. At the same time, consensus
 156 approaches for data analysis are emerging, further encouraging reuse of existing code [30,31,32,33,34].

157 The [Getting started developing workflows](#) section contains strategies for modifying and developing
158 workflows for your own analyses.

159 **Wrangling Scientific Software**

160 Analysis workflows commonly rely on multiple software packages to generate final results. These tools
161 are heterogeneous in nature: they are written by researchers working in different coding languages, with
162 varied approaches to software design and optimization, and often for specific analysis goals. Each
163 program has a number of other programs it depends upon to function (“dependencies”), and as software
164 changes over time to meet research needs, the results may change, even when run with identical
165 parameters. As a result, it is critical to take an organized approach to installing, managing, and keeping
166 track of software and software versions. To meet this need, most workflow managers integrate with
167 software management systems like conda, singularity, and docker [[11,35,36](#)].

168 Software management systems perform some combination of software installation, management, and
169 packaging that alleviate problems that arise from dependencies and facilitate documentation of software
170 versions. On many compute systems, system-wide software management is overseen by system
171 administrators, who ensure commonly-used and requested software is installed into a “module” system
172 available to all users. Unfortunately, this system does not lend itself well for exploring new workflows
173 and software, as researchers do not have permission to install software themselves. The Conda package
174 manager has emerged as a leading solution, largely because it handles both cluster permission and version
175 conflict issues with a user-based software environment system, and features a straightforward “recipe”
176 system which simplifies the process of making new software installable (**Figure 2**). Conda enables
177 lightweight software installation and can be used with the same commands across platforms, but can still
178 be impacted by differences in the host operating system. Alternatively, wrapping software environments
179 in “containers” that capture and reproduce all other aspects of the runtime environment can enhance

180 reproducibility over time [3]. Container-based software installation via docker and singularity is common
181 for production-level workflows.

182

183 *Figure 2: The conda package and environment manager simplifies software installation and*
184 *management. A. Conda Recipe Repositories: Each program distributed via Conda has a “recipe”*
185 *describing all software dependencies needed for installation using Conda (each of which must also be*
186 *installable via Conda). Recipes are stored and managed in the cloud in separate “channels”, some of*
187 *which specialize in particular fields or languages (e.g. the “bioconda” channel specializes in*
188 *bioinformatic software, the “r” channel specializes in R language packages) [11]. B. Use Conda*
189 *Environments to Avoid Installation Conflicts: Conda does not require root privileges for software*
190 *installation, thus enabling use by researchers working on shared cluster systems. However, even user-*
191 *based software installation can encounter dependency conflicts. For example, you might need to use*
192 *python2 to install and run a program (e.g. older scripts written by members of your lab), while also using*
193 *snakemake to execute your workflows (requires python \geq 3.5). By installing each program into an*
194 *isolated “environment” that contains only the software required to run that program, you can ensure all*
195 *programs used throughout your analysis will run without issue. Using small, separate environments for*
196 *your software and building many simple environments to accommodate different steps in your workflow*
197 *also reduces the amount of time it takes conda to resolve dependency conflicts between different software*
198 *tools (“solve” an environment). Conda virtual environments can be created and installed either on the*
199 *command line, or via an environment YAML file, as shown. In this case, the environment file also*
200 *specifies which Conda channels to search and download programs from. When specified in a YAML file,*
201 *conda environments are easily transferable between computers and operating systems. Further, because*
202 *the version of each package installed in an environment is recorded, workflow reproducibility is*
203 *enhanced. Although portions of Conda may be superseded by alternative solutions [37], this model of*
204 *software installation and management will likely persist.*

205 **Getting started with software management**

206 *Using software without learning management systems* While package managers and containers greatly
207 increase reproducibility, there are a number of ways to test software before needing to worry about
208 installation. Some software packages are available as web-based tools and through a series of data upload
209 and parameter specifications, allow the user to interact with a tool that is running on a back-end server.
210 Integrated development environments (IDE) like PyCharm and RStudio can manage software installation
211 for language-specific tools, and can be very helpful when writing analysis code. These approaches are
212 ideal for testing a tool to determine whether it produces useful output on your data before integration with
213 your reproducible workflow.

214 **Integrating software management within workflows** Workflow systems provide seamless integration
215 with software management tools. Each workflow system requires different specification for initiation of
216 software management, but typically requires about one additional line of code per step that requires the
217 use of software. If the software management tool is installed locally, the workflow will automatically
218 download and install the specified environment or container and use it for specified step.

219 In our experience, the complete solution for using scientific software involves starting with a combination
220 of interactive and exploratory analyses in IDEs and local conda installation to develop an analysis
221 strategy and create an initial workflow. This is then followed by workflow-integrated software
222 management via conda, singularity, or docker for executing the resulting workflow on many samples.

223 **Workflow-Based Project Management**

224 Project management, the strategies and decisions used to keep a project organized, documented,
225 functional, and shareable, is foundational to any research program. Clear organization and management is
226 a learned skill that takes time to implement. Workflow systems both simplify and improve computational

227 project management, but even workflows that are fully specified in workflow systems require additional
228 investment to stay organized, documented, and backed up.

229 **Systematically document your workflows**

230 Pervasive documentation provides indispensable context for biological insights derived from an analysis,
231 facilitates transparency in research, and increases reusability of the analysis code. Good documentation
232 covers all aspects of a project, including file and results organization, clear and commented code, and
233 accompanying explanatory documents for design decisions and metadata. Workflow systems facilitate
234 building this documentation, as each analysis step (with chosen parameters) and the links between those
235 steps are completely specified within the workflow syntax. This feature streamlines code documentation,
236 particularly if you include as much of the analysis as possible within the automated workflow framework.
237 Outside of the analysis itself, applying consistent organizational design can capitalize on the structure and
238 automation provided by workflows to simplify the generation of quality documentation for all aspects of
239 your project. Below, we discuss project management strategies for building reproducible workflow-
240 enabled biological analyses.

241 **Use consistent, self-documenting names**

242 Using consistent and descriptive identifiers for your files, scripts, variables, workflows, projects, and even
243 manuscripts helps keep your projects organized and interpretable for yourself and collaborators. For
244 workflow systems, this strategy can be implemented by tagging output files with a descriptive identifier
245 for each analysis step, either in the filename or by placing output files within a descriptive output folder.
246 For example, the file shown in **Figure 3** has been preprocessed with a quality control trimming step. For
247 large workflows, placing results from each step of your analysis in isolated, descriptive folders can be
248 essential for keeping your project workspace clean and organized.

249

250 Figure 3: **Consistent and informative file naming improves organization and interpretability.** For
251 ease of grouping and referring to input files, it is useful to keep unique sample identification in the
252 filename, often with a metadata file explaining the meaning of each unique descriptor. For analysis
253 scripts, it can help to implement a numbering scheme, where the name of first file in the analysis begins
254 with “00”, the next with “01”, etc. For output files, it can help to add a short, unique identifier to output
255 files processed with each analysis step. This particular file is a RAD sequencing fastq file of a fish species
256 that has been preprocessed with a fastq quality trimming tool.

257 **Store workflow metadata with the workflow**

258 Developing biological analysis workflows can involve hundreds of small decisions: What parameters
259 work best for each step? Why did you use a certain reference file for annotation as compared with other
260 available files? How did you finally manage to get around the program or installation error? All of these
261 pieces of information contextualize your results and may be helpful when sharing your findings. Keeping
262 information about these decisions in an intuitive and easily accessible place helps you find it when you
263 need it. To capitalize on the utility of version control systems described below, it is most useful to store
264 this information in plain text files. Each main directory of a project should include notes on the data or
265 scripts contained within, so that a collaborator could look into the directory and understand what to find
266 there (especially since that “collaborator” is likely to be you, a few months from now!). Code itself can
267 contain documentation - you can include comments with the reasoning behind algorithm choice or include
268 a link to the seqanswers post that helped you decide how to shape your differential expression analysis.
269 Larger pieces of information can be kept in “README” or notes documents kept alongside your code
270 and other documents. For example, a GitHub repository documenting the reanalysis of the Marine
271 Microbial Eukaryote Transcriptome Sequencing Project uses a README alongside the code to document

272 the workflow and digital object identifiers for data products [38,39]. While this particular strategy cannot
273 be automated, it is critical for interpreting the final results of your workflow.

274 **Document data and analysis exploration using computational notebooks**

275 Computational notebooks allow users to combine narrative, code, and code output (e.g. visualizations) in
276 a single location, enabling the user to conduct analysis and visually assess the results in a single file (see
277 **Figure 4**). These notebooks allow for fully documented iterative analysis development, and are
278 particularly useful for data exploration and developing visualizations prior to integration into a workflow
279 or as a report generated by a workflow that can be shared with collaborators.

280

281 *Figure 4: **Examples of computational notebooks.** Computational notebooks allow the user to mix text,*
282 *code, and results in one document. **Panel A.** shows an RMarkdown document viewed in the RStudio*
283 *integrated development environment, while **Panel B.** shows a rendered HTML file produced by knitting*
284 *the RMarkdown document [40]. **Panel C.** shows a Jupyter Notebook, where code, text, and results are*
285 *rendered inline as each code chunk is executed [41]. The second grey chunk is a raw Markdown chunk*
286 *with text that will be rendered inline when executed. Both notebooks generate a histogram of a metadata*
287 *feature, number of generations, from a long-term evolution experiment with Escherichia coli [42].*
288 *Computational notebooks facilitate sharing by packaging narrative, code, and visualizations together.*
289 *Computational notebooks can be further packaged with tools like Binder [43]. Binder builds an*
290 *executable environment (capable of running RStudio and Jupyter notebooks) out of a GitHub repository*
291 *using package management systems and docker to build reproducible and executable software*
292 *environments as specified in the repository. Binders can be shared with collaborators (or students in a*
293 *classroom setting), and analysis and visualization can be ephemerally reproduced or altered from the*
294 *code provided in computational notebooks.*

295

296 **Visualize your workflow**

297 Visual representations can help illustrate the connections in a workflow and improve the readability and
298 reproducibility of your project. At the highest level, flowcharts that detail relationships between steps of a
299 workflow can help provide big-picture clarification, especially when the pipeline is complicated. For
300 individual steps, a graphical representation of the output can show the status of the project or provide
301 insight on additional analyses that should be added. For example, **Figure 5** exhibits a modified
302 Snakemake workflow visualization from an RNA-seq quantification pipeline [44].

303

304 *Figure 5: A directed acyclic graph (DAG) that illustrates connections between all steps of a sequencing*
305 *data analysis workflow. Each box represents a step in the workflow, while lines connect sequential steps.*
306 *The DAG shown in this figure illustrates a real bioinformatics workflow for RNA-seq quantification was*
307 *generated by modifying the default Snakemake workflow DAG. While the workflow is complex, it is*
308 *coordinated by a workflow system that alleviates the need for a user to directly manage file*
309 *interdependencies.*

310 **Version control your project**

311 As your project develops, version control allows you to keep track of changes over time. You may
312 already do this in some ways, perhaps with frequent hard drive backups or by manually saving different
313 versions of the same file - e.g. by appending the date to a script name or appending “version_1” or
314 “version_FINAL” to a manuscript draft. For computational workflows, using version control systems
315 such as Git or Mercurial can be used to keep track of all changes over time, even across multiple systems,
316 scripting languages, and project contributors (see **Figure 6**). If a key piece of a workflow inexplicably
317 stops working, consistent version control can allow you to rewind in time and identify differences from
318 when the pipeline worked to when it stopped working. Backing up your version controlled analysis in an

319 online repository such as GitHub, GitLab, or Bitbucket provides critical insurance as you iteratively
320 modify and develop your workflow.

321

322 *Figure 6: **Version Control** Version control systems (e.g. Git, Mercurial) work by storing incremental*
323 *differences in files from one saved version (“commit”) to the next. To visualize the differences between*
324 *each version, text editors such as Atom and online services such as GitHub, GitLab and Bitbucket use red*
325 *highlighting to denote deletions, and green highlighting to denote additions. In this trivial example, a*
326 *typo in version 1 (in red) was corrected in version 2 (in green). These systems are extremely useful for*
327 *code and manuscript development, as it is possible to return to the snapshot of any saved version. This*
328 *means that version control systems save you from accidental deletions, preserve code you thought you no*
329 *longer needed and preserve a record of project changes over time.*

330 When combined with online backups, version control systems also facilitate code and data availability
331 and reproducibility for publication. For example, to preserve the version of code that produced published
332 results, you can create a “release”: a snapshot of the current code and files in a GitHub repository. You
333 can then generate a digital object identifier (DOI) for that release using a permanent documentation
334 service such as Zenodo ([\[45\]](#)) and make it available to reviewers and beyond (see “sharing” section,
335 below).

336 **Share your workflow and analysis code**

337 Sharing your workflow code with collaborators, peer reviewers, and scientists seeking to use a similar
338 method can foster discussion and review of your analysis. Sticking to a clear documentation strategy,
339 using a version control system, and packaging your code in notebooks or as a workflow prepare them to
340 be easily shared with others. To go one step further, you can package your code with tools like Binder,
341 ReproZip, or Whole Tale, or make interactive visualizations with tools like Shiny apps or Plotly. These

342 approaches let others run the code on cloud computers in environments identical to those in which the
343 original computation was performed (**Figure 4, Figure 7**) [43,46,47]. These tools substantially reduce
344 overhead associated with interacting with code and data, and in doing so, make it fast and easy to rerun
345 portions of the analysis, check accuracy, or even tweak the analysis to produce new results. If you also
346 share your code and workflows publicly, you will also help contribute to the growing resources for open
347 workflow-enabled biological research.

348

349 *Figure 7: Interactive visualizations facilitate sharing and repeatability. A. Interactive visualization*
350 *dashboard in the Pavian Shiny app for metagenomic analysis [48,49]. Shiny allows you to build*
351 *interactive web pages using R code. Data is manipulated by R code in real-time in a web page, producing*
352 *analysis and visualizations of a data set. Shiny apps can contain user-specifiable parameters, allowing a*
353 *user to control visualizations or analyses. As seen above, sample “PT1” is selected, and taxonomic ranks*
354 *class and order are excluded. Shiny apps allow collaborators who may or may not know R to modify R*
355 *visualizations to fit their interests. B. Plotly heatmap of transcriptional profiling in human brain samples*
356 *[50]. Hovering over a cell in the heatmap displays the sample names from the x and y axis, as well as the*
357 *intensity value. Plotting tools like plotly and vega-lite produce single interactive plots that can be shared*
358 *with collaborators or integrated into websites [51,52]. Interactive visualizations are also helpful in*
359 *exploratory data analysis.*

360 **Getting started developing workflows**

361 In our experience, the best way to have your workflow system work *for* you is to include as much of your
362 analysis as possible within the automated workflow framework, use self-documenting names, include
363 analysis visualizations, and keep rigorous documentation alongside your workflow that enables you to
364 understand each decision and entirely reproduce any manual steps. Some of the tools discussed above will

365 inevitably change over time, but these principles apply broadly and will help you design clear, well-
366 documented, and reproducible analyses. Ultimately, you will need to experiment with strategies that work
367 for you – what is most important is to develop a clear set of strategies and implement them tenaciously.
368 Below, we provide a few practical strategies to try as you begin developing your own workflows.

369 **Start with working code** When building a workflow for the first time, creating an initial workflow based
370 on a subset of your sample data can help verify that the workflow, tools, and command line syntax
371 function at a basic level. This functioning example code then provides a reliable workflow framework
372 free of syntax errors which you can customize for your data without the overhead of generating correct
373 workflow syntax from scratch. **Table 1** provides links to official repositories containing tutorials and
374 example biological analysis workflows, and workflow tutorials and code sharing websites like GitHub,
375 GitLab, and Bitbucket have many publicly available workflows for other analyses. If a workflow is
376 available through Binder, you can test and experiment with workflow modification on Binder’s cloud
377 system without needing to install a workflow manager or software management tool on your local
378 compute system [43].

379 **Test with subsampled data** While a workflow may run on test data, this is not a guarantee it will run on
380 all data. After verifying your chosen example workflow is functional, try running it with your own data or
381 some public data related to your species or condition of interest. If your analysis allows, trying the
382 workflow on a small subset of the data first can save time, energy, and computational resources. For
383 example, if working with FASTQ data, you can subsample the first million lines of a file (first 250k
384 reads) by running:

```
385 head -n 1000000 FASTQ_FILE.fq > test_fastq.fq
```

386 While there are many more sophisticated ways to subsample reads, this technique should be sufficient for
387 testing each step of a most workflows prior to running your full dataset. In specific cases, such as
388 eukaryotic genome assembly, you may need to be more intentional with how you subsample reads.

389 **Document your process** Document your changes, explorations, and errors as you develop. We
390 recommend using the Markdown language so your documentation is in plain text to facilitate version
391 control, but can still include helpful visual headings, code formatting, and embedded images. Markdown
392 editors with visual previewing, such as HackMD, can greatly facilitate notetaking, and Markdown
393 documents are visually rendered properly within your online version control backups on services such as
394 GitHub [\[53\]](#).

395 **Develop your workflow** From your working code, iteratively modify and add workflow steps to meet
396 your data analysis needs. This strategy allows you to find and fix mistakes on small sections of the
397 workflow. Periodically clean your output directory and rerun the entire workflow, to ensure all steps are
398 fully interoperable (using small test data will improve the efficiency of this step!). If possible, using mock
399 or control datasets can help you verify that the analysis you are building actually returns correct biological
400 results. Tutorials and tool documentation are useful companions during development; as with any
401 language, remembering workflow-specific syntax takes time and practice.

402 **Assess your results** Evaluate your workflow results as you go. Consider what aspects (e.g. tool choice,
403 program parameters) can be evaluated rigorously, and assess each step for expected behavior. Other
404 aspects (e.g. filtering metadata, joining results across programs or analysis, software and workflow bugs)
405 will be more difficult to evaluate. Wherever possible, set up positive and negative controls to ensure your
406 analysis is performing the desired analysis properly. If you're certain an analysis is executing as designed,
407 tracking down unusual results may reveal interesting biological differences.

