

**A peer-reviewed version of this preprint was published in PeerJ on 31 October 2018.**

[View the peer-reviewed version](https://peerj.com/articles/5853) (peerj.com/articles/5853), which is the preferred citable publication unless you specifically need to cite this preprint.

Li J, Cui B, Dai Y, Bai L, Huang J. 2018. BioInstaller: a comprehensive R package to construct interactive and reproducible biological data analysis applications based on the R platform. PeerJ 6:e5853 <https://doi.org/10.7717/peerj.5853>

# BioInstaller: a comprehensive R package to construct interactive and reproducible biological data analysis applications based on the R platform

Jianfeng Li<sup>1</sup>, Bowen Cui<sup>1</sup>, Yuting Dai<sup>1,2</sup>, Ling Bai<sup>1</sup>, Jinyan Huang<sup>Corresp. 1</sup>

<sup>1</sup> State Key Laboratory of Medical Genomics, Shanghai Institute of Hematology, National Research Center for Translational Medicine, Rui-Jin Hospital, Shanghai Jiao Tong University School of Medicine, Shanghai Jiao Tong University, Shanghai, China

<sup>2</sup> School of Life Sciences and Biotechnology, Shanghai Jiao Tong University, Shanghai, China

Corresponding Author: Jinyan Huang  
Email address: jinyan@shsmu.edu.cn

The increase in bioinformatics resources such as tools/scripts and databases poses a great challenge for users seeking to construct interactive and reproducible biological data analysis applications. Here, we propose an open-source, comprehensive, flexible R package named BioInstaller that consists of the R functions, Shiny application, the HTTP representational state transfer (REST) application programming interfaces (APIs), and a docker image. BioInstaller can be used to collect, manage and share various types of bioinformatics resources and perform interactive and reproducible data analyses based on the extendible Shiny application with Tom's Obvious, Minimal Language (TOML) and SQLite format databases. The source code of BioInstaller is freely available at our lab website, <http://bioinfo.rjh.com.cn/labs/jhuang/tools/bioinstaller>, the popular package host GitHub, <https://github.com/JhuangLab/BioInstaller>, and the Comprehensive R Archive Network (CRAN), <https://CRAN.R-project.org/package=BioInstaller>. In addition, a docker image can be downloaded from DockerHub (<https://hub.docker.com/r/bioinstaller/bioinstaller>).

1 **BioInstaller: a comprehensive R package to construct interactive and reproducible biological**  
2 **data analysis applications based on the R platform**

3 Jianfeng Li<sup>1,2,\*</sup>, Bowen Cui<sup>1,\*</sup>, Yuting Dai<sup>2</sup>, Ling Bai<sup>1</sup>, and Jinyan Huang<sup>1</sup>

4 <sup>1</sup> State Key Laboratory of Medical Genomics, Shanghai Institute of Hematology, National  
5 Research Center for Translational Medicine, Rui-Jin Hospital, Shanghai Jiao Tong University  
6 School of Medicine, Shanghai Jiao Tong University, Shanghai, China

7 <sup>2</sup> School of Life Sciences and Biotechnology, Shanghai Jiao Tong University, Shanghai, China

8 \* These authors contributed equally to this work.

9 Corresponding author: Jinyan Huang, [jinyan@shsmu.edu.cn](mailto:jinyan@shsmu.edu.cn)

10

11 **ABSTRACT**

12 The increase in bioinformatics resources such as tools/scripts and databases poses a great challenge  
13 for users seeking to construct interactive and reproducible biological data analysis applications.  
14 Here, we propose an open-source, comprehensive, flexible R package named BioInstaller that  
15 consists of the R functions, Shiny application, the HTTP representational state transfer (REST)  
16 application programming interfaces (APIs), and a docker image. BioInstaller can be used to  
17 collect, manage and share various types of bioinformatics resources and perform interactive and  
18 reproducible data analyses based on the extendible Shiny application with Tom's Obvious,  
19 Minimal Language (TOML) and SQLite format databases. The source code of BioInstaller is  
20 freely available at our lab website, <http://bioinfo.rjh.com.cn/labs/jhuang/tools/bioinstaller>, the  
21 popular package host GitHub, <https://github.com/JhuangLab/BioInstaller>, and the Comprehensive  
22 R Archive Network (CRAN), <https://CRAN.R-project.org/package=BioInstaller>. In addition, a  
23 docker image can be downloaded from DockerHub  
24 (<https://hub.docker.com/r/bioinstaller/bioinstaller>).

25

26 **INTRODUCTION**

27 With the rapid development of new bioscience technology, particularly next-generation  
28 sequencing (NGS), volumes of “omics” data have been generated, such as the 1000 Genomes  
29 Project, The Cancer Genome Atlas (TCGA), and Genotype-Tissue Expression (GTEx) (Abecasis  
30 et al. 2012; Cancer Genome Atlas Research et al. 2013; Consortium 2013; Sanchez-Vega et al.

31 2018). The bioinformatics tools and databases required for the downstream data analysis are also  
32 increasing at a phenomenal rate. R language, as the most popular programming language for  
33 statistics, biological data analysis, and big data, has enabled diverse and free R packages (>14000)  
34 for different types of applications, such as high throughput sequencing data analysis (e.g.  
35 Bioconductor) (Gentleman et al. 2004) and the development of web applications (e.g. Shiny  
36 framework) (Chang et al. 2015). With the development of web technologies and the release of the  
37 R web developmental framework Shiny, the number of interfaces available to R users has  
38 increased. However, due to the lack of high-performance and open-source cloud platforms based  
39 on R (e.g., Galaxy for Python users) (Afgan et al. 2016), it is still difficult for R users, especially  
40 those without web development skills, to construct interactive and reproducible biological data  
41 analysis applications supporting the upload and management of files, long-time computation, task  
42 submission, tracking of output files, exception handling, logging, export of plots and tables, and  
43 extendible plugin systems.

44 Another common problem usually faced by R and other programming platform users (e.g., the  
45 team of Galaxy) (Afgan et al. 2016) is how to acquire and share certain bioinformatics resources  
46 quickly and accurately. Numerous bioinformatics tools (e.g., primer design, sequence alignment,  
47 variant calling and annotation) or scripts (e.g., data format conversion, text processing) are  
48 scattered around world web hosts. Biomedical databases are facing the same situation. For  
49 example, genome sequences (e.g., hg19/hg38 for human, mm9 and mm10 for mouse) are mainly  
50 deposited in the UCSC Genome Browser and National Center for Biotechnology Information  
51 (NCBI, <https://www.ncbi.nlm.nih.gov/>) (Tyner et al. 2017). The best-known gene and transcript  
52 annotation resources are provided by GENCODE and the RefSeq database (Derrien et al. 2012;  
53 O'Leary et al. 2016). Genetic variants annotation databases, mainly cancer and Mendelian disorder  
54 related, are hosted by the original projects, e.g. TCGA, and various down-stream tools, e.g.,  
55 ANNOVAR, Variant Effect Predictor (VEP), Oncotator (McLaren et al. 2016; Ramos et al. 2015;  
56 Wang et al. 2010). Bioconductor is a popular bioinformatics R community for sharing genetic  
57 variants and other types of bioinformatics annotation databases via R package (Gentleman et al.  
58 2004), but it is difficult for users to share many types of tools/scripts and databases if they do not  
59 have the capability of packing their own tools/scripts and databases. In most cases, these resources  
60 are isolated and can only be accessed via a command line tool such as rsync  
61 (<https://rsync.samba.org/>) or wget (<http://www.gnu.org/software/wget/>), to request the

62 corresponding Uniform Resource Locators (URLs). Software distribution tools that do not demand  
63 root privileges, such as conda (<https://github.com/conda/conda>) and spack (Gamblin et al. 2015),  
64 have greatly improved the acquisition of bioinformatics software. However, considering the huge  
65 growth of tools/scripts and databases required for bioinformatics data analysis, the resources  
66 supported by these software distribution tools are far from sufficient. Users also need more  
67 experience to use these different package management tools under command line environment.  
68 Here, we present an open-source, comprehensive, flexible bioinformatics platform named  
69 BioInstaller that can be used to collect, manage and share various types of bioinformatics resources  
70 and to perform interactive and reproducible data analyses. By utilizing a simplified and standard  
71 TOML format configuration file with extra parse functions, the developers and users can freely  
72 and unreservedly share their public or internal bioinformatics tools/scripts and databases online on  
73 the GitHub repository or other hosts. In addition, users can easily obtain access to pooled  
74 bioinformatics resources via the diverse interfaces of BioInstaller, which includes R functions, the  
75 Shiny application (Chang et al. 2015) and HTTP representational state transfer (REST) application  
76 programming interfaces (APIs) that are rarely adopted in other similar tools. As a practical  
77 demonstration, we collected 157 tools/scripts and 110 databases specifically related to genetic  
78 variants annotation using the BioInstaller-defined configuration files. Notably, we developed a  
79 Shiny application to support functions including system monitoring, the logging system, file  
80 management, the queue system, and so on. This application can easily be reused in other Shiny  
81 applications. We expect the BioInstaller package and the practices in this work to reduce the  
82 difficulty of constructing the interactive and reproducible biological data analysis applications for  
83 R users, and to further improve the interactivity and reproducibility of bioinformatics data analysis.

