

SOFTWARE

Open Access



PyIR: a scalable wrapper for processing billions of immunoglobulin and T cell receptor sequences using IgBLAST

Cinque Soto^{1,2}, Jessica A. Finn³, Jordan R. Willis¹, Samuel B. Day¹, Robert S. Sinkovits⁴, Taylor Jones¹, Samuel Schmitz⁵, Jens Meiler⁵, Andre Branchizio¹ and James E. Crowe Jr^{1,2,3*} 

* Correspondence: james.crowe@vumc.org

¹Vanderbilt Vaccine Center, Vanderbilt University Medical Center, Nashville, TN 37232, USA

²Department of Pediatrics, Vanderbilt University Medical Center, Nashville, TN 37232, USA
Full list of author information is available at the end of the article

Abstract

Background: Recent advances in DNA sequencing technologies have enabled significant leaps in capacity to generate large volumes of DNA sequence data, which has spurred a rapid growth in the use of bioinformatics as a means of interrogating antibody variable gene repertoires. Common tools used for annotation of antibody sequences are often limited in functionality, modularity and usability.

Results: We have developed PyIR, a Python wrapper and library for IgBLAST, which offers a minimal setup CLI and API, FASTQ support, file chunking for large sequence files, JSON and Python dictionary output, and built-in sequence filtering.

Conclusions: PyIR offers improved processing speed over multithreaded IgBLAST (version 1.14) when spawning more than 16 processes on a single computer system. Its customizable filtering and data encapsulation allow it to be adapted to a wide range of computing environments. The API allows for IgBLAST to be used in customized bioinformatics workflows.

Keywords: Immune repertoires, Antibody, Illumina, CDR3, IgBLAST

Background

The diverse population of rearranged immunoglobulin and T cell receptor (TCR) variable gene sequences within an individual is referred to as their adaptive immune repertoire and is responsible for recognition and neutralization of a potentially unlimited number of pathogenic targets. Next-generation sequencing (NGS) technology has become the ideal method for probing diversity in adaptive immune repertoires [1–5]. With continued advances in NGS technology yielding more sequence reads in less time and at lower cost, the number of researchers interested in analyzing adaptive immune repertoires using different bioinformatics processing pipelines continues to grow [6]. Thus, the demand for efficient and easy-to-use bioinformatics tools for processing and analyzing NGS data from immune repertoire sequencing has never been higher.



© The Author(s). 2020 **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>. The Creative Commons Public Domain Dedication waiver (<http://creativecommons.org/publicdomain/zero/1.0/>) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

Two popular programs for processing immune repertoire sequencing data are IgBLAST [7] and IMGT/HighV-QUEST [8]. IgBLAST is used by several bioinformatic pipelines [9–12]. IMGT/HighV-QUEST is a web utility and is part of the international ImMunoGeneTics (IMGT) information system [13], which provides several web-based tools for immunogenetic analysis. Both tools use template-based nucleotide alignments between curated sets of germline genes and sequencing reads to infer the most plausible recombined germline gene sequences encoding the antibody and to delineate the complementarity-determining regions (CDRs) of the immunoglobulin or T cell receptor (TCR) sequence.

IgBLAST is derived from the well-known BLAST [14] family of sequence alignment and search tools and is available as both a web-based service and a downloadable executable. While the web-based service for IgBLAST is limited by the number of sequences that can be processed at one time, the executable could in principle be used to process millions of sequences. However, the exact number of sequences that can be processed is limited by the capacity of the computer hardware used for the task. This limitation is usually not an issue for high-performance computing clusters, but for more conventional workstations, this factor can introduce a barrier to processing data sets with millions of sequences efficiently.

IMGT/HighV-QUEST provides a web-based interface only and accepts a maximum of up to 500,000 sequences per submission. The submission size, although large, can limit the study of very large adaptive immune repertoire sequencing datasets, which are increasingly common. For those cases in which immune repertoire sequences are processed using this tool, results are made available for download in a tab-separated value (TSV) format. IMGT/HighV-QUEST distinguishes itself from all the other adaptive immune repertoire sequencing tools by providing a wealth of information about each processed sequence. However, when processing very large datasets such as sequence sets produced in a single run on current generation NovaSEQ instruments (Illumina), analysis using the IMGT/HighV-QUEST web-portal becomes impractical.

