

1 Reproducible Bioinformatics Project: A community for reproducible 2 bioinformatics analysis pipelines

3 Neha Kulkarni¹, Luca Alessandri¹, Riccardo Panero¹, Maddalena Arigoni¹, Martina Olivero²,
4 Francesca Cordero^{3§}, Marco Beccuti³ and Raffaele A Calogero^{1§}

5

6 ¹Dept. of Molecular Biotechnology and Health Sciences, University of Torino, Torino, Italy

7 ²Dept. of Oncology, University of Torino, Candiolo, Italy

8 ³Dept. of Computer Sciences, University of Torino, Torino, Italy

9

10 Neha Kulkarni kulkarnineha220@gmail.com

11 Luca Alessandri alessandri.luca1991@gmail.com

12 Riccardo Panero riccardo.panero@gmail.com

13 Maddalena Arigoni maddalena.arigoni@unito.it

14 Martina Olivero martina.olivero@ircc.it

15 Francesca Cordero fcordero@di.unito.it

16 Marco Beccuti beccuti@di.unito.it

17 Raffaele A Calogero raffaele.calogero@unito.it

18

19 [§]Corresponding author

20

21 Abstract

22 **Background** Reproducibility of a research is a key element in the modern science and it is
23 mandatory for any industrial application. It represents the ability of replicating an experiment

24 independently by the location and the operator. Therefore, a study can be considered
25 reproducible only if all used data are available and the exploited computational analysis workflow
26 is clearly described. However, today for reproducing a complex bioinformatics analysis, the raw
27 data and a list of tools used in the workflow could be not enough to guarantee the reproducibility
28 of the results obtained. Indeed, different releases of the same tools and/or of the system libraries
29 (exploited by such tools) might lead to sneaky reproducibility issues.

30 **Results** To address this challenge, we established the *Reproducible Bioinformatics Project (RBP)*,
31 which is a non-profit and open-source project, whose aim is to provide a schema and an
32 infrastructure, based on docker images and R package, to provide reproducible results in
33 Bioinformatics. One or more Docker images are then defined for a workflow (typically one for each
34 task), while the workflow implementation is handled via R-functions embedded in a package
35 available at github repository. Thus, a bioinformatician participating to the project has firstly to
36 integrate her/his workflow modules into Docker image(s) exploiting an Ubuntu docker image
37 developed ad hoc by RPB to make easier this task. Secondly, the workflow implementation must
38 be realized in R according to an R-skeleton function made available by RPB to guarantee
39 homogeneity and reusability among different RPB functions. Moreover she/he has to provide the
40 R vignette explaining the package functionality together with an example dataset which can be
41 used to improve the user confidence in the workflow utilization.

42 **Conclusions** Reproducible Bioinformatics Project provides a general schema and an infrastructure
43 to distribute robust and reproducible workflows. Thus, it guarantees to final users the ability to
44 repeat consistently any analysis independently by the used UNIX-like architecture.

45 Keywords

46 Reproducible research, docker, whole transcriptome sequencing, miRNA sequencing, ChIP
47 sequencing, community, SNV.

48 Background

49 Recently Baker and Lithgow [1, 2] highlighted the problem of the reproducibility in research.
50 Reproducibility criticality affects to different extent a large portion of the science fields [1]. Since
51 nowadays bioinformatics plays an important role in many biological and medical studies [3], a
52 great effort must be put to make such computational analyses reproducible [4, 5]. Reproducibility
53 issues in bioinformatics might be due to the short half-life of the bioinformatics software, the
54 complexity of the pipelines, the uncontrolled effects induced by changes in the system libraries,
55 the incompleteness or imprecision in workflow description, etc. To deal with reproducibility issues
56 in Bioinformatics Sandve [5] suggested ten good practice rules for the development of a
57 computational workflow (Table 1). A community that fulfill some of the rules suggested by Sandve
58 is Bioconductor [6] project, which provides version control for a large amount of
59 genomics/bioinformatics packages. In this way, old releases of any Bioconductor package are kept
60 available for the users. However, Bioconductor does not cover all the steps of any possible
61 bioinformatics workflow, e.g. in RNAseq workflow fastq trimming and alignment steps are
62 generally done using tools not implemented in Bioconductor. BaseSpace [7, 8] and Galaxy [9]
63 represent an example of both commercial and open-source solutions, which partially fulfill
64 Sandve's roles. Furthermore, the workflows implemented in such environments cannot be heavily
65 customized, e.g. BaseSpace has strict rules for applications submission. Moreover, clouds
66 applications, as BaseSpace, have to cope with legal and ethical issues [10]. On the other hand,
67 Galaxy does not provide standardized metadata to annotate workflows.