## 84 MATERIAL AND METHODS

### 85 Design and development of BioInstaller

86 BioInstaller was designed as an interactive R package to collect, manage and share various types  
87 of bioinformatics resources and perform interactive and reproducible data analyses. BioInstaller  
88 contains the R functions and the Shiny application (Chang et al. 2015) and REST APIs (**Figure**  
89 **1**). Both R and other programming platform users can utilize the functions of BioInstaller, such as  
90 by downloading bioinformatics tools/scripts and databases and performing statistical analysis and  
91 visualization. The R and Shiny interfaces of BioInstaller were mainly developed in R language

92 and utilize the HTML/CSS and JavaScript languages. To run an instance of BioInstaller, the R  
93 program and extra dependent R packages are required. Travis CI (<https://www.travis-ci.org/>) was  
94 used to automatically test the R functions on Linux and MAC OSX platforms. Periodically, the  
95 tested and updated BioInstaller package is submitted to CRAN with an increased version number,  
96 e.g., from v3.3.3 to v3.3.4. Both the open and restricted bioinformatics resources can be integrated  
97 using the TOML format configuration file. The configuration files can also be used in other  
98 programming language platforms to access desired masteries by using a unique item name, such  
99 as ‘bwa’, ‘gatk’, ‘annovar’, ‘db\_annovar\_1000g’, ‘db\_annovar\_gtex’, etc. A hash value was  
100 generated using the item name and version for the unique ids of tools/scripts and databases. An  
101 autogenerated docker image containing all required R packages and the backend web service of  
102 BioInstaller have been deposited at the DockerHub  
103 (<https://hub.docker.com/r/bioinstaller/bioinstaller>).

#### 104 **GitHub API and custom values/functions for querying of version**

105 The querying of versions of bioinformatics tools/scripts and databases of a GitHub or non-GitHub  
106 project is the basic function of BioInstaller. For GitHub projects items, the GitHub APIs were used  
107 to access the projects version information, such as release, tags, and branches. All released versions  
108 will be used as the available versions and returned to BioInstaller (**Figure S1a**). However, the  
109 situation becomes more complicated if the resources have not been published on GitHub. Here,  
110 we propose two types of methods of parsing item versions. Method I: If the released versions are  
111 fixed, users can write it in the ‘version\_available’ field in the configuration file. Method II:  
112 Utilizing the R packages rvest (<https://CRAN.R-project.org/package=rvest>) and RCurl  
113 (<https://CRAN.R-project.org/package=RCurl>) (Wu et al. 2016), we established an R functions  
114 pool to dynamically query the version of items from the original release website (**Dataset S1**). The  
115 demo function to query the latest version of GMAP is shown in **Figure S1b**. This is useful for  
116 automating a pipeline to build the precompiled binary version.

#### 117 **Mirror resource for an invalid link**

118 Network transferring is a common problem in bioinformatics data analysis. A mirror resource is  
119 one option to partially resolve these problems, including an invalid link and network blocking.  
120 BioInstaller allows users to set any numbers of mirror URLs for their tools/scripts and databases

121 to avoid the possible problems caused by network transmission. As shown in **Figure S1c**, the  
122 mirror URLs of Miniconda (<https://conda.io/miniconda.html>) are separately provided by the  
123 official and our hosts. Notably, established mirror URLs of bioinformatics resources can be used  
124 in the spack (Gamblin et al. 2015) and other similar tools to build the cache files.

### 125 **TOML format configuration files**

126 Massive bioinformatics tools/scripts and databases have been integrated into BioInstaller. Tom's  
127 Obvious, Minimal Language (TOML) is a popular and human-readable configuration formats  
128 supporting comments. We uses standard TOML format configuration file to store required  
129 information of the included bioinformatics tools/scripts and databases. These configuration files  
130 can be reused in other bioinformatics software packages or data-analysis pipelines via online  
131 accession or as a file copy. We have provided six directories to store different types of TOML files  
132 including 'github', 'nongithub', 'database', 'web', 'docker', and 'shiny'. Due to the broad  
133 compatibility of BioInstaller, any resource published on docker, GitHub, Zenodo  
134 (<https://zenodo.org/>) or other platforms can be supported.

### 135 **Implementation of the Shiny application**

136 To increase the convenience of BioInstaller for nonprogramming users, a user-friendly web  
137 application was developed based on Shiny (Chang et al. 2015). The user interface (UI) of  
138 BioInstaller was constructed using the R package shinydashboard ([https://cran.r-](https://cran.r-project.org/package=shinydashboard)  
139 [project.org/package=shinydashboard](https://cran.r-project.org/package=shinydashboard)) and Shiny (Chang et al. 2015). Output tables were generated  
140 by the R package DT (<https://CRAN.R-project.org/package=DT>) and wrapped JavaScript library  
141 DataTables (<https://datatables.net/>). Charts were mainly generated by published R packages and  
142 in-house scripts or R packages that all support interactive update and export of PDF, SVG, and  
143 PNG format plots. The tab items of the BioInstaller Shiny application at the left side of the  
144 navigation bar can be used to switch among all available modules, including 'Introduction',  
145 'Dashboard', 'Upload', 'File Viewer', 'Pipeline', 'Instant', 'Installer', and 'Setting'. The detail  
146 usage guidelines are provided on our host  
147 (<http://bioinfo.rjh.com.cn/labs/jhuang/tools/BioInstaller/>), and R users can also use the browser  
148 vignettes functions in R to access these documents.

## 149 **RESULTS**

## 150 **Overview and practices of BioInstaller's functionalities**

151 A comprehensive R package was developed that could be used to quickly construct interactive and  
152 reproducible biological data analysis applications based on the R platform (**Figure 2**). The  
153 functionalities (**Table 1, Dataset S2**) of BioInstaller were divided into six parts based on whether  
154 users use BioInstaller or not: 1) deployment of resources, 2) collection of resources, 3) sharing of  
155 resources, 4) construction of pipelines, 5) construction of Shiny applications, and 6) reproducible  
156 data analysis. An example of a real project (annovarR, <https://github.com/JhuangLab/annovarR>,  
157 under development) is shown in **Figure 2** to illustrate the full workflow for BioInstaller utilization,  
158 which was designed to integrate various genetic variant annotation and visualization tools,  
159 including public command line tools, R packages and custom annotation and visualization  
160 functions. Using the code library, predefined TOML files (database resources and plugins), and  
161 the docker file of BioInstaller, we could easily customize the BioInstaller-established Shiny  
162 application to work on the genetic variants annotation tasks. If BioInstaller is not used, we need to  
163 develop the UI and server code of the Shiny application for a large number of universal functions,  
164 such as the file management system, background task submission and queue management, and  
165 tracking of the output log and files. The docker image of BioInstaller is also out-of-the-box and  
166 could be modified and applied to our own works. Based on the integrated installer (e.g., conda,  
167 spack, and BioInstaller) and simplified TOML files of BioInstaller, users can collect, share, and  
168 deploy genetic variant annotation databases and tools with one-stop service. As a real practice of  
169 BioInstaller, we collected and shared more than 157 tools/scripts and 110 databases (**Table 2, S1,**  
170 **S2**) in the configuration pool of BioInstaller, including genetic variant annotation databases and  
171 tools; the meta information is freely available and hosted on the public GitHub website  
172 (<https://github.com/JhuangLab/BioInstaller/tree/master/inst/extdata/config>). The raw files are  
173 stored on the original websites (e.g., <https://github.com>, <https://sourceforge.net/>,  
174 <http://annovar.openbioinformatics.org/>, etc.) and our host.