To address the need for processing very large data sets containing immunoglobulin or TCR sequences, we developed a software tool that we call PyIR. The software is a minimally dependent Python3 wrapper and library for IgBLAST [7] and can scale to process up to 1 billion sequences. Its basic functionality splits the input FASTX (FASTQ or FASTA file) file and performs batch execution on chunks of sequences. This approach avoids having to read all data in memory at once, allowing efficient processing of very large data sets on modest size workstations with multiple CPUs. PyIR parses all of the IgBLAST-generated fields from the web-based file format into fields that comply with Adaptive Immune Receptor Repertoire (AIRR) recommendations [15] and then outputs the results into a JSON file format. PyIR also provides options for sequence quality filtering that can be invoked from the command line. These sequence filters allow the user to remove poor quality data after IgBLAST processing. We also have included an application programming interface (API) with PyIR that allows users to incorporate this tool directly into their own Python scripts. PyIR will be a useful computational tool for the AIRR community [15] by allowing those researchers with minimal Python programming experience to easily interface with IgBLAST.

Implementation

PyIR is available for Linux or OSX (Darwin) environments running Python 3.6 or higher. Installation is managed by the Python pip3 module, which also installs PyIR dependencies as needed. PyIR ships with IgBLAST (version 1.14) and users implement the software with their own executable and germline gene databases. We provide current germline genes from IMGT in this distribution, but we recommend that users download the most up-to-date version of the germline genes from IMGT at: <http://www.imgt.org>.

PyIR will run on machines with modest processing power and memory. For example, we have used a MacBook Pro with a 2.7GHz 2 core/4 thread i5 CPU and 8GB RAM (Apple) for this type of analysis. However, some consideration must be made for disk space, since PyIR stores runtime data as temporary files. Output files generated when running PyIR using standard options are expected to be 4 to 5 times larger than the input FASTX files.

Results and discussion

PyIR processing time

The critical enhancement in PyIR versus multithreaded IgBLAST (version 1.14) is the addition of file chunking. At the beginning of execution, PyIR estimates the number of input sequences based on the FASTX file size and then splits the file into 'chunks' that are stored as temporary files on disk. The default PyIR behavior is to balance the number of sequences per chunk with the number of CPU threads available and the size of the input file, but the user also can manually override this option to set a specified chunk size. PyIR then distributes chunks continuously to available processors for IgBLAST to process.

To test the processing efficiency of PyIR, we processed 1 million synthetic immunoglobulin sequences [1] using PyIR and multithreaded IgBLAST (version 1.14) on a workstation with 4 hyperthreaded 8-core Opteron 6278 processors. Both PyIR and multithreaded IgBLAST (version 1.14) showed linear scaling out to 16 processes (Fig. 1a). However, when more than 16 processes are spawned only PyIR maintains this linear scaling out to 64 processes (Fig. 1a). We also looked at speedup, defined here as the performance gained by spawning additional processes. We computed speedup by normalizing the time in hours to process 1 million (or 1 billion) sequences by the time it takes to process 1 million (or 1 billion) sequences using two processes. Both PyIR and multithreaded IgBLAST (version 1.14) show linear speedups out to 16 processes, but when more than 16 processes were spawned, IgBLAST did not show speedup (Fig. 1b). Thus, PyIR maintains linear scaling beyond 16 processes using all of the threads available from the hardware.

We also tested the ability of PyIR to process very large data sets (i.e., cDNA sequence data sets from analysis on a NovaSEQ instrument) using file chunking. We ran a separate set of trials on a workstation with 4 hyperthreaded 28-core Xeon Platinum 8280 processors, clocking the total time required to analyze a series of FASTA files containing from 1 thousand to 1 billion synthetic immunoglobulin sequences. Multithreaded IgBLAST (version 1.14) failed at around 70 million sequences, while PyIR ran to completion (Fig. 1c). These results demonstrate the