408 **Back up early and often** As you write new code, back up your changes in an online repository such as
409 GitHub, GitLab, or Bitbucket. These services support both drag-and-drop and command line interaction.
410 Data backup will be discussed in the next section, Data and resource management for workflow-enabled
411 biology.

412 **Scale up your workflow** Bioinformatic tools vary in the resources they require: some analysis steps are
413 compute-intensive, other steps are memory intensive, and still others will have large intermediate storage
414 needs. If using high-performance computing system or the cloud, you will need to request resources for
415 running your pipeline, often provided as a simultaneous execution limit or purchased by your research
416 group on a cost-per-compute basis. Workflow systems provide built-in tools to monitor resource usage for
417 each step. Running a complete workflow on a single sample with resource monitoring enabled generates
418 an estimate of computational resources needed for each step. These estimates can be used to set
419 appropriate resource limits for each step when executing the workflow on your remaining samples.
420 Strategies for resource management will be addressed in the next section, Data and resource management
421 for workflow-enabled biology.

422 **Find a community and ask for help when you need it** Local and online users groups are helpful
423 communities when learning a workflow language. When you are first learning, help from more advanced
424 users can save you hours of frustration. After you've progressed, providing that same help to new users
425 can help you cement the syntax in your mind and tackle more advanced uses. Data-centric workflow
426 systems have been enthusiastically adopted by the open science community, and as a consequence, there
427 is a critical mass of tutorials and open access code, as well as code discussion on forums and via social
428 media, particularly Twitter. Post in the relevant workflow forums when you have hit a stopping point you
429 are unable to work through. Be respectful of people's time and energy and be sure to include appropriate
430 details important to your problem (see Strategic troubleshooting section).

431 **Data and resource management for workflow-enabled** 432 **biology**

433 Advancements in sequencing technologies have greatly increased the volume of data available for
434 biological query [54]. Workflow systems, by virtue of automating many of the time-intensive project

435 management steps traditionally required for data-intensive biology, can increase our capacity for data
436 analysis. However, conducting biological analyses at this scale requires a coordinated approach to data
437 and computational resource management. Below, we provide recommendations for data acquisition,
438 management, and quality control that have become especially important as the volume of data has
439 increased. Finally, we discuss securing and managing appropriate computational resources for the scale of
440 your project.

441 **Managing large-scale datasets**

442 Experimental design, finding or generating data, and quality control are quintessential parts of data
443 intensive biology. There is no substitute for taking the time to properly design your analysis, identify
444 appropriate data, and conduct sanity checks on your files. While these tasks are not automatable, many
445 tools and databases can aid in these processes.

446 **Look for appropriate publicly-available data**

447 With vast amounts of sequencing data already available in public repositories, it is often possible to begin
448 investigating your research question by seeking out publicly available data. In some cases, these data will
449 be sufficient to conduct your entire analysis. In others cases, particularly for biologists conducting novel
450 experiments, these data can inform decisions about sequencing type, depth, and replication, and can help
451 uncover potential pitfalls before they cost valuable time and resources.

452 Most journals now require data for all manuscripts to be made accessible, either at publication or after a
453 short moratorium. Further, the FAIR (findable, accessible, interoperable, reusable) data movement has
454 improved the data sharing ecosystem for data-intensive biology [[55](#),[56](#),[57](#),[58](#),[59](#),[60](#),[60](#),[61](#)]. You can find
455 relevant sequencing data either by starting from the “data accessibility” sections of papers relevant to
456 your research or by directly searching for your organism, environment, or treatment of choice in public
457 data portals and repositories. The International Nucleotide Sequence Database Collaboration (INSDC),

458 which includes the Sequence Read Archive (SRA), European Nucleotide Archive (ENA), and DataBank
459 of Japan (DDBJ) is the largest repository for raw sequencing data, but no longer accepts sequencing data
460 from large consortia projects [62]. These data are instead hosted in consortia-specific databases, which
461 may require some domain-specific knowledge for identifying relevant datasets and have unique download
462 and authentication protocols. For example, raw data from the Tara Oceans expedition is hosted by the
463 Tara Ocean Foundation [63]. Additional curated databases focus on processed data instead, such as gene
464 expression in the Gene Expression Omnibus (GEO) [64]. Organism-specific databases such as
465 **Wormbase** (*Caenorhabditis elegans*) specialize on curating and integrating sequencing and other data
466 associated with a model organism [65]. Finally, rather than focusing on certain data types or organisms,
467 some repositories are designed to hold any data and metadata associated with a specific project or
468 manuscript (e.g. Open Science Framework, Dryad, Zenodo [66]).

469 **Consider analysis when generating your own data**

470 If generating your own data, proper experimental design and planning are essential. For cost-intensive
471 sequencing data, there are a range of decisions about experimental design and sequencing (including
472 sequencing type, sequencing depth per sample, and biological replication) that impact your ability to
473 properly address your research question. Conducting discussions with experienced bioinformaticians and
474 statisticians, **prior to beginning your experiments** if possible, is the best way to ensure you will have
475 sufficient statistical power to detect effects. These considerations will be different for different types of
476 sequence analysis. To aid in early project planning, we have curated a series of domain-specific
477 references that may be useful as you go about designing your experiment (see **Table 2**). Given the
478 resources invested in collecting samples for sequencing, it's important to build in a buffer to preserve
479 your experimental design in the face of unexpected laboratory or technical issues. Once generated, it is
480 always a good idea to have multiple independent backups of raw sequencing data, as it typically cannot be
481 easily regenerated if lost to computer failure or other unforeseeable events.

482 *Table 2: References for experimental design and considerations for common sequencing chemistries.*

Sequencing type	Resources
RNA-sequencing	[30,67,68]
Metagenomic sequencing	[31,69,70]
Amplicon sequencing	[71,72,73]
Microbial isolate sequencing	[74]
Eukaryotic genome sequencing	[75,76,77,78]
Whole-genome resequencing	[79]
RAD-sequencing	[80,80,81,82,83,84]
single cell RNA-seq	[85,86]

483 As your experiment progresses, keep track of as much information as possible: dates and times of sample
 484 collection, storage, and extraction, sample names, aberrations that occurred during collection, kit lot used
 485 for extraction, and any other sample and sequencing measurements you might be able to obtain
 486 (temperature, location, metabolite concentration, name of collector, well number, plate number, machine
 487 your data was sequenced, on etc). This metadata allows you to keep track of your samples, to control for
 488 batch effects that may arise from unintended batching during sampling or experimental procedures and
 489 makes the data you collect reusable for future applications and analysis by yourself and others. Wherever
 490 possible, follow the standard guidelines for formatting metadata for scientific computing to limit
 491 downstream processing and simplify analyses requiring these metadata (see: [10]). We have focused here
 492 on sequencing data; for data management over long-term ecological studies, we recommend [87].

493 **Getting started with sequencing data**

494 **Protect valuable data**

495 Aside from the code itself, raw data are the most important files associated with a workflow, as they
496 cannot be regenerated if accidentally altered or deleted. Keeping a read-only copy of raw data alongside a
497 workflow as well multiple backups protects your data from accidents and computer failure. This also
498 removes the imperative of storing intermediate files as these can be easily regenerated by the workflow.

499 When sharing or storing files and results, data version control can keep track of differences in files such
500 as changes from tool parameters or versions. The version control tools discussed in the [Workflow-based](#)
501 [project management](#) section are primarily designed to handle small files, but repositories such as the Open
502 Science Framework, Figshare, Zenodo, and Dryad can be used for storing larger files and datasets.

503 The Open Science Framework (OSF; [\[66\]](#)) is a free service that provides powerful collaboration and
504 sharing tools, provides built-in version control, integrates with other storage and version control
505 repositories, guarantees data preservation, and enables you to keep projects private until they are ready to
506 share. Like other services geared towards data sharing, OSF also enables generation of a digital object
507 identifier (doi) for each project. While other services such as Git Large File Storage (LFS), Figshare [\[88\]](#),
508 Zenodo [\[45\]](#), and the Dryad Digital Repository [\[89\]](#) each provide important services for sharing and
509 version control, OSF provides the most comprehensive set of free tools for managing data storage and
510 backup. As free tools often limit the size of files that can be stored, a number of cloud backup and storage
511 services are also available for purchase or via university contract, including Google Drive, Box, Dropbox,
512 Amazon Web Services, and Backblaze. Full computer backups can be conducted to these storage
513 locations with tools like rclone [\[90\]](#).

514 **Ensure data integrity during transfers**

515 If you're working with publicly-available data, you may be able to work on a compute system where the
516 data are already available, circumventing time and effort required for downloading and moving the data.
517 Databases such as the Sequence Read Archive (SRA) are now available on commercial cloud computing
518 systems, and open source projects such as Galaxy enable working with SRA sequence files directly from
519 a web browser [12,91]. Ongoing projects such as the NIH Common Fund Data Ecosystem aim to develop
520 a data portal to make NIH Common Fund data, including biomedical sequencing data, more findable,
521 accessible, interoperable, and reusable (FAIR).

522 In most cases, you'll still need to transfer some data - either downloading raw data or transferring
523 important intermediate and results files for backup and sharing (or both). Transferring compressed files
524 (gzip, bzip2, BAM/CRAM, etc.) can improve transfer speed and save space, and checksums can be used
525 to to ensure file integrity after transfer (see **Figure 8**).

526

527 *Figure 8: Use Checksums to ensure file integrity* Checksum programs (e.g. md5, sha256) encode file size
528 and content in a single value known as a "checksum". For any given file, this value will be identical
529 across platforms when calculated using the same checksum program. When transferring files, calculate
530 the value of the checksum prior to transfer, and then again after transfer. If the value is not identical,
531 there was an error introduced during transfer (e.g. file truncation, etc). Checksums are often provided
532 alongside publicly available files, so that you can verify proper download. Tools like rsync and rclone
533 that automate file transfers use checksums internally to verify that files were transferred properly, and
534 some GUI file transfer tools (e.g. Cyberduck) can assess checksums when they are provided [90]. If you
535 generated your own data and received sequencing files from a sequencing center, be certain you also
536 receive a checksum for each of your files to ensure they download properly.

537 **Perform quality control at every step**

538 The quality of your input data has a major impact on the quality of the output results, no matter whether
539 your workflow analyzes six samples or six hundred. Assessing data at every analysis step can reveal
540 problems and errors early, before they waste valuable time and resources. Using quality control tools that
541 provide metrics and visualizations can help you assess your datasets, particularly as the size of your input
542 data scales up. However, data from different species or sequencing types can produce anomalous quality
543 control results. You are ultimately the single most effective quality control tool that you have, so it is
544 important to critically assess each metric to determine those that are relevant for your particular data.

545 **Look at your files** Quality control can be as simple as looking at the first few and last few lines of input
546 and output data files, or checking the size of those files (see **Table 3**). To develop an intuition for what
547 proper inputs and outputs look like for a given tool, it is often helpful to first run the test example or data
548 that is packaged with the software. Comparing these input and output file formats to your own data can
549 help identify and address inconsistencies.

550 *Table 3: Some bash commands are useful to quickly explore the contents of a file. By using these*
551 *commands, the user can detect common formatting problems or other abnormalities.*

command	function	example
ls -lh	list files with information in a human-readable format	ls -lh *fastq.gz
head	print the first 6 lines of a file to standard out	head samples.csv
tail	print the last 6 lines of a file to standard out	tail samples.csv
less	show the contents of a file in a scrollable screen	less samples.csv
zless	show the contents of a gzipped file in a scrollable screen	zless sample1.fastq.gz
wc -l	count the number of lines in a file	wc -l ecoli.fasta
cat	print a file to standard out	cat samples.csv

grep	find matching text and print the line to standard out	grep ">" ecoli.fasta
cut	cut columns from a table	cut -d"," -f1 samples.csv

552 **Visualize your data** Visualization is another powerful way to pick out unusual or unexpected patterns.
553 Although large abnormalities may be clear from looking at files, others may be small and difficult to find.
554 Visualizing raw sequencing data with FastQC (**Figure 9A**) and processed sequencing data with tools like
555 the Integrative Genome Viewer and plotting tabular results files using python or R can make aberrant or
556 inconsistent results easier to track down [93,94].

557

558 *Figure 9: Visualizations produced by MultiQC. MultiQC finds and automatically parses log files from*
559 *other tools and generates a combined report and parsed data tables that include all samples. MultiQC*
560 *currently supports 88 tools. A. MultiQC summary of FastQC Per Sequence GC Content for 1905*
561 *metagenome samples. FastQC provides quality control measurements and visualizations for raw*
562 *sequencing data from a single sample, and is a near-universal first step in sequencing data analysis*
563 *because of the insights it provides [93,94]. FastQC measures and summarizes 10 quality metrics and*
564 *provides recommendations for whether an individual sample is within an acceptable quality range.*
565 *Not all metrics readily apply to all sequencing data types. For example, while multiple GC peaks might*
566 *be concerning in whole genome sequencing of a bacterial isolate, we would expect a non-normal*
567 *distribution for some metagenome samples that contain organisms with diverse GC content. Samples like*
568 *this can be seen in red in this figure. B. MultiQC summary of Salmon quant reads mapped per sample for*
569 *RNA-seq samples [95]. In this figure, we see that MultiQC summarizes the number of reads mapped and*
570 *percent of reads mapped, two values that are reported in the Salmon log files.*

571 **Pay attention to warnings and log files** Many tools generate log files or messages while running. These
572 files contain information about the quantity, quality, and results from the run, or error messages about
573 why a run failed. Inspecting these files can be helpful to make sure tools ran properly and consistently, or

574 to debug failed runs. Parsing and visualizing log files with a tool like MultiQC can improve
575 interpretability of program-specific log files (**Figure 9** [96]).

576 **Look for common biases in sequencing data** Biases in sequencing data originate from experimental
577 design, methodology, sequencing chemistry, or workflows, and are helpful to target specifically with
578 quality control measures. The exact biases in a specific data set or workflow will vary greatly between
579 experiments so it is important to understand the sequencing method you have chosen and incorporate
580 appropriate filtration steps into your workflow. For example, PCR duplicates can cause problems in
581 libraries that underwent an amplification step, and often need to be removed prior to downstream analysis
582 [97,98,99,100,101].

583 **Check for contamination** Contamination can arise during sample collection, nucleotide extraction,
584 library preparation, or through sequencing spike-ins like PhiX, and could change data interpretation if not
585 removed [102,103,104]. Libraries sequenced with high concentrations of free adapters or with low
586 concentration samples may have increased barcode hopping, leading to contamination between samples
587 [105].

588 **Consider the costs and benefits of stringent quality control for your data** Good quality data is
589 essential for good downstream analysis. However, stringent quality control can sometimes do more harm
590 than good. For example, depending on sequencing depth, stringent quality trimming of RNA-sequencing
591 data may reduce isoform discovery [106]. To determine what issues are most likely to plague your
592 specific data set, it can be helpful to find recent publications using a similar experimental design, or to
593 speak with experts at a sequencing core.

594 Because sequencing data and applications are so diverse, there is no one-size-fits-all solution for quality
595 control. It is important to think critically about the patterns you expect to see given your data and your
596 biological problem, and consult with technical experts whenever possible.

597 **Securing and managing appropriate computational resources**

598 Sequence analysis requires access to computing systems with adequate storage and analysis power for
 599 your data. For some smaller-scale datasets, local desktop or even laptop systems can be sufficient,
 600 especially if using tools that implement data-reduction strategies such as minhashing [107]. However,
 601 larger projects require additional computing power, or may be restricted to certain operating systems
 602 (e.g. linux). For these projects, solutions range from research-focused high performance computing
 603 systems to research-integrated commercial analysis platforms. Both research-only and commercial
 604 clusters provide avenues for research and educational proposals to enable access to their computing
 605 resources (see **Table 4**). In preparing for data analysis, be sure to allocate sufficient computational
 606 resources and funding for storage and analysis, including large intermediate files and resources required
 607 for personnel training. Note that workflow systems can greatly facilitate faithful execution of your
 608 analysis across the range of computational resources available to you, including distribution across cloud
 609 computing systems.

610 *Table 4: **Computing Resources** Bioinformatic projects often require additional computing resources. If a*
 611 *local or university-run high-performance computing cluster is not available, computing resources are*
 612 *available via a number of grant-based or commercial providers.*

Provider	Access Model	Restrictions
Amazon Web Services	Paid	
Bionimbus Protected Data Cloud	Research allocation	users with eRA commons account
Cyverse Atmosphere	Free with limits	storage and compute hours
EGI federated cloud	Access by contact	European partner countries
Galaxy	Free with storage limits	data storage limits

Google Cloud Platform	Paid	
Google Colab	Free	computational notebooks, no resource guarantees
Microsoft Azure	Paid	
NSF XSEDE	Research allocation	USA researchers or collaborators
Open Science Data Cloud	Research allocation	
Wasabi	Paid	data storage solution only

613 **Getting started with resource management**

614 As the scale of data increases, the resources required for analysis can balloon. Bioinformatic workflows
615 can be long-running, require high-memory systems, or involve intensive file manipulation. Some of the
616 strategies below may help you manage computational resources for your project.

617 **Apply for research units if eligible** There are a number of cloud computing services that offer grants
618 providing computing resources to data-intensive researchers (**Table 4**). In some cases, the resources
619 provided may be sufficient to cover your entire analysis.

620 **Develop on a local computer when possible** Since workflows transfer easily across systems, it can be
621 useful to develop individual analysis steps on a local laptop. If the analysis tool will run on your local
622 system, test the step with subsampled data, such as that created in the [Getting started developing](#)
623 [workflows](#) section. Once working, the new workflow component can be run at scale on a larger
624 computing system. Workflow system tool resource usage reporting can help determine the increased
625 resources needed to execute the workflow on larger systems. For researchers without access to free or
626 granted computing resources, this strategy can save significant cost.

