copepodTCR: Identification of Antigen-Specific T Cell Receptors with combinatorial peptide pooling

Vasilisa A. Kovaleva [1], David J. Pattinson [2], Carl Barton [3], Sarah R. Chapin [1], Anastasia A. Minervina [4], Katherine A. Richards [5], Andrea J. Sant [5], Paul G. Thomas [4]✉, Mikhail V. Pogorelyy [4]✉, Hannah V. Meyer [1]✉

[1]Simons Center for Quantitative Biology, Cold Spring Harbor Laboratory, Cold Spring Harbor, NY 11724, USA; [2]Department of Pathobiological Sciences, School of Veterinary Medicine, University of Wisconsin-Madison, Madison, WI 53711, USA; [3]Birkbeck, University of London, WC1E 7HX London, UK; [4]Department of Immunology, St. Jude Children's Research Hospital, Memphis, TN 38105, USA; [5]David H. Smith Center for Vaccine Biology and Immunology, Department of Microbiology and Immunology, University of Rochester Medical Center, Rochester, NY 14642, USA

Abstract

T cell receptor (TCR) repertoire diversity enables the orchestration of antigen-specific immune responses against the vast space of possible pathogens. Identifying TCR/antigen binding pairs from the large TCR repertoire and antigen space is crucial for biomedical research. Here, we introduce *copepodTCR*, an open-access tool for the design and interpretation of high-throughput experimental assays to determine TCR specificity. *copepodTCR* implements a combinatorial peptide pooling scheme for efficient experimental testing of T cell responses against large overlapping peptide libraries, useful for "deorphaning" TCRs of unknown specificity. The scheme detects experimental errors and, coupled with a hierarchical Bayesian model for unbiased results interpretation, identifies the response-eliciting peptide for a TCR of interest out of hundreds of peptides tested using a simple experimental set-up. We experimentally validated our approach on a library of 253 overlapping peptides covering the SARS-CoV-2 spike protein. We provide experimental guides for efficient design of larger screens covering thousands of peptides which will be crucial for the identification of antigen-specific T cells and their targets from limited clinical material.

Introduction

Identifying antigen-specific T cell responses and their targets is crucial in numerous applications, including vaccine research[1], the development of cancer immunotherapy[2], and autoimmunity treatment[3]. A T cell's ability to mount a targeted immune response is defined by the specificity of its T cell receptor (TCR). TCRs are generated semi-stochastically by somatic rearrangement of germline-encoded gene segments[4]. This process generates TCRs which will be able to respond with unique specificity against their cognate epitope. For conventional T cells, these are short peptide sequences presented on major histocompatibility complexes (MHC). MHC molecules are present on the surface of all nucleated cells in the body (MHC class I) and, in addition, on professional antigen presenting cells (MHC class II). While there are large efforts to predict peptide-MHC binding[5] and TCR-epitope specificity computationally[6], these are still limited due to lack of unbiased, true positive and negative training data[7] and also vary greatly depending peptide-MHC context[8]. Thus, there is still a critical role for efficient, empirical epitope discovery for T cells.

Identifying the peptide specificity of a given TCR requires an experimental assay in which T cells expressing the TCR of interest interact with antigen presenting cells that express candidate peptide(s)-MHC complexes on their surface, coupled with a quantitative read out of T cell activation upon this peptide:MHC encounter. An ideal experiment will allow high-throughput screening of a large peptide library against a given TCR, with mechanisms to detect experimental errors (false positives and false negatives), and a simple experimental setup. The design of the peptide libraries proves crucial to address these points.

First, to detect activating epitopes and ensure confidence in negative results, the library should tile i.e. cover the whole protein or proteome space of potential antigenic peptides. Comprehensive coverage can be achieved by using a 'sliding window' approach where the protein space of interest is sliced into overlapping peptides of defined length and overlap that together cover the whole space (Figure 1A). However, while optimal for coverage, the use of overlapping peptides introduces a significant challenge in designing and interpreting a high throughput T cell activation assay (as outlined below), particularly because the epitope recognized by the T cell will be present in multiple peptides of the library.

The most simple experimental design where each peptide is individually tested against a given TCR is time- and reagent-consuming (e.g. as in[9]; Supplementary Fig 1A). An alternative approach called matrix pooling[10] partially overcomes these limitations. In matrix pooling (Supplementary Fig 1B), a n row $\times$ m column experimental plate with single peptide per well are combined into n row and m column peptide pools. In this design, each peptide is present in one row and one column pool, and a TCR-specific peptide should thus lead to the activation of two pools. These can then be mapped back as the intersection of the respective row and column pool to identify the single peptide. Matrix pooling is more efficient compared to testing each peptide individually as it requires less time and fewer reagents. To further reduce the number of peptide pools that need to be tested, combinatorial peptide pooling (CPP) using a table of addresses for combining peptides into pools instead of the simple row and column mixing[11,12] (Supplementary Fig 1C) can be used. Each peptide is added to a unique subset of pools (binary "address"), which leads to matching activation patterns in T cells stimulated by combinatorial pools. The benefit of this approach is that a large number of peptides can be encoded by a few pools. However, for overlapping peptide libraries, a TCR will recognize more than one peptide which can make the results hard to interpret.

Here, we developed *copepodTCR*– Combinatorial Peptide Pooling Design for TCR specificity –an algorithm for TCR deorphaning that addresses the three critical challenges outlined above: it i) considerably reduces experimental resources and time required for testing by using a CPP approach, ii) accounts for overlapping peptides, and iii) enables the detection of errors in the experimental process and even in case of the error still significantly narrows down the list of peptide candidates. We demonstrate its ease of application and robustness of experimental design across a wide range of parameters, simulate erroneous experimental outcomes and their implications for experimental validation, and show its power in reducing experimental complexity. We experimentally validated our approach on a library of 253 overlapping peptides covering the SARS-CoV-2 spike organized into 12 pools, and provide a pooling scheme for a larger 1879 overlapping peptide library covering the entire SARS-CoV-2 proteome with 18 pools. Our algorithm is open-source, and available as a Python package. For ease of use, we have developed a ShinyApp web service to design custom experimental peptide pooling assays against any antigen and aid in the interpretation of the acquired experimental results.

