BIOINFORMATICS APPLICATIONS NOTE Vol. 30 no. 15 2014, pages 2204–2205
doi:10.1093/bioinformatics/btu185

Genome analysis Advance Access publication April 9, 2014

Rgb: a scriptable genome browser for R

Sylvain Mareschal[1,2,3,*], Sydney Dubois[1,2,3], Thierry Lecroq[2,3,4] and Fabrice Jardin[1,2,3,*]
[1]Centre Henri Becquerel, INSERM UMR 918, 76038 Rouen Cedex 1, France, [2]Normandy University, University of Rouen, 76821 Mont-Saint-Aignan, France, [3]Institute for Research and Innovation in Biomedicine (IRIB), Haute-Normandie, 76183 Rouen Cedex, France and [4]LITIS, INSA EA 4108, 76801 Saint-Etienne-du-Rouvray, France

Associate Editor: Inanc Birol

ABSTRACT

Summary: Thanks to its free licensing and the development of initiatives like Bioconductor, R has become an essential part of the bioinformatics toolbox in the past years and is more and more confronted with genomically located data. While separate solutions are available to manipulate and visualize such data, no R package currently offers the efficiency required for computationally intensive tasks such as interactive genome browsing. The package proposed here fulfills this specific need, providing a multilevel interface suitable for most needs, from a completely interfaced genome browser to low-level classes and methods. Its time and memory efficiency have been challenged in a human dataset, where it outperformed existing solutions by several orders of magnitude.

Availability and implementation: R sources and packages are freely available at the CRAN repository and dedicated Web site: http://bioinformatics.ovsa.fr/Rgb. Distributed under the GPL 3 license, compatible with most operating systems (Windows, Linux, Mac OS) and architectures.

Contact: maressyl@gmail.com or fabrice.jardin@chb.unicancer.fr

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on December 20, 2013; revised on March 3, 2014; accepted on April 2, 2014

1 INTRODUCTION

The growing demand from the biology community for statistically robust approaches has made the R statistics-oriented scripting language an essential part of the bioinformatics toolbox. Its graphical capabilities make it a valuable tool to produce publication-grade complex figures, whereas its computational efficiency allows it to handle huge datasets, as currently required in fields like transcriptomics or next-generation sequencing. These qualities come with an open-source licensing and various operating system ports that make it available virtually everywhere. Thanks to the Bioconductor initiative (Gentleman *et al.*, 2004), large amount of software is freely available as R packages for tasks as diverse as microarray processing (Smyth, 2005; Venkatraman and Olshen, 2007), feature annotation (Zhang *et al.*, 2003) or sequence analysis (Anders and Huber, 2010).

Much of this software generates genomic data, i.e. lists of chromosome regions defined by starting and ending coordinates. Such data are usually subset using chromosomal coordinates rather than row indexes, a paradigm R was not developed to deal with. Bioconductor historically addressed this issue with the *RangedData* class from the IRanges package, handling genomic regions as ranges of integers (base positions). Its flexibility and efficiency were extended a few years later by the GenomicRanges package, making direct use of IRanges components for the subsetting. For visualization, Bioconductor provides two solutions: rtracklayer and GViz. The former sends data to visualize to the UCSC web genome browser (Kent *et al.*, 2002), a model that implies frequent comings and goings between programs and consequent network burden. The latter is more integrated within Bioconductor and produces static graphics from the classes described above, in delays incompatible with user interactivity and intensive computing.

The package described here reconciles these two aspects in a coherent way, thus offering an interactive interface responsive enough for comfortable browsing and atomic operations suitable for computer-intensive algorithms.

2 IMPLEMENTATION

Rgb is implemented as an R package, providing various classes and methods for scripts. It makes use of the Reference class system, which offers an object-oriented framework similar to what can be found in other languages such as Java or C++. It is flexible enough to suit the needs of the four categories of R users:

- **R beginners** will find in Rgb a complete graphical user interface (GUI) allowing them to convert and visualize genomic data without any command line. Stand-alone builds of Rgb can even make the R dependency totally transparent.
- **Script writers**, needing an exclusive Command Line Interface to automate analysis, will find in Rgb classes and methods able to handle their genomic data and produce high-quality graphics from them.
- **Console users** who use R as an exploratory tool without a special need for reproducibility, will find Rgb's ability to mix the two interfaces particularly time saving.
- **Package developers** may be interested in extending classes to handle new data storage modes or representations, a process greatly facilitated by the object-oriented design of Rgb.

The core of the subsetting system consists of a collection of C functions making direct use of R libraries, and the graphical interface relies on Tcl-tk. Both of them are natively managed

*To whom correspondence should be addressed.