68 Recently container technology, a lightweight OS-level virtualization, was explored in the area of
69 Bioinformatics to make easier the distribution, the utilization and the maintenance of
70 bioinformatics software [11-13]. Indeed, since applications and their dependencies are packaged
71 together in the container image, the users have not to download and install all the dependencies
72 required by an application, thus avoiding all the cases where the dependencies are not well
73 documented or not available at all. Moreover, problems related to versions conflicts or updates of
74 the system libraries do not occur, because the containers are isolated from the rest of the
75 operating system.

76 Among the available container platforms, Docker (<http://www.docker.com>) is becoming *de facto*
77 the standard environment to quickly compose, create, deploy, scale and oversee containerized
78 applications under Linux. Its strengths are the high degree of portability, which allows users to
79 register and share containers over various hosts in private and public repositories; a more
80 effective resource use and a faster deployment compared with other software.

81 Although, Menegidio [13], da Veiga [11] and Kim [12] provided a large collection of bioinformatics
82 instruments based on Docker technology, today we are missing a community delivering to
83 bioinformaticians a controlled, but flexible framework to distribute Docker based workflows under
84 the umbrella of a reproducibility framework. Here, we describe the implementation of the
85 Reproducible Bioinformatics Project (RBP, <http://reproducible-bioinformatics.org/>), aiming to
86 distribute to the bioinformatics community docker-based applications under the reproducibility
87 framework proposed by Sandve [5]. RBP accepts simple docker implementations of *bioinformatics*
88 *software* (e.g. a docker embedding bwa aligner tool), implementation of *complex pipelines*
89 involving the use of multiple dockers images (e.g. a RNAseq workflow providing all the steps for an
90 analysis starting from the quality control of the fastq to differential expression), as well as

91 *demonstrative workflows* (i.e. docker images embedding the full bioinformatics workflow used in a
92 publication) intended to provide the ability to reproduce published data.

93 Implementation

94 The Reproducible Bioinformatics Project (RBP) reference web page is [reproducible-](#)
95 [bioinformatics.org](#). The project is based on three modules (Figure 1): (i) *docker4seq R package*
96 (<https://github.com/kendomaniac/docker4seq>), (ii) *dockers images*
97 (<https://hub.docker.com/u/repbioinfo/>), and (iii) *4SeqGUI*
98 (<https://github.com/mbeccuti/4SeqGUI>).

99 *Docker4seq* package provides the connection between users and docker containers. *Docker4seq* is
100 organized in two branches: stable and development. The transition between development and
101 stable branch is done when a module (R function(s)/docker container(s)) fulfills the 10 rules
102 suggested by Sandve [5] for good bioinformatics practice (Table 1):

103 The function *skeleton.R* in *docker4seq* provides a prototype to build a docker controlling function.
104 Acknowledgments of the developer work is provided within the structure of the *skeleton.R*. In
105 *skeleton.R* there is a field indicating developer affiliation and email for contacts. In docker images
106 repository [docker.io/repbioinfo](#) is available an Ubuntu image, as prototype for the creation of a
107 docker image compliant with the RBP specifications. Developer is free to decide to use this
108 prototype or to adapt a different Linux docker distribution for his/her application. Docker images
109 designed by the core developers of RBP are located in [docker.io/repbioinfo](#) (docker.com), the
110 images developed by third parties can be instead placed in any public-access docker repository.
111 RBP requires that any operation, implying the use of any R/Bioconductor packages or the use of an
112 external software, has to be implemented in a docker container. Only reformatting actions, e.g.
113 table assembly, data reordering, etc., can be handled outside a docker image.

114 Any new RBP module (R function(s)/docker image(s)) must be associated with an explanatory
115 vignette, accessible online as html document, and to a set of test data, also accessible online.
116 Thus, all instruments needed to acquire confidence on module functionalities are provided to the
117 final user.
118 Docker images are labelled with the extension YYYY.NN, where YYYY is the year of insertion in the
119 stable version and NN a progressive number. YYYY changes only if any update on the program(s),
120 implemented in the docker image, is done. This because any of such updates will affect the
121 reproducibility of the workflow. Previous version(s) will be also available in the repository. NN
122 refers to changes in the docker image, which do not affect the reproducibility of the workflow.
123 A new module can be submitted to the info@reproducible-bioinformatics.org and RBP core team
124 will verify the compliance with Sandve [5] rules. Ones validated, the R functions controlling the
125 new module are inserted in *docker4seq* stable release. Partially validated modules will be placed in
126 development branch and moved to stable one when compliance with Sandve's rules is fulfilled.
127 4SeqGUI is a Java based graphical interface to *docker4seq* functions. It is designed to provide a
128 GUI to users having limited knowledge of R scripting. Currently the GUI embeds only general-
129 purpose workflows, such as RNAseq, miRNAseq and Chip-seq workflow.