Results

Requirements on a combinatorial peptide pooling scheme tiling the whole protein space

For the development of the protocol, we translated the three experimental design considerations outlined above into computational constraints that we address in our algorithm design.

Unique addresses and uniqueness of their unions allow overlapping peptide design. Tiling the peptide space with overlapping peptides leads to epitope sharing in successive peptides, thus an epitope evoking T cell activation will do so in all pools the successive peptides are distributed to. We describe the pools that each peptide is added to as its address, encoded as a binary string where each position (bit) in the string indicates a pool, with 1 for presence and 0 for absence of the peptide from the pool. For each peptide, we require a unique address, and to accommodate overlapping peptides, we also require that the union of addresses for successive peptides is unique. This ensures that each peptide overlap results in a distinct combination of activated pools, allowing for a clear interpretation of results.

Error detection by maintaining a constant number of activated pools for any given epitope. The expectation in the T cell activation assay for a tiled peptide space is the activation of the T cell by two overlapping peptides containing the T cell's cognate epitope. However, activation assays may yield false

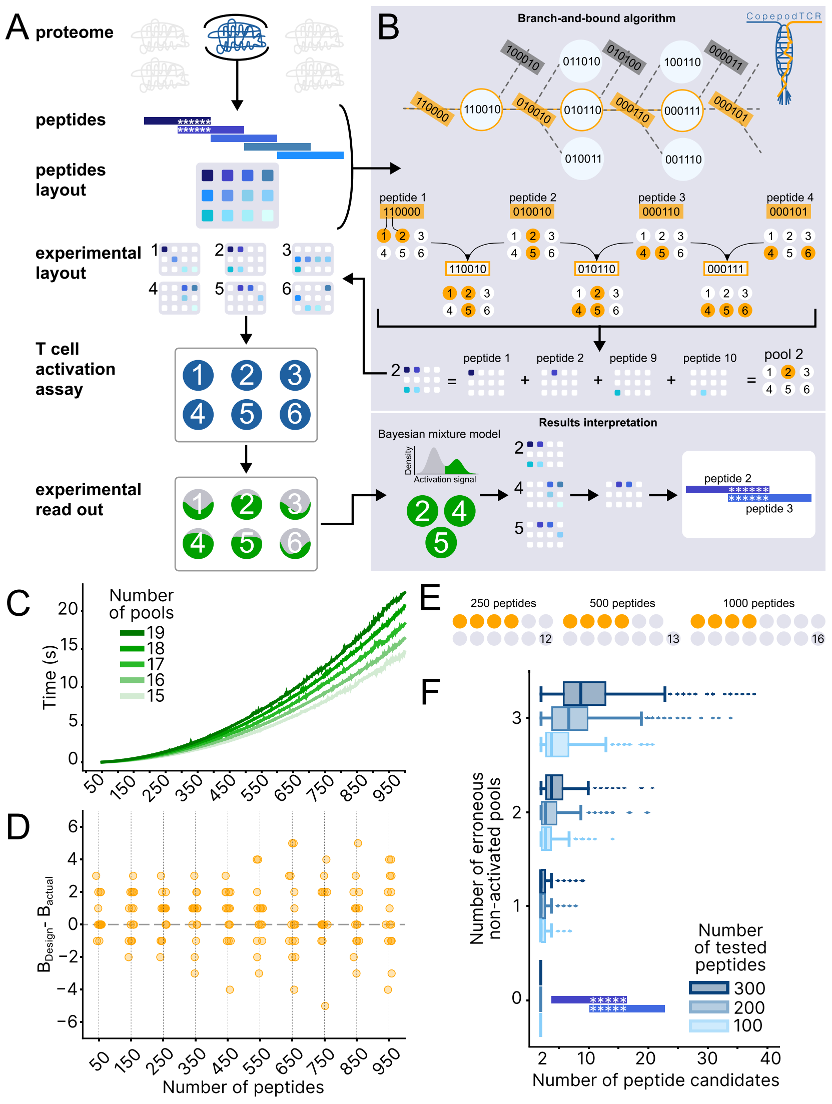

Figure 1. Combinatorial peptide pooling design. A. Experimental setup: synthetic peptides, with predefined overlap from a protein/proteome of interest are mixed according to the pooling algorithm (B) and tested for their ability to elicit a T cell response in a T cell activation assay. The results of the activation assay indicate a pair of overlapping peptides, with the overlapping section (asterisks) identified as the cognate antigen. B. Computational design: a branch-and-bound algorithm (BBA) creates a peptide mixing scheme by optimizing the peptide distribution into a predefined number of pools n (here $n = 6$). The distribution of each peptide is encoded into an address (edges in the graph), which connect nodes in the graph (circles) that represent a union between two addresses. The peptide mixing scheme constitutes the path through these unions and connecting addresses that ensure a balanced pool design. The algorithm provides the experimental layout, and uses a Bayesian mixture model to identify activated pools in the T cell activation assay (experimental read out), and returns the corresponding peptides and the overlapping antigen (B, lower panel). C. Empirical runtime analyses showing that the BBA algorithm depends on the number of pools and peptides (displayed for constant number of pools per peptide = 4). D. Deviation of BBA-derived experimental balance (B_{actual}) from optimal balance (B_{design}). The difference is displayed for a simulation of 16 pools and a constant number of pools per peptide = 4. E. Examples of the peptide pooling setup, with the indicated number of peptides and pools (number in the last circle), where yellow circles indicate the number of pools an individual peptide is added to. F. The number of possible candidates depends on the number of tested peptides and the number of erroneous non-activated pools. With any number of tested peptides, if the number of erroneously non-activated pools equals zero, the algorithm returns two peptides whose shared sequence contains the cognate epitope. Results display a simulation with 18 pools and a constant number of pools per peptide = 6.

