Vaxrank: A computational tool for designing personalized cancer vaccines

Alexander Rubinsteyn¹, Isaac Hodes¹, Julia Kodysh¹, Jeffrey Hammerbacher¹,²

¹ Department of Genetics and Genomic Sciences at Icahn School of Medicine at Mount Sinai
² Department of Microbiology and Immunology, Medical University of South Carolina
† Contact: alex@hammerlab.org

Abstract

Therapeutic vaccines targeting mutant tumor antigens (“neoantigens”) are an increasingly popular form of personalized cancer immunotherapy. Vaxrank is a computational tool for selecting neoantigen vaccine peptides from tumor mutations, tumor RNA data, and patient HLA type. Vaxrank is freely available at www.github.com/hammerlab/vaxrank under the Apache 2.0 open source license and can also be installed from the Python Package Index.

1 Introduction

Mutated cancer proteins recognized by T-cells have become known as “neoantigens” and are considered an essential component of a tumor-specific immune response (Finnigan et al., 2015; Gubin et al., 2015; Schumacher and Schreiber, 2015). Therapeutic vaccination against neoantigens is an emerging experimental cancer therapy that attempts to mobilize an antigen-specific immune response against mutated tumor proteins (Türeci et al., 2016; Zhang et al., 2017). Since few tumor mutations are shared between patients, neoantigen vaccines must be personalized therapies. A common approach for achieving personalization is high-throughput sequencing of tumor and normal patient samples followed by in-silico prioritization of mutated peptides that are likely to be presented on the surface of tumor cells by MHC (major histocompatibility complex) molecules.

Vaxrank is a tool for selecting mutated peptides for personalized therapeutic cancer vaccination. Vaxrank determines which peptides should be used in a vaccine from tumor-specific somatic mutations, tumor RNA sequencing data, and a patient’s HLA type. These peptides can then be synthesized and combined with an adjuvant to attempt to elicit an anti-tumor T-cell response in a patient.

The sequence of each mutated protein is determined by assembling variant RNA reads. Mutant protein sequences are ranked using a scoring system which seeks to satisfy two objectives: choosing mutations that are abundant in the tumor and choosing those whose translated amino acid sequences contain likely MHC ligands. Additionally, Vaxrank considers surrounding non-mutated residues in a peptide to prioritize vaccine peptide candidates and to improve the odds of successful synthesis.

Vaxrank was designed for and is currently being used in the Personalized Genomic Vaccine Phase I clinical trial at the Icahn School of Medicine at Mount Sinai (NCT02721043) (Rubinsteyn et al., 2016a).
Figure 1: Users provide tumor mutations, tumor RNA sequence data, and patient HLA type. These are used to determine mutant protein sequences and rank them according to expression and predicted MHC affinity.

## 2 Running Vaxrank

To generate a Vaxrank vaccine report, the user must provide one or more files containing somatic variants (in VCF, MAF, or JSON format), aligned tumor RNA-seq reads (as an indexed BAM), and the HLA alleles to be used for MHC binding prediction:

```
vaxrank
   --vcf somatic-variants.vcf
   --bam tumor-rna.bam
   --mhc-predictor netmhc
   --mhc-alleles H2-Kb,H2-Db
   --mhc-peptide-lengths 8-10
   --vaccine-peptide-length 21
   --min-alt-rna-reads 3
   --output-pdf-report vaccine-peptides.pdf
```

The `--mhc-predictor` argument controls which program is used to predict the affinity between a peptide-MHC pair. Vaxrank supports the use of locally installed instances of NetMHC (Andreatta and Nielsen, 2016), NetMHCpan (Nielsen et al., 2007), NetMHCCons (Karosiene et al., 2012), MHCflurry (Rubinsteyn et al., 2016b), or a variety of web-based predictors through IEDB (Vita et al., 2015). The `--min-alt-rna-reads` argument controls the minimum number of RNA reads supporting a variant required to include that variant in the output report. In addition to quantifying tumor expression of a mutations, the RNA reads are used to phase adjacent variants when reconstructing the mutated coding sequence. A more complete list of options for input data, filtering, and output formats can be seen by running `vaxrank --help`. Vaxrank’s output can be formatted as PDF, plain-text, HTML, or an Excel spreadsheet. The output lists variants in ranked order along with vaccine peptide(s) containing that variant, predicted MHC ligands, number of supporting RNA reads, and sequence properties that affect manufacturability.

## 3 Ranking Mutations

A patient’s coding mutations are ranked according to a score that combines each mutation’s degree of expression and aggregate affinity of overlapping mutant peptides for that patient’s MHC alleles.
\[ \text{RankingScore} = \text{ExpressionScore} \cdot \text{TotalBindingScore} \]

\[ \text{ExpressionScore} = \sqrt{\# \text{ RNA reads supporting variant allele}} \]

\[ \text{TotalBindingScore} = \sum_{s \in \text{subsequences}} \sum_{a \in \text{alleles}} \text{BindingScore}(s, a) \]

The \textit{BindingScore} function is, by default, a logistic transformation of the peptide-MHC binding affinity that loosely approximates the probability of T-cell response (Sette \textit{et al.}, 1994). Alternatively, binding predictions can be scored using an affinity threshold (commonly \( \leq 500\text{nM} \)) or a threshold on the percentile rank of the affinity. Only subsequences which overlap mutant residues and do not occur in the reference proteome are considered as part of the \textit{TotalBindingScore}.

4 Manufacturability

Vaxrank was designed under the assumption that its output will be used to make long peptides, due to their favorable immunological properties (Rosalia \textit{et al.}, 2013). Unfortunately, long peptides are also more difficult to synthesize using traditional solid phase chemistry (Bodanszky, 1988). To avoid known difficulties in synthesis, Vaxrank selects a window of amino acids around each mutation that minimizes the following undesirable properties:

1. total number of cysteine residues
2. \( \max(0, \text{mean hydrophobicity of } 7 \text{ residues at C-terminus}) \)
3. \( \max(0, \text{mean hydrophobicity of any } 7 \text{ amino acid window}) \)
4. glutamine, glutamic acid, or cysteine at N-terminus
5. cysteine at C-terminus
6. proline at C-terminus
7. asparagine at N-terminus
8. total number of asparagine-proline bonds

Manufacturability optimization does not affect the ranking of mutations but is only used for selecting which surrounding residues should be included. In cases where a mutation spans a “difficult” sequence (e.g. long hydrophobic stretch), minimizing these criteria may fail to salvage manufacturability.

\textit{Funding}: This work has been supported by the Icahn Institute and the Parker Institute for Cancer Immunotherapy.

References