130 Results

131 The stable branch of *docker4seq R package* contains all the R functions required to handle all the
132 steps of RNAseq workflow (Fig. 2A), ChIPseq workflow (Fig. 2B), and miRNAseq workflow (Fig. 2C).
133 *Docker4seq* also provides a wrapper function for the *bcl2fastq* Illumina tool to convert the Illumina
134 sequencer output in demultiplexed fastq files (Fig. 2). Then, the fastq files can be handled with any
135 of the three different workflows. The counts table produced by RNAseq or miRNAseq workflows
136 can be used for data visualization (*pca*, principal component analysis function), to evaluate the

137 statistical power of the experiment (*experimentPower* function), to define the optimal sample size
138 of the experiment for the detection of differentially expressed genes (*sampleSize* function) and to
139 detect differentially expressed genes/transcripts (*wrapperDeseq2* function). Sample size/statistical
140 power estimation of the experiment and differential expression are calculated respectively via
141 *RnaSeqSampleSize* [14] and *DESeq2* Bioconductor packages [15].

142 In the development branch, the main effort of the core developers is focused in providing
143 workflows for DNA and RNA somatic variant calling. The DNA variant calling workflow embeds the
144 pre-processing procedure suggested by the GATK best practice (Fig. 3A). RNAseq data preparation
145 for variant calling (Fig. 3C) requires the use of STAR 2 step procedure [16], which provides
146 significantly increased sensitivity to novel splice junctions. Then, after sorting and duplicates
147 marking, OPOSSUM [17] is used to remove intronic regions and to merge overlapping reads. We
148 have also implemented a specific procedure (Fig. 3B), based on xenome software [18], to
149 discriminate between human reads and mouse host reads in the sequences produced by the
150 analysis of patients derived xenografts (PDX, [19]). As part of the somatic variant calling workflow
151 we are implementing MUTECT 1 and 2 [20] (Fig. 4A) to call somatic variants as well as PLATYPUS
152 [21] for extracting information of joined-samples SNVs (Fig. 4B).

153 We are also expanding the RNAseq module adding the reference-free Salmon aligner [22], which
154 employs less memory for the alignment task than STAR, but providing similar results [23].

155 Finally, HashClone framework (Accepted for publication in BMC Bioinformatics), a new suite of
156 bioinformatics tools providing B-cells clonality assessment and minimal residual disease (MRD)
157 monitoring over time from deep sequencing data, was integrated in the *Docker4seq* package. In
158 particular, a parallel version of the standard HashClone workflow (Fig. 5) was developed exploiting
159 the docker architecture.

160 All the modules described above are implemented in 18 docker images deposited in the *docker*
161 hub (<https://hub.docker.com/u/repbioinfo/>).
162 As part of the RBP we have also developed a GUI, 4SeqGUI
163 (<https://github.com/mbeccuti/4SeqGUI>). The GUI is implemented in JAVA and can be exploited to
164 perform whole transcriptome sequencing workflow (Fig. 2A), CHIP sequencing workflow (Fig. 2B),
165 and miRNA sequencing workflow (Fig. 2C).

166 Discussion

167 Bioinformatics workflows are becoming an essential part of many research papers. However,
168 absence of clear and well-defined rules on the code distribution make the results of most
169 published researches unreproducible [24]. Recently, Almugbel and coworkers [25] described an
170 interesting infrastructure to embed Bioconductor based packages. However, Bioconductor does
171 not cover all steps of any possible bioinformatics workflow, thus providing a limited framework for
172 developing complex pipelines. Differently, RBP represents a new instrument, which expands the
173 idea of Almugbel [25], providing a more flexible infrastructure allowing the bioinformatics
174 community to spread their work under the guidance of rules, which guarantee inter-laboratory
175 reproducibility and do not limit docker implementations to Bioconductor packages. RBP core
176 developers created frameworks for RNA/miRNA quantification and analysis. CHIPseq workflow was
177 also developed and variant calling workflows for DNA and RNA are under active development. A
178 peculiar feature of RBP is the acceptance of *demonstrative workflows*, i.e. bioinformatics
179 procedures described in a biological/medical paper. A demonstrative workflow is wrapped in a
180 docker image and it is supported by a tutorial, which describes step by step how the analysis is
181 done to guarantee the reproducibility of published data.