627 **Gain quick insights using sketching algorithms** Understanding the basic structure of data, the
628 relationship between samples, and the approximate composition of each sample can very helpful at the
629 beginning of data analysis, and can often drive analysis decisions in different directions than those
630 originally intended. Although most bioinformatics workflows generate these types of insights, there are a
631 few tools that do so rapidly, allowing the user to generate quick hypotheses that can be further tested by
632 more extensive, fine-grained analyses. Sketching algorithms work with compressed approximate
633 representations of sequencing data and thereby reduce runtimes and computational resources. These
634 approximate representations retain enough information about the original sequence to recapitulate the
635 main findings from many exact but computationally intensive workflows. Most sketching algorithms
636 estimate sequence similarity in some way, allowing you to gain insights from these comparisons. For
637 example, sketching algorithms can be used to estimate all-by-all sample similarity which can be
638 visualized as a Principle Component Analysis or a multidimensional scaling plot, or can be used to build
639 a phylogenetic tree with accurate topology. Sketching algorithms also dramatically reduce the runtime for
640 comparisons against databases (e.g. all of GenBank), allowing users to quickly compare their data against
641 large public databases.

642 Rowe 2019 [108] reviewed programs and genomic use cases for sketching algorithms, and provided a
643 series of tutorial workbooks (e.g. Sample QC notebook: [109]).

644 **Use the right tools for your question** RNA-seq analysis approaches like differential expression or
645 transcript clustering rely on transcript or gene counts. Many tools can be used to generate these counts by
646 quantifying the number of reads that overlap with each transcript or gene. For example, tools like STAR
647 and HISAT2 produce alignments that can be post-processed to generate per-transcript read counts
648 [110,111]. However, these tools generate information-rich output, specifying per-base alignments for
649 each read. If you are only interested in read quantification, quasi-mapping tools provide the desired
650 results while reducing the time and resources needed to generate and store read count information
651 [112,113].

652 **Seek help when you need it** In some cases, you may find that your accessible computing system is ill-
653 equipped to handle the type or scope of your analysis. Depending on the system, staff members may be
654 able to help direct you to properly scale your workflow to available resources, or guide you in tailoring
655 computational unit allocations or purchases to match your needs.

656 **Strategies for troubleshooting**

657 Workflows, and research software in general, invariably require troubleshooting and iteration. When first
658 starting with a workflow system, it can be difficult to interpret code and usage errors from unfamiliar
659 tools or languages [2]. Further, the iterative development process of research software means
660 functionality may change, new features may be added, or documentation may be out of date [114]. The
661 challenges of learning and interacting with research software require time and patience [4].

662 One of the largest barriers to surmounting these challenges is learning how, when, and where to ask for
663 help. Below we outline a strategy for troubleshooting that can help build your own knowledge while
664 respecting both your own time and that of research software developers and the larger bioinformatic
665 community. In the “where to seek help” section, we also recommend locations for asking general
666 questions around data-intensive analysis, including discussion of tool choice, parameter selection, and
667 other analysis strategies. Beyond these tips, workshops and materials from training organizations such as
668 the Carpentries, R-Ladies, RStudio can arm you with the tools you need to start troubleshooting and
669 jump-start software and data literacy in your community [115]. Getting involved with these workshops
670 and communities not only provides educational benefits but also networking and career-building
671 opportunities.

672 **How to help yourself: Try to pinpoint your issue or error**

673 Software errors can be the result of syntax errors, dependency issues, operating system conflicts, bugs in
674 the software, problems with the input data, and many other issues. Running the software on the provided
675 test data can help narrow the scope of error sources: if the test data successfully runs, the command is
676 likely free of syntax errors, the source code is functioning, and the tool is likely interacting appropriately
677 with dependencies and the operating system. If the test data runs but the tool still produces an error when
678 run with your data and parameters, the error message can be helpful in discovering the cause of the error.
679 In many cases, the error you've encountered has been encountered many times before, and searching for
680 the error online can turn up a working solution. If there is a software issue tracker for the software (e.g. on
681 the GitHub, GitLab, or Bitbucket repository), or a Gitter, Slack, or Google Groups page, performing a
682 targeted search with the error message may provide additional context or a solution for the error. If
683 targeted searches do not return a results, Googling the error message with the program name is a good
684 next step. Searching with several variants and iteratively adding information such as the type of input
685 data, the name of the coding language or computational platform, or other relevant information, can
686 improve the likelihood that a there will be a match. There are a vast array of online resources for
687 bioinformatic help ranging from question sites such as Stack Overflow and BioStars, to personal or
688 academic blogs and even tutorials and lessons written by experts in the field [116]. This increases the
689 discoverability of error messages and their solutions.

690 Sometimes, programs fail without outputting an error message. In cases like these, the software's help
691 (usually accessible on the command line via `tool-name --help`) and official documentation may provide
692 clues or additional example use cases that may be helpful in resolving an error. Syntax errors are
693 extremely common, and typos as small as a single, misplaced character or amount of whitespace can
694 affect the code. If a command matches the documentation and appears syntactically correct, the software
695 version (often accessible at the command line `tool-name --version`) may be causing the error.

696 Best practices for software development follow “semantic versioning” principles, which aim to keep the
697 arguments and functionality the same for all minor releases of the program (e.g. 1.1 to 1.2) and only
698 change functions with major releases (e.g. 1.x to 2.0).

699 **How to seek help: include the right details with your question**

700 When searching for the error message and reading the documentation do not resolve an error, it is usually
701 appropriate to for seek help either from the software developers or from a bioinformatics community.
702 When asking for help, it’s essential to provide the right details so that other users and developers can
703 understand the exact conditions that produced the error. At minimum, include the name and version of the
704 program, the method used to install it, whether or not the test data ran, the exact code that produced the
705 error, the error message, and the full output text from the run (if any is produced). The type and version of
706 the operating system you are using is also helpful to include. Sometimes, this is enough information for
707 others to spot the error. However, if it appears that there may bug in the underlying code, specifying or
708 providing the minimum amount of data required to reproduce the error (e.g. reproducible example
709 [[117](#),[118](#)]) enables other to reproduce and potentially solve the error at hand. Putting the effort into
710 gathering this information both increases your own understanding of the problem and makes it easier and
711 faster for others to help solve your issue. Furthermore, it signals respect for the time that these developers
712 and community members dedicate to helping troubleshoot and solve user issues.

713 **Where to seek help: online and local communities of practice**

714 Online communities and forums are a rich source of archived bioinformatics errors with many helpful
715 community members. For errors with specific programs, often the best place to post is the developers’
716 preferred location for answering questions and solving errors related to their program. For open source
717 programs on GitHub, GitLab, or Bitbucket, this is often the “Issues” tab within the software repository,
718 but it could alternatively be a Google groups list, gitter page, or other specified forum. Usually, the

719 documentation indicates the best location questions. If question is more general, such as asking about
720 program choice or workflows, forums relevant to your field such as Stack Overflow, BioStars, or
721 SEQanswers are good choices, as posts here are often seen by a large community of researchers. Before
722 posting, search through related topics to double check the question has not already been answered. As
723 more research software development and troubleshooting is happening openly in online repositories, it is
724 becoming more important than ever to follow a code of conduct that promotes open and harassment-free
725 discussion environment [119]. Look for codes of conduct in the online forums you participate in, and
726 make sure you do your part to help ensure a welcoming community for participants of all backgrounds
727 and computational competencies.

728 While there is lots of help available online, there is no substitute for local communities. Local
729 communities may come in the form of a tech meetup, a users group, a hacky hour, or an informal meetup
730 of researchers using similar tools. While this may seem like just a local version of Stack Overflow, the
731 local, member-only nature can help create a safe and collaborative online space for troubleshooting
732 problems often encountered by your local bioinformatics community. The benefit to beginners is clear:
733 learning the best way to post questions and the important parts of errors, while getting questions answered
734 so they can move forward in their research. Intermediate users may actually find these communities most
735 useful, as they can also accelerate their own troubleshooting skills by helping others solve issues that they
736 have already struggled through. While it can be helpful to have some experts available to help answer
737 questions or to know when to escalate to Stack Overflow or other communities, a collaborative
738 community of practice with members at all experience levels can help all its members move their science
739 forward faster.

740 If such a community does not yet exist in your area, building this sort of community (discussed in detail
741 in [120]), can be as simple as hosting a seminar series or starting meetup sessions for data analysis co-
742 working. In our experience, it can also be useful to set up a local online forum (e.g. discourse) for group
743 troubleshooting.

744 **Conclusion**

745 Bioinformatics-focused workflow systems have reshaped data-intensive biology, reducing execution
746 hurdles and empowering biologists to conduct reproducible analyses at the massive scale of data now
747 available. Shared, interoperable research code is enabling biologists to spend less time rewriting common
748 analysis steps, and more time on interesting biological questions. We believe these workflow systems will
749 become increasingly important as dataset size and complexity continue to grow. This manuscript provides
750 a directed set of project, data, and resource management strategies for adopting workflow systems to
751 facilitate and expedite reproducible biological research. While the included data management strategies
752 are tailored to our own experiences in high-throughput sequencing analysis, we hope that these principles
753 enable biologists both within and beyond our field to reap the benefits of workflow-enabled data-intensive
754 biology.

755

756

757 **Declarations**

758 **Acknowledgements**

759 Thank you to all the members and affiliates of the Lab for Data-Intensive Biology at UC Davis for
760 providing valuable feedback on earlier versions of this manuscript and growing these practices alongside
761 us. We also thank the Carpentries Community for fundamentally shaping many of the ideas and practices
762 we cover in this manuscript.

763 Author Contributions

Author	Contributions
TER	Conceptualization; Methodology; Writing - Original Draft; Writing - Review and Editing; Visualization
PTB**	Methodology; Writing - Review and Editing
LCI**	Methodology; Writing - Review and Editing
SEJ**	Methodology; Visualization; Writing - Review and Editing
CMR**	Methodology; Writing - Review and Editing
CSW**	Methodology; Writing - Review and Editing
CTB	Methodology; Writing - Review and Editing; Supervision; Funding Acquisition
NTP	Conceptualization; Methodology; Writing - Original Draft; Writing - Review and Editing; Visualization; Supervision; Funding Acquisition

764 **co-equal contributions

765 Availability of Data and Materials

766 All materials for this manuscript can be found at <https://github.com/dib-lab/2020-workflows-paper>.

767

768 Competing Interests

769 The authors declare that no competing interests exist.

770 References

771 1. The nf-core framework for community-curated bioinformatics pipelines

772 Philip A. Ewels, Alexander Peltzer, Sven Fillinger, Harshil Patel, Johannes Alneberg, Andreas Wilm,

773 Maxime Ulysse Garcia, Paolo Di Tommaso, Sven Nahnsen

774 *Nature Biotechnology* (2020-02-13) <https://doi.org/ggk3qh>

775 DOI: [10.1038/s41587-020-0439-x](https://doi.org/10.1038/s41587-020-0439-x) · PMID: [32055031](https://pubmed.ncbi.nlm.nih.gov/32055031/)

776 2. Unmet needs for analyzing biological big data: A survey of 704 NSF principal investigators

777 Lindsay Barone, Jason Williams, David Micklos

778 *PLOS Computational Biology* (2017-10-19) <https://doi.org/gcgdcc>

779 DOI: [10.1371/journal.pcbi.1005755](https://doi.org/10.1371/journal.pcbi.1005755) · PMID: [29049281](https://pubmed.ncbi.nlm.nih.gov/29049281/) · PMCID: [PMC5654259](https://pubmed.ncbi.nlm.nih.gov/PMC5654259/)

780 3. Practical Computational Reproducibility in the Life Sciences

781 Björn Grüning, John Chilton, Johannes Köster, Ryan Dale, Nicola Soranzo, Marius van den Beek, Jeremy

782 Goecks, Rolf Backofen, Anton Nekrutenko, James Taylor

783 *Cell Systems* (2018-06) <https://doi.org/ggdv3z>

784 DOI: [10.1016/j.cels.2018.03.014](https://doi.org/10.1016/j.cels.2018.03.014) · PMID: [29953862](https://pubmed.ncbi.nlm.nih.gov/29953862/) · PMCID: [PMC6263957](https://pubmed.ncbi.nlm.nih.gov/PMC6263957/)

- 785 **4. A graduate student perspective on overcoming barriers to interacting with open-source software**
786 Oihane Cereceda, Danielle E. A. Quinn
787 *FACETS* (2020-01-01) <https://doi.org/ggw7f3>
788 DOI: [10.1139/facets-2019-0020](https://doi.org/10.1139/facets-2019-0020)
- 789 **5. Scientific workflows: Past, present and future**
790 Malcolm Atkinson, Sandra Gesing, Johan Montagnat, Ian Taylor
791 *Future Generation Computer Systems* (2017-10) <https://doi.org/ggs77s>
792 DOI: [10.1016/j.future.2017.05.041](https://doi.org/10.1016/j.future.2017.05.041)
- 793 **6. Rule-based workflow management for bioinformatics**
794 John S. Conery, Julian M. Catchen, Michael Lynch
795 *The VLDB Journal* (2005-09-09) <https://doi.org/fhzqvp>
796 DOI: [10.1007/s00778-005-0153-9](https://doi.org/10.1007/s00778-005-0153-9)
- 797 **7. Robust Cross-Platform Workflows: How Technical and Scientific Communities Collaborate to**
798 **Develop, Test and Share Best Practices for Data Analysis**
799 Steffen Möller, Stuart W. Prescott, Lars Wirzenius, Petter Reinholdtsen, Brad Chapman, Pjotr Prins, Stian
800 Soiland-Reyes, Fabian Klötzl, Andrea Bagnacani, Matúš Kalaš, ... Michael R. Crusoe
801 *Data Science and Engineering* (2017-11-16) <https://doi.org/ggwrrk>
802 DOI: [10.1007/s41019-017-0050-4](https://doi.org/10.1007/s41019-017-0050-4)
- 803 **8. Best Practices for Scientific Computing**
804 Greg Wilson, D. A. Aruliah, C. Titus Brown, Neil P. Chue Hong, Matt Davis, Richard T. Guy, Steven H.
805 D. Haddock, Kathryn D. Huff, Ian M. Mitchell, Mark D. Plumbley, ... Paul Wilson
806 *PLoS Biology* (2014-01-07) <https://doi.org/qtt>
807 DOI: [10.1371/journal.pbio.1001745](https://doi.org/10.1371/journal.pbio.1001745) · PMID: [24415924](https://pubmed.ncbi.nlm.nih.gov/24415924/) · PMCID: [PMC3886731](https://pubmed.ncbi.nlm.nih.gov/PMC3886731/)
- 808 **9. Computing Workflows for Biologists: A Roadmap**
809 Ashley Shade, Tracy K. Teal
810 *PLOS Biology* (2015-11-24) <https://doi.org/f72r9m>
811 DOI: [10.1371/journal.pbio.1002303](https://doi.org/10.1371/journal.pbio.1002303) · PMID: [26600012](https://pubmed.ncbi.nlm.nih.gov/26600012/) · PMCID: [PMC4658184](https://pubmed.ncbi.nlm.nih.gov/PMC4658184/)
- 812 **10. Good enough practices in scientific computing**
813 Greg Wilson, Jennifer Bryan, Karen Cranston, Justin Kitzes, Lex Nederbragt, Tracy K. Teal
814 *PLOS Computational Biology* (2017-06-22) <https://doi.org/gbkbwp>
815 DOI: [10.1371/journal.pcbi.1005510](https://doi.org/10.1371/journal.pcbi.1005510) · PMID: [28640806](https://pubmed.ncbi.nlm.nih.gov/28640806/) · PMCID: [PMC5480810](https://pubmed.ncbi.nlm.nih.gov/PMC5480810/)
- 816 **11. Bioconda: sustainable and comprehensive software distribution for the life sciences**
817 Björn Grüning, Ryan Dale, Andreas Sjödin, Brad A. Chapman, Jillian Rowe, Christopher H. Tomkins-
818 Tinch, Renan Valieris, Johannes Köster, The Bioconda Team
819 *Nature Methods* (2018-07-02) <https://doi.org/gd2xzp>
820 DOI: [10.1038/s41592-018-0046-7](https://doi.org/10.1038/s41592-018-0046-7) · PMID: [29967506](https://pubmed.ncbi.nlm.nih.gov/29967506/)
- 821 **12. The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018**
822 **update**
823 Enis Afgan, Dannon Baker, Bérénice Batut, Marius van den Beek, Dave Bouvier, Martin Čech, John

824 Chilton, Dave Clements, Nate Coraor, Björn A Grüning, ... Daniel Blankenberg
825 *Nucleic Acids Research* (2018-07-02) <https://doi.org/gdwxpr>
826 DOI: [10.1093/nar/gky379](https://doi.org/10.1093/nar/gky379) · PMID: [29790989](https://pubmed.ncbi.nlm.nih.gov/29790989/) · PMCID: [PMC6030816](https://pubmed.ncbi.nlm.nih.gov/PMC6030816/)

827 **13. Data Commons to Support Pediatric Cancer Research**
828 Samuel L. Volchenbom, Suzanne M. Cox, Allison Heath, Adam Resnick, Susan L. Cohn, Robert
829 Grossman
830 *American Society of Clinical Oncology Educational Book* (2017) <https://doi.org/ggv5zs>
831 DOI: [10.14694/edbk_175029](https://doi.org/10.14694/edbk_175029) · PMID: [28561664](https://pubmed.ncbi.nlm.nih.gov/28561664/)

832 **14. MGnify: the microbiome analysis resource in 2020**
833 Alex L Mitchell, Alexandre Almeida, Martin Beracochea, Miguel Boland, Josephine Burgin, Guy
834 Cochrane, Michael R Crusoe, Varsha Kale, Simon C Potter, Lorna J Richardson, ... Robert D Finn
835 *Nucleic Acids Research* (2019-11-07) <https://doi.org/ggctnm>
836 DOI: [10.1093/nar/gkz1035](https://doi.org/10.1093/nar/gkz1035) · PMID: [31696235](https://pubmed.ncbi.nlm.nih.gov/31696235/) · PMCID: [PMC7145632](https://pubmed.ncbi.nlm.nih.gov/PMC7145632/)

837 **15. nf-core/rnaseq**
838 GitHub
839 <https://github.com/nf-core/rnaseq>

840 **16. metagenome-atlas/atlas**
841 GitHub
842 <https://github.com/metagenome-atlas/atlas>

843 **17. ATLAS: a Snakemake workflow for assembly, annotation, and genomic binning of metagenome
844 sequence data**
845 Silas Kieser, Joseph Brown, Evgeny M. Zdobnov, Mirko Trajkovski, Lee Ann McCue
846 *bioRxiv* (2019-08-20) <https://doi.org/ggv5zm>
847 DOI: [10.1101/737528](https://doi.org/10.1101/737528)