negative results if the activation of a given pool fails. False positive results might be observed in for instance pipetting errors, or in rarer cases, might be perceived false positives if two epitopes in the tiled space evoke an activation signal. To robustly detect these errors, we require that the number of pools each peptide in the tiling assay is added to is known and remains constant. Any observed deviation in the number of activated pools therefore indicates a false positive or negative result. To achieve a constant number of expected activated pools, while maintaining the uniqueness of addresses and their unions (see above), each address and its successive unions should differ from their successors by exactly one pool, corresponding to a constant Hamming distance of two for the binary encoding of successive addresses and their unions.

Balanced pool design for low and uniform error rates. T cell activation based on TCR-pMHC interaction is influenced by a number of factors including the TCR binding strength and the 'hit rate' of encountering the cognate antigen. While TCR binding strength will be specific for each TCR and cognate peptide, the hit rate can be influenced in the experimental setup by ensuring sufficient peptide presentation to the tested T cells. Crucially, in a multiplexed assay as proposed here, the hit rate should be constant across the multiplexing axis, i.e. pools, to ensure consistent peptide exposure across experimental units. For the algorithmic design, we therefore require that the number of peptides per pool should be consistent and balanced across pools. Such balance not only ensures uniform error rates across all pools but also minimizes the overall error probability, as the error rate is spread evenly among the pools. To achieve the peptides per pool balance, the algorithm keeps track of the positional changes in the addresses, where the 0/1 transitions per position are optimized to be constant across address length.

Consequently, the requirements for the produced address arrangement are: 1) each address should be unique; 2) the union of two neighboring addresses should be unique; 3) the Hamming distance between two neighboring addresses, and the unions thereof should be identical and constant throughout the entire address arrangement; 4) the number of peptides in each pool should be similar.

The algorithm we have developed as part of *copepodTCR* operates on the principle of searching for a Hamiltonian path in a graph space representing all possible unique addresses and unique unions thereof. For a fast implementation of this path search, we devised a branch-and-bound algorithm (BBA, Figure 1B) that traverses a graph with addresses as edges and the unions of addresses as nodes. While navigating the address-union space, the BBA also simultaneously checks for union and address uniqueness coupled with the balance of the produced scheme.

Using *copepodTCR* to test TCR specificity against a large, tiling peptide library

The experimental setup starts with defining the protein/proteome of interest and obtaining synthetic peptides tiling its space. This set of peptides, containing an overlap of a predefined length, is entered into *copepodTCR* (Figure 1A). The BBA part of *copepodTCR* (Figure 1B, *Branch-and-Bound algorithm*) generates a peptide pooling scheme and, optionally, provides the pipetting scheme to generate the desired pools as either 384-well plate layouts or punch card models which could be further 3D printed and overlay the physical plate or pipette tip box (see Methods for details). Following this scheme, the peptides are mixed, and the resulting peptide pools tested in a T cell activation assay. The activation of T cells is measured for each peptide pool (Figure 1A, *experimental layout*, *T cell activation assay*, and *experimental read out*) with the assay of choice, such as flow cytometry- or microscopy-based activation assays detecting transcription and translation of a reporter gene. The experimental measurements for each pool are entered back into *copepodTCR* which employs a Bayesian mixture model to identify activated pools. Based on the activation patterns, it returns the set of overlapping peptides leading to T cell activation (Figure 1B, *Results Interpretation*).

Consistent, fast performance of *copepodTCR* for peptide screen design

We first assessed empirical run times for designing peptide pooling schemes with *copepodTCR*. Its BBA allows for the design of complex peptide pooling libraries in less than a minute. For instance, to design a peptide pooling scheme with 1,000 overlapping peptides distributed over 16 pools and each peptide added to 4 pools takes 16 seconds (Figure 1C, E). BBA run time is not dependent on the number of pools per peptide (Supplementary Fig 2B).

We next tested *copepodTCR*'s ability to achieve balanced peptide pools. We assessed balance by comparing the theoretical balance expected given the total number of peptides, pools, and peptide occurrences across pools, with the actual balance generated by the pooling scheme. *copepodTCR* consistently achieves a near-perfect balance in the distribution of peptides across pools irrespective of the total number of peptides being tested (Figure 1D). For instance, in the above example with 16 pools, and 4 pools per peptide the number of peptides per pool deviates by no more than 6 peptides from the ideal balance. On average, we achieve as low as 0.75 and 1.75 peptides deviation from the theoretical number of peptides per pool for pooling schemes of 50 and 950 peptides, respectively.

The peptide pooling scheme produced by the BBA ensures that the number of activated pools remains constant for any given epitope (Figure 1F, number of erroneously non-activated pools equal to 0). This consistency is a crucial feature that facilitates straightforward error detection in the experimental process. When an error is detected, such as a pool not being activated when it should have been, the tool effectively narrows down the list of potential peptide candidates responsible for the activation. The extent to which the list of candidates is reduced is influenced by the number of peptides tested and the number of erroneous non-activated pools. For instance, with the presence of one erroneous non-activated pool, the algorithm can refine the list of potential peptide candidates to approximately 10 (Figure 1F, number of erroneously non-activated pools equal to 1).

copepodTCR peptide pooling scheme successfully validates known TCR peptide specificity

To test that *copepodTCR*'s experimental design allows for the robust detection of true positive and negative TCR specificity, we designed a CPP assay for an NFAT-GFP reporter Jurkat TCR-transgenic cell line with known peptide specificity.