182 Availability and requirements

183 **Project name:** Reproducible Bioinformatics Project

184 **Project home page:** <http://reproducible-bioinformatics.org>

185 **Operating system:** UNIX-like

186 **Programming language:** R

187 **Other requirements:** docker version 17.05.0-ce or higher

188 **License:** GPL.

189

190 Declarations

191 Competing interests

192 None

193

194 Funding

195 This work has been supported by the EPIGEN FLAG PROJECT

196

197 Authors' contributions

198 NK and LA equally contributed to the development of miRNA workflow and all the other tools. RP

199 and FC developed the RNAseq workflow and refined the ChIPseq workflow. MA and MO

200 performed applications testing. MB and RAC developed the rules to submit tools and workflows to

201 the Reproducible Bioinformatics community. RAC and MB equally supervised the overall work.

202

203 Figures caption

204

205 **Figure 1: Reproducible Bioinformatics Project structure.**

206

207 **Figure 2: Workflows available in the stable branch of docker4seq.** A) Whole transcriptome

208 sequencing workflow, B) ChIP sequencing workflow, and C) miRNA sequencing workflow. The

209 names followed by parenthesis are the docker4seq functions used to execute the analysis steps.

210 Black indicate elements in common among more than one workflow.

211

212 **Figure 3: Variant calling workflows under refinement in the development branch of docker4seq.**

213 A) SNVs calling in DNA workflow. The function *snvPreprocessing* requires that users provides its

214 own copy of the GATK software, because of Broad Institute license restrictions. This function

215 returns a bam file sorted, with duplicates marked after GATK indel realignment and quality

216 recalibration. B) Data preprocessing for samples derived by Patient Derived Xenografts (PDX).

217 The *xenome* function discriminates between the mouse host reads and the human tumor reads,

218 then DNA or RNA SNV calling workflows can be applied .C) SNVs calling in RNA workflow. The

219 function *star2steps* generates a sorted bam, where duplicates are marked and processed by

220 opossum for removal of intronic regions and merging of overlapping reads. The names followed by

221 parenthesis are the docker4seq functions used to execute the analysis steps. Black indicate

222 elements in common between more than one workflow.

223

224 **Figure 4: Variant calling workflows under development in the development branch of**

225 **docker4seq.** A) Somatic SNVs detection using GATK MUTECT 1 or 2. B) Platypus based join

226 mutations caller. Dashed blocks are not implemented, yet.

227

228 **Figure 5: HashClone pipeline.** The HashClone strategy is organized in three steps:
229 The first step (red box) is used to detect k-mer in all patients' samples. The second step (green
230 box) focus on the generation of sequence signatures leading to the identification of the set of
231 putative clones present in each of the patients' sample; the third step (blue box) is used to the
232 characterization and evaluation of the cancer clones.

233

234 References

- 235 1. Baker M: **1,500 scientists lift the lid on reproducibility.** *Nature* 2016, **533**(7604):452-454.
- 236 2. Lithgow GJ, Driscoll M, Phillips P: **A long journey to reproducible results.** *Nature* 2017,
237 **548**(7668):387-388.
- 238 3. Searls DB: **The roots of bioinformatics.** *PLoS computational biology* 2010, **6**(6):e1000809.
- 239 4. Kanwal S, Khan FZ, Lonie A, Sinnott RO: **Investigating reproducibility and tracking**
240 **provenance - A genomic workflow case study.** *BMC bioinformatics* 2017, **18**(1):337.
- 241 5. Sandve GK, Nekrutenko A, Taylor J, Hovig E: **Ten simple rules for reproducible**
242 **computational research.** *PLoS computational biology* 2013, **9**(10):e1003285.
- 243 6. Gentleman RC, Carey VJ, Bates DM, Bolstad B, Dettling M, Dudoit S, Ellis B, Gautier L, Ge Y,
244 Gentry J *et al*: **Bioconductor: open software development for computational biology and**
245 **bioinformatics.** *Genome biology* 2004, **5**(10):R80.
- 246 7. Colombo AR, J. Triche T J, Ramsingh G: **Arkas: Rapid reproducible RNAseq analysis.**
247 *F1000Res* 2017, **6**:586.
- 248 8. Van Neste C, Gansemans Y, De Coninck D, Van Hoofstat D, Van Criekeing W, Deforce D, Van
249 Nieuwerburgh F: **Forensic massively parallel sequencing data analysis tool:**
250 **Implementation of MyFLq as a standalone web- and Illumina BaseSpace((R))-application.**
251 *Forensic Sci Int Genet* 2015, **15**:2-7.

