Improving Modularity, Efficiency, and Functionality of APE-Gen2.0

Alexandra Shewchuk[1], Romanos Fasoulis[2], Maurício M. Rigo[2] and Lydia E. Kavraki[2]

[1] *Tufts University, Medford, MA 20155, USA; alexandra.shewchuk@tufts.edu*
[2] *Department of Computer Science, Rice University, Houston, TX 77005, USA*

Abstract

The Major Histocompatibility Complex (MHC) is a protein receptor in the adaptive immune system. MHCs bind to and display peptides at the surface of the cell for T-cells to recognize. Modeling and analyzing peptide-MHC (pMHC) bindings has the potential to predict novel pMHC interactions, which can offer insight into immune responses. Many tools exist to model pMHC binding, both sequence-based and structure-based. However, many structure-based approaches offer limited modeling parameters and are unable to handle molecular modifications to the proteins. This article describes APE-Gen2.0 (Anchored Peptide-MHC Ensemble Generator 2.0), a more robust version of APE-Gen that offers more modeling arguments, handles peptide post-translational modifications, creates a user-friendly web server, and restructures the code base in a modular fashion. Furthermore, this tool will model both Class I and Class II MHCs, whereas the original version exclusively modeled Class I MHCs.

Keywords

MHC class I; MHC class II; peptide binding; adaptive immunity; molecular docking

1 Background and Related Work

In the adaptive immune system, T-cells are responsible for monitoring other cells in the body to identify which are healthy and which should be targeted for an immune response [1]. To accomplish this, T-cells interact with peptides bound to Major Histocompatibility Complexes (MHCs) at the cell surface.

There are two classical types of MHCs: Class I (MHC-I) and Class II (MHC-II). MHC-I molecules are found in almost all nucleated cells and bind to intracellular peptides that are 8-11 residues in length [2]. The peptide-MHC (pMHC) complexes travel to the surface of the cell and will interact with T Cell Receptors (TCRs). If a T-cell recognizes the peptide as “non-self”, the cell will be marked for programmed cell death. Mutations or malfunctions related to pMHCs are associated with cancer and autoimmune diseases. Conversely, discovering peptides that can be used to trigger an immune response is the subject of many immunotherapy treatments and peptide vaccines [3].

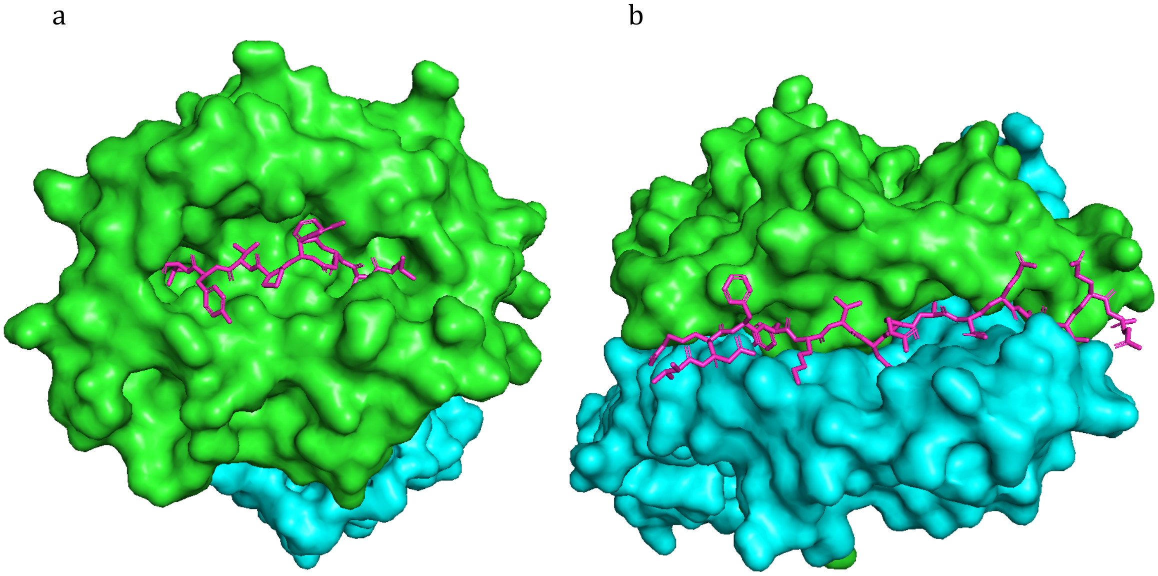

Figure 1: (a) a model of HLA-A*02:01, a Class I MHC, bound to peptide LLFGYPVYV;
(b) model of HLA-DR11, a Class II MHC, bound to peptide VDRFYKTLRAEQASQE;
Class I pMHC peptides embed within the α chain, whereas Class II pMHC peptides bind between the α and β chains

