|  |
| --- |
| Genome analysis  BioR: a Rapid and Flexible System for Genomics Annotation  Jean-Pierre A. Kocher1,#,\*, Daniel J. Quest1,#, Patrick Duffy, Michael A. Meiners, Raymond M. Moore, David Rider, Asif Hossain, Steven N. Hart and Valentin Dinu3  1Division of Biomedical Statistics and Informatics, Department of Health Sciences Research, Mayo Clinic, 200 First Street SW, Rochester, MN 55905, USA.  2Department of Research IT, Mayo Clinic, 200 Frist Street SW, Rochester, MN 55905, USA.  3Department of Biomedical Informatics, Arizona State University, 13212 East Shea Boulevard, Scottsdale, AZ 85259, USA.  Received on XXXXX; revised on XXXXX; accepted on XXXXX  Associate Editor: XXXXXXX |

[[1]](#footnote-2)\*abstract

**Motivation:** Variant annotation is a necessary step to facilitate interpretation of genomics results. Currently available standalone annotation tools cannot be easily updated and extended without the involvement of the authors. Here, we present BioR – a toolkit for annotating variants from both public and user-specified annotation sources stored into catalogs. The BioR toolkit provides the functionality to combine and retrieve annotation from catalogs via the command line interface. BioR Toolkit commands can be combined with other UNIX commands for advanced annotation processing and downstream analysis. We provide several catalogs from commonly used annotation sources and instructions for creating user-specific catalogs, giving users flexibility in defining their own annotation pipelines.

**Availability and Implementation:** BioR and documentation are available at <http://bioinformaticstools.mayo.edu>. The package is implemented in Java, and makes use of external tools written in Java and Perl. The toolkit can be executed on Mac OS X 10.5 and above or any Linux distribution.

**Supplementary information:** Supplementary materials are available at <http://bioinformaticstools.mayo.edu/> and at *Bioinformatics* online.

# introduction

Next Generation Sequencing (NGS) technology platforms are providing unprecedented opportunities to study genomic variants that are associated with clinical conditions and drug response. Using NGS technologies, researchers can identify mutations associated with rare diseases, characterize somatic variants in tumor for diagnostic or prognostic purpose, or guide therapeutic treatment. Although the large amount of data produced by NGS platforms and the time to process them is largely being addressed by expanding the IT infrastructure, high performance computing (HPC) and code optimization, the annotation process needed to interpret the thousands of variants found in individual genomes is still a challenging task. The annotation process requires extracting and combining information from disparate external and in-house annotation sources, or even command-line tools. Several applications such as ANNOVAR (Wang, K. *et al*. 2010) and TREAT (Asmann, Y.W. *et al*. 2012) have recently been developed to automate the annotation of genomics variants. However, these systems are restrictive since expansion and maintenance of the annotations depend upon the authors’ availability or willingness. Other approaches such as Bio2RDF (Belleau, F. *et al*. 2008) propose the conversion of annotation sources into RDF format that can be loaded into a triple store database for querying. This approach, although flexible because it allows independent integration of new annotation sources, presents scalability limitations. Under production loads, the number of searches to annotate variants can become extremely large. For instance, the annotation of about 30 million variants from 10 whole genome sequencing runs per day, with annotation extracted from 10 data sources would involve more than 300 million queries. Bio2RDF/triple stores also do not support the seamless integration with UNIX command-line tools that need to be invoked during the annotation process.

In this manuscript, we present BioR, a flexible and scalable infrastructure for the specific purposes of gene and variant annotation. BioR is built around a slightly modified version of the JSON format (<http://www.json.org/>), referred to as TJSON. TJSON is a compact, readable and hierarchical format that can be used to store one to many relationships present in relational annotation sources. To facilitate usability, BioR provides a toolkit (BioR Toolkit) that includes a set of UNIX command-line functions to facilitate catalog management and annotation extraction. The BioR Toolkit is engineered to work in HPC environments and scale to multiple simultaneous instances.

# methods and ResultS

## TJSON format

The TJSON format is used by catalogs and used as standard input/output for most of the functions of the BioR Toolkit. TJSON consists of a mix of tab delimited values and JSON strings (see example below). The JSON format was preferred over others like XML since in addition to being readable, it is relatively compact. Like XML, it can represent complex hierarchical data structures into a single text string. The hierarchical structures existing in relational data sources are therefore maintained in BioR catalogs. JSON strings can easily be extracted from a TJSON to be processed with functions from a JSON library accessible through most programing languages like Perl, C, Java, Python, and Ruby. These functions provide the necessary features to retrieve nested values from JSON strings. An example of the TJSON format, where “\t” is a tab character (typically non-displaying) acting as a column separator is here:

1024\t145.6\t{"\_type":"gene","\_strand":"+","\_minBP":10954,"\_maxBP":11507,"note":"similarity to: 1 Protein", "GeneID":"100506145"}\t12.334

## BioR Toolkit

The BioR toolkit includes set of commands for the management of catalogs, extraction of annotation based on genomics coordinates, variant or gene information. These standalone commands, that are executed like common UNIX commands, leverage 3rd party JSON libraries to process JSON strings. TJSON is intentionally used as standard input/output by most of the BioR commands to enable the concatenation of multiple BioR commands into a single UNIX command using standard piping syntax. Note the user can add functions to the toolkit or operate on their data using conventional UNIX tools as long as the function operates on TJSON strings.

The BioR Toolkit also includes commands to convert tab delimited input file into TJSON strings (such as VCF and BED files) or convert TJSON into tab delimited output file. Any metadata recorded in VCF or GFF style header (starting with “#”) in the input file will be carried through by the BioR toolkit functions to be recorded in the output file. The commands included in the BioR Toolkit are listed in Table S1.