- 252 9. Digan W, Countouris H, Barritault M, Baudoin D, Laurent-Puig P, Blons H, Burgun A, Rance
253 **B: An Architecture for Genomics Analysis in a Clinical Setting Using Galaxy and Docker.**
254 *Gigascience* 2017.
- 255 10. Dove ES, Joly Y, Tasse AM, Public Population Project in G, Society International Steering C,
256 International Cancer Genome Consortium E, Policy C, Knoppers BM: **Genomic cloud**
257 **computing: legal and ethical points to consider.** *European journal of human genetics* :
258 *EJHG* 2015, **23**(10):1271-1278.
- 259 11. da Veiga Leprevost F, Gruning BA, Alves Aflitos S, Rost HL, Uszkoreit J, Barsnes H, Vaudel M,
260 Moreno P, Gatto L, Weber J *et al*: **BioContainers: an open-source and community-driven**
261 **framework for software standardization.** *Bioinformatics* 2017, **33**(16):2580-2582.
- 262 12. Kim B, Ali T, Lijeron C, Afgan E, Krampis K: **Bio-Docklets: virtualization containers for**
263 **single-step execution of NGS pipelines.** *Gigascience* 2017, **6**(8):1-7.
- 264 13. Menegidio FB, Jabes DL, Costa de Oliveira R, Nunes LR: **Dugong: a Docker image, based on**
265 **Ubuntu Linux, focused on reproducibility and replicability for bioinformatics analyses.**
266 *Bioinformatics* 2017.
- 267 14. Ching T, Huang S, Garmire LX: **Power analysis and sample size estimation for RNA-Seq**
268 **differential expression.** *RNA* 2014, **20**(11):1684-1696.
- 269 15. Love MI, Huber W, Anders S: **Moderated estimation of fold change and dispersion for**
270 **RNA-seq data with DESeq2.** *Genome biology* 2014, **15**(12):550.
- 271 16. Dobin A, Davis CA, Schlesinger F, Drenkow J, Zaleski C, Jha S, Batut P, Chaisson M, Gingeras
272 TR: **STAR: ultrafast universal RNA-seq aligner.** *Bioinformatics* 2013, **29**(1):15-21.
- 273 17. Oikkonen L, Lise S: **Making the most of RNA-seq: Pre-processing sequencing data with**
274 **Opossum for reliable SNP variant detection.** *Wellcome Open Res* 2017, **2**:6.

- 275 18. Conway T, Wazny J, Bromage A, Tymms M, Sooraj D, Williams ED, Beresford-Smith B:
276 **Xenome--a tool for classifying reads from xenograft samples.** *Bioinformatics* 2012,
277 **28(12):i172-178.**
- 278 19. Siolas D, Hannon GJ: **Patient-derived tumor xenografts: transforming clinical samples into**
279 **mouse models.** *Cancer research* 2013, **73(17):5315-5319.**
- 280 20. Cibulskis K, Lawrence MS, Carter SL, Sivachenko A, Jaffe D, Sougnez C, Gabriel S, Meyerson
281 M, Lander ES, Getz G: **Sensitive detection of somatic point mutations in impure and**
282 **heterogeneous cancer samples.** *Nature biotechnology* 2013, **31(3):213-219.**
- 283 21. Rimmer A, Phan H, Mathieson I, Iqbal Z, Twigg SRF, Consortium WGS, Wilkie AOM, McVean
284 G, Lunter G: **Integrating mapping-, assembly- and haplotype-based approaches for calling**
285 **variants in clinical sequencing applications.** *Nature genetics* 2014, **46(8):912-918.**
- 286 22. Patro R, Duggal G, Love MI, Irizarry RA, Kingsford C: **Salmon provides fast and bias-aware**
287 **quantification of transcript expression.** *Nature methods* 2017, **14(4):417-419.**
- 288 23. Zhang C, Zhang B, Lin LL, Zhao S: **Evaluation and comparison of computational tools for**
289 **RNA-seq isoform quantification.** *BMC genomics* 2017, **18(1):583.**
- 290 24. Hothorn T, Leisch F: **Case studies in reproducibility.** *Briefings in bioinformatics* 2011,
291 **12(3):288-300.**
- 292 25. Almugbel R, Hung LH, Hu J, Almutairy A, Ortogero N, Tamta Y, Yeung KY: **Reproducible**
293 **Bioconductor workflows using browser-based interactive notebooks and containers.** *J Am*
294 *Med Inform Assoc* 2017.