848 **18. sunbeam-labs/sunbeam**
849 GitHub
850 <https://github.com/sunbeam-labs/sunbeam>

851 **19. Sunbeam: an extensible pipeline for analyzing metagenomic sequencing experiments**
852 Erik L. Clarke, Louis J. Taylor, Chunyu Zhao, Andrew Connell, Jung-Jin Lee, Bryton Fett, Frederic D.
853 Bushman, Kyle Bittinger
854 *Microbiome* (2019-03-22) <https://doi.org/gf9sd6>
855 DOI: [10.1186/s40168-019-0658-x](https://doi.org/10.1186/s40168-019-0658-x) · PMID: [30902113](https://pubmed.ncbi.nlm.nih.gov/30902113/) · PMCID: [PMC6429786](https://pubmed.ncbi.nlm.nih.gov/PMC6429786/)

856 **20. dib-lab/dammit**
857 GitHub
858 <https://github.com/dib-lab/dammit>

859 **21. dib-lab/elvers**
860 GitHub
861 <https://github.com/dib-lab/elvers>

- 862 22. **Snakemake—a scalable bioinformatics workflow engine**
863 J. Koster, S. Rahmann
864 *Bioinformatics* (2012-08-20) <https://doi.org/gd2xzq>
865 DOI: [10.1093/bioinformatics/bts480](https://doi.org/10.1093/bioinformatics/bts480) · PMID: [22908215](https://pubmed.ncbi.nlm.nih.gov/22908215/)
- 866 23. **Nextflow enables reproducible computational workflows**
867 Paolo Di Tommaso, Maria Chatzou, Evan W Floden, Pablo Prieto Barja, Emilio Palumbo, Cedric
868 Notredame
869 *Nature Biotechnology* (2017-04-11) <https://doi.org/gfj52z>
870 DOI: [10.1038/nbt.3820](https://doi.org/10.1038/nbt.3820) · PMID: [28398311](https://pubmed.ncbi.nlm.nih.gov/28398311/)
- 871 24. **Common Workflow Language, v1.0**
872 Peter Amstutz, Michael R. Crusoe, Nebojša Tijanić, Brad Chapman, John Chilton, Michael Heuer,
873 Andrey Kartashov, Dan Leehr, Hervé Ménager, Maya Nedeljkovich, ... Luka Stojanovic
874 *Figshare* (2016) <https://doi.org/gf6ppg>
875 DOI: [10.6084/m9.figshare.3115156.v2](https://doi.org/10.6084/m9.figshare.3115156.v2)
- 876 25. **Terra.Bio**
877 Terra.Bio
878 <https://terra.bio>
- 879 26. **The Seven Bridges Platform**
880 Seven Bridges
881 <https://www.sevenbridges.com/platform/>
- 882 27. **The drake R package: a pipeline toolkit for reproducibility and high-performance computing**
883 William Michael Landau
884 *The Journal of Open Source Software* (2018-01-26) <https://doi.org/gd876m>
885 DOI: [10.21105/joss.00550](https://doi.org/10.21105/joss.00550)
- 886 28. **Scalable Workflows and Reproducible Data Analysis for Genomics**
887 Francesco Strozzi, Roel Janssen, Ricardo Wurmus, Michael R. Crusoe, George Githinji, Paolo Di
888 Tommaso, Dominique Belhachemi, Steffen Möller, Geert Smant, Joep de Ligt, Pjotr Prins
889 *Methods in Molecular Biology* (2019) <https://doi.org/ggv5zh>
890 DOI: [10.1007/978-1-4939-9074-0_24](https://doi.org/10.1007/978-1-4939-9074-0_24) · PMID: [31278683](https://pubmed.ncbi.nlm.nih.gov/31278683/)
- 891 29. **“Sequana”: a Set of Snakemake NGS pipelines**
892 Thomas Cokelaer, Dimitri Desvillechabrol, Rachel Legendre, Mélissa Cardon
893 *The Journal of Open Source Software* (2017-08-30) <https://doi.org/ggv5zt>
894 DOI: [10.21105/joss.00352](https://doi.org/10.21105/joss.00352)
- 895 30. **A survey of best practices for RNA-seq data analysis**
896 Ana Conesa, Pedro Madrigal, Sonia Tarazona, David Gomez-Cabrero, Alejandra Cervera, Andrew
897 McPherson, Michał Wojciech Szcześniak, Daniel J. Gaffney, Laura L. Elo, Xuegong Zhang, Ali
898 Mortazavi
899 *Genome Biology* (2016-01-26) <https://doi.org/f3vhpi>
900 DOI: [10.1186/s13059-016-0881-8](https://doi.org/10.1186/s13059-016-0881-8) · PMID: [26813401](https://pubmed.ncbi.nlm.nih.gov/26813401/) · PMCID: [PMC4728800](https://pubmed.ncbi.nlm.nih.gov/PMC4728800/)

- 901 **31. Shotgun metagenomics, from sampling to analysis**
902 Christopher Quince, Alan W Walker, Jared T Simpson, Nicholas J Loman, Nicola Segata
903 *Nature Biotechnology* (2017-09-12) <https://doi.org/gbv6nf>
904 DOI: [10.1038/nbt.3935](https://doi.org/10.1038/nbt.3935) · PMID: [28898207](https://pubmed.ncbi.nlm.nih.gov/28898207/)
- 905 **32. Current best practices in single-cell RNA-seq analysis: a tutorial**
906 Malte D Luecken, Fabian J Theis
907 *Molecular Systems Biology* (2019-06-19) <https://doi.org/gf4f4d>
908 DOI: [10.15252/msb.20188746](https://doi.org/10.15252/msb.20188746) · PMID: [31217225](https://pubmed.ncbi.nlm.nih.gov/31217225/) · PMCID: [PMC6582955](https://pubmed.ncbi.nlm.nih.gov/PMC6582955/)
- 909 **33. Next-generation biology: Sequencing and data analysis approaches for non-model organisms**
910 Rute R. da Fonseca, Anders Albrechtsen, Gonçalo Espregueira Themudo, Jazmín Ramos-Madrigal, Jonas
911 Andreas Sibbesen, Lasse Maretty, M. Lisandra Zepeda-Mendoza, Paula F. Campos, Rasmus Heller,
912 Ricardo J. Pereira
913 *Marine Genomics* (2016-12) <https://doi.org/f9h2s2>
914 DOI: [10.1016/j.margen.2016.04.012](https://doi.org/10.1016/j.margen.2016.04.012) · PMID: [27184710](https://pubmed.ncbi.nlm.nih.gov/27184710/)
- 915 **34. Best practices for analysing microbiomes**
916 Rob Knight, Alison Vrbanac, Bryn C. Taylor, Alexander Aksenov, Chris Callewaert, Justine Debelius,
917 Antonio Gonzalez, Tomasz Kosciolk, Laura-Isobel McCall, Daniel McDonald, ... Pieter C. Dorrestein
918 *Nature Reviews Microbiology* (2018-05-23) <https://doi.org/gdj32p>
919 DOI: [10.1038/s41579-018-0029-9](https://doi.org/10.1038/s41579-018-0029-9) · PMID: [29795328](https://pubmed.ncbi.nlm.nih.gov/29795328/)
- 920 **35. Singularity: Scientific containers for mobility of compute**
921 Gregory M. Kurtzer, Vanessa Sochat, Michael W. Bauer
922 *PLOS ONE* (2017-05-11) <https://doi.org/f969fz>
923 DOI: [10.1371/journal.pone.0177459](https://doi.org/10.1371/journal.pone.0177459) · PMID: [28494014](https://pubmed.ncbi.nlm.nih.gov/28494014/) · PMCID: [PMC5426675](https://pubmed.ncbi.nlm.nih.gov/PMC5426675/)
- 924 **36. Docker: lightweight Linux containers for consistent development and deployment**
925 Dirk Merkel
926 *Linux Journal* (2014-03-01)
- 927 **37. TheSnakePit/mamba**
928 GitHub
929 <https://github.com/TheSnakePit/mamba>
- 930 **38. dib-lab/dib-MMETSP**
931 GitHub
932 <https://github.com/dib-lab/dib-MMETSP>
- 933 **39. Re-assembly, quality evaluation, and annotation of 678 microbial eukaryotic reference**
934 **transcriptomes**
935 Lisa K Johnson, Harriet Alexander, C Titus Brown
936 *GigaScience* (2019-04) <https://doi.org/ggrd6x>
937 DOI: [10.1093/gigascience/giy158](https://doi.org/10.1093/gigascience/giy158) · PMID: [30544207](https://pubmed.ncbi.nlm.nih.gov/30544207/) · PMCID: [PMC6481552](https://pubmed.ncbi.nlm.nih.gov/PMC6481552/)
- 938 **40. R Markdown** <https://rmarkdown.rstudio.com/>

- 939 41. **IOS Press Ebooks - Jupyter Notebooks – a publishing format for reproducible computational**
940 **workflows** <http://ebooks.iospress.nl/publication/42900>
- 941 42. **Tempo and mode of genome evolution in a 50,000-generation experiment**
942 Olivier Tenaillon, Jeffrey E. Barrick, Noah Ribeck, Daniel E. Deatherage, Jeffrey L. Blanchard, Aurko
943 Dasgupta, Gabriel C. Wu, Sébastien Wielgoss, Stéphane Cruveiller, Claudine Médigue, ... Richard E.
944 Lenski
945 *Nature* (2016-08-01) <https://doi.org/f8283v>
946 DOI: [10.1038/nature18959](https://doi.org/10.1038/nature18959) · PMID: [27479321](https://pubmed.ncbi.nlm.nih.gov/27479321/) · PMCID: [PMC4988878](https://pubmed.ncbi.nlm.nih.gov/PMC4988878/)
- 947 43. **Binder 2.0 - Reproducible, interactive, sharable environments for science at scale**
948 Project Jupyter, Matthias Bussonnier, Jessica Forde, Jeremy Freeman, Brian Granger, Tim Head, Chris
949 Holdgraf, Kyle Kelley, Gladys Nalvarte, Andrew Osheroﬀ, ... Carol Willing
950 *SciPy* (2018) <https://doi.org/gfwcm6>
951 DOI: [10.25080/majora-4af1f417-011](https://doi.org/10.25080/majora-4af1f417-011)
- 952 44. **ngs-docs/2020-ggg-201b-rnaseq**
953 GitHub
954 <https://github.com/ngs-docs/2020-ggg-201b-rnaseq>
- 955 45. **Zenodo - Research. Shared.** <https://zenodo.org/>
- 956 46. **Computing environments for reproducibility: Capturing the “Whole Tale”**
957 Adam Brinckman, Kyle Chard, Niall Gaffney, Mihael Hategan, Matthew B. Jones, Kacper Kowalik,
958 Sivakumar Kulasekaran, Bertram Ludäscher, Bryce D. Mecum, Jarek Nabrzyski, ... Kandace Turner
959 *Future Generation Computer Systems* (2019-05) <https://doi.org/ggv5zj>
960 DOI: [10.1016/j.future.2017.12.029](https://doi.org/10.1016/j.future.2017.12.029)
- 961 47. **ReproZip**
962 Fernando Chirigati, Rémi Rampin, Dennis Shasha, Juliana Freire
963 *Association for Computing Machinery (ACM)* (2016) <https://doi.org/ggxs3d>
964 DOI: [10.1145/2882903.2899401](https://doi.org/10.1145/2882903.2899401)
- 965 48. **Pavian** <https://fbreitwieser.shinyapps.io/pavian/>
- 966 49. **Pavian: interactive analysis of metagenomics data for microbiome studies and pathogen**
967 **identification**
968 Florian P Breitwieser, Steven L Salzberg
969 *Bioinformatics* (2019-09-25) <https://doi.org/ggnb3n>
970 DOI: [10.1093/bioinformatics/btz715](https://doi.org/10.1093/bioinformatics/btz715) · PMID: [31553437](https://pubmed.ncbi.nlm.nih.gov/31553437/)
- 971 50. **Visualizing Biological Data** <https://plotly.com/python/v3/ipython-notebooks/bioinformatics/>
- 972 51. **Plotly: The front-end for ML and data science models /**
- 973 52. **Vega-Lite: A Grammar of Interactive Graphics**
974 Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, Jeffrey Heer

- 975 *IEEE Transactions on Visualization and Computer Graphics* (2017-01) <https://doi.org/f92f32>
976 DOI: [10.1109/tvcg.2016.2599030](https://doi.org/10.1109/tvcg.2016.2599030) · PMID: [27875150](https://pubmed.ncbi.nlm.nih.gov/27875150/)
- 977 **53. HackMD - Collaborative Markdown Knowledge Base**
978 HackMD
979 <https://hackmd.io>
- 980 **54. Documentation** <https://www.ncbi.nlm.nih.gov/sra/docs/sragrowth/>
- 981 **55. The FAIR Guiding Principles for scientific data management and stewardship**
982 Mark D. Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton,
983 Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, ...
984 Barend Mons
985 *Scientific Data* (2016-03-15) <https://doi.org/bdd4>
986 DOI: [10.1038/sdata.2016.18](https://doi.org/10.1038/sdata.2016.18) · PMID: [26978244](https://pubmed.ncbi.nlm.nih.gov/26978244/) · PMCID: [PMC4792175](https://pubmed.ncbi.nlm.nih.gov/PMC4792175/)
- 987 **56. The international nucleotide sequence database collaboration**
988 Ilene Karsch-Mizrachi, Toshihisa Takagi, Guy Cochrane, on behalf of the International Nucleotide
989 Sequence Database Collaboration
990 *Nucleic Acids Research* (2018-01-04) <https://doi.org/ggwp7h>
991 DOI: [10.1093/nar/gkx1097](https://doi.org/10.1093/nar/gkx1097) · PMID: [29190397](https://pubmed.ncbi.nlm.nih.gov/29190397/) · PMCID: [PMC5753279](https://pubmed.ncbi.nlm.nih.gov/PMC5753279/)
- 992 **57. Public Microbial Resource Centers: Key Hubs for Findable, Accessible, Interoperable, and**
993 **Reusable (FAIR) Microorganisms and Genetic Materials**
994 P. Becker, M. Bosschaerts, P. Chaerle, H.-M. Daniel, A. Hellemans, A. Olbrechts, L. Rigouts, A.
995 Wilmotte, M. Hendrickx
996 *Applied and Environmental Microbiology* (2019-10-16) <https://doi.org/ggbpc9>
997 DOI: [10.1128/aem.01444-19](https://doi.org/10.1128/aem.01444-19) · PMID: [31471301](https://pubmed.ncbi.nlm.nih.gov/31471301/) · PMCID: [PMC6803313](https://pubmed.ncbi.nlm.nih.gov/PMC6803313/)
- 998 **58. Linking the International Wheat Genome Sequencing Consortium bread wheat reference**
999 **genome sequence to wheat genetic and phenomic data**
1000 Michael Alaux, Jane Rogers, Thomas Letellier, Raphaël Flores, Françoise Alfama, Cyril Pommier, Nacer
1001 Mohellibi, Sophie Durand, Erik Kimmel, Célia Michotey, ... International Wheat Genome Sequencing
1002 Consortium
1003 *Genome Biology* (2018-08-17) <https://doi.org/gf23xv>
1004 DOI: [10.1186/s13059-018-1491-4](https://doi.org/10.1186/s13059-018-1491-4) · PMID: [30115101](https://pubmed.ncbi.nlm.nih.gov/30115101/) · PMCID: [PMC6097284](https://pubmed.ncbi.nlm.nih.gov/PMC6097284/)
- 1005 **59. FAIR: A Call to Make Published Data More Findable, Accessible, Interoperable, and Reusable**
1006 Leonore Reiser, Lisa Harper, Michael Freeling, Bin Han, Sheng Luan
1007 *Molecular Plant* (2018-09) <https://doi.org/ggwp69>
1008 DOI: [10.1016/j.molp.2018.07.005](https://doi.org/10.1016/j.molp.2018.07.005) · PMID: [30076986](https://pubmed.ncbi.nlm.nih.gov/30076986/)
- 1009 **60. The Integrative Human Microbiome Project**
1010 The Integrative HMP (iHMP) Research Network Consortium
1011 *Nature* (2019-05-29) <https://doi.org/gf3wp9>
1012 DOI: [10.1038/s41586-019-1238-8](https://doi.org/10.1038/s41586-019-1238-8) · PMID: [31142853](https://pubmed.ncbi.nlm.nih.gov/31142853/) · PMCID: [PMC6784865](https://pubmed.ncbi.nlm.nih.gov/PMC6784865/)

