frontiers | Frontiers in Immunology

TYPE Methods
PUBLISHED 27 October 2022
DOI 10.3389/fimmu.2022.954078

OPEN ACCESS

EDITED BY
Ning Jiang,
University of Pennsylvania,
United States

REVIEWED BY
Philippe Robert,
Aix-Marseille Université, France
Jordi Villà-Freixa,
Universitat de Vic - Universitat Central de Catalunya, Spain

*CORRESPONDENCE
Serghei Mangul
serghei.mangul@gmail.com

SPECIALTY SECTION
This article was submitted to Systems Immunology, a section of the journal Frontiers in Immunology

RECEIVED 26 May 2022
ACCEPTED 05 October 2022
PUBLISHED 27 October 2022

CITATION
Peng K, Moore J, Vahed M, Brito J, Kao G, Burkhardt AM, Alachkar H and Mangul S (2022) pyTCR: A comprehensive and scalable solution for TCR-Seq data analysis to facilitate reproducibility and rigor of immunogenomics research. *Front. Immunol.* 13:954078. doi: 10.3389/fimmu.2022.954078

pyTCR: A comprehensive and scalable solution for TCR-Seq data analysis to facilitate reproducibility and rigor of immunogenomics research

Kerui Peng [1], Jaden Moore [1,2], Mohammad Vahed [1], Jaqueline Brito [1], Guoyun Kao [3], Amanda M. Burkhardt [1], Houda Alachkar[1] and Serghei Mangul [1]*

[1]Department of Clinical Pharmacy, School of Pharmacy, University of Southern California, Los Angeles, CA, United States, [2]Computer Science Department, Orange Coast College, Costa Mesa, CA, United States, [3]Department of Pharmacology and Pharmaceutical Sciences, School of Pharmacy, University of Southern California, Los Angeles, CA, United States

T cell receptor (TCR) studies have grown substantially with the advancement in the sequencing techniques of T cell receptor repertoire sequencing (TCR-Seq). The analysis of the TCR-Seq data requires computational skills to run the computational analysis of TCR repertoire tools. However biomedical researchers with limited computational backgrounds face numerous obstacles to properly and efficiently utilizing bioinformatics tools for analyzing TCR-Seq data. Here we report pyTCR, a computational notebook-based solution for comprehensive and scalable TCR-Seq data analysis. Computational notebooks, which combine code, calculations, and visualization, are able to provide users with a high level of flexibility and transparency for the analysis. Additionally, computational notebooks are demonstrated to be user-friendly and suitable for researchers with limited computational skills. Our tool has a rich set of functionalities including various TCR metrics, statistical analysis, and customizable visualizations. The application of pyTCR on large and diverse TCR-Seq datasets will enable the effective analysis of large-scale TCR-Seq data with flexibility, and eventually facilitate new discoveries.

KEYWORDS

TCR - T cell receptor, TCR-seq, immunogenomics, computational notebooks, TCR characterization, reproducibility

Introduction

T cell receptor (TCR) repertoire is a collection of all unique TCRs in an individual, which is formed through the process of V(D)J recombination after exposure to antigens and the activation of the adaptive immune response. With the growing understanding of TCR repertoire, researchers are able to leverage detailed TCR-Seq datasets to reveal the changes of TCR repertoires in a variety of human disease states such as cancer (1, 2), autoimmune diseases (3, 4), infectious diseases (5, 6), and neurodegenerative diseases (7, 8). Thus, these have helped the biomedical community to deepen the understanding of the roles of the adaptive immune system and adaptive immune responses. For example, studies have shown the usage and diversity of TCR repertoires could be utilized to help select the most suitable immunotherapy for cancer patients (9, 10). Thus, effective TCR profiling and analysis are informative to guide certain cancer treatments, which ultimately enables precision and personalized medicine.

With the rapid development of high-throughput sequencing techniques in the past decades, TCR-Seq has enabled researchers to effectively characterize TCR repertoires across various tissue types and diseases with high specificity and sensitivity by targeting TCR loci. Even with the available TCR profiling methods, TCR repertoire metrics such as diversity, gene usage, and motif enrichment cannot be easily interpreted directly from TCR-Seq data after initial TCR profiling. Post-analysis is required to calculate, visualize, and compare the sample level or population level TCR repertoire characteristics.

Existing bioinformatics tools for TCR repertoire post-analysis are available as R packages such as VDJTools (11), Immunarch (12), and HTML programs such as VisTCR (13) and Vidjil (14). These tools enable biomedical researchers to analyze TCR-Seq data, however, multiple barriers and limitations exist. First, as in any R package, users follow the instructions to enter commands in the command-line interface and the output will be presented in the summarized tables or figures. The analytical methods used for the particular step of the analysis are isolated, which can result in a limited understanding of the details of the analysis. This also increases the probability of human errors. Relying on the documentation of the tool is often not a reliable solution as it typically lacks details, has unclear and ambiguous wording, and can be outdated for future users. Second, the existing TCR-Seq analysis tools need to be installed and utilized with the command-line interface which can be a challenge for biomedical researchers who lack the required computational skill sets (15). Third, the output files are generated as individual files. Biomedical researchers need to work between different files and tools to finish the post-analysis, which leads to high chances of creating manual mistakes. Last, none of the existing tools cover all aspects of the TCR-Seq analysis. For example, researchers need to use multiple packages or additional tools for statistical analysis, which adds an additional burden for biomedical researchers.

Here, we present pyTCR (python TCR analysis), an easy-to-use, interactive, and scalable solution with a wider range of functionalities compared to existing tools. pyTCR utilizes interactive computational notebooks to facilitate reproducibility and rigor of performed TCR-Seq analysis. The availability of well-documented code, visualization, and results of the analysis in a single notebook will facilitate transparency and reproducibility of performed analysis, make the users more aware of the details of the metrics and thresholds being used in the analysis, as well as minimize the possibility of manual mistakes and misinterpretation of the TCR-Seq data analysis results. Notably, pyTCR provides statistical analysis for the first time in a TCR-Seq analysis tool, which is not available in the existing tools. We have demonstrated the utility of pyTCR by applying it to the COVID19-BWNW dataset containing 46 TCR-Seq samples. Additionally, we have compared the scalability of our tool with the existing tools and have demonstrated substantial improvement in running time.