## 175 **Comparison of BioInstaller with existing tools for the collection and sharing of** 176 **bioinformatics resources**

177 To better understand the advance provided by BioInstaller in terms of the collection and sharing  
178 of bioinformatics resources, we further compared BioInstaller with several existing tools,  
179 including Omictools (Henry et al. 2014) and Datasets2Tools (Torre et al. 2018) (**Table 1, 2**), the



180 two most comprehensive meta databases focused on bioinformatics tools. All provide a web forum  
181 to update the meta database of bioinformatics resources. However, BioInstaller offers an off-line  
182 way to develop the users' own meta databases via an unlimited configuration file pool (TOML and  
183 SQLite format) that is easy to carry and share and is independent of programming knowledge. In  
184 addition, the developed R functions and Shiny application can be used to query and download the  
185 linked or isolated file databases, such as appendix data from papers, annotation databases for  
186 genetic variants, genome sequences, etc. In most cases, it is suitable to tightly combine the meta  
187 database with the file database. Therefore, we designed and shared an upload module in the Shiny  
188 application to set the meta information for all files, and users can add the description, genome  
189 version, custom file types, and other customizable fields. Both Omictools and Dataset2Tools only  
190 include the items in their databases and do not integrate external resources. BioInstaller not only  
191 can be used to collect users own resources, but also can be used to integrate external resources.

## 192 **Summary of supported bioinformatics tools/scripts and databases**

193 For now, 157 tools/scripts and 110 databases are natively supported in BioInstaller (**Figure 1**,  
194 **Table 2, S1, S2**). First, we covered the most commonly used tools in each bioinformatics analysis  
195 process, including data quality control (n=17), alignment and assembly (n=27), variant detection  
196 (n=32) or annotation (n=12), high-throughput sequence (HTS) manipulation (n=17) and  
197 visualization libraries (n=11) (**Table 1, S1**), etc. Second, BioInstaller also provides abundant  
198 databases for annotating data or satisfying software dependencies. With BioInstaller, users can  
199 easily download UCSC sequence and annotation data (n=4995) (**Dataset S3**), blast databases  
200 (n=29) (**Table S2**), allele frequency databases (n=17), variant effect prediction databases (n=29),  
201 and disease-related (n=13), drug-related (n=4), noncoding region-related databases (n=15) (**Table**  
202 **2, S2**), among others. Notably, we collected and constructed 20 genetic variant annotation  
203 databases, which can be directly used in other variants annotation tools, including ANNOVAR  
204 (Wang et al. 2010), vcfanno (Pedersen et al. 2016), and annovarR  
205 (<https://github.com/JhuangLab/annovarR>).

206 BioInstaller has been released on CRAN for one and a half years and has accumulated a certain  
207 number of users, with a total of 19,912 downloads from CRAN (2018.8.3). In the recent release  
208 (v0.3.5), we provided the Shiny application and significantly expanded the supported tools/scripts  
209 and databases. The number of supported tools/scripts and databases is still increasing and is being

210 applied to other related projects, such as the integrated genetic variants annotation tool annovarR  
211 (<https://github.com/JhuangLab/annovarR>).

## 212 **Examples of BioInstaller R functions**

213 We have demonstrated the basic structure, functions, and web service of BioInstaller. The full help  
214 document is available at <http://bioinfo.rjh.com.cn/labs/jhuang/tools/BioInstaller/articles/>. Because  
215 most of the Shiny application UIs are wrapped with R functions, we use several use examples to  
216 illustrate the R functions of BioInstaller.

217 **Example #1:** Install packed or unpacked bioinformatics tools. We use the Ion Torrent Variant  
218 Caller (TVC) (Zook et al. 2014) and svaba (Wala et al. 2018) to show how to install or download  
219 the bioinformatics tools or scripts that are not supported by other package management tools.

```
220 > library(BioInstaller) # Library the R package
221 > set.biosoftwares.db("~/BioInstaller/info.yaml") # Store the installation information
222 > install.bioinfo(show.all.names = TRUE) # Get all items name supported by BioInstaller
223 > install.bioinfo(name = "tvc", show.all.versions = TRUE) # Get all available versions of tvc
224 > install.bioinfo(name = "svaba", show.all.versions = TRUE) # Get all available versions of
225 svaba
226 > install.bioinfo(name="tvc", download.dir = "/path/tool/tvc") # One-click install the tvc
227 > install.bioinfo(name="svaba", download.dir = "/path/tool/svaba") # One-click install the svaba
228 > show.installed() # Get all installed tools
229 > get.info("svaba") # Get the svaba installation information, such as update time and version
```

230

231 **Example #2:** Download genetic variants annotation databases. Genetic variants annotation is a  
232 common and high-demand task for most biomedicine research, especially for examining the  
233 correlations between phenotype and molecular features, such as germline and somatic mutations.  
234 The followed example describes how to download the genetic variants annotation databases  
235 dbSNP, CIViC, DisGeNET, and CancerHotspot (Chang et al. 2016; Griffith et al. 2017; Piñero et  
236 al. 2017).

```
237 > install.bioinfo("db_annovar_avsnp", extra.list = list(buildver = "hg19"), download.dir =
238 "/path/db/") # install the latest dbSNP from ANNOVAR website
```

```
239 > crawl.all.versions("db_annovar_avsnp") # Download all dbSNP to current directory
240 > install.bioinfo("db_civic", download.dir = "/tmp/db") # Download the nightly version of CIViC
241 database
242 > install.bioinfo("db_disgenet", download.dir = "/tmp/db") # Download the DisGeNET database
243 > install.bioinfo("db_cancer_hotspots", download.dir = "/tmp/db") # Download the DisGeNET
244 databaseß
245
246 Example #3: Download an annotation database based on the supplementary files of published
247 papers. The followed example is an epigenetic genes classification (e.g., reader, writer, eraser)
248 database only available in the papers supplementary file (Huether et al. 2014).
249 > install.bioinfo("db_annovar_epi_genes", extra.list = list(buildver = "hg19"), download.dir =
250 "/path/db/") # install the epigenetic genes database from our website
```

## 251 **User-interfaces and functions of the Shiny application**

252 **Introduction module.** Utilizing the Shiny function ‘includeMarkdown’, we generated the  
253 ‘Introduction’ module page from Markdown, a lightweight markup language, format document  
254 (**Figure S2a**).

255 **Dashboard module.** The ‘Dashboard’ module includes the system monitors, such as hardware  
256 (Disk and memory), queue tasks, task log, installed R packages, Python packages, conda  
257 environments, and the other information of the operating environments (**Figure S2b, c, S3**). The  
258 monitored data stream is automatically updated once every 10 seconds (**Figure S2b**). A demo table  
259 output in the dashboard lists all files in the environment variable ‘PATH’, where users can use the  
260 selector at the lower left quarter to customize the row numbers (5, 10, 25, 50, and all) (**Figure**  
261 **S2c**). All output tables in BioInstaller can be easily exported to CSV, XLS, PDF files or directly  
262 copied to the clipboard. Monitor plugins related to the information of the R system (**Figure S3a,**  
263 **b, c**), BioInstaller (**Figure S3d**), conda (**Figure S3e**) and spack (**Figure S3f**) are integrated in this  
264 work, which can reduce user input of extra command line commands and facilitate sharing with  
265 others.

266 **Upload module.** The ‘Upload’ module is used to upload files to the BioInstaller Shiny web  
267 platform. Optional fields, such as file type, genome version, and description, can be stored in the  
268 SQLite format database with the uploaded files path and the files md5 value (**Figure S2d**). When

269 uploading a file, users need to click the ‘Save’ button to confirm the upload behavior and update  
270 the database (**Figure S2e**). Before the confirmation click, users can preview the file and make a  
271 final decision (**Figure S2f**). Files with sizes ranging from 0.25 GB to 8 GB were tested on the  
272 Shiny application (**Table S3**). For files larger than 10 GB, we recommend using the rsync or FTP  
273 service to transfer files and then adding the corresponding description and records in TOML or  
274 SQLite databases.

275 **File viewer module.** The ‘File viewer’ module is used to manage all uploaded files in the  
276 BioInstaller Shiny application that supports view, delete and download, and all files can be used  
277 in the other plugins of the BioInstaller Shiny application, mainly in the ‘Pipeline’ and ‘Instant’  
278 modules (**Figure 3a, b**).