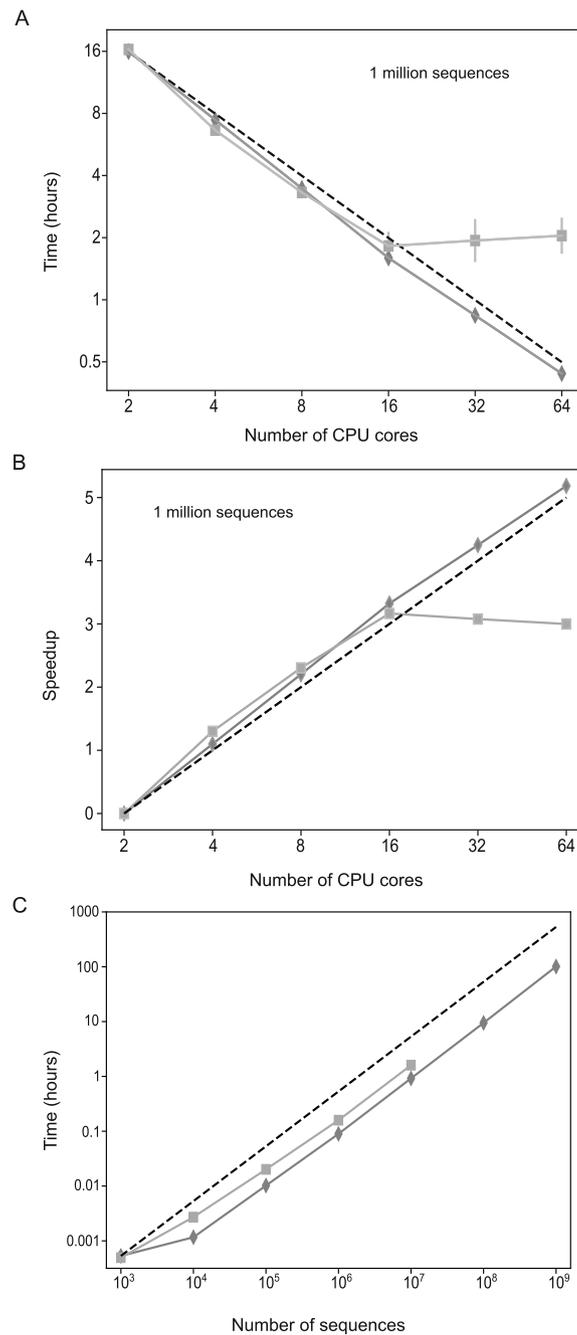


Fig. 1 Multiprocessing performance for PyIR and multithreaded IgBLAST (version 1.14). **a** One million synthetic immunoglobulin sequences were used to time PyIR (dark grey, \blacklozenge) against multithreaded IgBLAST (version 1.14) (grey, \blacksquare) as a function of the number of processes. Idealized timings are shown as a black dashed line. Average timings were measured over the three trial runs for 1 million sequences and computed separately for both IgBLAST and PyIR. Standard deviations appear as error bars for both methods. X and Y axes are in \log_2 space. **b** The speedup of PyIR relative to multithreaded IgBLAST (version 1.14) as a function of the number of simultaneous processes. Timings were done on a workstation equipped with 4 Opteron 6278 hyper-threaded 8-core processors for a total of 64 CPU threads using the average timings from (a). The X and Y axes are in \log_2 space. **c** One billion synthetic immunoglobulin sequences were used to determine the speedup PyIR achieved over multithreaded IgBLAST (version 1.14) as a function of the number of sequences. Idealized speedups are shown as a black dashed line. Timings were done on a workstation equipped 4 Xeon Platinum 8280 hyperthreaded 28-core processors for a total of 224 CPU threads. X and Y axes are in \log_{10} space

efficiency of PyIR and the underlying file chunking method to process very large data sets.

PyIR also contains options for sequence filtering, allowing users to remove sequences with stop codons or those that are out-of-frame according to IgBLAST. For example, we typically filter out sequences that: 1) fail to meet our minimum V and J gene assignment expectation value (E-value) threshold of 1.0E-6, 2) contain a stop codon in the translated nucleotide sequence, 3) are out-of-frame, or 4) lack a complementarity-determining region 3 (CDR3) region. These sequence filters can be enabled or disabled individually and any non-boolean filters (i.e., E-value filter) can be set directly by the user. We also have included a Phred score filter that can be applied to the CDR3 region of a sequence. This filter removes any sequence with a nucleotide in the CDR3 region with a Phred score below some user-defined threshold. The filtering options can be extended by users directly within PyIR or through the application programming interface (API).

PyIR application programming interface (API)

The API interface of PyIR can be used to call PyIR functions from within a Python script. This feature allows users to extend the functionality of PyIR or to use PyIR within a user's own adaptive immune repertoire processing pipeline. Several published methods for processing immune repertoire data lack an accessible API, which is limiting for some experimental immunologists who may be new to scripting in Python. An example of how one would incorporate PyIR within a Python script is given directly below:

```
from pyir import PyIR  
FILE = 'test.fasta'  
pyir = PyIR (query=FILE)  
result = pyir.run()  
print (result)
```

In this example, PyIR is imported directly into Python and used to process a FASTA formatted file called *test.fasta*. The Python object in the variable *result* is a Python dictionary containing all the parsed fields from the web-based output of IgBLAST. We provide five examples showing how one could make use of PyIR directly within a Python script (see <http://github.com/crowelab/PyIR>). The fifth example shows how one could use the PyIR API to generate a histogram of CDR3 lengths (see Additional file 1).