MHC-II molecules are found in antigen-presenting cells, such as B-cells and dendritic cells and bind to peptides (13-20 residues in length) originating from the extracellular environment [2]. These include proteins from viruses, bacteria, fungi, and other pathogens. As with pMHC-I, the pMHC-II complexes travel to the surface of the cell to be recognized by TCRs. However, in this case there is no cell apoptosis. If the peptide being presented is recognized as "non-self", the cell is assumed to be infected by an external pathogen and helper T-cells will become activated, triggering an immune pathway that culminates with the recruitment of other immune cells to help in the clearance of the infection. MHC-II molecules are used to build immunity against foreign pathogens, and thus continue to be extensively studied, especially in the wake of the COVID-19 pandemic. However, MHC-II are also involved in the rejection of organ transplants; for this reason, recipients of organ transplants must take immunosuppressants in order to accept the transplant [4].

Independently of MHC class, immune responses to both endogenous and exogenous substances rely on the binding of peptides to MHCs. Modeling, analyzing, and better understanding the three-dimensional structure of these pMHCs will allow researchers to identify the origins of diseases, diagnose patients, and develop new treatments.

Many tools exist to model peptide-MHCs interactions, as well as how strong these interactions will be. Binding affinity, which is the measure of how strong the peptide and MHC will bind, can be predicted with both sequence- and structure-based tools, while the models of these interactions are solely structure-based. Sequence-based peptide-MHC binding predictors include MHCFlurry and NetMHCPan/NetMHCPanII [8, 9]. These tools are developed using deep learning methods trained on sequence data for peptides and MHCs, and though they perform very well on pMHCs that are similar to the training data, these models are very susceptible to overfitting and often perform poorly on unseen conformations.

Structure-based approaches are gaining popularity as the ability to model and predict the shapes of three-dimensional molecules improves. Structure-based tools typically operate by reducing the energy of molecular interactions. Because stable, low-energy states are the biologically most likely states, energy reduction techniques can generate pMHC conformations that may be found in live cells. The metric commonly used to evaluate structure-based tools is the root mean square deviation (RMSD) between the heavy atoms in the backbone of the generated peptide and the peptide from the crystal structure.

More general protein-ligand modeling tools include Autodock Vina and DINC, which efficiently explore expansive conformation spaces [5, 6]. While these tools can often yield good results for peptide-MHC bindings, they tend to be significantly slower than more specialized tools [7]. APE-Gen is a specialized pMHC modeling tool [10]. APE-Gen models peptides within the binding cleft of Class I MHCs by starting with a template peptide conformation, anchoring two residues – typically the amino acids 1-2 positions from the peptide's termini. From there, the residues between the two anchors are removed, then the backbone of these removed amino acids is reconstructed. Reconstruction is performed using Random Coordinate Descent (RCD), an inverse kinematics algorithm that generates a backbone in different positions.