279 **Pipeline module.** The ‘Pipeline’ module is used to integrate complicated bioinformatics analysis  
280 workflows or other small scripts. An in-house interpreter R function was used to parse the plugin  
281 configuration files to generate the Shiny UI and server functions. A small script creating a data  
282 analysis directory structure was used as the demo for ‘Pipeline’ (**Dataset S4**). Users can input the  
283 project name and the parent directory to create a series of predefined directories. The R commands  
284 used in this task are editable at the bottom of the box (**Figure 3c**). After users click the ‘Submit’  
285 button, BioInstaller will generate a random character as the submitted task key. Users can use this  
286 key to retrieve the output information, such as files and logs, in the ‘Dashboard’ module (**Figure**  
287 **3c, d, e, f**). All submitted tasks enter the task queue supported by the SQLite database using the R  
288 package litseq (<https://CRAN.R-project.org/package=litseq>). Tasks in the queue are automatically  
289 checked by the activated workers (**Figure 3e, f**).

290 **Instant module.** The ‘Instant’ module is used to run the real-time plots and data analysis, and  
291 similar to the ‘Pipeline’ module, the UI and server were automatically generated via plugin  
292 configuration files (**Dataset S5**). We used the meta database query of BioInstaller, Datasets2tools  
293 (Torre et al. 2018), PubMed, and plots of Maftools (Mayakonda & Koeffler 2016), a cancer  
294 somatic mutations visualization tool, as the demo to demonstrate the function. Users can select the  
295 input files defined in the plugins configuration file (TOML) or user-uploaded files. The commands  
296 are stored in the bottom of the boxes and can be modified by the user. After clicking the ‘Run’  
297 button, all output box codes, such as output plots and tables, run on the server side and are returned  
298 in real time to the Shiny UI (**Figure 4**). We developed several plugins to query and access several  
299 meta databases related to bioinformatics, such as the BioInstaller meta databases (**Figure 4a, b**),

300 Datasets2tools (Torre et al. 2018) (**Figure 4c, d**), and PubMed (**Figure 4e, f**). The powerful  
301 visualization functions of R packages are also supported in the ‘Instant’ module. As shown in  
302 **Figure 4g, h**, users can obtain the demo output (PDF and PNG format) of Maftools. After running  
303 all box codes, a single box can be separately updated and exported by users.

304 **Installer module.** The ‘Installer’ module is the main Shiny interface of BioInstaller for  
305 downloading and installing bioinformatics tools/scripts and databases. We provide the Shiny  
306 interfaces of BioInstaller, conda and spack (**Figure S4**). The ‘Installer’ module is similar to the  
307 ‘Pipeline’ module, which is also needed to submit a task to the queue. The status and log  
308 information can be retrieved in the ‘Dashboard’ module. Three basic use cases of the BioInstaller  
309 Shiny application are available: 1) download db\_annoar\_refgene database (**Figure S4a**); 2) create  
310 conda environment (**Figure S4b**); 3) install ‘zlib’ using spack (**Figure S4c**).

311 **Setting module.** The setting module is the interface for setting the value of the variable used in  
312 the BioInstaller plugins or R files. Both a Shiny UI and a YAML editor are offered for users  
313 (**Figure S5a, b**). Any updates of the YAML editor (**Figure S5b**) can change and refresh the Shiny  
314 UI options (**Figure S5a**). It is helpful for users to manage various material related to BioInstaller  
315 and its plugins.

316 In most cases, through the one-click interface of BioInstaller, users can easily download and install  
317 the desired bioinformatics resources without any command line skills. Functions for automatic  
318 compiling from the source file with the dependent software or database are also supported in  
319 BioInstaller. However, for complicated software with high system dependence, we recommend  
320 using the interfaces of conda (<https://conda.io/docs/>) and spack (Gamblin et al. 2015).

### 321 **Portable message queue for background tasks based on SQLite**

322 Tasks with long-time costs are challenging in Shiny, which always blocks the other interactive  
323 operations simultaneously when the previous task has not been finished. Here, we utilized the R  
324 package litseq (<https://CRAN.R-project.org/package=litseq>) to submit and manage the background  
325 queue tasks. litseq is portable and lightweight. litseq does not require extra software or service  
326 from other programming platforms and can work on any clusters server running computing-  
327 intensive tasks. The developed queue worker in BioInstaller can be used for all other background  
328 tasks submitted by litseq. All litseq-submitted tasks of BioInstaller are assigned a unique  
329 identification id. All executed commands, output logs, and others are saved in the permanent files.

### 330 **Opencpu backend service improves reproducibility**

331 Opencpu (Ooms 2017) is an R package for reproducible research that can expose a web REST API  
332 interface with R, Latex and, Pandoc. The R functions of BioInstaller are invoked by the activated  
333 Opencpu R process or daemon service. For other programming platform users, this is one possible  
334 method for utilizing the R functions of BioInstaller (**Figure 1**). The output of JSON and text  
335 formats are returned when using the browser access (**Figure 5a**) or simulated requests. Three of  
336 the most basic APIs usages of BioInstaller were used to demonstrate how it works: 1) obtaining  
337 all supported tools/scripts and databases; 2) acquiring available versions of the appointed item; 3)  
338 installing a tool in a directory (**Figure 5b**). Notably, a random string, such as “x0a469794fa”, will  
339 be generated as the key of Opencpu to obtain the output of one R session. Both JSON and text  
340 format output can be returned by Opencpu backend APIs (**Figure 5c**).

### 341 **Docker container of BioInstaller**

342 A prebuilt docker image is available on the docker hub  
343 (<https://hub.docker.com/r/bioinstaller/bioinstaller>), and the latest code change of the BioInstaller  
344 repository can automatically trigger an update of the docker image. In the docker image, we  
345 integrated and configured three types of web services, including Opencpu, Shiny (Chang et al.  
346 2015; Ooms 2017), and the RStudio server (<https://www.rstudio.com/products/rstudio-server/>).  
347 The followed commands can be used to deploy and start the service of BioInstaller service.

```
348 $ docker pull bioinstaller/bioinstaller  
349 $ docker run -it -p 80:80 -p 8004:8004 bioinstaller/bioinstaller
```

350 Users can deploy a new instance host of BioInstaller and all other web services in a few minutes,  
351 and other tools/scripts and databases are also allowed to be embedded in this docker image using  
352 the Dockerfile (<https://github.com/JhuangLab/BioInstaller/blob/master/Dockerfile>).

### 353 **Use the GitHub forum to share, rate, and discuss the bioinformatics resources**

354 The full-text search is natively supported by the GitHub website with highlight and age forwarding  
355 functions (**Figure S6a, b**). To simplify the submitting of new items to BioInstaller, the GitHub  
356 repository issues page (<https://github.com/JhuangLab/BioInstaller/issues>) is recommended for  
357 other users to share, rate, and discuss bioinformatics tools/scripts and databases with a designated

358 label (**Figure S6c**). The ‘watching’ function of GitHub can allow users to receive notifications of  
359 all conversations on the BioInstaller. Another advantage of establishing a free sharing community  
360 based on the GitHub is that all history changes on the code and forum posts can be recorded and  
361 retrieved. A rating function for bioinformatics tools/scripts or databases is also feasible by  
362 calculating the points corresponding to thumbs up or down.

## 363 **DISCUSSION**

364 Bioinformatics tools/scripts and databases are widely used in various data analysis projects. The  
365 construction of interactive and reproducible biological data analysis applications is critical for  
366 most bioinformatics data analyses (Henry et al. 2014; McQuilton et al. 2016; Ohno-Machado et  
367 al. 2017). The integrative utilization of these resources is becoming increasingly important for  
368 improving integrated biosciences data analysis. R language, as the most popular programming  
369 language for statistics, biological data analysis, and big data, has provided massive useful R  
370 packages for various data analysis efforts, especially the NGS field. However, there has been no  
371 comprehensive and free R application that can support file upload and management, perform long-  
372 time computation with a tasks submission system, track and record the output of files and log,  
373 develop extendible plugins, add or remove functions of the application in real time, and respond  
374 for REST APIs. Another common problem for users of R and other programming platforms for  
375 biological data analysis is that massive bioinformatics resources are isolated and scattered, which  
376 significantly increases the difficulty of deploying, collecting and sharing these resources. Well-  
377 known software distribution tools that do not need root privileges, such as conda  
378 (<https://conda.io/docs>) and spack (Gamblin et al. 2015), were designed for comprehensive fields  
379 and usually lack support for life science resources. Bioconda is a fine example of the centralized  
380 installation of bioinformatics software (approximately 1900 items) that has significantly improved  
381 the reproducibility of bioinformatics data analysis (Gruning et al. 2018). However, this is not  
382 sufficient compared with the rapid increase in software and databases in the life sciences field.  
383 As described in this study, we present a comprehensive, free and open-source platform,  
384 BioInstaller, to construct the interactive and reproducible biological data analysis applications.  
385 BioInstaller contains the R functions, the Shiny application, REST APIs and the docker image.  
386 This platform and the practices described in this work are sufficient for most R users to  
387 conveniently and quickly develop an interactive and reproducible biological data analysis

388 application with diverse predefined functions (e.g., file management, task submission, plugin  
389 management system, logging, etc.), plugins, and files offered by BioInstaller. Moreover, based on  
390 the TOML format files, we have also integrated hundreds of bioinformatics resources required for  
391 the wide field of bioinformatics, such as sequence alignment, variant calling and annotation, and  
392 so on. We hope this newly presented open source platform for R users can reduce the difficulty of  
393 constructing the interactive and reproducible biological data analysis applications and further  
394 improve the interactivity and reproducibility of bioinformatics data analysis.