295

296 Tables

297

298 **Table 1:** Good practice bioinformatics rules, derived from Sandve et al. [5]

-
- 1** For Every Result, Keep Track of How It Was Produced

 - 2** Avoid Manual Data Manipulation Steps

 - 3** Archive the Exact Versions of All External Programs Used

 - 4** Version Control All Custom Scripts

 - 5** Record All Intermediate Results, When Possible in Standardized Formats

 - 6** For Analyses That Include Randomness, Note Underlying Random Seeds

 - 7** Always Store Raw Data behind Plots

 - 8** Generate Hierarchical Analysis Output, Allowing Layers of Increasing Detail to Be Inspected

 - 9** Connect Textual Statements to Underlying Results

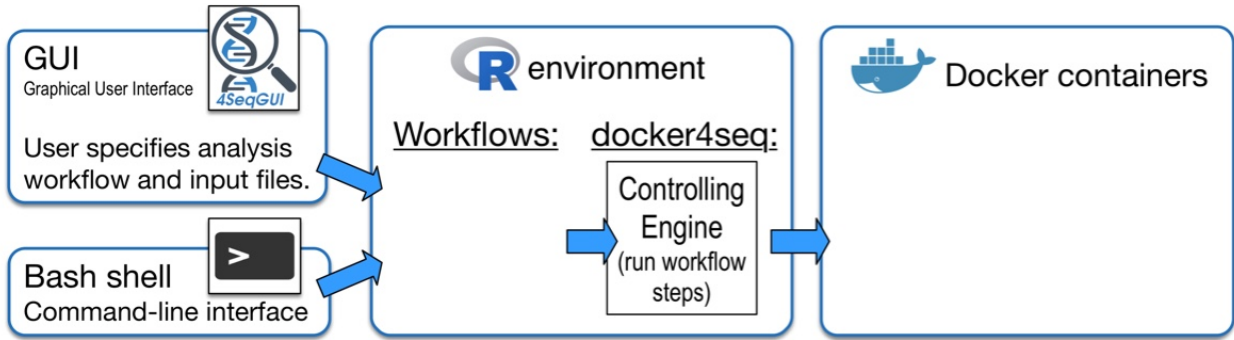
 - 10** Provide Public Access to Scripts, Runs, and Results