Results

pyTCR: a comprehensive and scalable solution for TCR-seq data analysis

pyTCR, an open-source, user-friendly tool that addresses the issues mentioned above, offers broader and more comprehensive TCR repertoire analysis with an increased number of types of analyses compared to the existing tools. Six types of analysis are contained in the pyTCR, which include basic analysis, clonality analysis, diversity analysis, overlap analysis, gene usage analysis, and motif analysis (Table 1). TCR repertoire metrics, visualization, and statistical analysis are included in all types of analyses (Figure 1, Table 1). Our tool for TCR-Seq data analysis uses interactive computational notebooks for post analysis and visualization of TCR-Seq data with a rich set of functionalities. Notably, we have used Google Colaboratory (Google Colab) to provide the cloud option, which is free to use and provides different subscriptions based on users' needs. No installation of the software to the local computers is required if the users choose to use the Google Colaboratory. However, large files can't be run on Google Colab if the users do not subscribe to Google Colab. The use of computational notebooks enables users to execute analysis and produce tabular output and customizable visualization so that users can use the desired features in their datasets to generate results.

The users can clone the GitHub repository to their local computer and utilize the pyTCR notebooks locally *via* Jupyter Notebook (16). Users who choose to use Google Colab (cloud option of pyTCR) can upload the data files from local

TABLE 1 TCR repertoire analysis functions in pyTCR.

Metric	Description
Basic analysis	
Read count	Number of reads in a sample
Clonotype count	Number of clonotypes in a sample
Mean frequency	Mean of clonotype frequency in a sample $\sum_{i=1}^{n}(pi) \div n$ pi= frequency of clonotype i n = number of unique clonotype in the sample
Geometric mean frequency	Geometric mean of clonotype frequency in a sample $[\prod_{i}^{n}(pi)]^{1/n}$ pi = frequency of clonotype i n = number of unique clonotype in the sample
Mean length of CDR3 nucleotide sequence	Mean length of CDR3 nucleotide sequence, weighted by clonotype frequency, in a sample $\sum_{i=1}^{n}(len(nt) \times pi)$ pi = frequency of clonotype i n = number of unique clonotype in the sample len(nt) = length of CDR3 nucleotide sequence
Convergence	Mean of unique CDR3 nucleotide sequences that code for the same CDR3 amino acid sequence
Spectratype	Frequency of clonotypes based on CDR3 nucleotide lengths
Clonality analysis	
The most or the least frequent clonotype	The most or the least frequent clonotype
1-Pielou index	The evenness of the distribution of the clonotypes $1 + \sum_{i=1}^{n}[(pi \times In(pi))]/In(n)$ pi= frequency of clonotype i n = number of unique clonotype in the sample
Clonal proportion	The number of distinct clonotypes that accounts for greater than or equal to percentage (customizable) of the total of sequencing reads
Relative abundance (in all repertoire, top clonotypes, rare clonotypes)	The proportion of repertoire account for clonal groups with specific abundances in a sample
Diversity analysis	
Shannon-Wiener index	$e^{\sum_{i=1}^{n} - (pi \times In(pi)}$ pi= frequency of clonotype i n = number of unique clonotype in the sample
Normalized Shannon-Wiener index	shannon-wiener index $\div In(n)$ = number of unique clonotype in the sample
Inverse Simpson index	Simpson index D = $\sum_{i=1}^{n} pi^2$ Inverse Simpson index: 1/D
Gini Simpson index	Gini Simpson index: 1-D
D50 index	The percentage of distinct clonotypes that accounts for greater than or equal to 50% of the total of sequencing reads
Chao1 estimate	Chao1 estimate = Chaol estimate = $S + \frac{f1(f1-1)}{2(f2+1)}$ Variation of Chao1 estimate = Variation of Chaol estimate = $f2[0.5*(\frac{f1}{f2})^2 + (\frac{f1}{f2})^3 + 0.25*(\frac{f1}{f2})^4]$ f1 = number of clonotypes with read of 1 f2 = number of clonotypes with read of 2 S = number of clonotypes
Gini coefficient	The ratio of the area that lies between the line of equality and the Lorenz curve over the total area under the line of equality
Gene usage analysis	

(Continued)