## 395 CONCLUSION

396 As described in this work, we established a new platform to construct interactive and reproducible  
397 biological data analysis applications based on R language. This platform contains diverse user  
398 interfaces, including the R functions and R Shiny application, REST APIs, and support for  
399 collecting, managing, sharing, and utilizing massive bioinformatics tools/scripts and databases.

## 400 ACKNOWLEDGEMENT

401 We would like to thank the many BioInstaller users who provided feedback and suggestions. We  
402 thank the Centre for HPC at Shanghai Jiao Tong University for supporting computing platforms.

## 403 REFERENCES

- 404 Abecasis GR, Auton A, Brooks LD, DePristo MA, Durbin RM, Handsaker RE, Kang HM, Marth GT, and  
405 McVean GA. 2012. An integrated map of genetic variation from 1,092 human genomes. *Nature*  
406 491:56-65. 10.1038/nature11632
- 407 Afgan E, Baker D, van den Beek M, Blankenberg D, Bouvier D, Cech M, Chilton J, Clements D, Coraor N,  
408 Eberhard C, Gruning B, Guerler A, Hillman-Jackson J, Von Kuster G, Rasche E, Soranzo N, Turaga  
409 N, Taylor J, Nekrutenko A, and Goecks J. 2016. The Galaxy platform for accessible, reproducible  
410 and collaborative biomedical analyses: 2016 update. *Nucleic Acids Res* 44:W3-W10.  
411 10.1093/nar/gkw343
- 412 Cancer Genome Atlas Research N, Weinstein JN, Collisson EA, Mills GB, Shaw KR, Ozenberger BA, Ellrott  
413 K, Shmulevich I, Sander C, and Stuart JM. 2013. The Cancer Genome Atlas Pan-Cancer analysis  
414 project. *Nat Genet* 45:1113-1120. 10.1038/ng.2764
- 415 Chang MT, Asthana S, Gao SP, Lee BH, Chapman JS, Kandoth C, Gao J, Socci ND, Solit DB, Olshen AB,  
416 Schultz N, and Taylor BS. 2016. Identifying recurrent mutations in cancer reveals widespread  
417 lineage diversity and mutational specificity. *Nat Biotechnol* 34:155-163. 10.1038/nbt.3391
- 418 Chang W, Cheng J, Allaire J, Xie Y, and McPherson J. 2015. Shiny: web application framework for R. *R*  
419 *package version 011* 1:106.
- 420 Consortium GT. 2013. The Genotype-Tissue Expression (GTEx) project. *Nat Genet* 45:580-585.  
421 10.1038/ng.2653



- 422 Derrien T, Johnson R, Bussotti G, Tanzer A, Djebali S, Tilgner H, Guernec G, Martin D, Merkel A, Knowles  
423 DG, Lagarde J, Veeravalli L, Ruan X, Ruan Y, Lassmann T, Carninci P, Brown JB, Lipovich L,  
424 Gonzalez JM, Thomas M, Davis CA, Shiekhhattar R, Gingeras TR, Hubbard TJ, Notredame C,  
425 Harrow J, and Guigo R. 2012. The GENCODE v7 catalog of human long noncoding RNAs: analysis  
426 of their gene structure, evolution, and expression. *Genome Res* 22:1775-1789.  
427 10.1101/gr.132159.111
- 428 Gamblin T, LeGendre M, Collette MR, Lee GL, Moody A, de Supinski BR, and Futral S. 2015. The Spack  
429 package manager: bringing order to HPC software chaos. Proceedings of the International  
430 Conference for High Performance Computing, Networking, Storage and Analysis: ACM. p 40.
- 431 Gentleman RC, Carey VJ, Bates DM, Bolstad B, Dettling M, Dudoit S, Ellis B, Gautier L, Ge Y, Gentry J,  
432 Hornik K, Hothorn T, Huber W, Iacus S, Irizarry R, Leisch F, Li C, Maechler M, Rossini AJ, Sawitzki  
433 G, Smith C, Smyth G, Tierney L, Yang JY, and Zhang J. 2004. Bioconductor: open software  
434 development for computational biology and bioinformatics. *Genome Biol* 5:R80. 10.1186/gb-  
435 2004-5-10-r80
- 436 Griffith M, Spies NC, Krysiak K, McMichael JF, Coffman AC, Danos AM, Ainscough BJ, Ramirez CA, Rieke  
437 DT, Kujan L, Barnell EK, Wagner AH, Skidmore ZL, Wollam A, Liu CJ, Jones MR, Bilski RL, Lesurf R,  
438 Feng YY, Shah NM, Bonakdar M, Trani L, Matlock M, Ramu A, Campbell KM, Spies GC, Graubert  
439 AP, Gangavarapu K, Eldred JM, Larson DE, Walker JR, Good BM, Wu C, Su AI, Dienstmann R,  
440 Margolin AA, Tamborero D, Lopez-Bigas N, Jones SJ, Bose R, Spencer DH, Wartman LD, Wilson  
441 RK, Mardis ER, and Griffith OL. 2017. CIViC is a community knowledgebase for expert  
442 crowdsourcing the clinical interpretation of variants in cancer. *Nat Genet* 49:170-174.  
443 10.1038/ng.3774
- 444 Gruning B, Dale R, Sjodin A, Chapman BA, Rowe J, Tomkins-Tinch CH, Valieris R, Koster J, and Bioconda T.  
445 2018. Bioconda: sustainable and comprehensive software distribution for the life sciences. *Nat*  
446 *Methods* 15:475-476. 10.1038/s41592-018-0046-7
- 447 Henry VJ, Bandrowski AE, Pepin AS, Gonzalez BJ, and Desfeux A. 2014. OMICtools: an informative  
448 directory for multi-omic data analysis. *Database,2014,(2014-01-01)* 2014:2091-2092.
- 449 Huether R, Dong L, Chen X, Wu G, Parker M, Wei L, Ma J, Edmonson MN, Hedlund EK, Rusch MC,  
450 Shurtleff SA, Mulder HL, Boggs K, Vadordaria B, Cheng J, Yergeau D, Song G, Becksfort J,  
451 Lemmon G, Weber C, Cai Z, Dang J, Walsh M, Gedman AL, Faber Z, Easton J, Gruber T, Kriwacki  
452 RW, Partridge JF, Ding L, Wilson RK, Mardis ER, Mullighan CG, Gilbertson RJ, Baker SJ, Zambetti  
453 G, Ellison DW, Zhang J, and Downing JR. 2014. The landscape of somatic mutations in epigenetic  
454 regulators across 1,000 paediatric cancer genomes. *Nat Commun* 5:3630. 10.1038/ncomms4630
- 455 Mayakonda A, and Koeffler HP. 2016. Maftools: Efficient analysis, visualization and summarization of  
456 MAF files from large-scale cohort based cancer studies. Available at  
457 <https://www.biorxiv.org/content/biorxiv/early/2016/05/11/052662.full.pdf> (accessed Sep.3  
458 2018).
- 459 McLaren W, Gil L, Hunt SE, Riat HS, Ritchie GR, Thormann A, Flicek P, and Cunningham F. 2016. The  
460 Ensembl Variant Effect Predictor. *Genome Biol* 17:122. 10.1186/s13059-016-0974-4
- 461 McQuilton P, Gonzalez-Beltran A, Rocca-Serra P, Thurston M, Lister A, Maguire E, and Sansone SA. 2016.  
462 BioSharing: curated and crowd-sourced metadata standards, databases and data policies in the  
463 life sciences. *Database (Oxford)* 2016. 10.1093/database/baw075
- 464 O'Leary NA, Wright MW, Brister JR, Ciufo S, Haddad D, McVeigh R, Rajput B, Robbertse B, Smith-White  
465 B, Ako-Adjei D, Astashyn A, Badretdin A, Bao Y, Blinkova O, Brover V, Chetvernin V, Choi J, Cox E,  
466 Ermolaeva O, Farrell CM, Goldfarb T, Gupta T, Haft D, Hatcher E, Hlavina W, Joardar VS, Kodali  
467 VK, Li W, Maglott D, Masterson P, McGarvey KM, Murphy MR, O'Neill K, Pujar S, Rangwala SH,  
468 Rausch D, Riddick LD, Schoch C, Shkeda A, Storz SS, Sun H, Thibaud-Nissen F, Tolstoy I, Tully RE,  
469 Vatsan AR, Wallin C, Webb D, Wu W, Landrum MJ, Kimchi A, Tatusova T, DiCuccio M, Kitts P,