PyIR output

After processing and sequence filtering, PyIR can return a zipped JSON file, a tab separated value file (TSV) or a Python dictionary (if PyIR is used as an API). We do note that the JSON output file can be large since PyIR stores the parsed results from the three best alignments. The user has the option of storing only the single best alignment, which reduces the size of the JSON file. Our primary focus for using JSON as the preferred output format was to allow for easy insertion into a MongoDB database. Several recent studies have been published that contain large adaptive immune repertoire sequencing datasets [1, 2, 16]. Facilitating the ability to process and store these data sets

locally into an industry standard database such as MongoDB or MariaDB motivated use of the JSON format.

Conclusion

PyIR was designed to extend the functionality of IgBLAST to allow for processing of very large datasets (> 100 million antibody or TCR recombined variable gene sequences). In terms of processing efficiency, we found that PyIR scaled linearly with the number of processes out to 1 billion sequences. Multithreaded IgBLAST (version 1.14) also scaled linearly but failed at around 70 million sequences. Our benchmarks suggest that PyIR can process about 2 million sequences per hour on a modest 64-core server and can process roughly 10 million sequences per hour on a 112-core machine that sits on the premium end of enterprise hardware. The API provides novice Python programmers with the ability to interface directly with IgBLAST and to explore using this tool in their own bioinformatics workflows. We do note that PyIR is not the only method for processing immune repertoire sequencing [17]. However, PyIR is easily adaptable and uses IgBLAST which has been extensively benchmarked against other methods [18]. We expect that PyIR will find use among the Adaptive Immune Receptor Repertoire (AIRR) Community.

Availability and requirements

Project name: PyIR

Project homepage: <http://github.com/crowelab/PyIR>

Archived version: <https://doi.org/10.5281/zenodo.3862746>

Operating systems: Linux, UNIX, OSX (Darwin)

Programming languages: Python

Other requirements: None

License: Free to academics

Any restrictions to use by non-academics: Yes; non academics should contact the author for permission to use the software or license options for incorporation into software that is being sold for profit.

Supplementary information

Supplementary information accompanies this paper at <https://doi.org/10.1186/s12859-020-03649-5>.

Additional file 1. Generating CDR3 length distributions with the PyIR API. Synthetic sequence data from Briney et al., was used to demonstrate the use of PyIRs API in generating a CDR3 length distribution.

Abbreviations

NGS: Next-generation sequencing; CDR3: Complementarity-determining region 3; JSON: JavaScript Object Notation; API: Application Programming Interface; CLI: Command-line interface; FASTA: File format for representing nucleotide sequence; FASTQ: File format for storing nucleotide sequence and quality scores

Acknowledgements

The authors thank Robin G. Bombardi, M. Luke Myers, Gopal Sapparapu and Andrew I. Flyak for helpful discussions and input on design.

Authors' contributions

JAF, JRW, AB and JEC conceived of the idea, AB, SBD, TJ, SS and JAF developed the package, and SBD and RSS tested and benchmarked the package and CS, SBD, JM and JEC wrote the paper. All authors edited the paper and approved the final version.

Funding

This work was supported by grants and contracts from the National Institutes of Health [the grant U19 AI117905 and the contract HHSN272201400024C] and a grant from the Human Vaccines Project. The NIH and HVP funding bodies had no role in the design of the study; collection, analysis, and interpretation of data; or in writing the manuscript.

Availability of data and materials

The datasets and source code supporting the conclusions of this article are available in the GitHub repository, <http://github.com/crowelab/PyIR>.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare they have no competing interests.

Author details

¹Vanderbilt Vaccine Center, Vanderbilt University Medical Center, Nashville, TN 37232, USA. ²Department of Pediatrics, Vanderbilt University Medical Center, Nashville, TN 37232, USA. ³Department of Pathology, Microbiology, and Immunology, Vanderbilt University, Nashville, TN 37232, USA. ⁴San Diego Supercomputer Center, University of California, San Diego, La Jolla, CA 92093, USA. ⁵Department of Chemistry, Vanderbilt University, Nashville, TN 37212, USA.