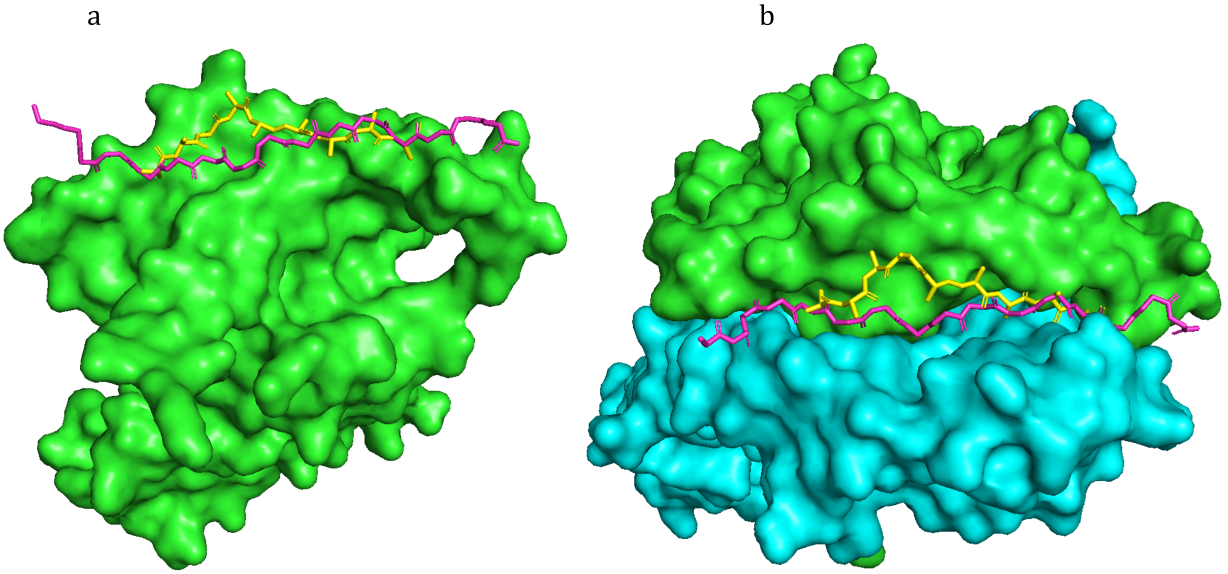

Figure 2: Ground truth peptide backbone, in pink, and an RCD reconstructed peptide backbone, in yellow, displayed by HLA-DR11, a human Class II MHC; (a) peptide side view against MHC α chain; (b) peptide top view against MHC α and β chains

This terminal-based tool can take in PDB-formatted pMHC structure files for redocking (also called self-docking) to refine existing structures, or it can generate new structures by taking in a MHC allotype and a peptide sequence, then modeling the peptide based on the MHC template.

2 Methods and Results

2.1 Improving modularity

When developing large-scale tools, oftentimes new features are incorporated quickly in order to create a functioning version. Software development best practices would recommend finding instances where code could be reworked to factor out repeated lines, attributes and methods could be added to objects in order to reduce the number arguments being passed into functions, and various other techniques to make the codebase more readable, modular, and easy to modify.

Below is an example of a common pattern repeated in several instances throughout the codebase:

Code Block 1

```
rechained = replace_chains(self.pdb_filename, "A", "C")
overwritten = ''.join(rechained)
with open(self.pdb_filename, 'w') as PTMed_file:
    PTMed_file.write(overwritten)
```

```
overwritten = replace_HETATM(peptide_file)
overwritten = ''.join(overwritten)
with open(peptide_file, 'w') as peptide_handler:
    peptide_handler.write(overwritten)
```

```
conect = replace_CONECT_fields(residue_name + '.conect', bonds_list)
conected = ''.join(conect)
conect_file = residue_name + '.pdb'
with open(conect_file, 'w') as conect_handler:
    conect_handler.write(conected)
file_list.append(conect_file)
```

These three code blocks all follow this pattern:
> Run a function that takes in an input file and arguments; store the returned object
> Convert this return object into a joined string
> Write this string to an export file
> Conditionally, store this export file in a variable to be used later on

Using functional programming paradigms, we constructed a higher-order function — `apply_function_to_file` — that takes in first-class functions:

Code Block 2

```
def apply_function_to_file(func, input_file, output_file="", **kwargs):
    if output_filename == "": output_filename = input_filename
    overwritten = func(input_file, **{key: value for key, value in kwargs.items() if
key
                                                          in
                                                   func.__code__.co_varnames})
    overwritten = ''.join(overwritten)
    with open(output_file, 'w') as output:
        output.write(overwritten)
    return output_file
```

With this this function, the repeated code in Code Block 1 is reduced to the following:

Code Block 3

```
apply_function_to_file(replace_chains,self.pdb_filename,chain_from="A", chain_to="C")
```

```
apply_function_to_file(replace_HETATM, peptide_file)
```

```
file_list.append(apply_function_to_file(replace_CONECT_fields,residue_name+ '.conect',
                                        output_filename=residue_name + '.pdb',
                                        bonds_list=bonds_list))
```

By reducing several lines of code into a single one, these unclear operations can be more easily understood by readers and the codebase becomes less cluttered.