- 1013 **61. The Pfam protein families database in 2019**
1014 Sara El-Gebali, Jaina Mistry, Alex Bateman, Sean R Eddy, Aurélien Luciani, Simon C Potter, Matloob
1015 Qureshi, Lorna J Richardson, Gustavo A Salazar, Alfredo Smart, ... Robert D Finn
1016 *Nucleic Acids Research* (2019-01-08) <https://doi.org/gfhx7r>
1017 DOI: [10.1093/nar/gky995](https://doi.org/10.1093/nar/gky995) · PMID: [30357350](https://pubmed.ncbi.nlm.nih.gov/30357350/) · PMCID: [PMC6324024](https://pubmed.ncbi.nlm.nih.gov/PMC6324024/)
- 1018 **62. The International Nucleotide Sequence Database Collaboration**
1019 Guy Cochrane, Ilene Karsch-Mizrachi, Toshihisa Takagi, International Nucleotide
1020 Sequence Database Collaboration
1021 *Nucleic Acids Research* (2016-01-04) <https://doi.org/gf28sk>
1022 DOI: [10.1093/nar/gkv1323](https://doi.org/10.1093/nar/gkv1323) · PMID: [26657633](https://pubmed.ncbi.nlm.nih.gov/26657633/) · PMCID: [PMC4702924](https://pubmed.ncbi.nlm.nih.gov/PMC4702924/)
- 1023 **63. Open science resources for the discovery and analysis of Tara Oceans data**
1024 Stéphane Pesant, Fabrice Not, Marc Picheral, Stefanie Kandels-Lewis, Noan Le Bescot, Gabriel Gorsky,
1025 Daniele Iudicone, Eric Karsenti, Sabrina Speich, Romain Troublé, ... Sarah Searson
1026 *Scientific Data* (2015-05-26) <https://doi.org/gd32xw>
1027 DOI: [10.1038/sdata.2015.23](https://doi.org/10.1038/sdata.2015.23) · PMID: [26029378](https://pubmed.ncbi.nlm.nih.gov/26029378/) · PMCID: [PMC4443879](https://pubmed.ncbi.nlm.nih.gov/PMC4443879/)
- 1028 **64. Gene Expression Omnibus: NCBI gene expression and hybridization array data repository**
1029 R. Edgar
1030 *Nucleic Acids Research* (2002-01-01) <https://doi.org/fttpkn>
1031 DOI: [10.1093/nar/30.1.207](https://doi.org/10.1093/nar/30.1.207) · PMID: [11752295](https://pubmed.ncbi.nlm.nih.gov/11752295/) · PMCID: [PMC99122](https://pubmed.ncbi.nlm.nih.gov/PMC99122/)
- 1032 **65. WormBase: a modern Model Organism Information Resource**
1033 Todd W Harris, Valerio Arnaboldi, Scott Cain, Juancarlos Chan, Wen J Chen, Jaehyoung Cho, Paul
1034 Davis, Sibyl Gao, Christian A Grove, Ranjana Kishore, ... Paul W Sternberg
1035 *Nucleic Acids Research* (2019-10-23) <https://doi.org/ggv5zk>
1036 DOI: [10.1093/nar/gkz920](https://doi.org/10.1093/nar/gkz920) · PMID: [31642470](https://pubmed.ncbi.nlm.nih.gov/31642470/) · PMCID: [PMC7145598](https://pubmed.ncbi.nlm.nih.gov/PMC7145598/)
- 1037 **66. Open Science Framework (OSF)**
1038 Erin D. Foster, MSLS, Ariel Deardorff, MLIS
1039 *Journal of the Medical Library Association* (2017-04-04) <https://doi.org/gfxvhq>
1040 DOI: [10.5195/jmla.2017.88](https://doi.org/10.5195/jmla.2017.88) · PMCID: [PMC5370619](https://pubmed.ncbi.nlm.nih.gov/PMC5370619/)
- 1041 **67. Erratum: How many biological replicates are needed in an RNA-seq experiment and which**
1042 **differential expression tool should you use?**
1043 Nicholas J. Schurch, Pietà Schofield, Marek Gierliński, Christian Cole, Alexander Sherstnev, Vijender
1044 Singh, Nicola Wrobel, Karim Gharbi, Gordon G. Simpson, Tom Owen-Hughes, ... Geoffrey J. Barton
1045 *RNA* (2016-09-16) <https://doi.org/ggv5zr>
1046 DOI: [10.1261/rna.058339.116](https://doi.org/10.1261/rna.058339.116) · PMID: [27638913](https://pubmed.ncbi.nlm.nih.gov/27638913/) · PMCID: [PMC5029462](https://pubmed.ncbi.nlm.nih.gov/PMC5029462/)
- 1047 **68. Power analysis and sample size estimation for RNA-Seq differential expression**
1048 Travers Ching, Sijia Huang, Lana X. Garmire
1049 *RNA* (2014-11) <https://doi.org/f6nstp>
1050 DOI: [10.1261/rna.046011.114](https://doi.org/10.1261/rna.046011.114) · PMID: [25246651](https://pubmed.ncbi.nlm.nih.gov/25246651/) · PMCID: [PMC4201821](https://pubmed.ncbi.nlm.nih.gov/PMC4201821/)

- 1051 **69. Unlocking the potential of metagenomics through replicated experimental design**
1052 Rob Knight, Janet Jansson, Dawn Field, Noah Fierer, Narayan Desai, Jed A Fuhrman, Phil Hugenholtz,
1053 Daniel van der Lelie, Folker Meyer, Rick Stevens, ... Jack A Gilbert
1054 *Nature Biotechnology* (2012-06-07) <https://doi.org/f32nds>
1055 DOI: [10.1038/nbt.2235](https://doi.org/10.1038/nbt.2235) · PMID: [22678395](https://pubmed.ncbi.nlm.nih.gov/22678395/) · PMCID: [PMC4902277](https://pubmed.ncbi.nlm.nih.gov/PMC4902277/)
- 1056 **70. Contamination in Low Microbial Biomass Microbiome Studies: Issues and Recommendations**
1057 Raphael Eisenhofer, Jeremiah J. Minich, Clarisse Marotz, Alan Cooper, Rob Knight, Laura S. Weyrich
1058 *Trends in Microbiology* (2019-02) <https://doi.org/gfpfkt>
1059 DOI: [10.1016/j.tim.2018.11.003](https://doi.org/10.1016/j.tim.2018.11.003) · PMID: [30497919](https://pubmed.ncbi.nlm.nih.gov/30497919/)
- 1060 **71. Consistent and correctable bias in metagenomic sequencing experiments**
1061 Michael R McLaren, Amy D Willis, Benjamin J Callahan
1062 *eLife* (2019-09-10) <https://doi.org/gf7wbr>
1063 DOI: [10.7554/elife.46923](https://doi.org/10.7554/elife.46923) · PMID: [31502536](https://pubmed.ncbi.nlm.nih.gov/31502536/) · PMCID: [PMC6739870](https://pubmed.ncbi.nlm.nih.gov/PMC6739870/)
- 1064 **72. From Benchtop to Desktop: Important Considerations when Designing Amplicon Sequencing**
1065 **Workflows**
1066 Dáithí C. Murray, Megan L. Coghlan, Michael Bunce
1067 *PLOS ONE* (2015-04-22) <https://doi.org/f7hjz7>
1068 DOI: [10.1371/journal.pone.0124671](https://doi.org/10.1371/journal.pone.0124671) · PMID: [25902146](https://pubmed.ncbi.nlm.nih.gov/25902146/) · PMCID: [PMC4406758](https://pubmed.ncbi.nlm.nih.gov/PMC4406758/)
- 1069 **73. Assessment of variation in microbial community amplicon sequencing by the Microbiome**
1070 **Quality Control (MBQC) project consortium**
1071 Rashmi Sinha, Galeb Abu-Ali, Emily Vogtmann, Anthony A Fodor, Boyu Ren, Amnon Amir, Emma
1072 Schwager, Jonathan Crabtree, Siyuan Ma, Christian C Abnet, ... The Microbiome Quality Control Project
1073 Consortium
1074 *Nature Biotechnology* (2017-10-02) <https://doi.org/gbzrdx>
1075 DOI: [10.1038/nbt.3981](https://doi.org/10.1038/nbt.3981) · PMID: [28967885](https://pubmed.ncbi.nlm.nih.gov/28967885/) · PMCID: [PMC5839636](https://pubmed.ncbi.nlm.nih.gov/PMC5839636/)
- 1076 **74. Completing bacterial genome assemblies: strategy and performance comparisons**
1077 Yu-Chieh Liao, Shu-Hung Lin, Hsin-Hung Lin
1078 *Scientific Reports* (2015-03-04) <https://doi.org/f686w8>
1079 DOI: [10.1038/srep08747](https://doi.org/10.1038/srep08747) · PMID: [25735824](https://pubmed.ncbi.nlm.nih.gov/25735824/) · PMCID: [PMC4348652](https://pubmed.ncbi.nlm.nih.gov/PMC4348652/)
- 1080 **75. Earth BioGenome Project: Sequencing life for the future of life**
1081 Harris A. Lewin, Gene E. Robinson, W. John Kress, William J. Baker, Jonathan Coddington, Keith A.
1082 Crandall, Richard Durbin, Scott V. Edwards, Félix Forest, M. Thomas P. Gilbert, ... Guojie Zhang
1083 *Proceedings of the National Academy of Sciences* (2018-04-24) <https://doi.org/gdh5vz>
1084 DOI: [10.1073/pnas.1720115115](https://doi.org/10.1073/pnas.1720115115) · PMID: [29686065](https://pubmed.ncbi.nlm.nih.gov/29686065/) · PMCID: [PMC5924910](https://pubmed.ncbi.nlm.nih.gov/PMC5924910/)
- 1085 **76. Opportunities and challenges in long-read sequencing data analysis**
1086 Shanika L. Amarasinghe, Shian Su, Xueyi Dong, Luke Zappia, Matthew E. Ritchie, Quentin Gouil
1087 *Genome Biology* (2020-02-07) <https://doi.org/ggkpv7>
1088 DOI: [10.1186/s13059-020-1935-5](https://doi.org/10.1186/s13059-020-1935-5) · PMID: [32033565](https://pubmed.ncbi.nlm.nih.gov/32033565/) · PMCID: [PMC7006217](https://pubmed.ncbi.nlm.nih.gov/PMC7006217/)

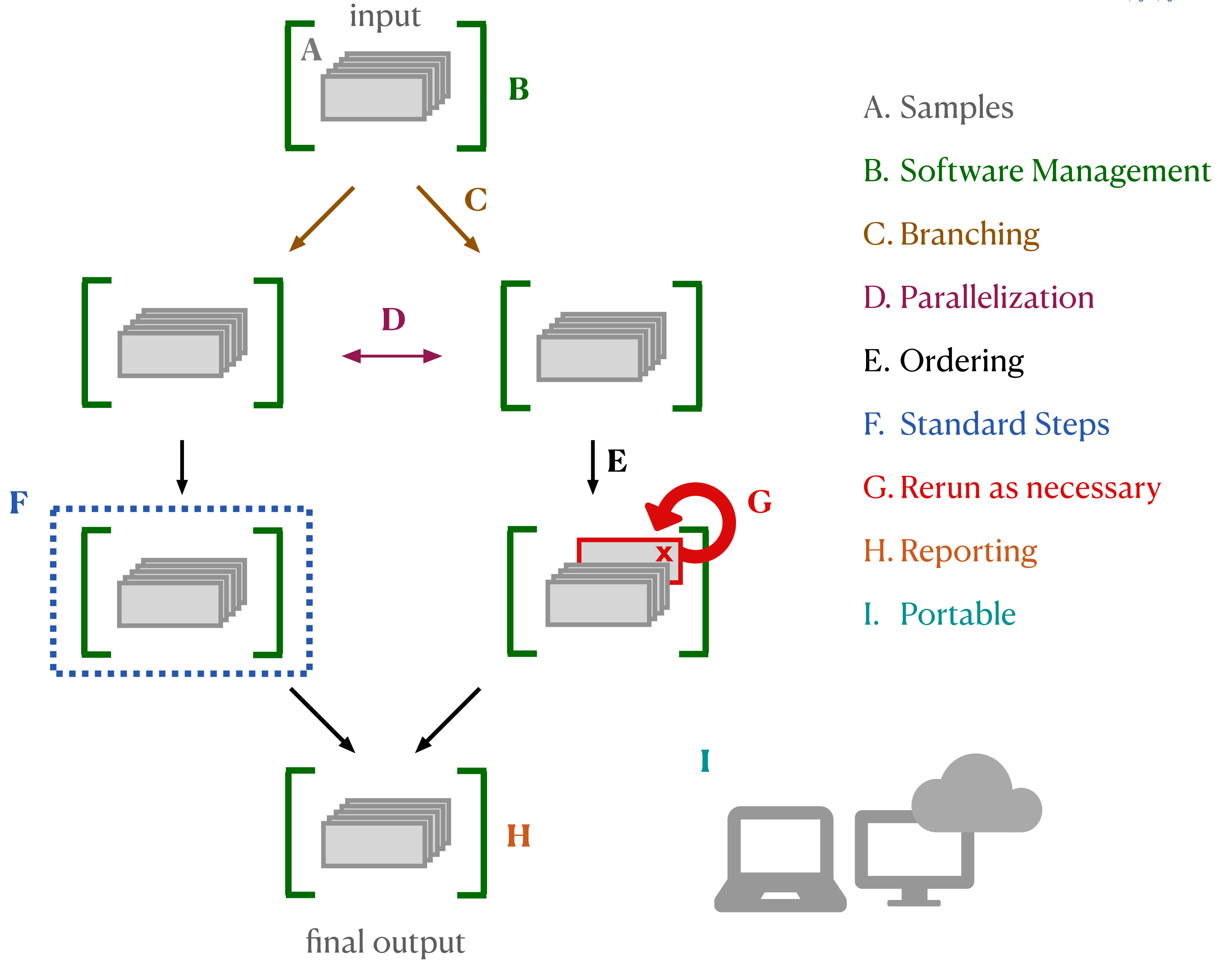
- 1089 **77. Whole-genome sequencing of eukaryotes: From sequencing of DNA fragments to a genome**
1090 **assembly**
1091 K. S. Zadesenets, N. I. Ershov, N. B. Rubtsov
1092 *Russian Journal of Genetics* (2017-07-12) <https://doi.org/gg26n5>
1093 DOI: [10.1134/s102279541705012x](https://doi.org/10.1134/s102279541705012x)
- 1094 **78. Ten steps to get started in Genome Assembly and Annotation**
1095 Victoria Dominguez Del Angel, Erik Hjerde, Lieven Sterck, Salvadors Capella-Gutierrez, Cederic
1096 Notredame, Olga Vinnere Pettersson, Joelle Amselem, Laurent Bourri, Stephanie Bocs, Christophe Klopp,
1097 ... Henrik Lantz
1098 *F1000Research* (2018-02-05) <https://doi.org/gdq9zi>
1099 DOI: [10.12688/f1000research.13598.1](https://doi.org/10.12688/f1000research.13598.1) · PMID: [29568489](https://pubmed.ncbi.nlm.nih.gov/29568489/) · PMCID: [PMC5850084](https://pubmed.ncbi.nlm.nih.gov/PMC5850084/)
- 1100 **79. Whole-genome sequencing approaches for conservation biology: Advantages, limitations and**
1101 **practical recommendations**
1102 Angela P. Fuentes-Pardo, Daniel E. Ruzzante
1103 *Molecular Ecology* (2017-10) <https://doi.org/gfzn9r>
1104 DOI: [10.1111/mec.14264](https://doi.org/10.1111/mec.14264) · PMID: [28746784](https://pubmed.ncbi.nlm.nih.gov/28746784/)
- 1105 **80. Bioinformatic processing of RAD-seq data dramatically impacts downstream population genetic**
1106 **inference**
1107 Aaron B. A. Shafer, Claire R. Peart, Sergio Tusso, Inbar Maayan, Alan Brelsford, Christopher W. Wheat,
1108 Jochen B. W. Wolf
1109 *Methods in Ecology and Evolution* (2017-08) <https://doi.org/gbrdv5>
1110 DOI: [10.1111/2041-210x.12700](https://doi.org/10.1111/2041-210x.12700)
- 1111 **81. Selecting RAD-Seq Data Analysis Parameters for Population Genetics: The More the Better?**
1112 Natalia Díaz-Arce, Naiara Rodríguez-Ezpeleta
1113 *Frontiers in Genetics* (2019-05-29) <https://doi.org/gg26n7>
1114 DOI: [10.3389/fgene.2019.00533](https://doi.org/10.3389/fgene.2019.00533) · PMID: [31191624](https://pubmed.ncbi.nlm.nih.gov/31191624/) · PMCID: [PMC6549478](https://pubmed.ncbi.nlm.nih.gov/PMC6549478/)
- 1115 **82. Harnessing the power of RADseq for ecological and evolutionary genomics**
1116 Kimberly R. Andrews, Jeffrey M. Good, Michael R. Miller, Gordon Luikart, Paul A. Hohenlohe
1117 *Nature Reviews Genetics* (2016-01-05) <https://doi.org/gd85wf>
1118 DOI: [10.1038/nrg.2015.28](https://doi.org/10.1038/nrg.2015.28) · PMID: [26729255](https://pubmed.ncbi.nlm.nih.gov/26729255/) · PMCID: [PMC4823021](https://pubmed.ncbi.nlm.nih.gov/PMC4823021/)
- 1119 **83. Unbroken: RADseq remains a powerful tool for understanding the genetics of adaptation in**
1120 **natural populations**
1121 Julian M. Catchen, Paul A. Hohenlohe, Louis Bernatchez, W. Chris Funk, Kimberly R. Andrews, Fred W.
1122 Allendorf
1123 *Molecular Ecology Resources* (2017-05) <https://doi.org/f92pff>
1124 DOI: [10.1111/1755-0998.12669](https://doi.org/10.1111/1755-0998.12669) · PMID: [28319339](https://pubmed.ncbi.nlm.nih.gov/28319339/)
- 1125 **84. Responsible RAD: Striving for best practices in population genomic studies of adaptation**
1126 David B. Lowry, Sean Hoban, Joanna L. Kelley, Katie E. Lotterhos, Laura K. Reed, Michael F. Antolin,
1127 Andrew Storfer