We first tiled and synthesized the SARS-CoV-2 spike protein into 253 17 amino acid (aa) long peptides with a 12 aa overlap between consecutive peptides. Peptides were mixed according to the CPP scheme generated by *copepodTCR*. The T cell activation assay was performed in triplicates for the NFAT-GFP reporter Jurkat 76.7 cell line stably expressing TCR6.3 from ref.[13] recognising the $S_{167-180}$ epitope in context of DBP1*04:(01/02). The activation signal for each pool was determined as the fraction of GFP+ (activated) T cells and measured by both flow cytometry (Figure 2A) and fluorescence microscopy (Supplementary Fig 3A).

To accurately identify which pools led to T cell activation, we applied the results interpretation module of *copepodTCR* based on a Bayesian mixture model (Supplementary Fig 3D). The model considers the activation signal to be drawn from two distinct distributions arising from the activated and non-activated pools (Figure 2B, Supplementary Fig 3B) and provides the probabilities that the value was drawn from either distribution as a criterion for pool classification (Figure 2C, Supplementary Fig 3C). The CPP scheme in conjunction with the mixture model correctly identified the known, cognate peptide of the TCR6.3 reporter cell line.

To assess the sensitivity of the approach, we mixed all 253 peptides in a single pool and then diluted the mixture 5- and 25-fold. The dilution caused a gradual decrease in the number of activated T cells, but the activation was still distinguishable when compared to the negative unstimulated control (Figure 2D). These data suggest that pools with a much larger number of peptides and thus lower concentration of each individual peptide will still trigger detectable activation. Thus, it allows one to use *copepodTCR* for larger overlapping peptide libraries covering entire viral proteomes. Applying *copepodTCR* to the 1879 peptides tiling the SARS-CoV-2 proteome resulted in 18 pools, on average containing 625 peptides each (SI table 1), but we have not validated this pooling scheme experimentally.

Lastly, we compared the experimental read out obtained by flow cytometry and fluorescent microscopy (Supplementary Fig 3A-C). Both methods yield consistent results with high correlation ($R^2 = 0.94$; Figure 2E).

Discussion

We have successfully demonstrated how *copepodTCR* can aid in the design of CPP screens and their interpretation. Here, we address some remaining challenges in the design of peptide tiling screens and potential approaches to overcome them.

Tiling the protein sequence by overlapping peptides leads to the first and last peptide in the protein only sharing overlapping sequences with their successor or predecessor, respectively. If the TCR-specific peptide sequence is located within these peptides, pools from only one address will be activated, contrary to all other peptide sequences leading to the activation of the union of two addresses. While this could be interpreted as a false negative result as one pool fewer than expected is activated, the careful design of the pooling scheme will allow the interpretation of this result within the context of the peptide library.