In a similar vein, several functions throughout the codebase took in several parameters that all pertained to a peptide representation. Rather than clutter each function call with several arguments, such as the index or the list of post-translational modifications, we added these as attributes to the Peptide object representation. This way, only the single Peptide object is passed into the function, which reduces the overhead of understanding what each function takes in.

2.2 Flexible residues speedup

In MHCs, there are certain residues that are considered to be flexible — that is, they can take on different conformations. To generate pMHC structures, the atoms of the flexible residues must be fixed, but in doing so, the atom labels are lost and must be recovered. This is accomplished by taking the fixed backbone carbon atoms, then generating constraints based on the bonds between these carbons and the other atoms of the residue. With these bonds, a constraint satisfaction solver can be used to determine the labels of the atoms.

For each pMHC conformation, dozens of flexible residues must be fixed. When generating several hundred potential pMHC structures, this operation ends up being very costly. Originally, the function for fixing residues ran in an average of 0.380 seconds per peptide (as averaged across 1000 function calls).

At its core, program profiling — understanding where bottlenecks and inefficiencies crop up — aims to reduce operations. A simple way to reduce program runtime is to eliminate unnecessary function calls; this way, it is possible to reduce the total number of instructions, and, because each function call creates its own stack frame, less time will be spent on machine-level operations. With this in mind, we simplified the flexible residue fixer by assigning variables within a parent function, rather than defining variables to the same values in each child function call. Additionally, by reducing the number of variables being reassigned, fewer cache writebacks need to occur, which can improve temporal locality. With only small changes, the average time spent in the residue fixer function decreased to an average of 0.347 seconds per peptide (as averaged across 1000 function calls).

This time reduction from 0.380 s to 0.347 s corresponds to a 1.10*x* speedup in latency.

2.3 Class II Loop Reconstruction

Like many pMHC modeling tools, the original version of APE-Gen models only Class I MHCs. Class I and Class II MHCs are functionally different in many ways, but structurally they differ in how they bind to their peptides. MHC-I molecules have a deeper binding cleft that strongly anchors the ends of the peptides while the middle arches outward; the peptide of MHC-II molecules have a more open cleft, and the peptide conformations have straighter backbones and longer ends that are not bound to the MHC.

To determine whether the APE-Gen2.0 workflow, which dealt originally with only Class I MHCs, would function on Class II MHCs, we performed the random coordinate descent (RCD) algorithm on a pMHC with a known peptide core. This generated the backbone of the peptide core, as well as its energy and heavy atom RMSD (as compared to the known peptide core).

Out of 100 peptide conformations, the lowest RMSD value was 1.073 Å. Because the RMSD value is under the accepted 2.00 Å threshold, this demonstrates that RCD is able to successfully reconstruct the peptide backbone. Thus, class II pMHCs can be incorporated into APE-Gen2.0.