TABLE 1 Continued

Metric	Description
V,D,J gene weighted usage	Mean of gene frequency weighted by clonotype frequency
V,D,J gene unweighted usage	Mean of gene frequency by the number of the reads
Overlap analysis	
Morisita-Horn index	$Ch = \frac{(2*\sum_{i=1}^{S} xiyi)}{(\frac{\sum_{i=1}^{s} xi^2}{X^2} + \frac{\sum_{i=1}^{S} yi^2}{Y^2})XY}$ x_i: the number of reads clonotype i is represented in the total number of reads X from one sample y_i: the number of reads clonotype i is represented in the total number of reads Y from another sample S: the index of overlapped clonotypes
Jaccard index	$\frac{A \cap B}{A \cup B}$ A: sample A, B: sample B Number of clonotypes that is in both samples divided by the number of clonotypes that is in either sample
Overlap coefficient	$\frac{A \cap B}{A}$ A: sample A (the sample with a smaller clonotype count), B: sample B Number of clonotypes that is in both samples divided by the number of clonotypes that is in the sample with a smaller clonotype count
Tversky index	$\frac{BothAB}{\alpha*onlyA + \beta*onlyB + botyhAB}$ α=β=0.5 (Sørensen–Dice coefficient) Both AB, number of clonotypes that is in both sample A and B, only A: number of clonotypes that is only in sample A, only B: number of clonotypes that is only in sample B
Cosine similarity	$\frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$ A_i: the frequency of clonotype i in sample A B_i: the frequency of clonotype i in sample B n: the index of overlapped clonotypes
Pearson correlation of clonotype frequencies	$Rij = \frac{\sum_{k=1}^{N} (\phi ik - \overline{\phi i}) - (\phi jk - \overline{\phi j})}{\sqrt{\sum_{k=1}^{N} (\phi ik - \overline{\phi i})^2 - \sum_{k=1}^{N} (\phi jk - \overline{\phi j})^2}}$ $\varnothing ik$: the frequency of clonotype k in sample i $\varnothing ik$: the frequency of clonotype k in sample j N: the index of overlapping clonotypes
Relative overlap diversity	$Dij = \frac{dij}{di*dj}$ dij: the number of clonotypes that is in both samples di: the number of clonotypes that is in sample i dj: the number of clonotypes that is in sample j
Geometric mean of relative overlap frequencies	$Fij = \sqrt{fij*fji}$ $fij = \sum_{k=1}^{N} \varnothing ik$:the total frequency of clonotypes that overlap between samples i and j in sample i $fji = \sum_{k=1}^{N} \varnothing jk$:the total frequency of clonotypes that overlap between samples i and j in sample j
Clonotype-wise sum of geometric mean frequencies	$F2ij = \sum_{k=1}^{N} \sqrt{\varnothing ik \varnothing jk}$ $\varnothing ik$:the frequency of clonotype k in sample i $\varnothing jk$:the frequency of clonotype k in sample j N: the index of overlapped clonotypes

(Continued)

TABLE 1 Continued

Metric	Description
Jensen-Shannon divergence	$F2ij = \sum_{k=1}^{N} \sqrt{\varnothing ik \varnothing jk}$ $\mathrm{KL}(P \,.\, Q) = \sum_{i} pi \quad \log 2 \frac{pi}{qi}$ p_i: the sum of overlapped variable segment (V) frequencies in sample 1 q_i:the sum of overlapped variable segment (V) frequencies in sample 2
Motif analysis	
Amino acid spectratype	Frequency of clonotypes based on CDR3 amino acid lengths
	Amino acid motif analysChao1 estimateis Number of counts of the amino acid motifs
Nucleotide sequence motif analysis	Number of counts of the nucleotide sequence motifs

We documented the name of the metrics (indicated in the column "Metric") and the description of the corresponding metrics (indicated in the column "Description").

computers, web-based drives, or GitHub repositories. The results can be downloaded and stored locally or into web-based drives by the code provided. pyTCR is capable of converting results from pre-processing software such as MiXCR and ImmunoSEQ to the pyTCR analysis format. The minimal clonotype information should include the counts of reads, clonotype frequency, CDR3 nucleotide sequence, CDR3 amino acid sequence, and the inferred V, D, and J genes in the input data files. pyTCR provides conversion to the corresponding format which consists of the columns of counts of reads (#count), frequency (freq), CDR3 sequence (cdr3nt), CDR3 amino acids (cdr3aa), V gene (v), D gene (d), J gene (j), features (if provided), sample name. These should be already filtered for the non-coding CDR3 by the upstream tools. If the metadata is not available, a notebook for combining individual files to a metadata file should be executed prior to any TCR-Seq analysis to reduce the burden of analyzing individual data files separately. In order to achieve this, all the sample files should be stored or uploaded in one folder prior to generating the metadata file. The notebook that combines individual sample data files to a metatable with all the files is provided.

pyTCR is able to perform basic analysis to characterize the TCR repertoire

The focus of the basic analysis is to group and provide the most fundamental TCR repertoire metrics in one place. The basic analysis performed by pyTCR estimates provides the number of reads, clonotype counts, mean clonotype frequency, the geometric mean of clonotype frequency, mean length of CD3 nucleotide sequence, convergence, spectratype as TCR repertoire metrics. The visualization is available for all the metrics (except for spectratype) in the basic analysis at the individual sample level and group level. The available plot types are violin plot, strip plot, swarm plot, box plot, boxen plot, point plot, and bar plot (Figure 1A, Supplementary Figure 1). We were able to detect that the mean reads count in the hospitalization group was lower than that in the non-hospitalization group (480844.1 and 554580.3, respectively; t-test: p = 0.229), and the mean clonotype count in the hospitalization group was lower than in the non-hospitalization group (271777.6 and 328980.1, respectively; t-test: p = 0.136) in the COVID19-BWNW dataset.

pyTCR is able to perform clonality analysis to assess the evenness of distribution of TCR clonotypes

The clonality analysis offers the measurements of clonality, which has been used to assess the evenness of distribution of the clonotypes based on the relative abundance of clonotypes in the sample. The metrics include the list of the most or the least frequent clonotypes, 1-Pielou index for evenness measure (0 means no evenness, 1 means complete evenness), clonal proportion, and the distribution of clonotype groups based on relative abundance. Specifically, clonal proportion presents the number of clonotypes that consist of a certain percentage of the clonotypes in the repertoire. In the COVID19-BWNW dataset, the number of clonotypes that counts for 10% of the clonotypes in the repertoire was smaller in the hospitalization group than in the non-hospitalization group (49.5 and 459, respectively; Wilcoxon rank-sum test, p = 0.596), the corresponding plots were presented in various types (Supplementary Figure 2). Additionally, the distribution of clonotype groups based on clonotype frequency or count in each sample can be presented in bar plots across all the clonotypes, the top clonotypes, and the rare clonotypes (Supplementary Figure 3). We presented the distribution of five clonotype groups (hyperexpanded, large, medium, small, and rare) across all clonotypes in Figure 1C, this categorization is similarly done in other existing tools. The users have full control of the thresholds of the clonotype groups in our tool.