- 470 Murphy TD, and Pruitt KD. 2016. Reference sequence (RefSeq) database at NCBI: current status,  
471 taxonomic expansion, and functional annotation. *Nucleic Acids Res* 44:D733-745.  
472 10.1093/nar/gkv1189
- 473 Ohno-Machado L, Sansone SA, Alter G, Fore I, Grethe J, Xu H, Gonzalez-Beltran A, Rocca-Serra P, Gururaj  
474 AE, Bell E, Soysal E, Zong N, and Kim HE. 2017. Finding useful data across multiple biomedical  
475 data repositories using DataMed. *Nat Genet* 49:816-819. 10.1038/ng.3864
- 476 Ooms J. 2017. opencpu: Producing and Reproducing Results. Available at [https://CRAN.R-](https://CRAN.R-project.org/package=opencpu)  
477 [project.org/package=opencpu](https://CRAN.R-project.org/package=opencpu) (accessed Sep. 3 2018).
- 478 Pedersen BS, Layer RM, and Quinlan AR. 2016. Vcfanno: fast, flexible annotation of genetic variants.  
479 *Genome Biol* 17:118. 10.1186/s13059-016-0973-5
- 480 Piñero J, Bravo À, Queralt-Rosinach N, Gutiérrez-Sacristán A, Deu-Pons J, Centeno E, García-García J,  
481 Sanz F, and Furlong LI. 2017. DisGeNET: a comprehensive platform integrating information on  
482 human disease-associated genes and variants. *Nucleic Acids Research* 45:D833-D839.  
483 10.1093/nar/gkw943
- 484 Ramos AH, Lichtenstein L, Gupta M, Lawrence MS, Pugh TJ, Saksena G, Meyerson M, and Getz G. 2015.  
485 Oncotator: Cancer Variant Annotation Tool. *Human Mutation* 36:2423-2429.
- 486 Sanchez-Vega F, Mina M, Armenia J, Chatila WK, Luna A, La KC, Dimitriadoy S, Liu DL, Kantheti HS,  
487 Saghafinia S, Chakravarty D, Daian F, Gao Q, Bailey MH, Liang WW, Foltz SM, Shmulevich I, Ding  
488 L, Heins Z, Ochoa A, Gross B, Gao J, Zhang H, Kundra R, Kandoth C, Bahceci I, Dervishi L,  
489 Dogrusoz U, Zhou W, Shen H, Laird PW, Way GP, Greene CS, Liang H, Xiao Y, Wang C, Iavarone A,  
490 Berger AH, Bivona TG, Lazar AJ, Hammer GD, Giordano T, Kwong LN, McArthur G, Huang C,  
491 Tward AD, Frederick MJ, McCormick F, Meyerson M, Cancer Genome Atlas Research N, Van  
492 Allen EM, Cherniack AD, Ciriello G, Sander C, and Schultz N. 2018. Oncogenic Signaling Pathways  
493 in The Cancer Genome Atlas. *Cell* 173:321-337 e310. 10.1016/j.cell.2018.03.035
- 494 Torre D, Krawczuk P, Jagodnik KM, Lachmann A, Wang Z, Wang L, Kuleshov MV, and Ma'ayan A. 2018.  
495 Datasets2Tools, repository and search engine for bioinformatics datasets, tools and canned  
496 analyses. *Sci Data* 5:180023. 10.1038/sdata.2018.23
- 497 Tyner C, Barber GP, Casper J, Clawson H, Diekhans M, Eisenhart C, Fischer CM, Gibson D, Gonzalez JN,  
498 Guruvadoo L, Haeussler M, Heitner S, Hinrichs AS, Karolchik D, Lee BT, Lee CM, Nejad P, Raney  
499 BJ, Rosenbloom KR, Speir ML, Villarreal C, Vivian J, Zweig AS, Haussler D, Kuhn RM, and Kent WJ.  
500 2017. The UCSC Genome Browser database: 2017 update. *Nucleic Acids Res* 45:D626-D634.  
501 10.1093/nar/gkw1134
- 502 Wala JA, Bandopadhyay P, Greenwald N, O'Rourke R, Sharpe T, Stewart C, Schumacher S, Li Y,  
503 Weischenfeldt J, Yao X, Nusbaum C, Campbell P, Getz G, Meyerson M, Zhang CZ, Imielinski M,  
504 and Beroukheim R. 2018. SvABA: genome-wide detection of structural variants and indels by local  
505 assembly. *Genome Res.* 10.1101/gr.221028.117
- 506 Wang K, Li M, and Hakonarson H. 2010. ANNOVAR: functional annotation of genetic variants from high-  
507 throughput sequencing data. *Nucleic Acids Res* 38:e164. 10.1093/nar/gkq603
- 508 Zook JM, Chapman B, Wang J, Mittelman D, Hofmann O, Hide W, and Salit M. 2014. Integrating human  
509 sequence data sets provides a resource of benchmark SNP and indel genotype calls. *Nat*  
510 *Biotechnol* 32:246-251. 10.1038/nbt.2835

511

**Figure 1**(on next page)

Overview of structure and functions of BioInstaller.

Bioinformatics tools, scripts and databases are supported by BioInstaller. Bootstrap and Shinydashbord are used to construct the front-end interface. The R functions, Shiny and Opencpu services and the SQLite and TOML databases were applied in the back-end.



**Figure 2** (on next page)

The relevance, applicability and a real example of BioInstaller

Non-programmer

Programmer

Meta database

File database

Pubmed  
 Datasets2Tools  
 BioInstaller  
 .....

GitHub  
 Zenodo  
 CRAN  
 Bioconductor  
 Conda  
 Spack  
 BioInstaler  
 .....

Pre-defined functions and files (BioInstaller)

R function, Shiny application, REST APIs, .....