- 1128 *Molecular Ecology Resources* (2017-05) <https://doi.org/gfzn6h>
1129 DOI: [10.1111/1755-0998.12677](https://doi.org/10.1111/1755-0998.12677) · PMID: [28382730](https://pubmed.ncbi.nlm.nih.gov/28382730/)
- 1130 **85. Design and computational analysis of single-cell RNA-sequencing experiments**
1131 Rhonda Bacher, Christina Kendziorski
1132 *Genome Biology* (2016-04-07) <https://doi.org/gc958g>
1133 DOI: [10.1186/s13059-016-0927-y](https://doi.org/10.1186/s13059-016-0927-y) · PMID: [27052890](https://pubmed.ncbi.nlm.nih.gov/27052890/) · PMCID: [PMC4823857](https://pubmed.ncbi.nlm.nih.gov/PMC4823857/)
- 1134 **86. A practical guide to single-cell RNA-sequencing for biomedical research and clinical**
1135 **applications**
1136 Ashraful Haque, Jessica Engel, Sarah A. Teichmann, Tapio Lönnberg
1137 *Genome Medicine* (2017-08-18) <https://doi.org/gfgppz>
1138 DOI: [10.1186/s13073-017-0467-4](https://doi.org/10.1186/s13073-017-0467-4) · PMID: [28821273](https://pubmed.ncbi.nlm.nih.gov/28821273/) · PMCID: [PMC5561556](https://pubmed.ncbi.nlm.nih.gov/PMC5561556/)
- 1139 **87. Developing a modern data workflow for evolving data**
1140 Glenda M. Yenni, Erica M. Christensen, Ellen K. Bledsoe, Sarah R. Supp, Renata M. Diaz, Ethan P.
1141 White, S. K. Morgan Ernest
1142 *bioRxiv* (2018-07-24) <https://doi.org/gdqbzn>
1143 DOI: [10.1101/344804](https://doi.org/10.1101/344804)
- 1144 **88. figshare - credit for all your research** <https://figshare.com/>
- 1145 **89. Dryad Home - Publish and Preserve your Data** <https://datadryad.org/stash>
- 1146 **90. RClone: a package to identify MultiLocus Clonal Lineages and handle clonal data sets in .**
1147 Diane Bailleul, Solenn Stoeckel, Sophie Arnaud-Haond
1148 *Methods in Ecology and Evolution* (2016-08) <https://doi.org/f82t65>
1149 DOI: [10.1111/2041-210x.12550](https://doi.org/10.1111/2041-210x.12550)
- 1150 **91. SRA in the Cloud** <https://www.ncbi.nlm.nih.gov/sra/docs/sra-cloud/>
- 1151 **92. Cyberduck | Libre server and cloud storage browser for Mac and Windows with support for**
1152 **FTP, SFTP, WebDAV, Amazon S3, OpenStack Swift, Backblaze B2, Microsoft Azure & OneDrive,**
1153 **Google Drive and Dropbox** <https://cyberduck.io/>
- 1154 **93. Babraham Bioinformatics - FastQC A Quality Control tool for High Throughput Sequence**
1155 **Data** <https://www.bioinformatics.babraham.ac.uk/projects/fastqc/>
- 1156 **94. Integrative Genomics Viewer (IGV): high-performance genomics data visualization and**
1157 **exploration**
1158 H. Thorvaldsdottir, J. T. Robinson, J. P. Mesirov
1159 *Briefings in Bioinformatics* (2012-04-19) <https://doi.org/f4sc43>
1160 DOI: [10.1093/bib/bbs017](https://doi.org/10.1093/bib/bbs017) · PMID: [22517427](https://pubmed.ncbi.nlm.nih.gov/22517427/) · PMCID: [PMC3603213](https://pubmed.ncbi.nlm.nih.gov/PMC3603213/)
- 1161 **95. Salmon provides fast and bias-aware quantification of transcript expression**
1162 Rob Patro, Geet Duggal, Michael I Love, Rafael A Irizarry, Carl Kingsford
1163 *Nature Methods* (2017-03-06) <https://doi.org/gcw9f5>
1164 DOI: [10.1038/nmeth.4197](https://doi.org/10.1038/nmeth.4197) · PMID: [28263959](https://pubmed.ncbi.nlm.nih.gov/28263959/) · PMCID: [PMC5600148](https://pubmed.ncbi.nlm.nih.gov/PMC5600148/)

- 1165 96. **MultiQC: summarize analysis results for multiple tools and samples in a single report**
1166 Philip Ewels, Måns Magnusson, Sverker Lundin, Max Källér
1167 *Bioinformatics* (2016-10-01) <https://doi.org/f3s996>
1168 DOI: [10.1093/bioinformatics/btw354](https://doi.org/10.1093/bioinformatics/btw354) · PMID: [27312411](https://pubmed.ncbi.nlm.nih.gov/27312411/) · PMCID: [PMC5039924](https://pubmed.ncbi.nlm.nih.gov/PMC5039924/)
- 1169 97. **Identifying and mitigating bias in next-generation sequencing methods for chromatin biology**
1170 Clifford A. Meyer, X. Shirley Liu
1171 *Nature Reviews Genetics* (2014-09-16) <https://doi.org/ggd9m9>
1172 DOI: [10.1038/nrg3788](https://doi.org/10.1038/nrg3788) · PMID: [25223782](https://pubmed.ncbi.nlm.nih.gov/25223782/) · PMCID: [PMC4473780](https://pubmed.ncbi.nlm.nih.gov/PMC4473780/)
- 1173 98. **The impact of amplification on differential expression analyses by RNA-seq**
1174 Swati Parekh, Christoph Ziegenhain, Beate Vieth, Wolfgang Enard, Ines Hellmann
1175 *Scientific Reports* (2016-05-09) <https://doi.org/f8kzcp>
1176 DOI: [10.1038/srep25533](https://doi.org/10.1038/srep25533) · PMID: [27156886](https://pubmed.ncbi.nlm.nih.gov/27156886/) · PMCID: [PMC4860583](https://pubmed.ncbi.nlm.nih.gov/PMC4860583/)
- 1177 99. **Detection and Removal of PCR Duplicates in Population Genomic ddRAD Studies by Addition**
1178 **of a Degenerate Base Region (DBR) in Sequencing Adapters**
1179 Hannah Schweyen, Andrey Rozenberg, Florian Leese
1180 *The Biological Bulletin* (2014-10) <https://doi.org/f6q76c>
1181 DOI: [10.1086/bblv227n2p146](https://doi.org/10.1086/bblv227n2p146) · PMID: [25411373](https://pubmed.ncbi.nlm.nih.gov/25411373/)
- 1182 100. **Elimination of PCR duplicates in RNA-seq and small RNA-seq using unique molecular**
1183 **identifiers**
1184 Yu Fu, Pei-Hsuan Wu, Timothy Beane, Phillip D. Zamore, Zhiping Weng
1185 *BMC Genomics* (2018-07-13) <https://doi.org/ggv5zp>
1186 DOI: [10.1186/s12864-018-4933-1](https://doi.org/10.1186/s12864-018-4933-1) · PMID: [30001700](https://pubmed.ncbi.nlm.nih.gov/30001700/) · PMCID: [PMC6044086](https://pubmed.ncbi.nlm.nih.gov/PMC6044086/)
- 1187 101. **Biased estimates of clonal evolution and subclonal heterogeneity can arise from PCR**
1188 **duplicates in deep sequencing experiments**
1189 Erin N Smith, Kristen Jepsen, Mahdieh Khosroheidari, Laura Z Rassenti, Matteo D'Antonio, Emanuela
1190 M Ghia, Dennis A Carson, Catriona HM Jamieson, Thomas J Kipps, Kelly A Frazer
1191 *Genome Biology* (2014-08-07) <https://doi.org/ggfhv7>
1192 DOI: [10.1186/s13059-014-0420-4](https://doi.org/10.1186/s13059-014-0420-4) · PMID: [25103687](https://pubmed.ncbi.nlm.nih.gov/25103687/) · PMCID: [PMC4165357](https://pubmed.ncbi.nlm.nih.gov/PMC4165357/)
- 1193 102. **Evidence for extensive horizontal gene transfer from the draft genome of a tardigrade**
1194 Thomas C. Boothby, Jennifer R. Tenlen, Frank W. Smith, Jeremy R. Wang, Kiera A. Patanella, Erin
1195 Osborne Nishimura, Sophia C. Tintori, Qing Li, Corbin D. Jones, Mark Yandell, ... Bob Goldstein
1196 *Proceedings of the National Academy of Sciences* (2015-12-29) <https://doi.org/9gs>
1197 DOI: [10.1073/pnas.1510461112](https://doi.org/10.1073/pnas.1510461112) · PMID: [26598659](https://pubmed.ncbi.nlm.nih.gov/26598659/) · PMCID: [PMC4702960](https://pubmed.ncbi.nlm.nih.gov/PMC4702960/)
- 1198 103. **No evidence for extensive horizontal gene transfer in the genome of the tardigrade *Hypsibius***
1199 ***dujardini***
1200 Georgios Koutsovoulos, Sujai Kumar, Dominik R. Laetsch, Lewis Stevens, Jennifer Daub, Claire Conlon,
1201 Habib Maroon, Fran Thomas, Aziz A. Aboobaker, Mark Blaxter
1202 *Proceedings of the National Academy of Sciences* (2016-05-03) <https://doi.org/f8nrtd>
1203 DOI: [10.1073/pnas.1600338113](https://doi.org/10.1073/pnas.1600338113) · PMID: [27035985](https://pubmed.ncbi.nlm.nih.gov/27035985/) · PMCID: [PMC4983863](https://pubmed.ncbi.nlm.nih.gov/PMC4983863/)

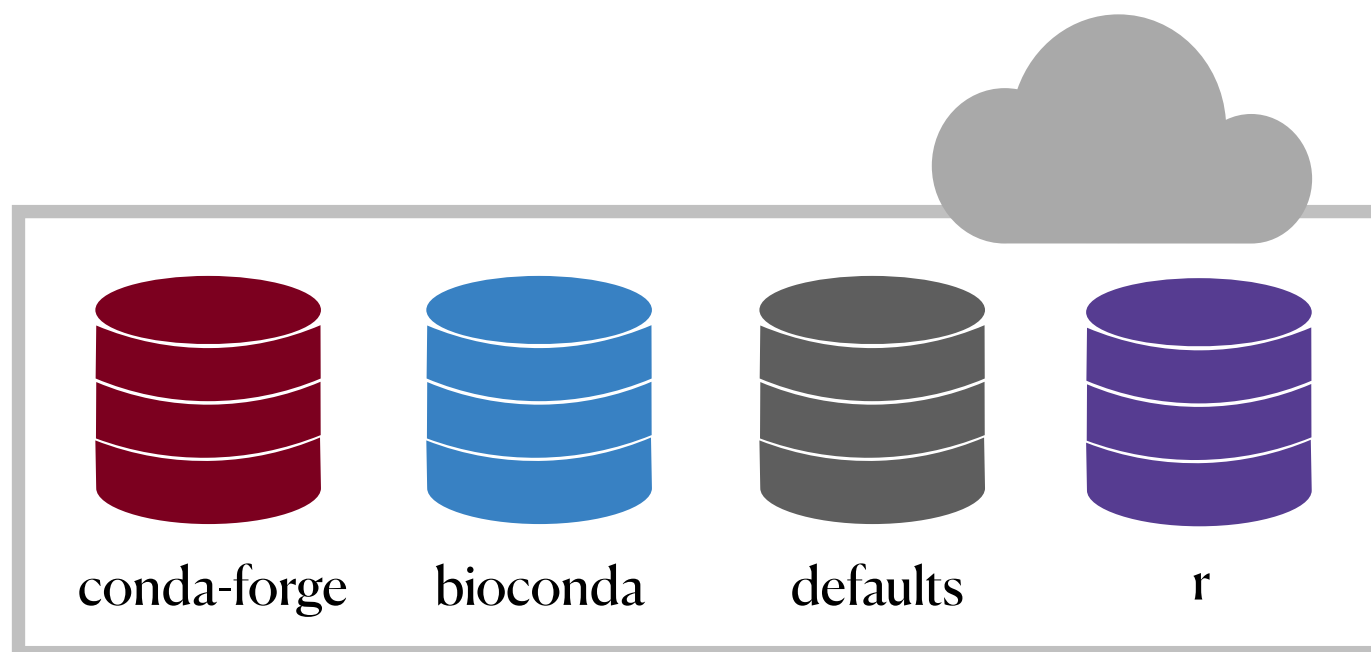
- 1204 104. **Large-scale contamination of microbial isolate genomes by Illumina PhiX control**
1205 Supratim Mukherjee, Marcel Huntemann, Natalia Ivanova, Nikos C Kyrpides, Amrita Pati
1206 *Standards in Genomic Sciences* (2015-03-30) <https://doi.org/ggv5zn>
1207 DOI: [10.1186/1944-3277-10-18](https://doi.org/10.1186/1944-3277-10-18) · PMID: [26203331](https://pubmed.ncbi.nlm.nih.gov/26203331/) · PMCID: [PMC4511556](https://pubmed.ncbi.nlm.nih.gov/PMC4511556/)
- 1208 105. **Index hopping on the Illumina HiseqX platform and its consequences for ancient DNA studies**
1209 Tom van der Valk, Francesco Vezzi, Mattias Ormestad, Love Dalén, Katerina Guschanski
1210 *Molecular Ecology Resources* (2019-05-05) <https://doi.org/gfwtvw>
1211 DOI: [10.1111/1755-0998.13009](https://doi.org/10.1111/1755-0998.13009) · PMID: [30848092](https://pubmed.ncbi.nlm.nih.gov/30848092/)
- 1212 106. **On the optimal trimming of high-throughput mRNA sequence data**
1213 Matthew D. MacManes
1214 *Frontiers in Genetics* (2014) <https://doi.org/gfn5g7>
1215 DOI: [10.3389/fgene.2014.00013](https://doi.org/10.3389/fgene.2014.00013) · PMID: [24567737](https://pubmed.ncbi.nlm.nih.gov/24567737/) · PMCID: [PMC3908319](https://pubmed.ncbi.nlm.nih.gov/PMC3908319/)
- 1216 107. **Streaming histogram sketching for rapid microbiome analytics**
1217 Will PM Rowe, Anna Paola Carrieri, Cristina Alcon-Giner, Shabhonam Caim, Alex Shaw, Kathleen Sim,
1218 J. Simon Kroll, Lindsay J. Hall, Edward O. Pyzer-Knapp, Martyn D. Winn
1219 *Microbiome* (2019-03-16) <https://doi.org/ggv5zq>
1220 DOI: [10.1186/s40168-019-0653-2](https://doi.org/10.1186/s40168-019-0653-2) · PMID: [30878035](https://pubmed.ncbi.nlm.nih.gov/30878035/) · PMCID: [PMC6420756](https://pubmed.ncbi.nlm.nih.gov/PMC6420756/)
- 1221 108. **When the levee breaks: a practical guide to sketching algorithms for processing the flood of**
1222 **genomic data**
1223 Will P. M. Rowe
1224 *Genome Biology* (2019-09-13) <https://doi.org/gf8bfj>
1225 DOI: [10.1186/s13059-019-1809-x](https://doi.org/10.1186/s13059-019-1809-x) · PMID: [31519212](https://pubmed.ncbi.nlm.nih.gov/31519212/) · PMCID: [PMC6744645](https://pubmed.ncbi.nlm.nih.gov/PMC6744645/)
- 1226 109. **will-rowe/genome-sketching**
1227 GitHub
1228 <https://github.com/will-rowe/genome-sketching>
- 1229 110. **STAR: ultrafast universal RNA-seq aligner**
1230 Alexander Dobin, Carrie A. Davis, Felix Schlesinger, Jorg Drenkow, Chris Zaleski, Sonali Jha, Philippe
1231 Batut, Mark Chaisson, Thomas R. Gingeras
1232 *Bioinformatics* (2013-01) <https://doi.org/f4h523>
1233 DOI: [10.1093/bioinformatics/bts635](https://doi.org/10.1093/bioinformatics/bts635) · PMID: [23104886](https://pubmed.ncbi.nlm.nih.gov/23104886/) · PMCID: [PMC3530905](https://pubmed.ncbi.nlm.nih.gov/PMC3530905/)
- 1234 111. **Graph-based genome alignment and genotyping with HISAT2 and HISAT-genotype**
1235 Daehwan Kim, Joseph M. Paggi, Chanhee Park, Christopher Bennett, Steven L. Salzberg
1236 *Nature Biotechnology* (2019-08-02) <https://doi.org/gf5395>
1237 DOI: [10.1038/s41587-019-0201-4](https://doi.org/10.1038/s41587-019-0201-4) · PMID: [31375807](https://pubmed.ncbi.nlm.nih.gov/31375807/)
- 1238 112. **Near-optimal probabilistic RNA-seq quantification**
1239 Nicolas L Bray, Harold Pimentel, Páll Melsted, Lior Pachter
1240 *Nature Biotechnology* (2016-04-04) <https://doi.org/f8nvsp>
1241 DOI: [10.1038/nbt.3519](https://doi.org/10.1038/nbt.3519) · PMID: [27043002](https://pubmed.ncbi.nlm.nih.gov/27043002/)

- 1242 113. **RapMap: a rapid, sensitive and accurate tool for mapping RNA-seq reads to transcriptomes**
1243 Avi Srivastava, Hirak Sarkar, Nitish Gupta, Rob Patro
1244 *Bioinformatics* (2016-06-15) <https://doi.org/f8vm2j>
1245 DOI: [10.1093/bioinformatics/btw277](https://doi.org/10.1093/bioinformatics/btw277) · PMID: [27307617](https://pubmed.ncbi.nlm.nih.gov/27307617/) · PMCID: [PMC4908361](https://pubmed.ncbi.nlm.nih.gov/PMC4908361/)
- 1246 114. **The Types, Roles, and Practices of Documentation in Data Analytics Open Source Software**
1247 **Libraries**
1248 R. Stuart Geiger, Nelle Varoquaux, Charlotte Mazel-Cabasse, Chris Holdgraf
1249 *Computer Supported Cooperative Work (CSCW)* (2018-05-29) <https://doi.org/gd4rhr>
1250 DOI: [10.1007/s10606-018-9333-1](https://doi.org/10.1007/s10606-018-9333-1)
- 1251 115. **Data Carpentry: Workshops to Increase Data Literacy for Researchers**
1252 Tracy K. Teal, Karen A. Cranston, Hilmar Lapp, Ethan White, Greg Wilson, Karthik Ram, Aleksandra
1253 Pawlik
1254 *International Journal of Digital Curation* (2015-03-18) <https://doi.org/gf5hjjw>
1255 DOI: [10.2218/ijdc.v10i1.351](https://doi.org/10.2218/ijdc.v10i1.351)
- 1256 116. **BioStar: An Online Question & Answer Resource for the Bioinformatics Community**
1257 Laurence D. Parnell, Pierre Lindenbaum, Khader Shameer, Giovanni Marco Dall’Olio, Daniel C. Swan,
1258 Lars Juhl Jensen, Simon J. Cockell, Brent S. Pedersen, Mary E. Mangan, Christopher A. Miller, Istvan
1259 Albert
1260 *PLoS Computational Biology* (2011-10-27) <https://doi.org/dhsz4k>
1261 DOI: [10.1371/journal.pcbi.1002216](https://doi.org/10.1371/journal.pcbi.1002216) · PMID: [22046109](https://pubmed.ncbi.nlm.nih.gov/22046109/) · PMCID: [PMC3203049](https://pubmed.ncbi.nlm.nih.gov/PMC3203049/)
- 1262 117. **How to create a Minimal, Reproducible Example - Help Center**
1263 Stack Overflow
1264 <https://stackoverflow.com/help/minimal-reproducible-example>
- 1265 118. **FAQ: How to do a minimal reproducible example (replex) for beginners**
1266 RStudio Community
1267 (2020-03-25) [https://community.rstudio.com/t/faq-how-to-do-a-minimal-reproducible-example-replex-](https://community.rstudio.com/t/faq-how-to-do-a-minimal-reproducible-example-replex-for-beginners/23061)
1268 [for-beginners/23061](https://community.rstudio.com/t/faq-how-to-do-a-minimal-reproducible-example-replex-for-beginners/23061)
- 1269 119. **Code of conduct in open source projects**
1270 Parastou Tourani, Bram Adams, Alexander Serebrenik
1271 *Institute of Electrical and Electronics Engineers (IEEE)* (2017-02) <https://doi.org/gfzbs6>
1272 DOI: [10.1109/saner.2017.7884606](https://doi.org/10.1109/saner.2017.7884606)
- 1273 120. **Building a local community of practice in scientific programming for life scientists**
1274 Sarah L. R. Stevens, Mateusz Kuzak, Carlos Martinez, Aurelia Moser, Petra Bleeker, Marc Galland
1275 *PLoS Biology* (2018-11-28) <https://doi.org/gfpwkb>
1276 DOI: [10.1371/journal.pbio.2005561](https://doi.org/10.1371/journal.pbio.2005561) · PMID: [30485260](https://pubmed.ncbi.nlm.nih.gov/30485260/) · PMCID: [PMC6287879](https://pubmed.ncbi.nlm.nih.gov/PMC6287879/)