10.3389/fimmu.2022.954078

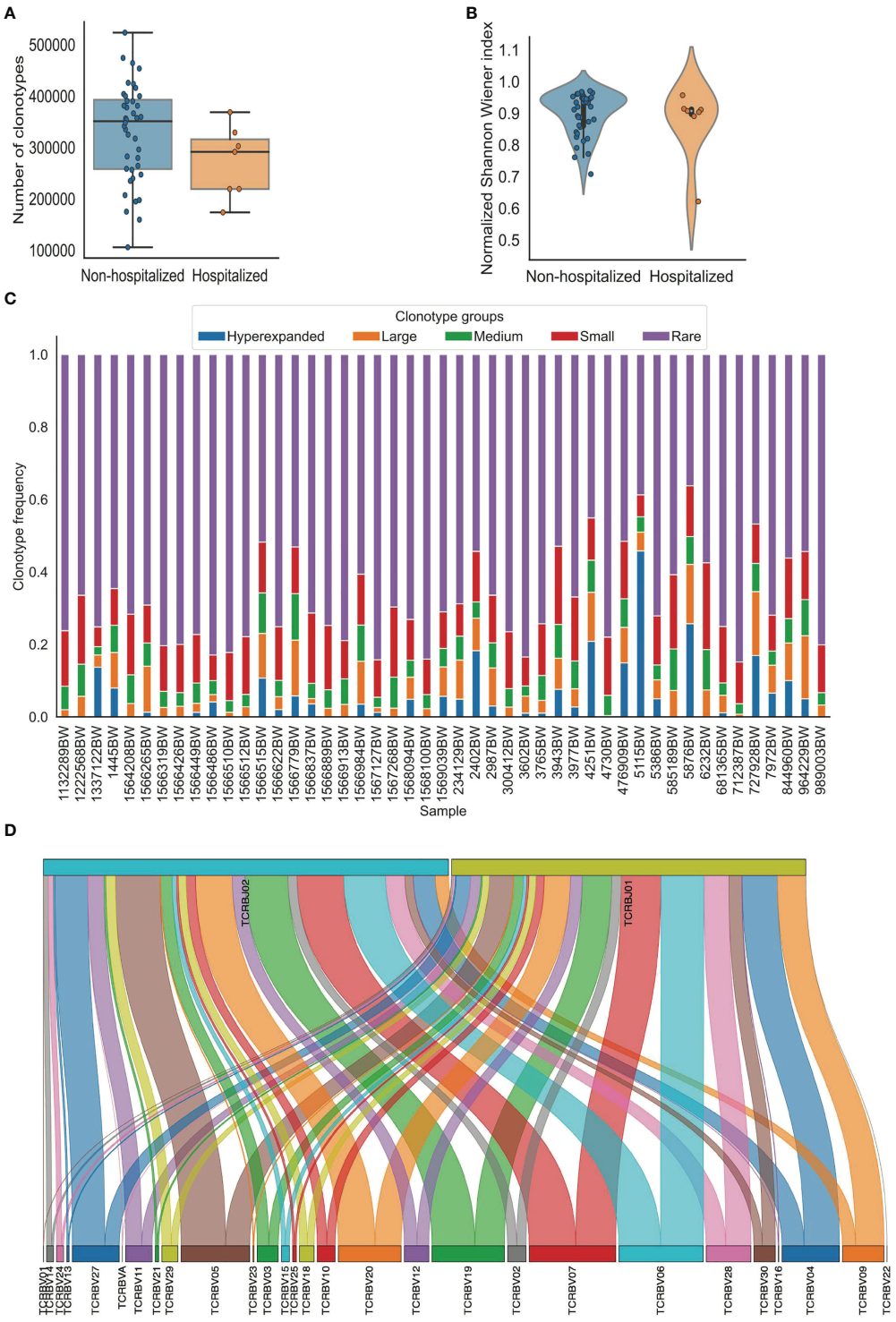

FIGURE 1
Visualization of TCR repertoire metrics generated using pyTCR. **(A)** The clonotype counts of each sample grouped by hospitalization status were presented as a box plot and strip plot. **(B)** The normalized Shannon-Wiener index of each sample grouped by hospitalization status was presented as a violin plot. **(C)** The distribution of clonotype groups in each sample was presented as a stacked bar plot. The clonotypes were categorized into five groups based on the clonotype frequencies. Hyperexpanded clonotypes were the clones with frequencies between 0.01 to 1, large clonotypes were the clones with frequencies between 0.001 to 0.01, medium clonotypes were the clones with frequencies between 0.0001 to 0.001, small clonotypes were the clones with frequencies between 0.00001 to 0.0001, rare clonotypes were the clones with frequencies between 0 to 0.00001. **(D)**. The V-J combinations with V and J gene frequencies in sample 1445BW were presented as a Sankey plot.

pyTCR is able to perform gene usage analysis to detect over and underrepresented TCR genes across the samples

Gene usage analysis provides the weighted and unweighted V/D/J gene usage calculations. For gene usage analysis, V gene usage, D gene usage, and J gene usage, both weighted (which is based on clonotype frequency) and unweighted (which is based on clonotype count) are provided as TCR repertoire metrics. Heatmap and hierarchically clustered heatmap are the available visualizations (Supplementary Figure 4A, B). Sankey plot is also available to visualize the V-J combinations (Figure 1D, Supplementary Figure 4C), this is not provided by other existing tools. We observed higher V gene weighted usage of TRBV05-05*01 (0.0084 and 0.0066, respectively) and TRBV13-01*01 (0.0069 and 0.0042, respectively) in the non-hospitalization group. In comparison, we observed higher V gene weighted usage of TRBV20 (0.0638 and 0.0588, respectively) in the hospitalization group in the COVID19-BWNW dataset. We also observed higher V gene unweighted usage of TRBV18-01*01 (0.035 and 0.031) and TRBV30-01*01 (0.025 and 0.019) in the hospitalization group. After the Bonferroni correction to account for the multiple comparisons, according to the adjusted p values, the differences mentioned above were not statistically significant.