Deployment of resources

Collection of resources

Sharing of resources

Construction of pipelines

Construction of Shiny application

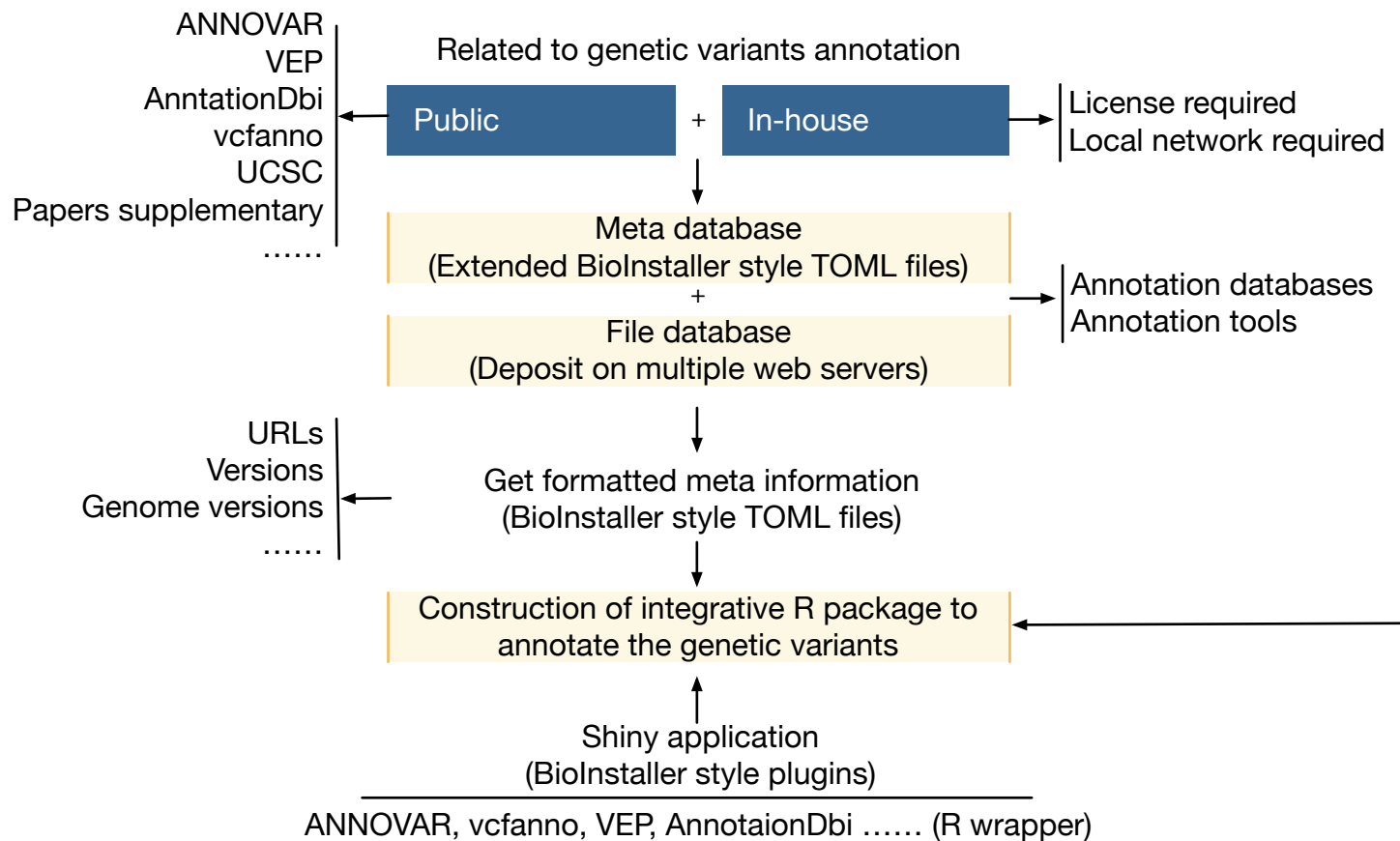
Reproducible data analysis

Users

Resources types

Service provider

Applications



**Figure 3**(on next page)

Shiny application modules of file viewer and pipeline.

(A) Uploaded files are showed on the table where view, deletion, and download function are provided. (B) The interface of the preview result. (C) Easy project was used as the demo in pipeline module, which could be used to create a series of directories via submitting a queue task with two parameters: project name and parent directory. (D) The dialog box displays a prompt message with a queue character key. (E) Task queue and queue information can be requested by the character key in the dashboard module. (F) Function to get the output log of the submitted task.

A

File List **Peer Preprints**

Copy CSV Excel PDF Print Search:

All files stored in shinyapp Web service

ID	Action	file_name	file_path	file_size	file
11		rap-2pg-cv.pdf	/Users/ljff.annovarR/upload/11	91516	pdf
12		ir-15-546.pdf	/Users/ljff.annovarR/upload/12	1666944	pdf
13		rap-2pg-cv.pdf	/Users/ljff.annovarR/upload/13	91516	pdf
14		dat.txt	/Users/ljff.BioInstaller/upload/14	66500	txt

Show 5 entries

Showing 11 to 14 of 14 entries Previous 1 2 3 Next

B

File Preview **NOT PEER-REVIEWED**

Copy CSV Excel PDF Print Search:

All files stored in shinyapp Web service

date	EMS	neonatal	infant	kid	child	adolescent	youth	middle age
20160101	426	0	0	7	0	6	61	4.
20160102	424	0	1	4	1	4	55	3.
20160103	393	0	1	0	1	0	47	4.
20160104	420	0	0	1	0	3	52	4.
20160105	397	0	2	3	0	1	46	4.

Show 5 entries

Showing 1 to 5 of 366 entries Previous 1 2 3 4 5 ... 74 Next

C

easy\_project

New bioinformatics analysis project

Parameters:

Project Name

Project parent dir

Submit

R commands for task

```

1 parent_dir <- normalizePath(parent_dir, mustWork = FALSE)
2 project_dir <- normalizePath(file.path(parent_dir, project_name), mustWork
3 analysis_dirs <- c(paste0("rnaseq/", c("exp", "fusion", "splicing", "mutat
4 paste0("dnaseq/", c("wes/mutation", "wgs/mutation", "ch
5 data_dirs <- c("fastq", "fasta", "bam", "vcf", "meta")
6 meta_json <- list(project_dir = project_dir, project_name = project_name,
7 create_time = format(Sys.time(), "%Y-%m-%d %H:%M:%S"),
8 creator = Sys.getenv("USER"))
9 meta_json$project_id <- git2r::hash(sprintf("%s,%s",
10 meta_json$project_name, meta_json$crea
11 sapply(sprintf("%s/%s", project_dir, c(file.path("analysis", analysis_dirs
12 file.path("data", data_dirs))),
13 function(x) {dir.create(x, recursiv
14 configr::write.config(meta_json, file.path(project_dir, "project.json"), w
15

```

D

Submit success!

Please copy and backup the followed messages. It is required to access the output or run in your command line client.

You can access the result via input key in the dashboard page:

S0uG5j0KN355S1J0FL6ZIOHds9l0viefBa43otseYa3unhRS9I

Commands and paramters:

```

{"input2var":{"project_name":["input_project_name"],"parent_dir":
["input_parent_dir"],"input":{"input_project_name":["B-ALL"],"in
put_parent_dir":["/tmp"],"req_pkgs":[""],"qqcommand":[""],"qqkey
":["S0uG5j0KN355S1J0FL6ZIOHds9l0viefBa43otseYa3unhRS9I"],"qqcomma
nd_type":["R"],"boxes":["new_proj"],"last_cmd":["parent_dir <- no
rmalizePath(parent_dir, mustWork = FALSE)\nproject_dir <- normali
zePath(file.path(parent_dir, project_name), mustWork = FALSE)\nan
alysis_dirs <- c(paste0("\rnaseq/", c("exp", "fusion", "spl
icing", "mutation")),\n paste0("\dnaseq/",
c("\wes/mutation", "\wgs/mutation", "chip/peak")))\ndata_dirs
<- c("\fastq", "\fasta", "\bam", "\vcf", "\meta")\nmeta_jso
n <- list(project_dir = project_dir, project_name = project_name,
\n create_time = format(Sys.time(), "%Y-%m-%d %H:%M:%S"),\ncrea
tor = Sys.getenv("USER"))\nmeta_json$project_id <- git2r::hash
(sprintf("\%s,%s",\n\n meta_jso
n$project_name, meta_json$create_time))\nsapply(sprintf("\%s/%s"
, project_dir, c(file.path("\analysis", analysis_dirs,\n
file.path("data", data_dirs)),
\n function(x) {dir.create(
x, recursive = TRUE))\nconfigr::write.config(meta_json, file.pat
h(project_dir, "\project.json"), write.type = "\json")\n"}]

```

E

Task table query

Key S0uG5j0KN355S1J0FL6ZIOHds9l0viefBa43otseYa3unhRS9I

Query

Task information

Copy CSV Excel PDF Print Search:

Task information

id	msgid	key	status	log
98	106	S0uG5j0KN355S1J0FL6ZIOHds9l0viefBa43otseYa3unhRS9I	FINISHED	~/BioInst

Show 5 entries

Showing 1 to 1 of 1 entries

PeerJ Preprints | <https://doi.org/10.7287/peerj.preprints.2721.v1>

F

Output of log

```

----- ~/BioInstaller/log/S0uG5j0KN355S1J0FL6ZIOHds9l0viefBa43otseYa3unhRS9I
.log
project_name =>
[1] "B-ALL"
parent_dir =>
[1] "/tmp"
parent_dir <- normalizePath(parent_dir, mustWork = FALSE)
project_dir <- normalizePath(file.path(parent_dir, project_name), mustWork = FALSE)
analysis_dirs <- c(paste0("rnaseq/", c("exp", "fusion", "splicing", "mutation")),
paste0("dnaseq/", c("wes/mutation", "wgs/mutation", "chip/peak")))
data_dirs <- c("fastq", "fasta", "bam", "vcf", "meta")
meta_json <- list(project_dir = project_dir, project_name = project_name,
create_time = format(Sys.time(), "%Y-%m-%d %H:%M:%S"),
creator = Sys.getenv("USER"))
meta_json$project_id <- git2r::hash(sprintf("%s,%s",
meta_json$project_name, meta_json$create_time))
sapply(sprintf("%s/%s", project_dir, c(file.path("analysis", analysis_dirs),
function(x) {dir.create(x, recursive = TRUE)}))
configr::write.config(meta_json, file.path(project_dir, "project.json"), write.type = "json")