299

300

301 Figures

302



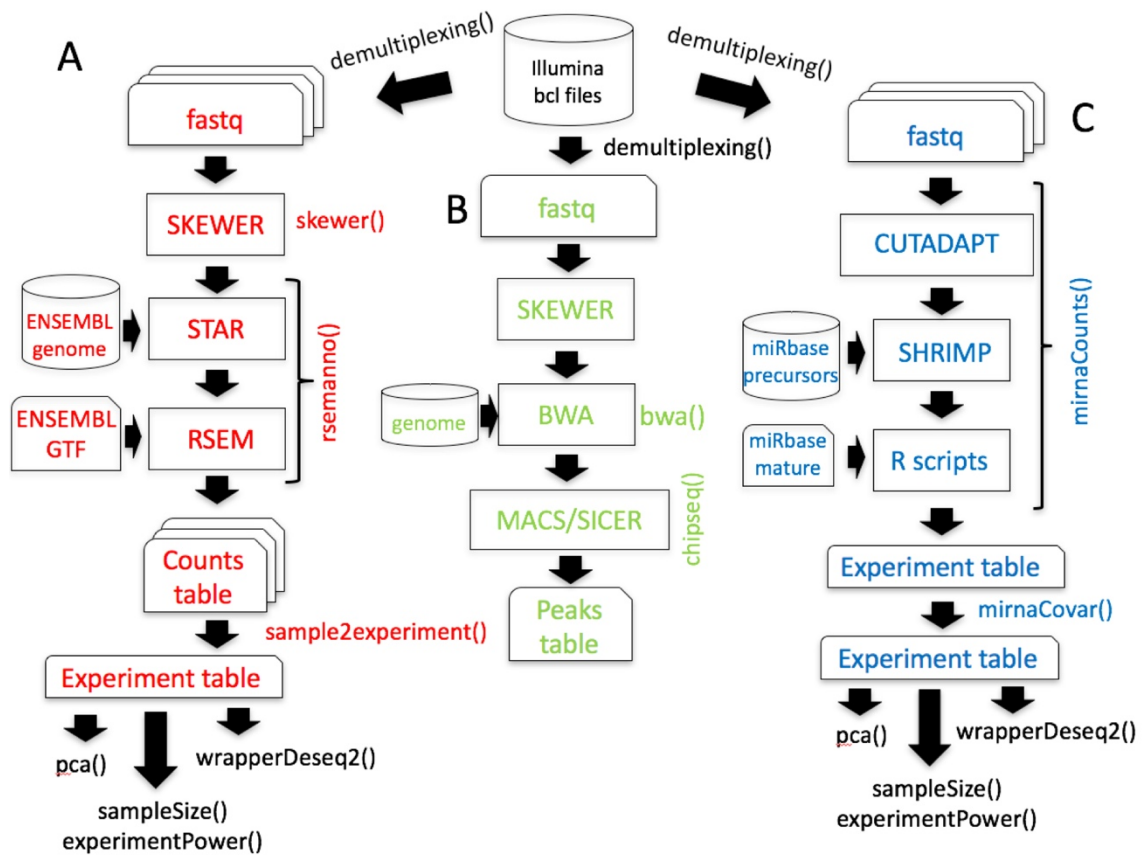
303

304

305

Figure 1

306



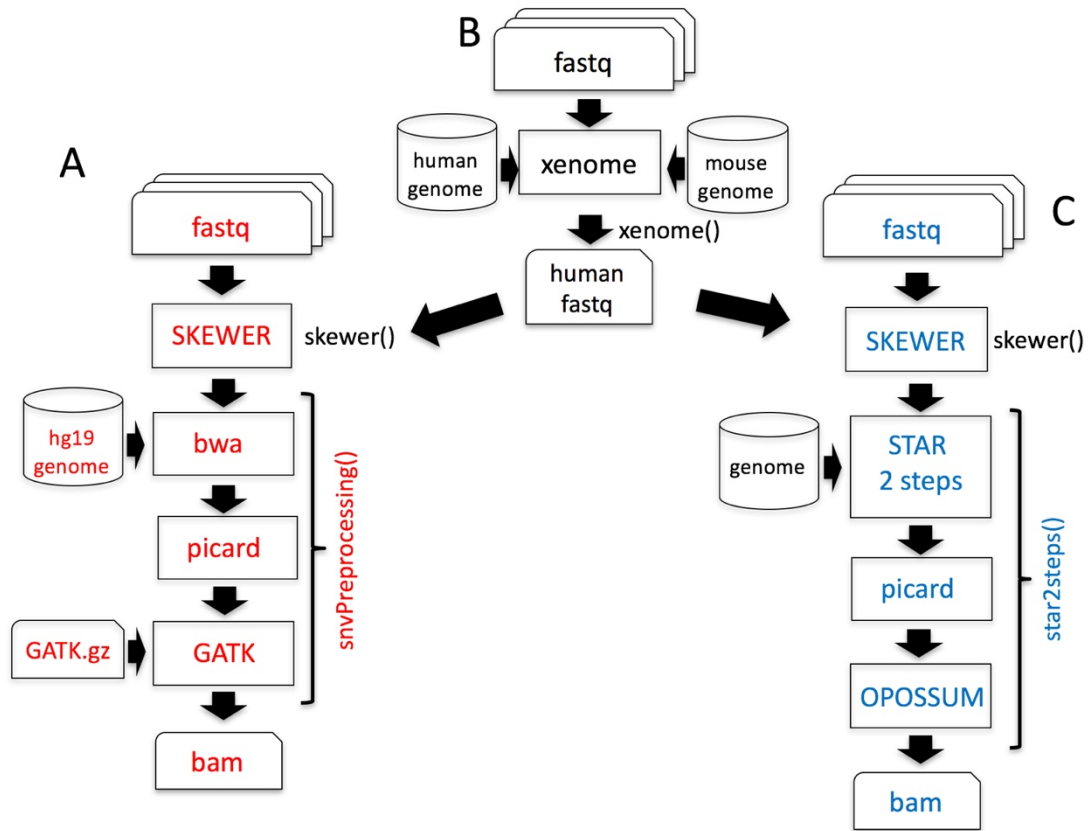
307

308

309

Figure 2