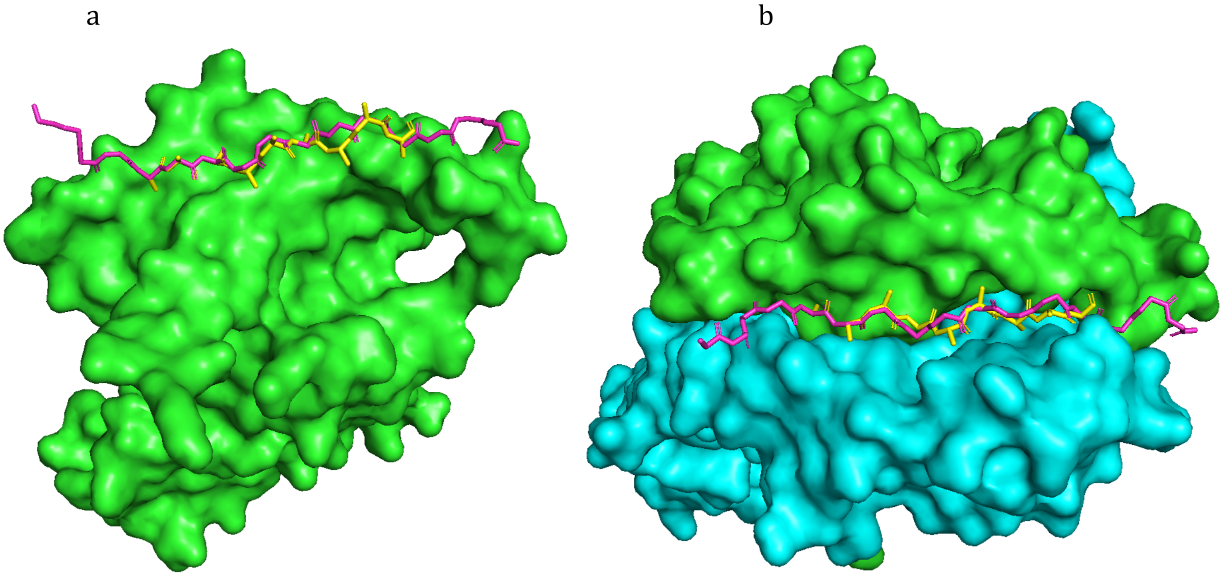

Figure 3: Ground truth peptide backbone, in pink, and the lowest RMSD reconstructed peptide backbone, in yellow, displayed by HLA-DR 11, a human Class II MHC; (a) peptide side view against MHC α chain; (b) peptide top view against MHC α and β chains

2.4 Class II Data Collection

Once the RCD trial demonstrated that Class II MHCs can be incorporated into the APE-Gen2.0 workflow, more Class II pMHC ground truth data needed to be cleaned and verified. From a PDB bank, we downloaded 47 PDB files, and, because the α, β, and peptide chain IDs were not consistent across all PDB files, we hand-determined which chain IDs needed to be extracted and renumbered. This was performed using a simple script and external PDB processing tools [11].

3 Discussion and Next Steps

To incorporate Class II pMHC modeling into APE-Gen2.0, there are several considerations that must be made. Namely, identifying the peptide cores, which are 9-residue motifs that identify anchors within the longer peptide; adding Class II templates; and thorough testing and comparison to other Class II pMHC tools.

3.1 Peptide Cores

The peptides that bind to Class II MHCs are longer than those that bind to Class I MHCs and do not show consistent anchors across all peptides of the same length. However, it appears that all peptides have a 9-mer core, with anchors in positions 1, 4, 6, and 9. Though these values have not been computationally proven, Figure 4 shows the cores as visually apparent through visualization tools.

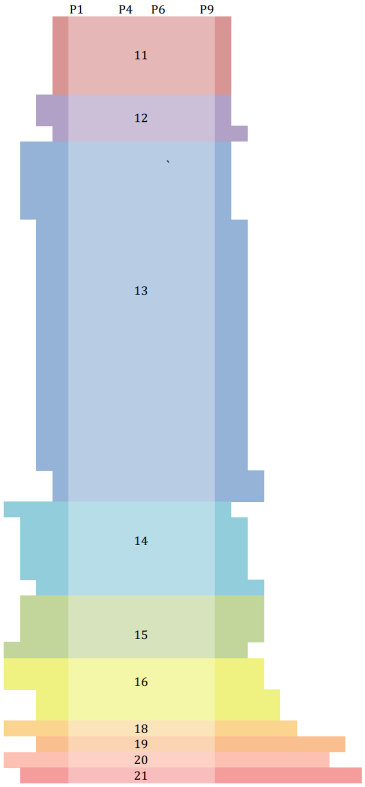

Figure 4: The values in the center of this chart show the length of the peptides, grouped by color; (e.g. all the 13-mer peptides are shown in blue) The lighter middle regions are the 9-mer cores. The darker flanking regions show the residues and their positions outside the core. While there is always at least one residue on either flank, no other clear core identification rules exist, so the present strategy of determining anchors – by finding residues closest to a beta-pleat on the MHC – may prove useful for determining Class II peptide anchors.

3.2 Templates

As with Class I MHCs, many templates can be included in the APE-Gen source code so that MHCs with unknown structures or binding motifs can be approximated through homology modeling. Firstly, deeper research into non-canonical anchors should be performed, and secondly, homology modeling for Class II MHCs should be reviewed and the modeled structures should be validated against ground truth structures. Additionally, to reduce time spent searching for close matches, it may be valuable to quickly differentiate Class II and Class I structures so not all sequences are compared.