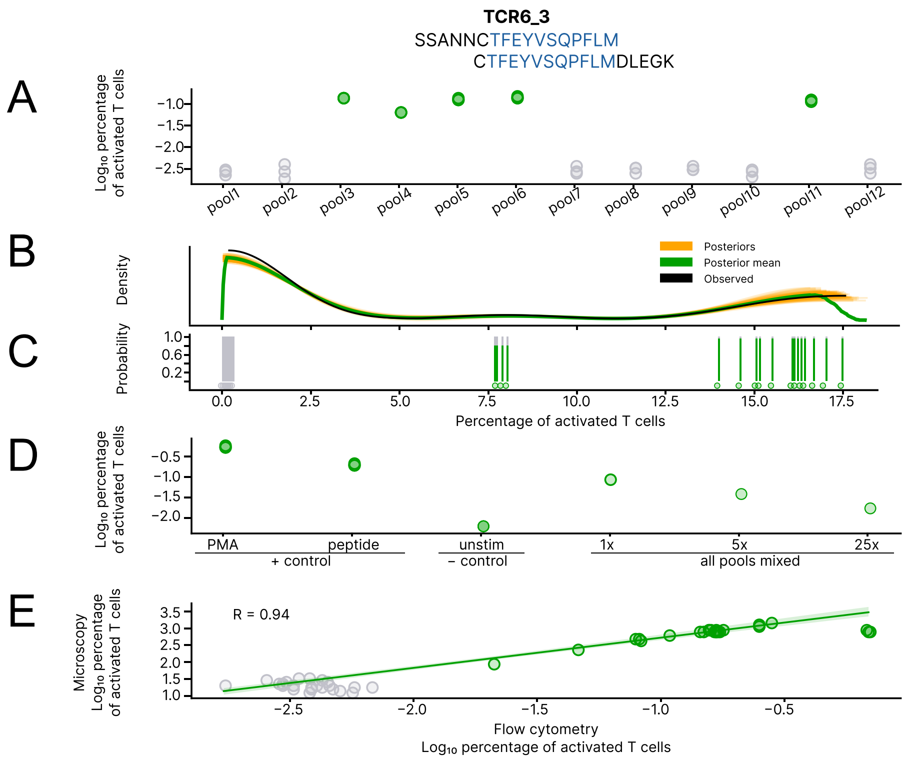

Figure 2. CopepodTCR design successfully recapitulates known TCR specificity. *CopepodTCR*'s experimental design was tested for its ability to confidently identify the known, cognate peptide (blue letters in peptide sequence) of the T cell reporter cell line expressing MHC class II restricted TCR6.3. The algorithm correctly determined that the cognate peptide is located in the overlap of the two peptides, that were uniquely identified in the activated pool scheme. A. Flow cytometry read out as percentage of GFP-producing (activated) cells in the T cell activation assay (y-axis) for each pool in the design (x-axis), with pools detected as activated/non-activated in green and grey, respectively (as in C.). B. Bayesian mixture model (green and orange) of observed (black) pool activation signal for the read out obtained by flow cytometry. C. The probability of each measurement coming from the distribution of activated pools (green) and from a distribution of non-activated pools (gray). D. Sensitivity of the assay. Left: activation of TCR6.3 cell line by PMA/ionomycin, single cognate peptide and baseline activation levels without peptide stimulation. Right: The activation signal from all pools mixed together (1x, all pools mixed), and a 5-fold (5x), or 25-fold (25x) dilution thereof. All dilutions show activation levels above unstimulated negative control. E. Correlation of flow cytometry and microscopy-based read out of the T cell activation assay, each represented by percentage of activated cells and color-coded as in A.

CopepodTCR will alert the user if the activation scheme corresponds to an address associated with an end-position peptide.

Another limitation arises when dealing with peptide sets that do not have a consistent overlap length across all peptides. This inconsistency can lead to variable numbers of pools being activated, with more activated pools for increased epitope sharing across peptides and vice versa. Such inconsistency complicates the interpretation of experimental results, as these scenarios look indistinguishable from false positives or false negatives in the assay. However, *copepodTCR* contains an inconsistency check in the experimental set-up, will alert the user of such issues and adjusts the interpretation of the results as addresses assigned to these peptides are known.

Potential peptide cross-reactivity poses a more significant challenge. In the case of cross-reactivity, several peptides containing similar epitopes would lead to the activation of the tested T cell. Cross-reactive epitopes often trigger different degrees of activation, which might aid in identifying these in post-hoc analyses. While cross-reactivity of TCRs in general is not a rare scenario, with potentially millions of different peptides recognized by a single T cell[14], it is still unlikely that two of these peptides will co-occur in a relatively small viral proteome.

For a large number of peptides to be tested, either the number of required pools or the number of peptides per pool will increase (Supplementary Fig 2A). For instance, testing 50,000 possible overlapping peptides would require a minimum of 35 pools, where each pool consists of approximately 5,700 peptides. For a stock peptide concentration of 10mM as we tested in this study, the individual peptide concentration after pooling in this set-up would be 1.75 μM. At this low concentration, the hit rate of a T cell to encounter and thus get activated by its true cognate peptide is significantly decreased. To avoid this potential problem, the number of pools or the number of pools per peptide in the experimental setup can be adjusted to have a lower number of peptides per pool. However, from our dilution experiment (shown in Figure 2D), we were able to detect activation of the MHC-class II restricted T cell line even with a 25-fold dilution for a pool with 253 peptides, corresponding to an individual peptide concentration per pool of 1.58 μM, which is lower than the estimated concentration in the large screen exemplified above.

The application of the CPP method for the deorphanization of an MHC-class I restricted T cell line remains to be explored. Peptides presented on MHC class I molecules are shorter than those presented on MHC class II molecules[15,16]. Prior to MHC loading, antigen presenting cells use a multi-step process including proteases and transporter molecules to process the pooled peptides and achieve the required lengths. Differences in processing efficiency for peptides of short lengths might result in a weakened activation signal, potentially lowering the assay efficiency for MHC class I restricted TCRs.

In this work we validated the algorithm on a single MHC class II restricted cell line with known specificity, but our approach can scale to multiple cell lines, especially if the fraction of activated cells is measured by fluorescence microscopy. CPPs generated with *copepodTCR* could also be used for investigating the activation of primary cells, as suggested for a subset of non-overlapping SARS-CoV-2 peptides in ref.[17]. In this protocol, sequencing of TCRs from T cells expressing activation surface markers after stimulation with peptide pools allows the identification of T cell clones co-occurring with peptide addresses, allowing identification of hundreds of TCR-peptide pairs in a single experiment. We expect that the small number of combinatorial pools combined with optimal assignment of peptides and robust error-correction enabled by *copepodTCR* algorithm will be beneficial for this assay.

Methods

Algorithmic constraints for the combinatorial peptide pooling scheme

Let's consider a single protein E. We create a library of peptides by a sliding window approach across its sequence, i.e., each peptide overlaps with its predecessor and successor by the constant number of amino acids. The only exception is the first and last peptide in the protein, which only overlap with the successor and predecessor respectively. In our experiment, we have a total number of M overlapping peptides E_j derived from E:

$$E = \{E_1, \ldots, E_M\}, \text{ with } overlap(E_j, E_{j+1}) = const \tag{1}$$

for which we want to find a mixing scheme, into n pools p:

$$p = \{p_1, \ldots, p_n.\} \tag{2}$$

The algorithm needs to find the optimal distribution of peptides into the pools, such that the number of pools n is minimized, the total occurrence of each peptide across pools equals x, where x is minimized, and the total number of peptides per pool is approximately constant.

We consider the distribution of peptides E_j into pools p_i as assigning addresses a_j to each peptide E_j. An address a_j is represented by a binary string with n number of digits, digit b_i encoding the presence of a peptide E_j in pool p_i. The number of digits equal to 1 in any address a_j should equal x:

$$\forall j : \sum_{i=1}^{n} b_{ji} = x \tag{3}$$

The construction of the pools is limited by the combination of the following constraints:

- The number of peptides per pool should be approximately the same for each pool:

$$\overline{\overline{p_i}} \approx w \text{ where } w = \frac{M * x}{n} \tag{4}$$

- Each address a_j differs only in one pool from its successor a_{j+1}:

$$a_{j+1} \text{ with } D_H\,(a_j,\ a_{j+1}) = 2 \tag{5}$$

  where D_H is the Hamming distance.
- For all other addresses, the Hamming distance is greater than 2:

$$D_H\,(a_j,\ a_k) > 2 \text{ where } |j - k| \geq 2 \tag{6}$$

- The Hamming distance between the union of two adjacent addresses and any other union of adjacent addresses is equal or greater than 2:

$$\forall j, k\ :\ D_H\,(a_j \cup a_{j+1},\ a_k \cup a_{k+1}) \geq 2 \tag{7}$$

Branch-and-bound algorithm

The algorithm searches for an arrangement of addresses A:

$$A = \{a_1, a_2, a_3, \ldots, a_M\} \tag{8}$$

where M is the total number of tested peptides. Each address a_j corresponds to a peptide E_j. The produced arrangement should meet all criteria described above.