310



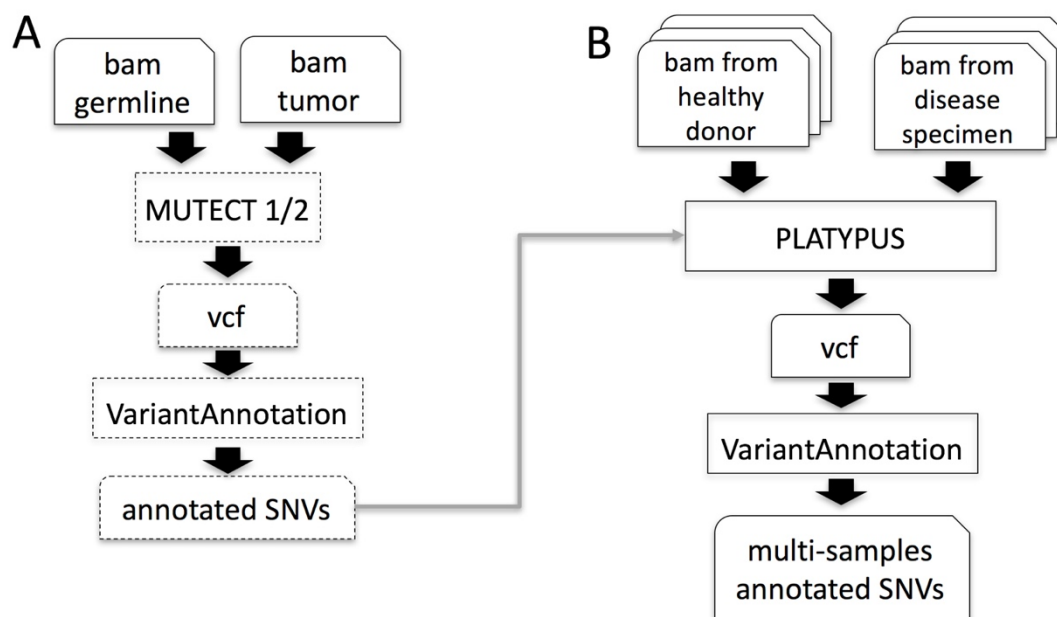
311

312

313

Figure 3

314



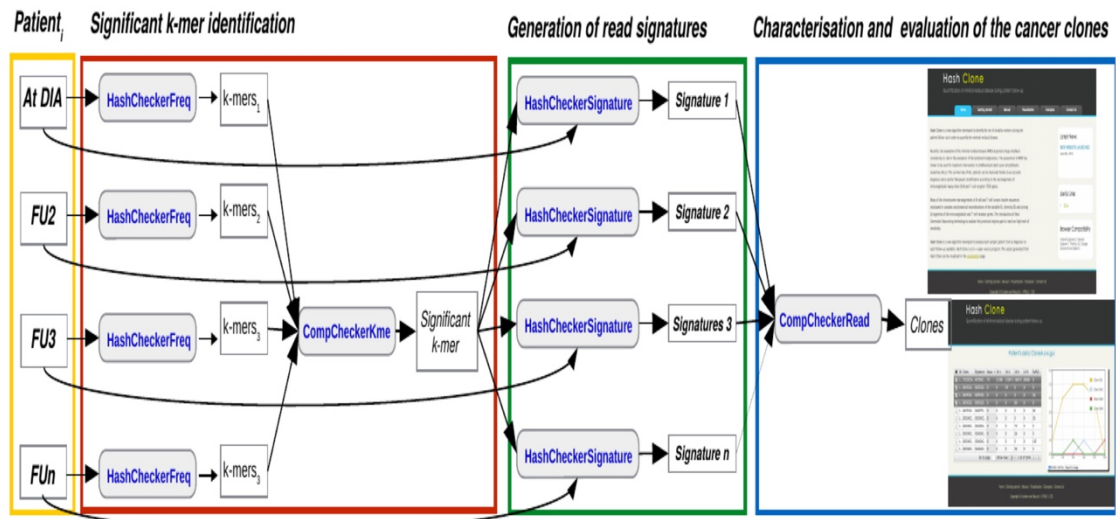
315

316

317

Figure 4

318



319

320

321

Figure 5