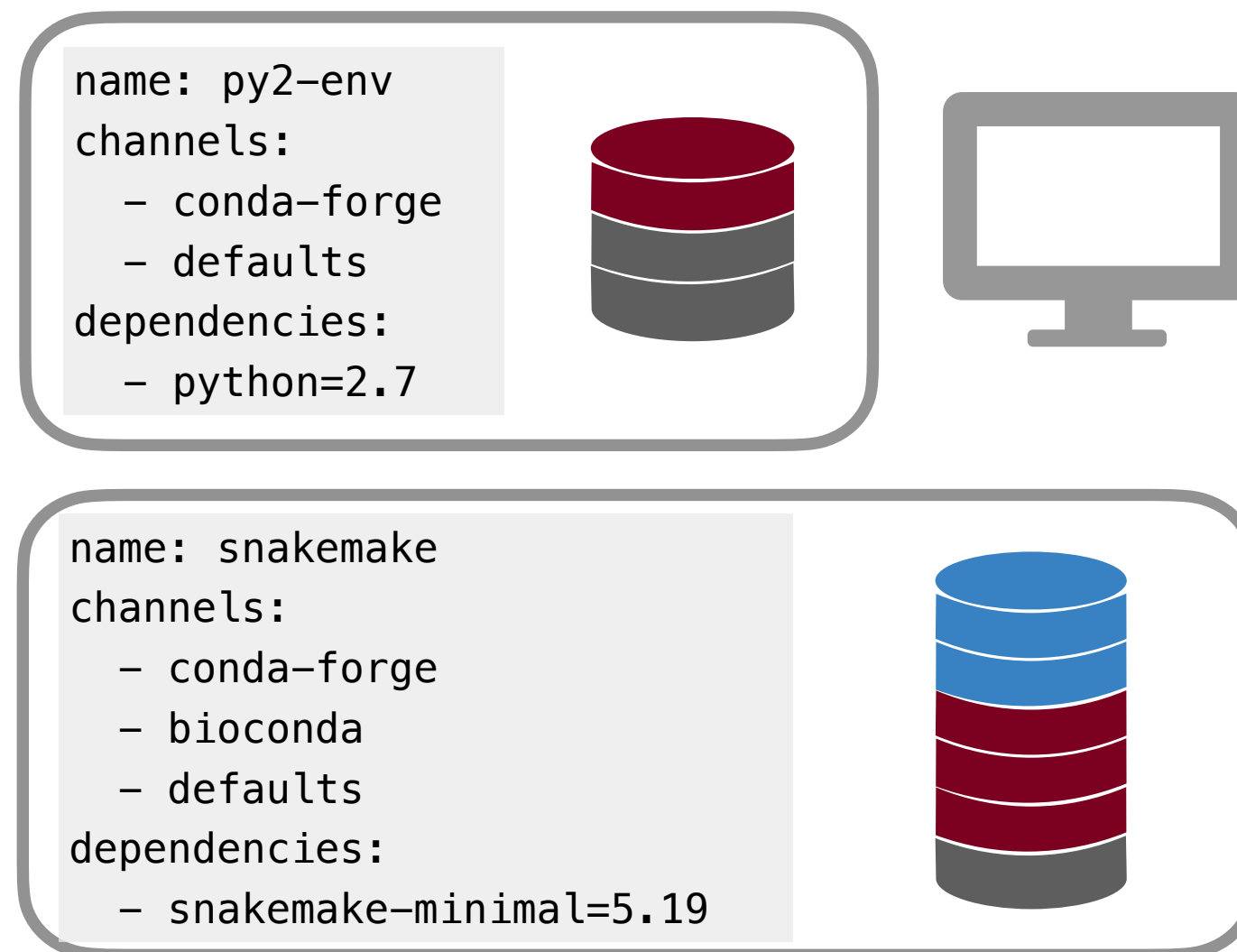


- A. Samples
- B. Software Management
- C. Branching
- D. Parallelization
- E. Ordering
- F. Standard Steps
- G. Rerun as necessary
- H. Reporting
- I. Portable

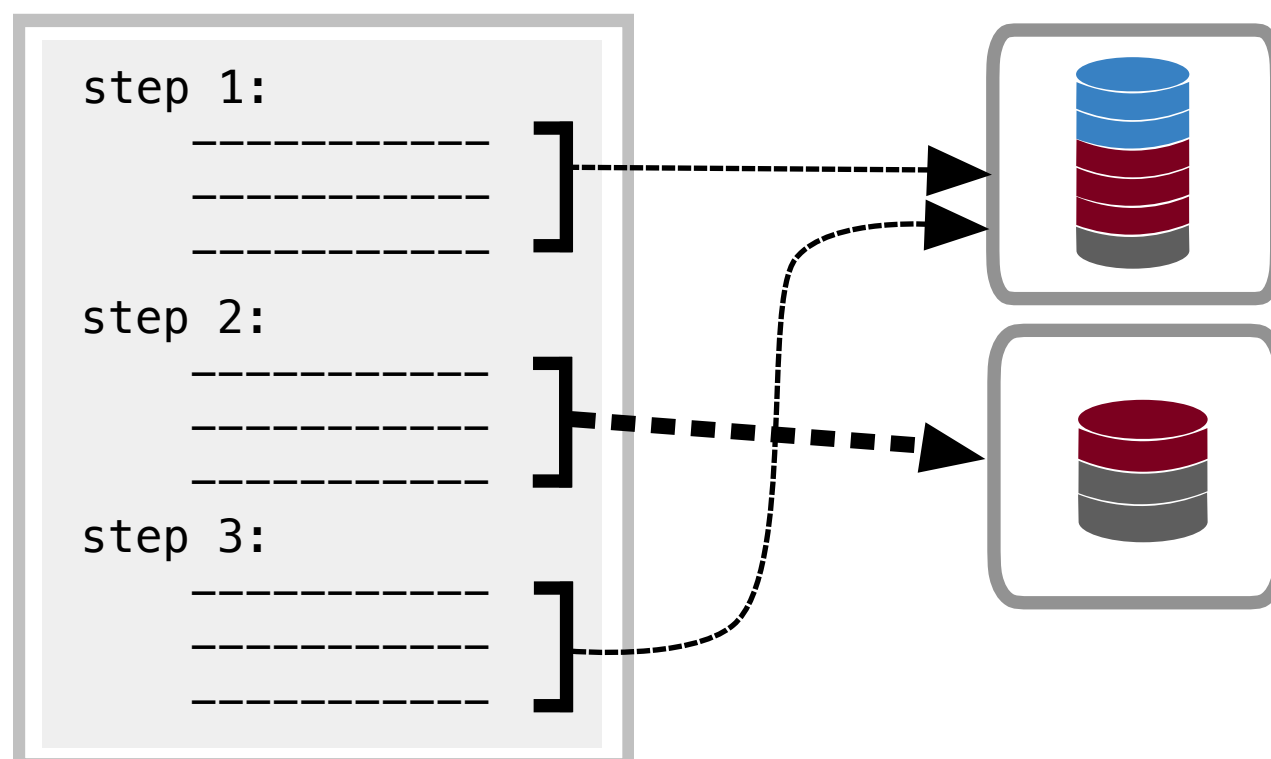
A.

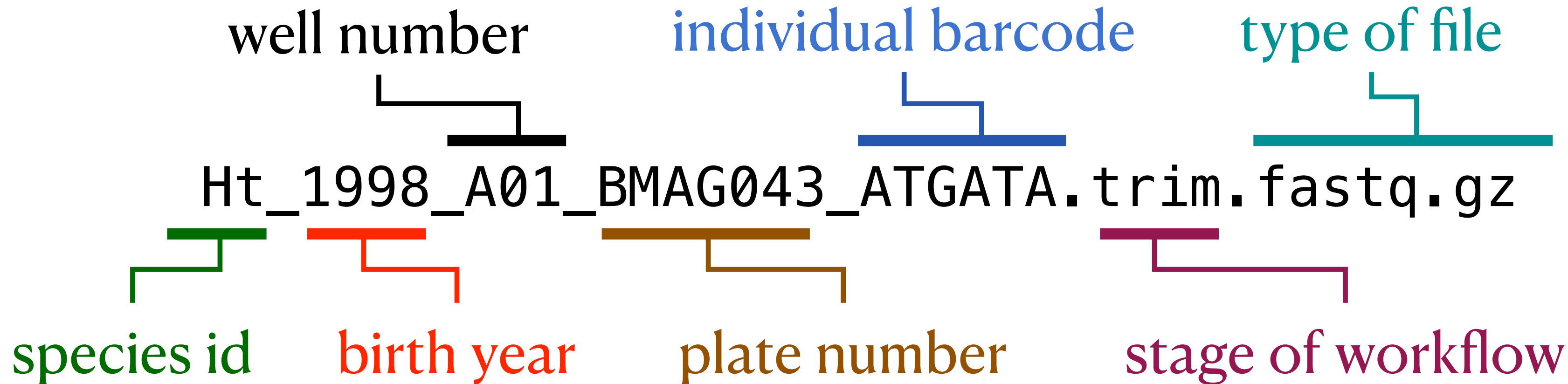


B.



C.





A title: "Distribution of generations in a long-term evolution experiment"

output: html_document

```
---  
````{r setup, include=FALSE}  
knitr::opts_chunk$set(echo = TRUE, messages = FALSE, warnings = F)
````
```

```
````{r}  
library(ggplot2)
````
```

This plot shows the number of samples sequenced at each generation in a long-term evolution experiment.

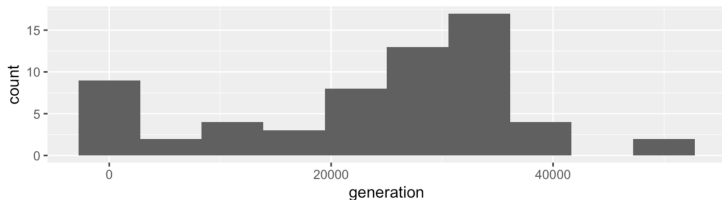
```
````{r, fig.height = 2}  
metadata <- read.csv("Ecoli_metadata_composite.tsv", sep = "\t")
ggplot(metadata, aes(x = generation)) +
 geom_histogram(bins = 10)
````
```

B Distribution of generations in a long-term evolution experiment

```
library(ggplot2)
```

This plot shows the number of samples sequenced at each generation in a long-term evolution experiment.

```
metadata <- read.csv("Ecoli_metadata_composite.tsv", sep = "\t")  
ggplot(metadata, aes(x = generation)) +  
  geom_histogram(bins = 10)
```



C Distribution of generations in a long-term evolution experiment

```
In [1]: import pandas as pd  
import matplotlib
```

This plot shows the number of samples sequenced at each generation in a long-term evolution experiment.

```
In [2]: metadata = pd.read_csv("Ecoli_metadata_composite.tsv",  
                               sep = "\t")  
plt = metadata['generation'].plot(kind = "hist")  
plt.set_xlabel("Generation")
```

```
Out[2]: Text(0.5, 0, 'Generation')
```

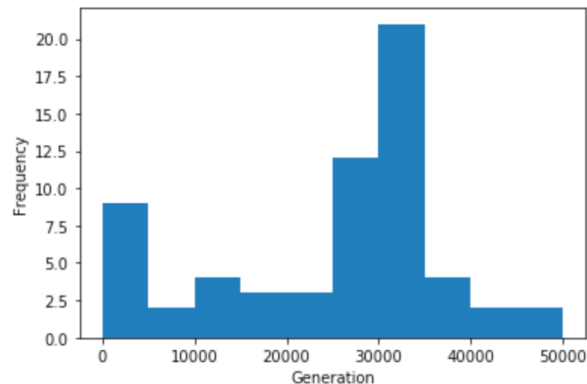
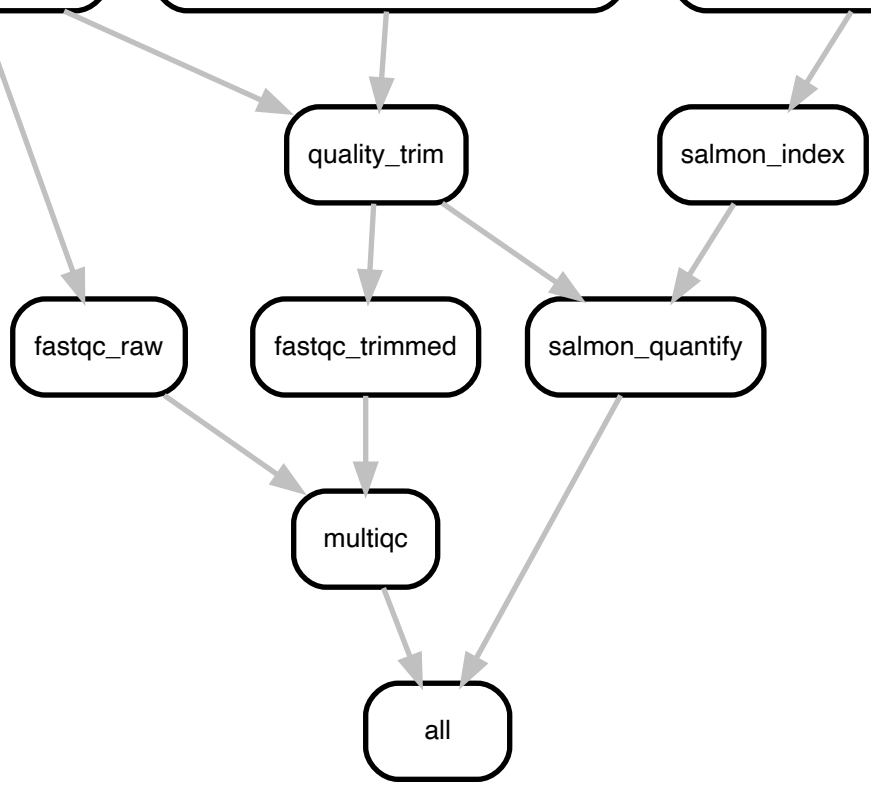


