





Peer Community In Mathematical & Computational Biology

An accelerated Vidjil algorithm: up to 30X faster identification of V(D)J recombinations via spaced seeds and Aho-Corasick pattern matching

Giulio Ermanno Pibiri  based on peer reviews by **Sven Rahmann**  and 1
anonymous reviewer

Cyprien Borée, Mathieu Giraud, Mikaël Salson (2024) Alignment-free detection and seed-based identification of multi-loci V(D)J recombinations in Vidjil-algo. HAL, ver. 2, peer-reviewed and recommended by Peer Community in Mathematical and Computational Biology. <https://hal.science/hal-04361907>

Submitted: 01 January 2024, Recommended: 23 July 2024

Cite this recommendation as:

Pibiri, G. (2024) An accelerated Vidjil algorithm: up to 30X faster identification of V(D)J recombinations via spaced seeds and Aho-Corasick pattern matching. *Peer Community in Mathematical and Computational Biology*, 100268. [10.24072/pci.mcb.100268](https://doi.org/10.24072/pci.mcb.100268)

Published: 23 July 2024

Copyright: This work is licensed under the Creative Commons Attribution 4.0 International License. To view a copy of this license, visit <https://creativecommons.org/licenses/by/4.0/>

VDJ recombination is a crucial process in the immune system, where a V (variable) gene, a D (diversity) gene, and a J (joining) gene are randomly combined to create unique antigen receptor genes. This process generates a vast diversity of antibodies and T-cell receptors, essential for recognizing and combating a wide array of pathogens. By identifying and quantifying these VDJ recombinations, we can gain a deeper and more precise understanding of the immune response, enhancing our ability to monitor and manage immune-related conditions.

It is therefore important to develop efficient methods to identify and extract VDJ recombinations from large sequences (e.g., several millions/billions of nucleotides). The work by Borée, Giraud, and Salson [2] contributes one such algorithm. As in previous work, the proposed algorithm employs the Aho-Corasick automaton to simultaneously match several patterns against a string but, differently from other methods, it also combines the efficiency of spaced seeds. Working with seeds rather than the original string has the net benefit of speeding up the algorithm and reducing its memory usage, sometimes at the price of a modest loss in accuracy. Experiments conducted on five different datasets demonstrate that these features grant the proposed method excellent practical performance compared to the best previous methods, like Vidjil [3] (up to 5X faster) and MiXCR [1] (up to 30X faster), with no quality loss.

The method can also be considered an excellent example of a more general trend in scalable algorithmic design: adapt “classic” algorithms (in this case, the Aho-Corasick pattern matching algorithm) to work in sketch space (e.g., the spaced seeds used here), trading accuracy for efficiency. Sometimes, this compromise is necessary for the sake of scaling to very large datasets using modest computing power.

References:

- [1] D. A. Bolotin, S. Poslavsky, I. Mitrophanov, M. Shugay, I. Z. Mamedov, E. V. Putintseva, and D. M. Chudakov (2015). “MiXCR: software for comprehensive adaptive immunity profiling.” *Nature Methods* 12, 380–381. ISSN: 1548-7091. <https://doi.org/10.1038/nmeth.3364>
- [2] C. Borée, M. Giraud, M. Salson (2024) “Alignment-free detection and seed-based identification of multi-loci V(D)J recombinations in Vidjil-algo”. <https://hal.science/hal-04361907v2>, version 2, peer-reviewed and recommended by Peer Community In Mathematical and Computational Biology.
- [3] M. Giraud, M. Salson, M. Duez, C. Villenet, S. Quief, A. Caillault, N. Gardel, C. Roumier, C. Preudhomme, and M. Figeac (2014). “Fast multiclonal clusterization of V(D)J recombinations from high-throughput sequencing”. *BMC Genomics* 15, 409. <https://doi.org/10.1186/1471-2164-15-409>.

Reviews

Evaluation round #1

DOI or URL of the preprint: <https://hal.science/hal-04361907>

Version of the preprint: 1

Authors’ reply, 08 July 2024

[Download author’s reply](#)

[Download tracked changes file](#)

Decision by [Giulio Ermanno Pibiri](#) , posted 08 March 2024, validated 09 March 2024

This work contributes an efficient algorithm to extract the so-called “V(D)J junctions” from raw sequencing data, using an approach based on the Aho-Corasick automaton and spaced seeds.

Both reviewers found merits in the technical contribution of this paper and agreed that the paper is generally well-written and easy to follow, even for non experts. Furthermore, the experiments are convincing and the code is open-source.

This preprint merits a revision to address the comments of the reviewers.

While one of them has minor comments and suggestions on how to improve some script in the software repository, the other reviewer asks for some clarifications regarding the experiments.

In particular, the authors are kindly requested to address the following points in the revised version of their work:

- A discussion on how the seeds are built.
- Inclusion of the method RTCR as another suitable baseline to compare against.
- A discussion on why the previous version of the algorithm performs better in some circumstances.

The authors present a novel engineered method to detect and designate V(D)J recombinations from raw sequence data.

The novelty of the approach lies not so much in new methods or data structures, but in their efficient combination (e.g., spaced seeds with Aho-Corasick automata).

The evaluation of the method is comprehensive, carefully done, and described in detail.

I found reading the paper, especially the introduction, very enjoyable. The introduction addresses also non-specialists of the field and concisely explains the challenges with just the right amount of detail.

Also, the remainder of the paper can be read without problems.

The evaluation gives the impression that the challenges concerning VCJ recombination are now essentially solved; perhaps the authors can comment on whether this is true or not, or what else should be done in the future in this field (except further small optimizations).