Each address is encoded as a binary string with n bits, where n is equal to the total number of pools. Each digit b_i in an address a_j corresponds to a pool p_i, where $b_i = 1$ means the presence of a peptide E_j in pool p_i, and $b_i = 0$ means the absence of a peptide E_j in pool p_i. Two consecutive addresses a_j and a_{j+1} form a union u_j:

$$a_j \cup a_{j+1} = u_j \tag{9}$$

Together, alternating addresses and unions form an extended arrangement AU:

$$AU = \{a_1, u_1, a_2, u_2, a_3, u_3, \ldots, a_{M-1}, u_{M-1}, a_M\} \tag{10}$$

The union u_j also is encoded as a binary string with each digit b_i corresponding to a pool p_i. $b_i = 1$ in u_j means that $overlap(E_j, E_{j+1})$ is present in pool p_i, and $b_i = 0$ indicates its absence in pool p_i.

The arrangement of addresses A is characterized by its balance:

$$B = \{w_1, w_2, w_3, \ldots, w_n\} \tag{11}$$

where n is the total number of pools, and w_i represents total number of peptides in pool p_i and is calculated as:

$$w_i = \sum_{k=1}^{M} a_k[i] \tag{12}$$

where M is the total number of peptides, a_k is an address, and $a_k[i]$ is its i bit.

The algorithm works as follows:

1. initiate with arbitrary a_1 and u_1:

$$AU = \{a_1, u_1\} \tag{13}$$

2. produce all possible next addresses that can together with a_1 form u_1, and discard addresses already present in A;
3. calculate the distortion D to balance B introduced by adding each address to A with the addition of a small random number ϵ:
$$D = \mathrm{Var}(B_{new}) - \mathrm{Var}(B) + \epsilon \tag{14}$$
where $0.0 \leq \epsilon < 1.0$.
4. sort the list of possible addresses based on D and add the first one to the arrangement A and to the extended arrangement AU:
$$A = \{a_1, a_2\} \tag{15}$$
$$AU = \{a_1, u_1, a_2\} \tag{16}$$
5. take the two last components of AU (u_1 and a_2), generate a list of all possible unions based thereon and discard unions already present in AU;
6. calculate the distortion D to balance B introduced by adding each union to A with the addition of a small random number ϵ.
7. sort the list of possible unions based on D and add the first one to the extended arrangement AU:
$$AU = \{a_1, u_1, a_2, u_2\} \tag{17}$$
8. take the two last components of AU and continue with 2., which leads to traversing through space of all possible addresses and unions.
9. stop when the length of the arrangement A is equal to M:
$$\overline{\overline{A}} = M \tag{18}$$

In *copepodTCR*, the algorithm is implemented recursively, allowing for backtracking if it reaches a dead end, enabling it to select another address in the list of next addresses, and another union in the list of next unions.

Peptide mixture scheme

The address a_i from the produced arrangement A corresponds to peptide E_i. Based on a_i, the peptide pooling scheme is generated, where an address indicates to which pool p_i each peptide is added.

To aid with the experimental procedure, *copepodTCR* offers the creation of STL files, encoding the pipetting scheme necessary to create peptide pools based on the arrangement of experimental peptides E_i in a 384-well plate. STL files are generated in Python using *Trimesh* (v3.23.5)[18]. Each card represents one pool, with holes positioned at the coordinates corresponding to the peptides designated for addition to that pool.

Bayesian mixture model

We developed a hierarchical Bayesian mixture model (Supplementary Fig 3D) to interpret the results of pooled T cell activation assays. As input, the model requires an activation signal, Act, expressed as a percentage of activated cells. We measured activation by either flow cytometry or microscopy and computed the percentage of activated cells divided by all cells in one sample (flow cytometry) or the number of activated cells per sample divided by total number of activated cells across all samples (microscopy).

We assume Act is composed of two distributions, reflecting activated and non-activated pools. We model these as two truncated Normal distributions μ_{Pos} and μ_{Neg} with parameters μ, σ and lower truncation limit a:

$$\mu_{\mathrm{Pos}} \sim \mathrm{TruncatedNormal}(\mu, \sigma, a = 0) \tag{19}$$
$$\mu_{\mathrm{Neg}} \sim \mathrm{TruncatedNormal}(\mu, \sigma, a = 0) \tag{20}$$

For each replicate of the pool experiments, we model Act as a TruncatedNormal distribution with parameters μ_{pool}, σ_{pool}, and lower truncation limit a:

$$Act \sim \mathrm{TruncatedNormal}(\mu_{pool}, \sigma_{pool}, a = 0) \tag{21}$$

μ_{pool} is drawn from the truncated Normal distribution assigned to each pool with parameters μ_{Pos} or μ_{Neg}, σ, and lower truncation limit a:

$$\mu_{pool} \sim \text{TruncatedNormal}(\mu_{\text{Neg or Pos}}, \sigma, a = 0) \tag{22}$$

μ_{Pos} or μ_{Neg} are assigned to each pool according to L following the Bernoulli distribution:

$$L \sim \text{Bernoulli}(p = 0.5, q = 0.5) \tag{23}$$

To assign a pool activation status, we compute the probability of being drawn from the distribution of non-activated pools L and of being drawn from the distribution of activated pools $1 - L$.