3.3 Testing

Once Class II MHCs have been incorporated into the workflow, the generated pMHC structures should be compared to ground truth structures to ensure reasonable results. Additionally, the RMSD values of the generated structures compared to the ground truth structures should be compared against other Class II pMHC binding prediction tools.

4 Conclusion and Additional Resources

Improvements to the workflow, features, and modularity of APE-Gen will continue to be made, and the results will be shared with the public in the coming months.

For a presentation of the work described in this paper, please refer to this link:

DREU Presentation

5 Acknowledgements

Special thanks to Anja Conev and Maurício Rigo for their invaluable feedback during the review of this paper.

6 References

1. Antunes, D.A.; Abella, J.R.; Devaurs, D.; Rigo, M.M.; Kavraki, L.E. Structure-based methods for binding mode and binding affinity prediction for peptide-MHC complexes. *Curr. Top. Med. Chem.* 2018, 18, 2239–2255.
2. Wieczorek M., Abualrous E.T., Sticht J., Álvaro-Benito M., Stolzenberg S., Noé F., Freund C. Major Histocompatibility Complex (MHC) Class I and MHC Class II Proteins: Conformational Plasticity in Antigen Presentation. *Front Immunol*. 2017 Mar 17;8:292. doi: 10.3389/fimmu.2017.00292. PMID: 28367149; PMCID: PMC5355494.
3. Janeway C.A. Jr, Travers P., Walport M., et al. Immunobiology: The Immune System in Health and Disease. 5th edition. New York: *Garland Science*; 2001. The major histocompatibility complex and its functions. https://www.ncbi.nlm.nih.gov/books/NBK27156/
4. Yuzefovych, Y., Valdivia, E., Rong, S., Hack, F., Rother, T., Schmitz, J., Bräsen, J. H., Wedekind, D., Moers, C., Wenzel, N., Gueler, F., Blasczyk, R., & Figueiredo, C. (2020). Genetic Engineering of the Kidney to Permanently Silence MHC Transcripts During *ex vivo* Organ Perfusion. *Frontiers in immunology*, *11*, 265. https://doi.org/10.3389/fimmu.2020.00265
5. O. Trott, A. J. Olson, AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization and multithreading, *Journal of Computational Chemistry* 31 (2010) 455-461
6. D. Devaurs, D. A. Antunes, S. Hall-Swan, N. Mitchell, M. Moll, G. Lizée, and L. E. Kavraki, "Using parallelized incremental meta-docking can solve the conformational sampling issue when docking large ligands to proteins," *BMC Molecular and Cell Biology*, vol. 20, no. 1, p. 42, 2019.
7. Antunes, D.A., Devaurs, D., Moll, M. *et al.* General Prediction of Peptide-MHC Binding Modes Using Incremental Docking: A Proof of Concept. *Sci Rep* 8, 4327 (2018). https://doi.org/10.1038/s41598-018-22173-4
8. T. O'Donnell, A. Rubinsteyn, U. Laserson. "MHCflurry 2.0: Improved pan-allele prediction of MHC I-presented peptides by incorporating antigen processing," *Cell Systems*, 2020. https://doi.org/10.1016/j.cels.2020.06.010

9. Birkir Reynisson, Bruno Alvarez, Sinu Paul, Bjoern Peters, Morten Nielsen, NetMHCpan-4.1 and NetMHCIIpan-4.0: improved predictions of MHC antigen presentation by concurrent motif deconvolution and integration of MS MHC eluted ligand data, *Nucleic Acids Research*, Volume 48, Issue W1, 02 July 2020, Pages W449–W454, https://doi.org/10.1093/nar/gkaa379
10. Abella, J.R.; Antunes, D.A.; Clementi, C.; Kavraki, L.E. APE-Gen: A Fast Method for Generating Ensembles of Bound Peptide-MHC Conformations. *Molecules* 2019, *24*, 881. https://doi.org/10.3390/molecules24050881
11. Rodrigues JPGLM, Teixeira JMC, Trellet M and Bonvin AMJJ. pdb-tools: a swiss army knife for molecular structures. F1000Research 2018, 7:1961. https://doi.org/10.12688/f1000research.17456.1