Minor Suggestions and questions:

- It may be better to move Fig. 1 to the bottom of the page.
- 46: "Afzal et al. (2019) did a comparison of several of those software ." -> software tools.
- The term "affectation vector" does not sound right to me, but then I am not a native speaker. Maybe one could use "label sequence"?
- Notation: Please write p-value instead of p -value and E-value instead of E -value in running text.
- 175: the word p-value is not correct here. You mean a 99.9% confidence interval?
- Fig. 5: The titles of the subfigures are too far away from the actual figure. I first mis-interpreted the leftmost figure as the detection results and could not make sense of the statements on lines 291ff.
- Figures: It should also be said that vidjil-old is probably 2018.02 and -new is the "development" version.
- Fig. 8: I suggest to place the color legend to the right of the figures, not below.
- Fig. 8: IGK/vidjil-new: How/why is the designation bar higher than the detection bar?

Comments on the software:

I highly appreciate that the authors provide a Snakemake workflow that starts by installing the software(s). I have a few suggestions for improvement.

- The config file could have reasonable defaults for the directories (results/, benchmarks/, software/).
- MixCR can be skipped, e.g. by using `-keep-going` on Snakemake, but it would be nicer if there was an option to disable it.
- I do get warnings during compilation.
- The workflow outputs a lot in information to the screen. This could be written into log files (using a logs/ directory).
- There are many errors during the Snakefile. Not all are related to MixCR. The software doesn't seem to build properly. Here is a list of the jobs still to be done, i.e. none of the jobs below completes successfully, in spite of `-keep-going`.

Job stats:

job	count
afzal_summary	1
afzal_summary_per_type	2
all	1
gather_classic_results	13
gather_random_vdj_results	3
gather_should_vdj_results	4
install_mixcr_custom_library	1
install_vidjil_dev	1
install_vidjil_old	1

mixcr_to_vdj	4	
random_sequences		1
random_vdj	3	
results_to_plot	23	
run_mixcr3	49	
run_mixcr3_align	2	
run_mixcr4	49	
run_mixcr4_align	2	
run_vidjil	89	
run_vidjil_align	1	
run_vidjil_align_old	2	
should_vdj_dataset	1	
shouldvdj_export_results	2	
vdj_to_assign_detection_results		8
vidjil_detect_to_vdj	4	
vidjil_to_vdj	4	
total	271	

Reviewed by anonymous reviewer 1, 20 February 2024

In this work the authors present an improved version of the Vidjil tool, previously developed for processing high-throughput sequencing data in order to extract so-called V(D)J junctions. V(D)J recombinations in lymphocytes are essential for immunological diversity but could also serve as useful markers of pathologies. The authors propose a multi-loci seed-based method based on an Aho-Corasick-like automaton and spaced seeds extracted from V, D, and J genes. The algorithm was benchmarked against MiXCR on five datasets, evaluating both specificity and sensitivity of detection, as well as the correctness of designation. Results show the newly implemented algorithms bring significant speedups (up to 30x) along with a smaller memory footprint and comparable accuracy.

Overall the manuscript is very well written, extremely clear, and easy to follow in all of its sections. Results are generally convincing and the datasets and experimental benchmarks seem to have been adequately designed in order to evaluate the performance of the algorithms.

Nevertheless, I still have some remarks that I would like to address to the authors and that might improve the quality of the manuscript:

- One of the core components of the method is an automaton built from spaced seeds extracted from V, D, and J genes. However, it is not clear how these seeds were extracted. I think it is important to discuss it a bit more in depth.
- It seems that experiments are focused on data affected by substitution errors (dataset B). Are datasets C, D, and E affected by indels? I guess the method is mainly thought to be used with sequences that are primarily affected by substitution errors but I think it could be interesting to see how it performs in presence of indels.
- I am not fully convinced about the choice of MiXCR as the only tool to compare Vidjil-algo with. The authors justify this as MiXCR being the most balanced tool in terms of flexibility and accuracy, according to a previous systematic review. From the same review however it seems that MiXCR is not the only balanced tool. As a matter of fact RTCR also seems to offer a good balance but is also the one with the most consistent performance across datasets. For this reason, I would suggest also to include RTCR in the comparison, in particular to see how it compares against Vidjil-algo in the challenging dataset E.

- The authors showed the new version of Vidjil-algo provides better results in almost all datasets considered. However in the most challenging one (dataset E), the previous approach actually performs better in some cases. I wonder if the authors investigated this behavior and could discuss the possible reasons. It might be worthwhile also to mention the specific algorithmic differences between the old and new method in the introduction.

Minor remarks/suggestions:

- In the introduction it is mentioned that the new designed algorithm for detection has a time complexity of $O(n)$ compared to the previous version which has $O(l \cdot n)$ complexity. Since also the designation algorithm has been improved it could also make sense to briefly discuss its complexity with respect to the previous implementation.
- The authors define as “designation” the problem of determining the specific germline V, D, and J genes that have undergone recombination, as well as identifying nucleotide deletions/insertions that may have occurred at the junctions of these genes. It is not clear to me if both of these problems are taken into account when evaluating the correct designations or if it is only based on the identification of the specific germline genes.
- Indexation: in the definition of $P(g,u)$ I would replace $|g| \dots i + |u| - 1$ with $|g|, i + |u| - 1$ to be consistent with the “factor” definition introduced above.
- “Section 2 - Locus estimation and recombination detection” paragraph: I would replace t with q in δ and δ^{\prime} definitions to be consistent with Figure 3’s pseudocode.
- It seems that the value δ in Figure 4 is not discussed anywhere. I would add some words about the value actually used by Vidjil-algo in section 3.
- Line 265: Supplementary figure 2 seems a bit redundant as it shows the same information of Table 2 and 3.

Typos:

- Figure 3: a in the pseudocode should be a_q
- Figure 4: (at the end of the caption) “previously” -> “previously”