pyTCR is able to assess the diversity of TCR repertoires

Diversity analysis offered by pyTCR includes all the widely adopted indices to characterize the diversity of TCR repertoire, which contains Shannon-Wiener index, normalized Shannon-Wiener index, inverse Simpson index, Gini Simpson index, D50 index, Chao1 estimate, Gini coefficient (Table 1). High Shannon-Wiener index, low normalized Shannon-Wiener index, high inverse Simpson index, high Gini Simpson index, high Chao1 estimate, and high Gini coefficient represent high clonal diversity. Additionally, the D50 index represents the percentage of unique clonotypes that account for greater than 50% of the total number of sequences. The visualization is available for all the diversity metrics at the sample or group level as violin plot, strip plot, swarm plot, box plot, boxen plot, point plot, and bar plot (Figure 1B, Supplementary Figure 5). In the COVID19-BWNW dataset, the median Shannon-Wiener index, the median inverse Simpson index, and the median Gini Simpson index were all lower in the hospitalization group than in the non-hospitalization group. Even though none of the diversity indices was statistically significant, most of the diversity indices showed the trend that patients in the non-hospitalization group have more diverse TCR clonotypes than patients in the hospitalization group. This finding was consistent with the results observed in the previously published studies, that severe COVID-19 patients had reduced TCR diversity than moderate COVID-19 patients (17, 18).

pyTCR is able to effectively compare clonotypes and motifs across samples

The overlap analysis offers a comprehensive list of overlap metrics for comparing the clonotype frequencies between two samples. These metrics include the Jaccard index, overlap coefficient, Morisita-Horn index, Tversky index, Cosine similarity, Pearson correlation of clonotype frequencies, relative overlap diversity, the geometric mean of relative overlap frequencies, the clonotype-wise sum of geometric mean frequencies, and Jensen-Shannon divergence. For overlap analysis, the visualization is shown in the heatmaps (Supplementary Figure 6). Currently, existing tools only accept one sample per file for overlap comparisons which can be difficult to manage if the data already contains multiple samples per file. pyTCR allows for an unlimited amount of samples per file which enables more flexibility and less file management.

Motif analysis provides enriched nucleotide and amino acid motif discovery with customized length. For motif analysis, the amino acid motif counts and nucleotide motif counts in each sample are provided. The users are able to customize the length of the motif and visualize the distribution of the motifs in each sample or each group. In the COVID19-BWNW dataset, we observed amino acid motifs NTEAFF, YNEQFF, CASSLG, TDTQYF, NQPQHF, TGELFF, SYEQYF were the most abundant ones in both the hospitalization and non-hospitalization groups (Supplementary Figure 7A). We also observed nucleotide motifs such as TCTGTG, CTGTGC, TGTGCC, GTGCCA, TGCCAG, GCCAGC, CCAGCA, CAGCAG were the most abundant ones in both hospitalization and non-hospitalization groups (Supplementary Figure 7B).

pyTCR offers several advantages compared to the existing tools

pyTCR provides more comprehensive functionalities compared with VDJtools, Immunarch, and VisTCR (Supplementary Table 1). Notably, pyTCR includes several innovations that have not been implemented in the VDJtools, Immunarch, or VisTCR before.. First, statistical analyses for TCR-Seq datasets are embedded in pyTCR computational notebooks. No additional software or platform is needed for statistical analysis. Second, pyTCR has the most comprehensive list of overlap indices, which enable the thorough comparison between clonotypes across samples. Third, pyTCR offers enriched motif detection for both amino acid sequences as well as nucleotide sequences. Furthermore, we have compared

the results produced by pyTCR with VDJtools on the types of analysis that are provided by VDJtools on the COVID19-BWNW dataset. The results are consistent across our tool and VDJtools.

We also evaluated the scalability of pyTCR by varying the number of samples in the TCR-Seq data input files. After subsampling the COVID19-BWNW dataset into data files containing 2 to 46 samples, we recorded the central processing unit (CPU) time and memory usage required by pyTCR, VDJtools, and Immunarch when running overlap analysis. We observed that pyTCR required a significantly less amount of CPU time across all the subsamples compared to VDJtools and Immunarch (Figure 2A). For example, on average, pyTCR used 5 minutes and 25 seconds to process the overlap analysis for 46 samples, while Immunarch used 1 hour 49 minutes and 46 seconds to process the same number of samples. In terms of memory usage, pyTCR had reduced memory usage for most of the subsamples (Figure 2B). According to our benchmark results, we observe that pyTCR has up to 22 times faster performance than existing TCR-seq analysis tools, especially for datasets with larger numbers of samples.

Discussion

We have presented pyTCR, a comprehensive and scalable computational notebook-based solution for TCR-Seq analysis and visualization with a rich set of functionalities. For the cloud-based version, we use Google Colaboratory (Google Colab). Google Colab, as a user-friendly, free with no installation needed prior to use service for Google account holders, is suitable for biomedical researchers with a limited computational background. Using interactive computational notebooks promotes high transparency for biomedical researchers because the steps of analyzing and visualizing are recorded and saved, which are easy to be shared with the scientific community. The goal of pyTCR is to provide straightforward scripts for useful analysis which can be easily modified and complemented by users instead of stand-alone software tools with well-structured code. The availability of the code as part of the notebook allows the users to document all the steps of the analysis and share them in a reproducible and transparent way.

pyTCR offers several advantages compared to the existing tools. First, pyTCR includes more comprehensive measurements than existing tools to analyze TCR-Seq data. The enriched measurements can provide users with more options to effectively characterize TCR repertoires and compare across various phenotypes. Furthermore, pyTCR provides code and analysis jointly together. Users can understand the definition of measurements and interpret results easily with pyTCR, as the explanation of the code and the math equations are available in the notebooks. Additionally, pyTCR allows users to adjust parameters easily and directly in the notebooks. Unlike other traditional bioinformatics tools, changing parameters that generate separate files which leads to high error rates by analyzing across different files, pyTCR provides all the analysis to be performed in the cloud where the files are automatically saved with the updated parameters and no generation of different files is needed. Last, our tool is more scalable as it requires less computational time for analysis.