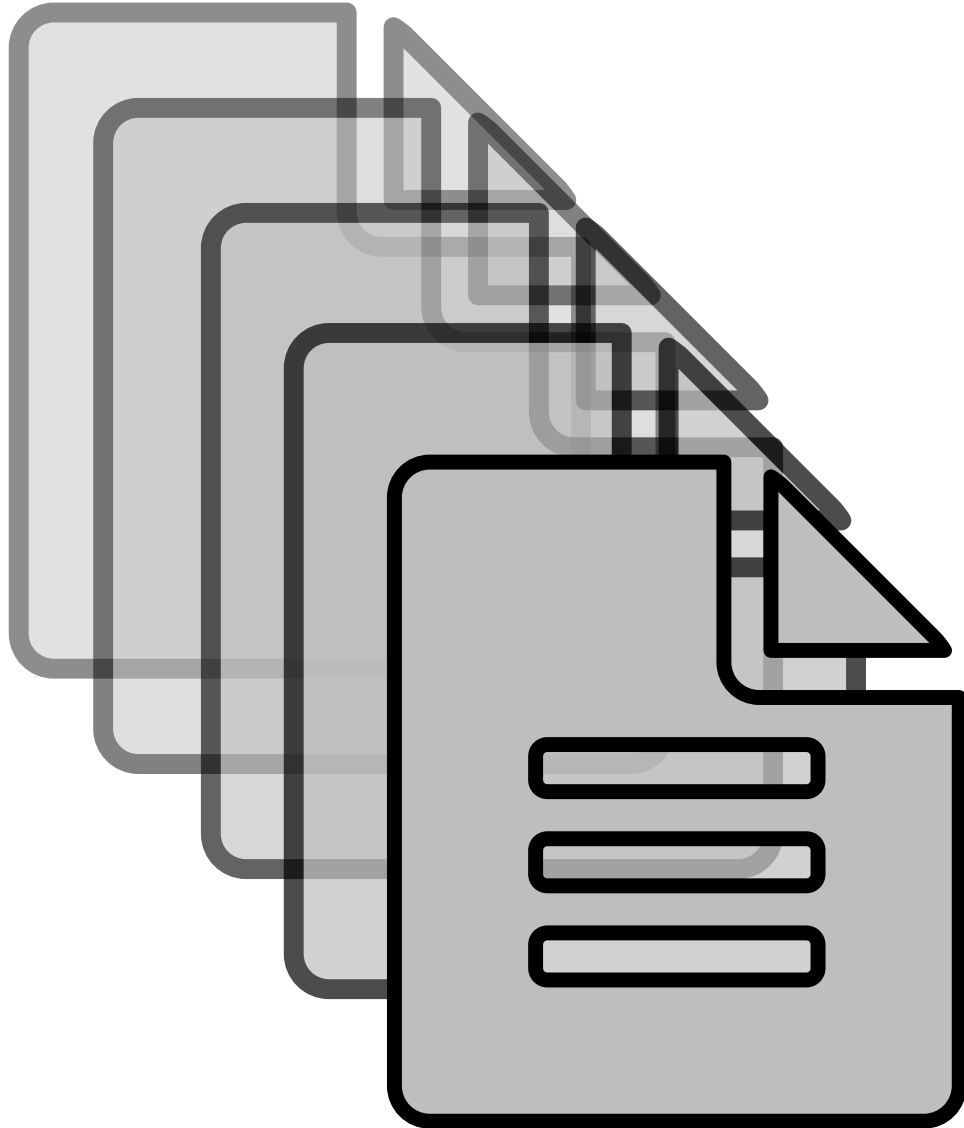
Figure 5: rnaseq dag
download_reads
sample: ERR458493

download_trimmomatic_adapter_file

download_yeast_transcriptome

[Click here to access/download;Figure;Figure5-rnaseq-dag.pdf](#)





```
Hello world  
This code is great  
The best doce ever  
Here is my code
```

commit 4c38c2c



```
Hello world  
This code is great  
The best code ever  
Here is my code
```

commit 5c6c28d

A

Select sample

PT1

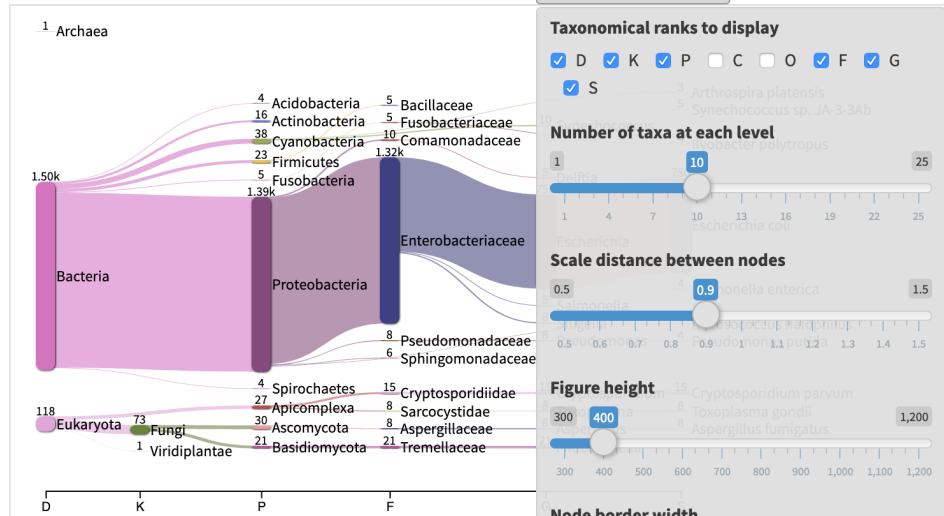
Sankey visualization

Table

Text

Hover over a node to see the abundance of the taxon in other samples.

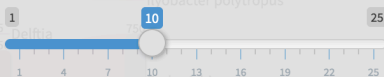
Configure Sankey ...



Taxonomical ranks to display

- D
- K
- P
- C
- O
- F
- G
- S

Number of taxa at each level



Scale distance between nodes

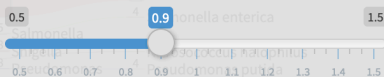
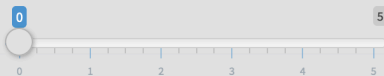


Figure height

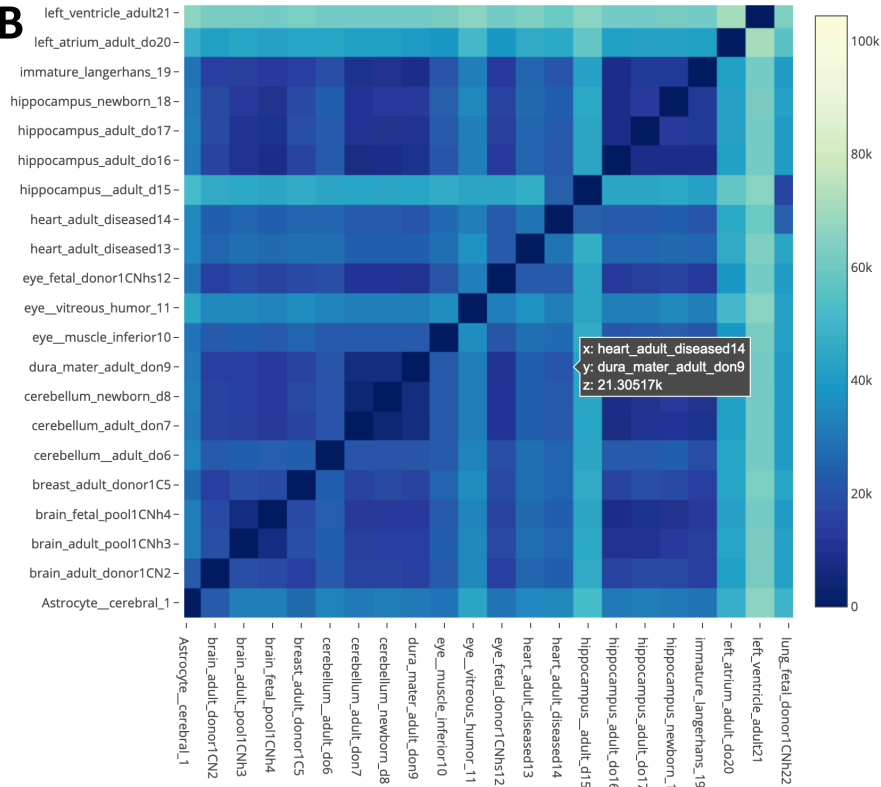


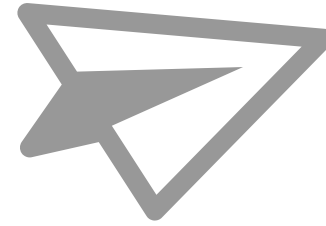
Node border width




Save Network


B






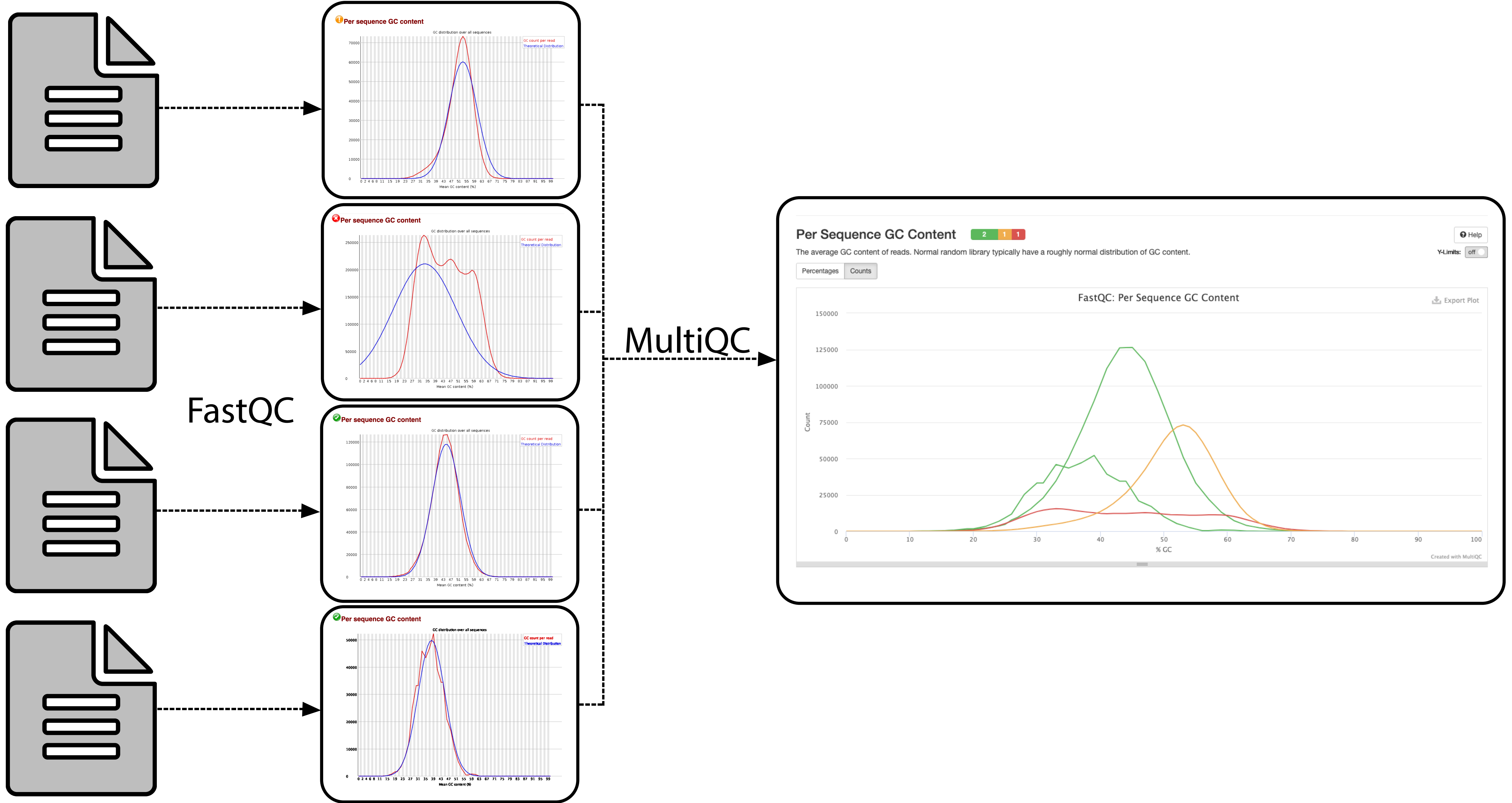
 = 758a89ad4f61

 = 98h1v48wnw3

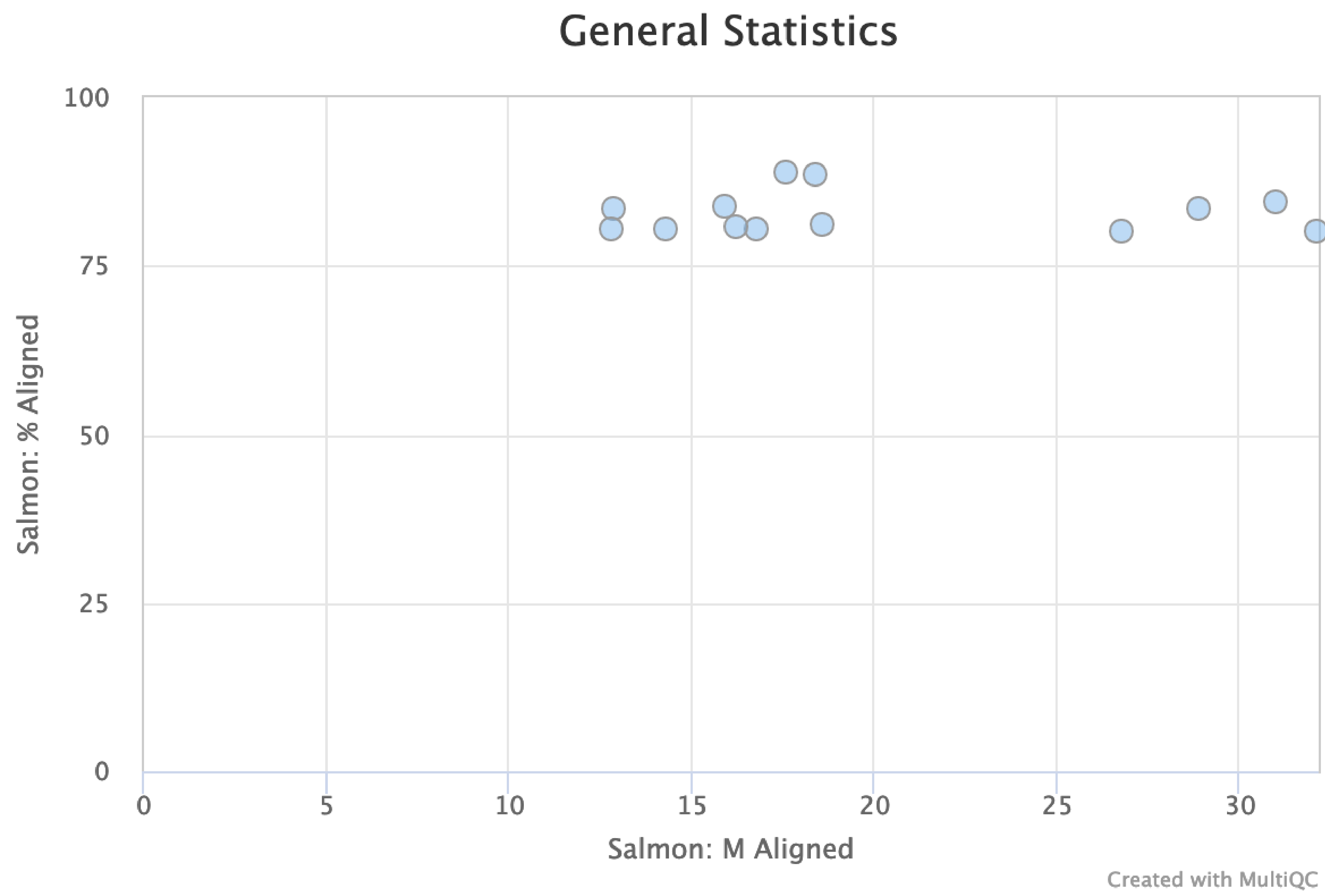
 = 758a89ad4f61 ✓

 = pq98wy4hvps ✗

A.



B.





UNIVERSITY OF CALIFORNIA, DAVIS

BERKELEY • DAVIS • IRVINE • LOS ANGELES • RIVERSIDE • SAN DIEGO • SAN FRANCISCO



SANTA BARBARA • SANTA CRUZ

DEPARTMENT OF POPULATION HEALTH
AND REPRODUCTION
SCHOOL OF VETERINARY MEDICINE
(530) 752-1358
FAX (530) 752-4278

ONE SHIELDS AVENUE
DAVIS, CALIFORNIA 95616-8743

Dear Editors,

Enclosed please find our manuscript entitled "**Streamlining Data-Intensive Biology with Workflow Systems**" by Taylor Reiter, Phillip T. Brooks, Luiz Irber, Shannon E.K. Joslin, Charles M. Reid, Camille Scott, C. Titus Brown, and N. Tessa Pierce, for consideration for publication in GigaScience. Our pre-print is on biorxiv (<https://doi.org/10.1101/2020.06.30.178673>), but this article is not submitted elsewhere.

This manuscript provides a directed set of project, data, and resource management strategies for adopting computational workflow systems to facilitate and expedite reproducible research in biology.

The biological research community has expressed a strong commitment to the idea that all life sciences research - including data and analysis workflows - should be Findable, Accessible, Interoperable, and Reusable (FAIR). These ideals are readily achievable for computational analyses, but implementing them in practice has proven difficult, particularly for biologists with limited computational training. Data-centric workflow systems offer a solution by internally managing computational resources, software, and conditional execution of analysis steps using a declarative specification of the workflow. These systems are reshaping the landscape of biological data analysis and empowering researchers to conduct reproducible analyses at scale. Online communities for sharing reusable workflow code have proliferated, mitigating the initial costs of learning and implementing workflow code, enhancing reproducibility, and leading to faster time-to-insight.

There is broad interest in workflows for biological analyses, recently profiled in Nature Methods ("When Computational Pipelines go 'clank' ") and Nature Toolbox ("Workflow systems turn raw data into scientific knowledge"; see citations at end of letter), but few practical resources exist. In this manuscript, we build upon strong "best" and "good-enough" practice recommendations for biological computing (articles below) to **provide what we believe to be the first detailed guidance for employing actual workflow systems in data-intensive biology research.**

The following three articles are most similar to ours, each providing specific guidance for biological computing. None address workflow systems.

Wilson G, Bryan J, Cranston K, Kitzes J, Nederbragt L, Teal TK (2017) Good enough practices in scientific computing. PLoS Comput Biol 13(6): e1005510.
<https://doi.org/10.1371/journal.pcbi.1005510>

Shade A, Teal TK (2015) Computing Workflows for Biologists: A Roadmap. PLoS Biol 13(11): e1002303. <https://doi.org/10.1371/journal.pbio.1002303>

Wilson G, Aruliah DA, Brown CT, Chue Hong NP, Davis M, Guy RT, et al. (2014) Best Practices for Scientific Computing. PLoS Biol 12(1): e1001745. <https://doi.org/10.1371/journal.pbio.1001745>

Data-centric workflow systems are expanding our capacity to conduct end-to-end FAIR analyses in computational biology, maximizing the value of both analysis code and generated data. The strategies presented here are designed to accelerate structured adoption of these systems for the benefit of individual researchers and the research community as a whole.

Thank you for your consideration.

Sincerely,



N. Tessa Pierce (corresponding author)
NSF Postdoctoral Fellow
Department of Population Health and Reproduction
University of California, Davis

Additional citations:

Marx, Vivien. "When computational pipelines go 'clank'." Nature Methods (2020): 1-4. <https://doi.org/10.1038/s41592-020-0886-9>

Perkel, Jeffrey M. "Workflow systems turn raw data into scientific knowledge." Nature 573.7772 (2019): 149-150. <http://dx.doi.org/10.1038/d41586-019-02619-z>

Suggested Reviewers:

Neil Chue Hong
Software Sustainability Institute
n.chuehong@epcc.ed.ac.uk

Karen Cranston, PhD
Agriculture and Agri-Food Canada
karen.cranston@gmail.com

Kate Hertweck, PhD
Fred Hutchinson Cancer Research Center
k8hertweck@gmail.com

Ethan White, PhD
University of Florida
ethanwhite@ufl.edu

Greg Wilson, PhD
RStudio
greg.wilson@rstudio.com