```

CC BY 4.0 Open Access | rec: 17 Sep 2018 | doi: 10.7287/peerj.preprints.2721.v1



**Figure 4**(on next page)

Shiny application 'Instant' module

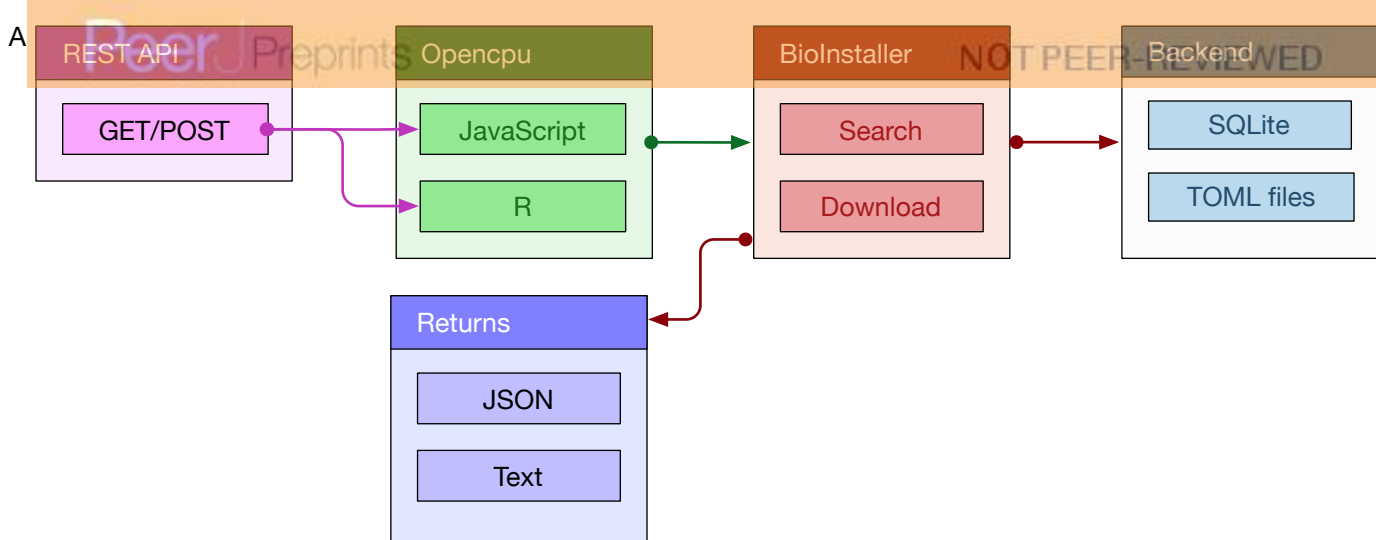
(A)(B) A demonstration show how Shiny 'Installer' module works by downloading 'db\_ucsc\_refgene'. Dynamic and interactive manipulations are supported. Log information of submitted download/install job can be recalled using given random characters. (C)(D) The input box and output log when a new softwares environment is created by the conda plugin of 'Installer' module. (E)(F) The input box and output log when 'zlib' is installed by the spack plugin of 'Installer' module.



**Figure 5**(on next page)

REST APIs of BioInstaller.

(A) Workflow of REST APIs of BioInstaller that JSON and TEXT returns through the GET/POST query. (B) Using curl to invoke background R functions of BioInstaller. (C) The key character with GET method is provided to get the background R session output.



B

```

curl http://localhost:5656/ocpu/library/BioInstaller/R/install.bioinfo -d \
  "show.all.names=TRUE" -X POST
  
```

Show all items supported by BioInstaller

```

curl http://localhost:5656/ocpu/library/BioInstaller/R/install.bioinfo \
  -d "name='bwa', show.all.versions=TRUE" -X POST
  
```

Show all versions of Bwa

```

curl http://localhost:5656/ocpu/library/BioInstaller/R/install.bioinfo \
  -d "name='bwa', destdir='/opt/aliner/bwa'" -X POST
  
```

Download and install latest Bwa

```

curl http://localhost:5656/ocpu/library/BioInstaller/R/install.bioinfo \
  -d "name='db_annovar_1000g', destdir='/opt/annovardb', \
  extra.list=list(buildver='hg19')" -X POST
  
```

Download 1000 Genome Project annotation database

C

```

curl http://localhost:5656/ocpu/tmp/{key}/R.val/json
  
```

Get the JSON format value of returned output

```

curl http://localhost:5656/ocpu/tmp/{key}/R.val/text
  
```

Get the text format value of returned output

```

curl http://localhost:5656/ocpu/tmp/{key}/R.val/print
  
```

Get the function 'print' output of returned value

PeerJ Preprints | <https://doi.org/10.7287/peerj.preprints.27221v1> | CC BY 4.0 Open Access | rec: 17 Sep 2018, publ: 17 Sep 2018

**Table 1** (on next page)

List of the relevance and applicability of BioInstaller

	With BioInstaller	Without BioInstaller
<b>Deployment of resources</b>		
User-interfaces	R functions, Shiny UI, REST APIs (Conda, Spack, and other tools/scripts)	Command-line tools (Conda, Spack, and custom tools)
Retrieve installed packages	Integrated Shiny dashboard page including R packages, conda and Python packages, Spack packages, and BioInstaller resource	Multiple command line operations
<b>Collection of resources</b>		
Local development	Yes	No
Need to register an account	Not need	Need
Type of backend databases	Default use TOML and SQLite (portable purpose) Plugins for other types	MySQL
Resources hosts	No limitation	Centralized
File sizes	No limitation	Limited
PubMed query	Integrated R codes with secret key (no limited access) Shiny UI with formatted table	Isolated R codes without secret key (limited access, $n \leq 20$ ) Online version without formatted table
<b>Sharing of resources</b>		
Medium	Simplified TOML format files	Form or configuration file required more skills
Download service	Local Shiny application	Centralized web service or command line tools
<b>Construction of pipelines</b>		
Store of meta information (e.g. URL and version)	Pre-defined TOML file	De novo source code (e.g. ANNOVAR and fusioncatcher)
<b>Construction of Shiny application</b>		

Pre-defined pages	Pre-defined Shiny UI and server (Dashboard, file management, task submission, logging, export and update of plots exception handling, setting)	Isolated examples UI and server codes
Difficulty	Easy to construct the Shiny application (Plugins + optional R codes)	Relatively complicated (Require R codes for UI and server)
<b>Reproducible data analysis</b>		
Logging	Support	Manual
Docker image	Pre-defined docker image with Shiny, Rstudio, and Opencpu services	Most not

1

**Table 2** (on next page)

Overview comparison of BioInstaller and existing tools on the collection and share of bioinformatics resources



	BioInstaller	Omictools	Datasets2Tools
<b>Infrastructure and Utilities</b>			
Programing language	R, JavaScript	HTML/CSS/JavaScri pt	HTML/CSS/JavaScri pt
Chrome extension	No	No	Yes
Web service	R Shiny	Web	Web
R functions	Yes	No	No
REST APIs	Yes	No	Yes
Backend database	TOML and SQLite	Not available	MySQL
Docker image	Yes	No	No
<b>Functionality</b>			
Access and collect meta database	Yes	Yes	Yes
Access and collect file database	Yes	No	No
Integration of external resources	Yes	No	No
PubMed query	Yes	No	No
Dataset query	Yes	No	Yes
Number of supported resources	Integrated	High	Medium
Version query	Yes	No	No
Download service	Yes	No	No
Local branch and development	Yes	No	No
<b>Input and output</b>			
Input	R functions, Web text, APIs	Web text only	Web text + APIs
Output	Text, table, plots, and Web page (PNG, SVG and PDF)	Web page	Text and Web page

**Table 3** (on next page)

Summary of BioInstaller included tools/scripts and databases

Category	Number
<b>1, Tools/scripts</b>	
Alignment and assembly	27
Quality control	17
HTS manipulation	17
Association analysis	6
Genetic variants annotation	12
Detection of SNVs, INDELs and SVs	32
Immunity-associated	2
Isoform analysis	3
Gene expression analysis	9
Network analysis	3
Visualization libraries	11
System dependence	18
<b>2, Databases</b>	
<b>Variant-level</b>	
Allele frequency	17
Variants Effect prediction	29
Disease-related	6
<b>Gene-level</b>	
Basic information	8
Gene function	3
Disease-related	7
<b>Drug related</b>	4
<b>Noncoding RNA related</b>	15
<b>Reference genome</b>	9
<b>Protein related</b>	4
<b>Others</b>	8

1