Received: 23 February 2020 Accepted: 9 July 2020

Published online: 16 July 2020

References

- Soto C, Bombardi RG, Branchizio A, Kose N, Matta P, Sevy AM, Sinkovits RS, Gilchuk P, Finn JA, Crowe JE Jr. High frequency of shared clonotypes in human B cell receptor repertoires. *Nature*. 2019;566(7744):398–402.
- Briney B, Inderbitzin A, Joyce C, Burton DR. Commonality despite exceptional diversity in the baseline human antibody repertoire. *Nature*. 2019;566(7744):393–7.
- Weinstein JA, Jiang N, White RA 3rd, Fisher DS, Quake SR. High-throughput sequencing of the zebrafish antibody repertoire. *Science*. 2009;324(5928):807–10.
- Briney BS, Willis JR, Crowe JE Jr. Location and length distribution of somatic hypermutation-associated DNA insertions and deletions reveals regions of antibody structural plasticity. *Genes Immun*. 2012;13(7):523–9.
- Zhu J, Ofek G, Yang Y, Zhang B, Louder MK, Lu G, McKee K, Pancera M, Skinner J, Zhang Z, et al. Mining the antibodyome for HIV-1-neutralizing antibodies with next-generation sequencing and phylogenetic pairing of heavy/light chains. *Proc Natl Acad Sci U S A*. 2013;110(16):6470–5.
- Smakaj E, Babrak L, Ohlin M, Shugay M, Briney B, Tosoni D, Galli C, Grobelsek V, D'Angelo I, Olson B, et al. Benchmarking immunoinformatic tools for the analysis of antibody repertoire sequences. *Bioinformatics*. 2020;36(6):1731–39.
- Ye J, Ma N, Madden TL, Ostell JM. IgBLAST: an immunoglobulin variable domain sequence analysis tool. *Nucleic Acids Res*. 2013;41(Web Server issue):W34–40.
- Alamyar E, Duroux P, Lefranc MP, Giudicelli V. IMGT((R)) tools for the nucleotide analysis of immunoglobulin (IG) and T cell receptor (TR) V-(D)-J repertoires, polymorphisms, and IG mutations: IMGT/V-QUEST and IMGT/HighV-QUEST for NGS. *Methods Mol Biol*. 2012;882:569–604.
- Briney B, Le K, Zhu J, Burton DR. Clonify: unseeded antibody lineage assignment from next-generation sequencing data. *Sci Rep*. 2016;6:23901.
- Vander Heiden JA, Yaari G, Uduman M, Stern JN, O'Connor KC, Hafler DA, Vigneault F, Kleinstein SH. pRESTO: a toolkit for processing high-throughput sequencing raw reads of lymphocyte receptor repertoires. *Bioinformatics*. 2014;30(13):1930–2.
- Kwong PD, Chuang GY, DeKosky BJ, Gindin T, Georgiev IS, Lemmin T, Schramm CA, Sheng Z, Soto C, Yang AS, et al. Antibodyomics: bioinformatics technologies for understanding B-cell immunity to HIV-1. *Immunol Rev*. 2017;275(1):108–28.
- Corcoran MM, Phad GE, Vazquez Bernat N, Stahl-Hennig C, Sumida N, Persson MA, Martin M, Karlsson Hedestam GB. Production of individualized V gene databases reveals high levels of immunoglobulin genetic diversity. *Nat Commun*. 2016;7:13642.
- Lefranc MP, Giudicelli V, Duroux P, Jabado-Michaloud J, Folch G, Aouinti S, Carillon E, Duvergey H, Houles A, Paysan-Lafosse T, et al. IMGT((R)), the international ImMunoGeneTics information system((R)) 25 years on. *Nucleic Acids Res*. 2015; 43(Database issue):D413–22.
- Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. *J Mol Biol*. 1990;215(3):403–10.
- Adaptive Immune Receptor Repertoire (AIRR) Community of the Antibody Society. <http://docs.airr-community.org/en/latest/>. Accessed 27 Jan 2020.
- Guo Y, Chen K, Kwong PD, Shapiro L, Sheng Z. cAb-rep: a database of curated antibody repertoires for exploring antibody diversity and predicting antibody prevalence. *Front Immunol*. 2019;10:2365.
- López-Santibáñez-Jácóme L, Eréndira Avendaño-Vázquez S, Flores-Jasso CF. The pipeline repertoire for Ig-Seq analysis. *Front Immunol*. 2019;10:899.
- Smakaj E, Babrak L, Ohlin M, Shugay M, Briney B, Tosoni D, Galli C, Grobelsek V, D'Angelo I, Olson B, et al. Benchmarking immunoinformatic tools for the analysis of antibody repertoire sequences. *Bioinformatics*. 2020;36(6):1731–9.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.