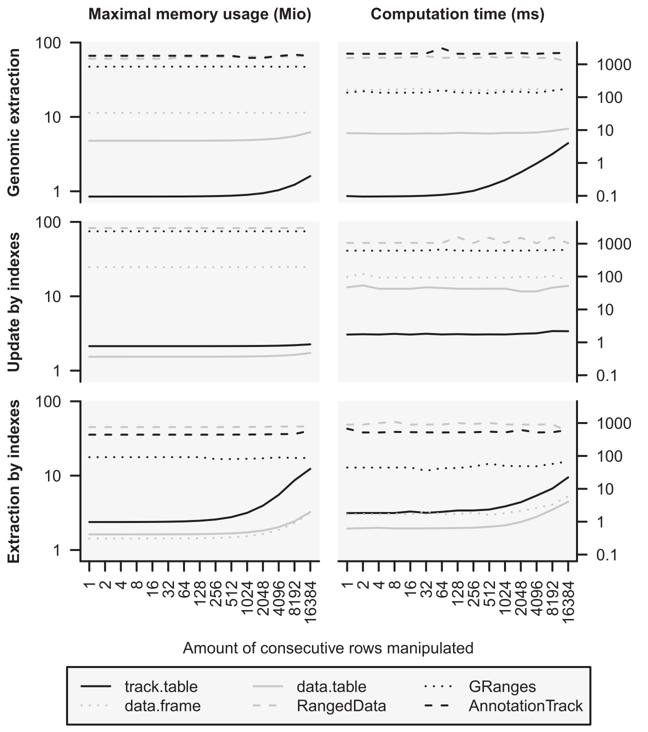

Fig. 1. Performance comparison of Rgb's *track.table* class with existing software. Three atomic tasks were benchmarked on a common dataset, in terms of maximal memory usage and computing time, as recorded by R

by R, and have been successfully compiled and tested on Windows and Linux operating systems. BAM file support can be added by the Rsamtools package, which is part of Bioconductor.

3 PERFORMANCES

To assert Rgb's *track.table* class suitability for computer-intensive tasks such as responsive genome browsing or overlap computation, its time and maximal memory consumption on usual atomic tasks were compared with existing solutions: standard R *data.frame*, the more efficient *data.table*, IRanges' *RangedData*, GenomicRanges' *GRanges* and GViz's *AnnotationTrack*. Three tasks were monitored as the most common atomic operations in such data: genomic extraction by chromosomal coordinates, extraction and modification of consecutive rows by indexes. The 291 128 exons recorded in the CCDS database (Pruitt *et al.*, 2009) for the human genome were used as a common dataset. Benchmarking was performed on a mid-range desktop computer (3.1 GHz Intel i3-2100 processor with 8 GB RAM) running R 3.0.2 in a 64-bit Fedora 18 distribution. Each measure was made with R functions *proc.time* and *gc* in a fresh session, to normalize garbage collection effects (scripts and dataset available as Supplementary Data).

With the genomic extraction task (Fig. 1), GViz and *RangedData* proved their unsuitability to intensive tasks, with computing times nearing the second for a single extraction (typical genome browser representations contain several tracks to be subset), and extravagant memory usage (hundreds of Mio for a dataset of 30 Mio). *GRanges* and generic R solutions perform better, but are still outperformed by Rgb by a magnitude of 50–1500 on small genomic extractions.

Similar results can be observed with the modification task, with Rgb outperforming R generic classes by a factor of 100 in time consumption and 10 in memory usage. It still outperforms bioinformatic solutions with the last task, but is overtaken here by generic solutions such as *data.frame* and *data.table*. However, as this kind of extraction is far less justified in a genomic context than the previous one, this average performance can be accepted.

4 CONCLUSION

Rgb provides several entry points to a consistent genome browsing system, which may prove equally useful to users looking for an interactive genome browser able to handle their R datasets and developers needing efficient genomic subsetting capabilities for their scripts. Its object-oriented paradigm and open-source licensing make it easily extendable, and its performances over existing solutions have been proven in a real genomic dataset.

ACKNOWLEDGEMENTS

The authors thank Christian Bastard for his helpful comments and feedback during Rgb development and the many CGH-array datasets he provided for testing.

Funding: This work was supported by a PhD grant from the Région Haute-Normandie (France).

Conflict of Interest: none declared.

REFERENCES

Anders,S. and Huber,W. (2010) Differential expression analysis for sequence count data. *Genome Biol.*, **11**, R106.
Gentleman,R.C. *et al.* (2004) Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol.*, **5**, R80.
Kent,W.J. *et al.* (2002) The human genome browser at UCSC. *Genome Res.*, **12**, 996–1006.
Pruitt,K.D. *et al.* (2009) The consensus coding sequence (CCDS) project: identifying a common protein-coding gene set for the human and mouse genomes. *Genome Res.*, **19**, 1316–1323.
Smyth,G.K. (2005) limma: linear models for microarray data. In: Gentleman,R. *et al.* (ed.) *Bioinformatics and Computational Biology Solutions Using R and Bioconductor, Statistics for Biology and Health*. Springer, New York, NY, pp. 397–420.
Venkatraman,E.S. and Olshen,A.B. (2007) A faster circular binary segmentation algorithm for the analysis of array CGH data. *Bioinformatics*, **23**, 657–663.
Zhang,J. *et al.* (2003) An extensible application for assembling annotation for genomic data. *Bioinformatis*, **19**, 155.