A pool is considered to be activated if L is less than 0.5. The model was developed and fitted on the observed data using PyMC (version 5.9.2)[19].

Peptide pooling

Individual 17-mer peptides, overlapping by 12 amino acids from the SARS-CoV-2 Spike protein (Uniprot acc. P0DTC2, synthesized by Mimotopes), were diluted to 10mM in a matching solvent according to the manufacturer's solubility test. The diluted peptides were transferred to a 384-well plate (951020702, Eppendorf).

For the dilution experiment, we mixed all 253 peptides in a single pool and then diluted the mixture 5- and 25-fold, corresponding to concentrations for each individual peptide in these pools of 39.5 μM, 7.91 μM, and 1.58 μM, respectively.

We used *copepodTCR* to design a peptide pooling scheme and corresponding STL files for punched cards matching an empty 12.5 μl GripTIPS box (Integra), generated G-code with Cura v. 5.1.0 (Ultimaker), and printed them with PLA on a 3D printer (Ender V3, Creality). Each punched card encoded the pooling strategy for a single combinatorial pool: we placed a card on top of an empty tip box, filled open holes with tips, and then use this patterned pipette tip array to transfer 2.5 μl from the source plate to the lid of the tip box using a ViaFLO 384-channel electronic pipette (Integra). The tip box lid was then pulse centrifuged at 1000g to collect droplets in the corner and pooled peptides were transferred into an 8-tube strip and stored at -80°C before use.

Stimulation of reporter T cell lines with combinatorial pools

For validation of our approach, we used a NFAT-GFP reporter Jurkat 76.7 cell lines from ref.[13] (TCR6.3) with known specificity. Jurkat cells (10^5) were co-cultured in a round-bottom 96-well cell culture plate (Corning 3799) in 100 μl RPMI-1640 media (Gibco) supplemented with 10% FBS, 1% penicillin-streptomycin, containing 1 μg/ml anti-CD28 antibody (BD Biosciences, 555725) and 1 μg/ml of anti-CD49d antibody (BD Biosciences, 555501), with 10^5 PBMCs from a healthy HLA-DPB04:02+ donor pulsed with 1 μl of the given combinatorial pool for 16 hours. For positive controls we used PMA/Ionomycin cocktail (Invitrogen, 00-4970-93) according to manufacturer protocol, or single CTFEYVSQPFLMDLEGK peptide (known TCR6.3 target epitope) at final concentration in media 1μM. Each coculture experiment was performed in triplicates. Plates were washed with 300 μl of FACS buffer (DPBS with 0.05% BSA and 2mM EDTA) and resuspended in 300 μl of FACS buffer. Subsequently, 150 μl were transferred into a 96-well flat-bottom plate (Corning 3596) and acquired on the Incucyte S3 (Sartorius) with a 4x objective. Green object count for each well was determined using Incucyte Analysis Software with Surface Fit segmentation, no edge split, 2.0 GCU Threshold and $< 750\mu m^2$ Area filter. Simultaneously, a second plate was acquired on BD Symphony A2 with a high throughput sampler (Supplementary Fig 4 for gating strategy). Flow cytometry data was analysed with FlowJo (v. 10.8.1).

Results were interpreted using the Bayesian Mixture model of *copepodTCR*.

copepodTCR Python package

A python package for *copepodTCR* is available via *pip*. It relies on python packages CVXPY (version 1.3.2)[20], pandas (version 1.5.3)[21], NumPy (version 1.23.5)[22], *Trimesh* (version 3.23.5)[18], and PyMC (version 5.9.2)[19]. Its detailed documentation be accessed at copepodTCR.readthedocs.io.

copepodTCR tool

We developed an interactive user interface for *copepodTCR* using Shiny for Python (version 0.5.1)[23]. Besides the *copepodTCR* Python package, the tool uses Seaborn (version 0.12.2)[24], pandas (version 1.5.3)[21], and

Matplotlib (version 3.8.0)[25].

CopepodTCR assists the user in all stages of the combinatorial peptide pooling experiment. It helps to select the appropriate number of pools and peptide occurrence, checks the list of given peptides for overlap consistency and simulates activated pools for any epitope of a given length; it generates the peptide pooling scheme and produces STL files for peptide mixing; it analyzes the experimental results, and returns a set of peptides responsible for those results.

CopepodTCR can be accessed on https://copepodtcr.cshl.edu/.

Author contributions

VAK, PGT, MVP, and HVM conceptualized the work; VAK developed the software and implemented the user interface together with SRC; CB advised on algorithm design; VAK, DJP and HVM developed the hierarchical model; VAK conducted the formal analyses; MVP, AAM, KAR, AJS and VAK performed the experiments; VAK, MVP, and HVM wrote the original draft; all authors reviewed & edited the final draft; MVP, PGT, and HVM supervised the work.

Code availability

Custom analysis code was written in python (version $\geq$ 3.10.11). Code and data to re-produce all figures in the manuscript are freely available on GitHub: https://github.com/meyer-lab-cshl/copepodTCR_paper.git.

Competing interests

PGT has consulted for Pfizer, JNJ, Cytoagents, 10X, and Illumina, and serves on the SAB for Shennon Bio and Immunoscape. PGT, AAM, and MVP have patents related to TCR amplification, cloning, and applications thereof. The authors declare no other competing interests.

Acknowledgments

The authors thank Jeremy Shaw, Beiyun Liu, Luigi Mari and R.K. Subbarao Malireddi for their help with microscopy assays, and Anastasia Troshina for the *copepodTCR* logo design.