We recognized that there are other tools available for TCR-Seq analysis, however, these tools may not share the same

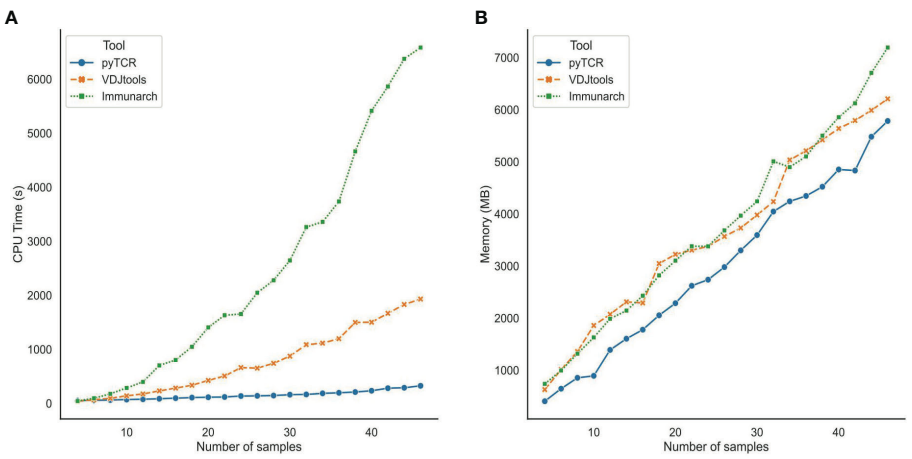

FIGURE 2
Central Processing Unit (CPU) time **(A)** and Memory usage **(B)** for subsamples of the COVID19-BWNW dataset for overlap analysis. Each dot on the line plot represented the average of 10 runs for different input sizes.

purposes as pyTCR. For example, both VisTCR (13) and Vidjil (14) are web-based tools for TCR-Seq analysis that uses fastq files as input files. While pyTCR utilizes the tsv or csv files generated by pre-processing tools such as MiXCR to conduct post-analysis. The sample files that we used in the manuscript were generated by Adaptive Biotechnologies, unfortunately, the users do not have access to the raw sequencing data per user policy. VDJviz is a web-based tool as well and it uses VDJtools as a back-end.

However, there are limitations of pyTCR including the possibility of accidentally modifying the code resulting in generating errors, limited available types of analysis, and storage and processing speed limits from the Google Colab platform. For example, users with limited experience with Python scripting may be prone to generate errors with the availability of code and results in interactive notebooks. Additionally, pyTCR cannot be used directly on raw sequencing files (such as fastq format).

In conclusion, our tool offers broader and more powerful functions in TCR repertoire research. We expect the computational notebook-based tool to be adopted by the broad biomedical community as it carries benefits that are superior or comparable to R packages.

Method

TCR-Seq data

We used the COVID19-BWNW dataset from the Adaptive ImmuneRace study to demonstrate the functionality of pyTCR. COVID19-BWNW dataset contains 46 convalescent COVID-19 patient samples collected at Bloodworks Northwest. Demographic and clinical features including age, gender, smoking status, ICU admit status, birth year, blood type, CMV at donation, days from the last symptom to sample date, ethnicity, race, height, weight, and hospitalization status are reported. The extracted genomic DNA was sequenced based on Multiplex PCR and only for the TCR beta chain by using the MiSeq platform.

TCR-Seq data preprocessing

We downloaded 46 TCR-Seq data samples and the file containing demographic and clinical features in the tab-separated values (tsv) format. All the demographic and clinical features were listed in the sample_tags column in the file. The features were split into one in each column for further analysis.

Statistical analysis

The statistical analysis is available for comparing numerical values across two groups. We first examine whether the datasets are normally distributed. If the dataset is normally distributed, we use the student's t-test to evaluate the statistical significance. Otherwise, we use Wilcoxon rank-sum test to evaluate the statistical significance. Bonferroni correction is also included to count for the multiple comparisons across different genes.

Jupyter notebooks

Jupyter notebooks (https://jupyter.org) is a web-based interactive computing platform that contains code, markdown text, and visualizations. These features enable users to conduct reproducible and transparent data analysis. We develop pyTCR based on Jupyter notebooks. In each Jupyter cell, we include code for either calculation for analysis or visualization. Users can easily change the parameters in the code to generate the results of their interests. Markdown text is used for instructions and explanations.

pyTCR data structure and software implementation

The individual input data file can be text files, tab-separated values (tsv) files, or comma-separated values (csv) files. pyTCR uses a small suite of robust open-source python libraries to facilitate complex data analysis and visualizations. Python libraries including Pandas, Numpy, Matplotlib, Seaborn, and Scipy are used to provide rich functionality with limited amounts of code. Pandas is used to convert tabularly formatted TCR-Seq data into python data frames for notebooks to utilize. Numpy is used to perform complex mathematical operations across python data frames. Matplotlib and Seaborn are then used in tandem to generate rich data visualizations from the resultant data.

One critical component of pyTCR functionality is the overlap analysis between two samples. Such operations are unavoidably expensive in terms of computing power. Yet, pyTCR employs multiple different technical optimizations by default to provide the most optimal performance for researchers. In the case of any piecewise comparison between two samples, we first index and group the data into a key-value pair hash table for instantaneous look-up time. We then uniquely merge samples for comparison and index them using a hash table in the same manner. By utilizing this method, we are able to negate a large amount of computing time that would otherwise be

associated with searching for the correct samples in the data set. This method allows us to instantly retrieve the required samples for future look-ups which shifts most of the computing time from slow searches, back onto the piecewise comparisons.