Finally, the BioR Toolkit supports two command line utilities for annotating variants: 1) bior\_snpeff which integrates SnpEff annotations (Cingolani *et al.* 2012) and 2) bior\_vep to annotate files using Ensemble’s variant effect predictor (www.ensembl.org/info/docs/variation/vep/).

## BioR Catalogs

BioR catalogs are flat files in a fixed TJSON format containing the annotation information and corresponding genomic coordinates. The first tab-delimited field is used to store the origin of the sequence (usually a chromosome). The next two fields record the start and end coordinates of a genomic interval for position-dependent annotations. These two fields are otherwise set to 0. The last field is a JSON string that contains all the data from the original source.

To reduce storage footprint and accelerate coordinate based searches, catalogs are compressed using the open source BGZip (Danecek, P. *et al*. 2011) and indexed using Tabix (Li, H. 2011). The Tabix index file is stored in the same directory as the related catalog. BioR Toolkit takes advantage of the Tabix library to perform coordinate-based overlap searches. BioR can also perform searches on identifiers that can be indexed using a BioR toolkit command for fast querying. Finally, to accelerate coordinate-base and variant matching searches, a set of semantically consistent identifiers called ‘Golden Identifiers’ automatically index and are implicitly used by some of BioR commands (Table S2).

## Building BioR Catalogs

The complexity of creating BioR catalogs depends on the organization of data in the annotation source. Commonly, the data can be extracted in a tab delimited text format that can be readily converted to a BioR catalog using a provided command. When annotations are extracted from complex structures such as relational databases, programming is required to reformat the data, and BioR catalogs must be created for each relationship the user wants to use. This data has to be parsed and formatted to comply with BioR catalog format including Golden Identifier’s nomenclature to enable searching.

## BioR Catalog Library

BioR includes 19 highly vetted and documented catalogs from the most common data sources (see Table S3) and an extensive list of catalogs built from UCSC Genome Browser tracks (Kent, W.J. *et al*. 2002). To increase clinical applicability, pharmacogenomics catalogs that integrate annotations from PharmGKB, DrugBank, and Therapeutic Target Database is provided.

## Example

The following example illustrates how sample variant rsIDs stored in the file rsID.txt can be annotated with European frequency from the 1000 Genomes Project. First, using the “bior\_lookup” command, rsIDs in the rsID.txt file are matched to entries in the dbSNP.tsv.bgz catalog containing the identifier ‘ID’. Matching entries in JSON format are piped to the function “bior\_same\_variant”. This function uses the Gold Identifiers present in the JSON string to look-up allele frequencies in the KGenomes.tsv.gz catalog. Finally, the function “bior\_drill” and the Unix command “cut” reformat the TJSON string into a tab-delimited output.

$ cat rsIDs.txt |

bior\_lookup -p ID –d dbSNP.tsv.bgz |

bior\_same\_variant -d KGenomes.tsv.gz |

bior\_drill -c -1 -p INFO.EUR\_AF |

cut -f 1,3

This macro annotates 100,000 rsIDs in 2:23 minutes on a MacBook Pro 2.3 GHz Intel Core i7 with solid state drive and 8G RAM.

# results

BioR is an open and flexible gene and variant annotation tool. BioR includes a toolkit with a base set of commands needed to build and index catalogs as well as retrieve annotations. Annotations can be retrieved based on location (genomic coordinates) or identifiers. The TJSON format is used for catalogs and as input/output for most of the toolkit functions facilitating the assembly of complex pipelines. Since the TJSON format is readable users can design their own scripts to extract annotation from catalogs. Scripts can also be intermixed with toolkit commands as long as the TSJON format is maintained. This stream-based approach on which BioR is based significantly reduces memory footprint. In addition, the BioR toolkit is inherently parallel and can be configured to take advantage of computers with multi-core architectures. BioR catalogs can easily be combined into new catalogs to decrease retrieval time by avoiding multiple cross-catalog queries. In conclusion, BioR is a rapid and flexible system for annotating high-throughput genomics experiments.

acknowledgements

*Funding*: The authors thank the Center for Individualized Medicine at Mayo Clinic for funding the development of BioR.

References

Wang, K. *et al*. (2010) ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data. Nucleic acids research, **38**(16): p. e164-e164.

Asmann, Y.W. *et al*. (2012) TREAT: a bioinformatics tool for variant annotations and visualizations in targeted and exome sequencing data. Bioinformatics, **28**(2): p. 277-278.

Belleau, F. *et al*. (2008) Bio2RDF: towards a mashup to build bioinformatics knowledge systems. Journal of biomedical informatics, **41**(5): p. 706-716.

Cingolani, P. *et al.* (2012) A program for annotating and predicting the effects of single nucleotide polymorphisms, SnpEff: SNPs in the genome of Drosophila melanogaster strain w1118; iso-2; iso-3. Fly (Austin). **6**(2) :p. 80-92.

Danecek, P. *et al*. (2011) The variant call format and VCFtools. Bioinformatics, **27**(15): p. 2156-2158.

Li, H. (2011) Tabix: fast retrieval of sequence features from generic TAB-delimited files. Bioinformatics, **27**(5): p. 718-719.

Kent, W.J. *et al*. (2002) The human genome browser at UCSC. Genome Research, **12**(6): p. 996-1006.

1. \*To whom correspondence should be addressed.

   #: these two authors contributed equally [↑](#footnote-ref-2)