Funding

The research was supported by the Simons Center for Quantitative Biology at Cold Spring Harbor Laboratory; US National Institutes of Health Grants S10OD028632-01, U01AI150747, and R01AI136514. This work was discussed in part at the Aspen Center for Physics, which is supported by National Science Foundation grant PHY-2210452. The funders had no role in the template design or decision to publish.

References

1. Sbai, H, Mehta, A & DeGroot, A. Use of T cell epitopes for vaccine development. *Current drug targets-Infectious disorders* **1** (2001).
2. Waldman, AD, Fritz, JM & Lenardo, MJ. A guide to cancer immunotherapy: from T cell basic science to clinical practice. *Nature Reviews Immunology* **20** (2020).
3. Prinz, JC. Immunogenic self-peptides-the great unknowns in autoimmunity: Identifying T-cell epitopes driving the autoimmune response in autoimmune diseases. *Frontiers in Immunology* **13** (2023).
4. Davis, MM & Bjorkman, PJ. T-cell antigen receptor genes and T-cell recognition. *Nature* **334** (1988).
5. Zhao, W & Sher, X. Systematically benchmarking peptide-MHC binding predictors: From synthetic to naturally processed epitopes. *PLOS Computational Biology* **14** (2018).
6. Meysman, P, Barton, J, Bravi, B, Cohen-Lavi, L, Karnaukhov, V, Lilleskov, E, Montemurro, A, Nielsen, M, Mora, T, Pereira, P, *et al.* Benchmarking solutions to the T-cell receptor epitope prediction problem: IMMREP22 workshop report. *ImmunoInformatics* **9** (2023).
7. Dens, C, Laukens, K, Bittremieux, W & Meysman, P. The pitfalls of negative data bias for the T-cell epitope specificity challenge. *Nature Machine Intelligence* **5** (2023).
8. Chaves, FA, Lee, AH, Nayak, JL, Richards, KA & Sant, AJ. The Utility and Limitations of Current Web-Available Algorithms To Predict Peptides Recognized by CD4 T Cells in Response to Pathogen Infection. *The Journal of Immunology* **188** (2012).
9. Joglekar, AV & Li, G. T cell antigen discovery. *Nature methods* **18** (2021).
10. Fiore-Gartland, A, Manso, BA, Friedrich, DP, Gabriel, EE, Finak, G, Moodie, Z, Hertz, T, De Rosa, SC, Frahm, N, Gilbert, PB, *et al.* Pooled-peptide epitope mapping strategies are efficient and highly sensitive: an evaluation of methods for identifying human T cell epitope specificities in large-scale HIV vaccine efficacy trials. *PloS one* **11** (2016).
11. Klinger, M, Pepin, F, Wilkins, J, Asbury, T, Wittkop, T, Zheng, J, Moorhead, M & Faham, M. Multiplex identification of antigen-specific T cell receptors using a combination of immune assays and immune receptor sequencing. *PloS one* **10** (2015).
12. Snyder, TM, Gittelman, RM, Klinger, M, May, DH, Osborne, EJ, Taniguchi, R, Zahid, HJ, Kaplan, IM, Dines, JN, Noakes, MT, *et al. Magnitude and Dynamics of the T-Cell Response to SARS-CoV-2 Infection at Both Individual and Population Levels* 2020.
13. Mudd, PA, Minervina, AA, Pogorelyy, MV, Turner, JS, Kim, W, Kalaidina, E, Petersen, J, Schmitz, AJ, Lei, T, Haile, A, *et al.* SARS-CoV-2 mRNA vaccination elicits a robust and persistent T follicular helper cell response in humans. *Cell* **185** (2022).
14. Sewell, AK. Why must T cells be cross-reactive? *Nature Reviews Immunology* **12** (2012).
15. Szeto, C, Lobos, CA, Nguyen, AT & Gras, S. TCR recognition of peptide–MHC-I: Rule makers and breakers. *International journal of molecular sciences* **22** (2020).
16. La Gruta, NL, Gras, S, Daley, SR, Thomas, PG & Rossjohn, J. Understanding the drivers of MHC restriction of T cell receptors. *Nature Reviews Immunology* **18** (2018).
17. Snyder, TM, Gittelman, RM, Klinger, M, May, DH, Osborne, EJ, Taniguchi, R, Zahid, HJ, Kaplan, IM, Dines, JN, Noakes, MT, *et al.* Magnitude and dynamics of the T-cell response to SARS-CoV-2 infection at both individual and population levels. *MedRxiv* (2020).
18. Dawson-Haggerty et al. *trimesh* version 3.2.0. https://trimsh.org/.
19. Oriol, AP, Virgile, A, Colin, C, Larry, D, J., FC, Maxim, K, Ravin, K, Jupeng, L, C., LC, A., MO, *et al.* PyMC: A Modern and Comprehensive Probabilistic Programming Framework in Python. *PeerJ Computer Science* **9** (2023).
20. Diamond, S & Boyd, S. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research* **17** (2016).
21. McKinney, W. *Data Structures for Statistical Computing in Python* in *Proceedings of the 9th Python in Science Conference* (2010).
22. Harris, CR, Millman, KJ, van der Walt, SJ, Gommers, R, Virtanen, P, Cournapeau, D, Wieser, E, Taylor, J, Berg, S, Smith, NJ, *et al.* Array programming with NumPy. *Nature* **585** (2020).
23. Development team, TS. *Shiny for Python* https://github.com/rstudio/py-shiny.
24. Waskom, ML. seaborn: statistical data visualization. *Journal of Open Source Software* **6** (2021).
25. Hunter, JD. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering* **9** (2007).