Comparison with other methods

We used the COVID19-BWNW dataset to compare pyTCR to VDJtools and Immunarch for benchmarking purposes. We subsampled the COVID19-BWNW dataset to files containing 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46 TCR-Seq samples. We then ran the overlap analysis of each tool ten times and computed the average CPU time and RAM usage for each. For the comparison, we utilized a high-performance computing cluster (HPCC) to acquire the most accurate benchmarking results. However, in the comparison of 2 TCR-Seq samples for VDJtools and Immunarch, the HPCC was unable to record the results due to the short nature of the task. That is, the benchmark ended too quickly for the HPCC to accurately record the results. Thus, the results for 2 TCR-Seq samples were not taken with the average of ten runs. Instead, we recorded the results once by introducing an artificial stall in the benchmark such that the HPCC had time to record, and then we subtracted the artificial stall time from the final CPU time. The RAM usage remains unchanged with this workaround.

Data availability statement

COVID19-BWNW dataset was part of the ImmuneRACE Study, which was downloaded from the ImmuneCODETM database (https://immunerace.adaptivebiotech.com/data). To improve the discoverability of the data, we have copied the data to (https://github.com/Mangul-Lab-USC/pyTCR_publication/tree/main/data/samples). All data required to produce the figures and analysis performed in this paper are freely available at (https://github.com/Mangul-Lab-USC/pyTCR)_publication. pyTCR is freely available at (https://github.com/Mangul-Lab-USC/pyTCR). pyTCR is distributed under the terms of the MIT License. The user manual for pyTCR can be downloaded at (https://github.com/Mangul-Lab-USC/pyTCR/blob/main/User_Manual_pyTCR.pdf). All code required to produce the figures and analysis performed in this paper is freely available at (https://github.com/Mangul-Lab-USC/pyTCR_publication).

Ethics statement

The ImmuneRACE study involving human participants was reviewed and approved by the Western Institutional Review Board (reference number 1-1281891-1) on March 17th, 2020. The participants provided their written informed consent to participate in the ImmuneRACE study.

Author contributions

KP designed and developed the tool. KP wrote the manuscript with input from all other authors. JM, MV, and JB optimized the tool. GK assisted in creating tables. AB and HA revised the manuscript. SM conceived and supervised the study. All authors contributed to the article and approved the submitted version.

Funding

KP and SM are supported by the National Science Foundation grants 2041984 and 2135954.

Acknowledgments

We would like to thank Yesha Patel for the helpful suggestions for statistical analysis. The authors acknowledge the Center for Advanced Research Computing (CARC) at the University of Southern California for providing computing resources that have contributed to the research results reported within this publication. URL: https://carc.usc.edu.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fimmu.2022.954078/full#supplementary-material

SUPPLEMENTARY FIGURE 1
The clonotype counts were shown in groups by hospitalization status. The patients that were not hospitalized were shown in blue while the patients that were hospitalized were shown in orange. Different types of plots were shown as follows: **(A)** violin plot **(B)** strip plot **(C)** swarm plot **(D)** box plot **(E)** boxen plot **(F)** point plot **(G)** bar plot.

SUPPLEMENTARY FIGURE 2
Clonal portion grouped by hospitalization status. The y-axis presented the number of clonotypes that counted for 10% of all the clonotypes in the repertoire. The patients that were not hospitalized were shown in blue while the patients that were hospitalized were shown in orange. Different types of plots were shown as follows: **(A)** violin plot **(B)** strip plot **(C)** swarm plot **(D)** box plot **(E)** boxen plot **(F)** point plot **(G)** bar plot.

SUPPLEMENTARY FIGURE 3
The distribution of clonotype groups in each sample. **(A)** The distribution of clonotype groups was based on the clonotype frequency across all the clonotypes. Hyperexpanded clonotypes (blue) were the clones with frequencies between 0.01 to 1, large clonotypes (orange) were the clones with frequencies between 0.001 to 0.01, medium clonotypes (green) were the clones with frequencies between 0.0001 to 0.001, small clonotypes (red) were the clones with frequencies between 0.00001 to 0.0001, rare clonotypes (purple) were the clones with frequencies between 0 to 0.00001. **(B)** The distribution of clonotype groups based on the clonotype count across the top 100 clonotypes. The clonotypes with clone counts between 1001-5000 were presented in blue, the clonotypes with clone counts between 101-1000 were resented in orange, and the clonotypes with clone counts between 11-100 were presented in green. **(C)** The distribution of clonotype groups based on the clonotype count across the rare 100 clonotypes. The clonotypes with clone count of 1 were presented in blue.

SUPPLEMENTARY FIGURE 4
Heatmap of the weighted V gene usage in each sample and the Sankey plot for V-J combinations. **(A)** The heatmap of V gene weighted usage **(B)** The hierarchically-clustered heatmap of V gene weighted usage. x-axis represented each sample, y-axis represented different V genes. The shade of the color corresponded to the V gene frequency. **(C)** The V-J combinations with V and J gene frequencies in sample 3602BW were presented as a Sankey plot.

SUPPLEMENTARY FIGURE 5
The diversity indices were shown in groups by hospitalization status. The patients that were not hospitalized were shown in blue while the patients that were hospitalized were shown in orange. **(A)** violin plot of the Shannon Wiener index **(B)** strip plot of normalized Shannon Wiener index **(C)** swarm plot of inverse Simpson index **(D)** box plot of Gini Simpson index **(E)** boxen plot of D50 index **(F)** point plot of Chao1 estimate **(G)** bar plot of Gini coefficient.

SUPPLEMENTARY FIGURE 6
The overlap indices across samples. Heatmaps of each overlap index as shown above. The x-axis and the y-axis presented each sample. **(A)** Jaccard index **(B)** Overlap coefficient **(C)** Morisita-Horn index **(D)** Tversky index **(E)** Cosine similarity **(F)** Pearson correlation based on clonotype counts **(G)** Pearson correlation based on clonotype frequency **(H)** Relative overlap diversity **(I)** Geometric mean of relative overlap frequencies **(J)** Clonotype-wise sum of geometric mean frequencies **(K)** Jensen-Shannon divergence of variable gene usage distributions.

SUPPLEMENTARY FIGURE 7
The number of highly presented motifs across samples grouped by hospitalization status. The x-axis presented motifs and the y-axis presented the count of the motifs. **(A)** amino acid motifs (k=6) that had numbers of counts of no less than 9,999 and were presented in more than 2 samples in hospitalization and non-hospitalization groups were shown in the strip plot **(B)** nucleotide motifs (k=6) that had numbers of counts of more than 150,000 and were presented in more than 2 samples in hospitalization and non-hospitalization group were shown in the strip plot.

References

1. Aoki H, Shichino S, Matsushima K, Ueha S. Revealing clonal responses of tumor-reactive T-cells through T cell receptor repertoire analysis. *Front Immunol* (2022) 13:807696. doi: 10.3389/fimmu.2022.807696

2. Liu XS, Mardis ER. Applications of immunogenomics to cancer. *Cell* (2017) 168:600–12. doi: 10.1016/j.cell.2017.01.014

3. Simnica D, Schliffke S, Schultheiß C, Bonzanni N, Fanchi LF, Akyüz N, et al. High-throughput immunogenetics reveals a lack of physiological T cell clusters in patients with autoimmune cytopenias. *Front Immunol* (2019) 10:1897. doi: 10.3389/fimmu.2019.01897

4. Liu X, Zhang W, Zhao M, Fu L, Liu L, Wu J, et al. T Cell receptor β repertoires as novel diagnostic markers for systemic lupus erythematosus and rheumatoid arthritis. *Ann Rheumatol Dis* (2019) 78:1070–8. doi: 10.1136/annrheumdis-2019-215442

5. Emerson RO, DeWitt WS, Vignali M, Gravley J, Hu JK, Osborne EJ, et al. Immunosequencing identifies signatures of cytomegalovirus exposure history and HLA-mediated effects on the T cell repertoire. *Nat Genet* (2017) 49:659–65. doi: 10.1038/ng.3822

6. Luo L, Liang W, Pang J, Xu G, Chen Y, Guo X, et al. Dynamics of TCR repertoire and T cell function in COVID-19 convalescent individuals. *Cell Discov* (2021) 7:89. doi: 10.1038/s41421-021-00321-x

7. Gate D, Saligrama N, Leventhal O, Yang AC, Unger MS, Middeldorp J, et al. Clonally expanded CD8 T cells patrol the cerebrospinal fluid in alzheimer's disease. *Nature* (2020) 577:399–404. doi: 10.1038/s41586-019-1895-7

8. Patas K, Willing A, Demiralay C, Engler JB, Lupu A, Ramien C, et al. T Cell phenotype and T cell receptor repertoire in patients with major depressive disorder. *Front Immunol* (2018) 9:291. doi: 10.3389/fimmu.2018.00291

9. Hogan SA, Courtier A, Cheng PF, Jaberg-Bentele NF, Goldinger SM, Manuel M, et al. Peripheral blood TCR repertoire profiling may facilitate patient stratification for immunotherapy against melanoma. *Cancer Immunol Res* (2019) 7:77–85. doi: 10.1158/2326-6066.CIR-18-0136

10. Schrama D, Ritter C, Becker JC. T Cell receptor repertoire usage in cancer as a surrogate marker for immune responses. *Semin Immunopathol* (2017) 39:255–68. doi: 10.1007/s00281-016-0614-9

11. Shugay M, Bagaev DV, Turchaninova MA, Bolotin DA, Britanova OV, Putintseva EV, et al. VDJtools: Unifying post-analysis of T cell receptor repertoires. *PloS Comput Biol* (2015) 11:e1004503. doi: 10.1371/journal.pcbi.1004503

12. Nazarov V, Tsvetkov V, Rumynskiy E, Popov A, Balashov I, Samokhina M, et al. *Immunarch: Bioinformatics analysis of T-Cell and B-Cell immune repertoires*. (2022). Available at: https://immunarch.com/, https://github.com/immunomind/immunarch

13. Ni Q, Zhang J, Zheng Z, Chen G, Christian L, Grönholm J, et al. VisTCR: An interactive software for T cell repertoire sequencing data analysis. *Front Genet* (2020) 11. doi: 10.3389/fgene.2020.00771

14. Duez M, Giraud M, Herbert R, Rocher T, Salson M, Thonier F. Vidjil: A web platform for analysis of high-throughput repertoire sequencing. *PloS One* (2016) 11:e0166126. doi: 10.1371/journal.pone.0166126

15. Mangul S, Mosqueiro T, Abdill RJ, Duong D, Mitchell K, Sarwal V, et al. Challenges and recommendations to improve the installability and archival stability of omics computational tools. *PloS Biol* (2019) 17:e3000333. doi: 10.1371/journal.pbio.3000333

16. Randles BM, Pasquetto IV, Golshan MS, Borgman CL. *Using the jupyter notebook as a tool for open science: An empirical study*. New York, NY, United States: Institute of Electrical and Electronics Engineers (IEEE) (2017) p. 1–2. doi: 10.1109/JCDL.2017.7991618

17. Chang C-M, Feng PH, Wu TH, Alachkar H, Lee KY, Chang WC. Profiling of T cell repertoire in SARS-CoV-2-Infected COVID-19 patients between mild disease and pneumonia. *J Clin Immunol* (2021) 41:1131–45. doi: 10.1007/s10875-021-01045-z

18. Zhang F, Gan R, Zhen Z, Hu X, Li X, Zhou F, et al. Adaptive immune responses to SARS-CoV-2 infection in severe versus mild individuals. *Signal Transduction Targeting Ther* (2020) 5:156. doi: 10.1038/s41392-020-00263-y