Epitope Threader Manual

Author: René Staritzbichler, Date: 19.12.2013

Goal: a fast estimation of binding energies of peptide sequences with protein.

Background

Being presented on the MHC is a key step for a peptide to induce an immune reaction. However, not all peptides that are presented on the MHC are supposed to cause an immune cascade. Not only foreign but also their own content is presented. Thus MHC binding is a necessary but not a sufficient step.

Adaptive immune reactions are induced by TCRs binding to the MHC-peptide. MHC-peptide binding is generally provided by anchor positions and thus is not expected to be as strong as possible, because the key step is the subsequent binding to the TCR.

There are two approaches to predict binding of peptides to MHC or MHC-TCR:

1. Data driven. If many epitopes are known for a given MHC allele one can train learning algorithms on that data. This requires epitopes to be known, which is the case only for a few alleles.
2. Structure based. Given a 3D model of the molecules one can calculate the binding energies of candidate epitopes with MHC or MHC-TCR. There are a number of structures available and these show a high level of conservation (little variation between their structures) and thus homology is expected to result in high quality models for alleles where no structure is available.
3. Combinations. Dynapred performs first structure based energy calculations and then uses that information as input for the training.

EpitopeThreader is a structure based approach for a fast calculation of binding energies of query peptide sequences to a given MHC or MHC-TCR. Threading is a method where a protein sequence is placed onto a 3D structure of another protein. Practically that means to replace the side-chains of the template with the ones of the query. This becomes very efficient on a residue level, where side-chains are neglected.

EpitopeThreader is designed to compare different kinds of energy calculations:

1. Centroid based. Place all side-chain atoms on position of either C-alpha or C-beta atom. The simplest model, but very robust. Based on CHARMM force field.
2. Knowledge based potentials. Probaibility distributions can be transformed into potentials. EpitopeThreader provided functionalities to derive potentials for:
    a. Amino acid pair potential
    b. Amino acid solvation
    c. Backbone geometry
3. Derived from molecular dynamics. The sampling of MD provides the most accurate energies at the price of long calculation times. However, the resulting energies can be fitted by residue level energy functions which in turn are fast in their application.

Input:

1. PDB file containing MHC-peptide or MHC-peptide-TCR structure
2. A list of query sequences
3. Topology and parameter file of CHARMM (part of distribution)

Note:

1. The method is written for binding of candidate epitopes to MHC or MHC-TCR. However, the workflow can be applied to any protein or peptide chains.
2. The input PDB file can be either a x-ray structure or a homology model.

Key features

1. Binding energy calculation for peptide sequences to given protein. The peptides are the candidate epitopes, the protein is either the MHC or the MHC-TCR complex.
2. Deriving knowledge based potentials.

External tools:

1. 'psfgen' from the NAMD package.

Main workflow (each a separate step, described individually below):

1. Extract so called 'default amino acids' from CHARMM force field
2. Sort chains in molecule such that epitope is chain A
3. Clean PDB using psfgen
4. Translate PDB into internal format
5. Create Calpha or Cbeta centroids
6. Build scoring matrix
7. Scan query peptide sequences

Additionally:

8. Derive knowledge based potentials

Masterscript: 'run_epitope_threader.py' encompasses steps 1) - 7)

```
./run_epitope_threader.py   IN_MODELS  IN_EPITOPES  EPITOPE_LENGTH
OUT_PATH  OUT_SCAN_SCORES
```

detailed description is below the next section

Individual steps:

1) extract default amino acids

```
epitope_threader.exe -default_amino_acids PARAMETER  TOPOLOGY  OUTFILE
```

PARAMETER: e.g.: par_all27_prot_lipid.prm
TOPOLOGY: e.g.: top_all27_prot_lipid.rtf

2) Sort and split chains

```
epitope_threader.exe -sort_chains  IN-PDB  EPITOPE-LENGTH  OUTPATH
```

IN-PDB: file with coordinates for MHC-peptide or MHC-peptide-TCR
EPITOPE-LENGTH: number of residues of peptide in IN-PDB

Every chain will be written into its own PDB file, the epitope will be placed into NAME_A.pdb. The individual PDBs for each chain are needed in step 3)

3) Clean pdb, add hydrogens

write the following into a file, e.g.   NAME.psf:

```
PATH/psfgen << ENDMOL
topology  PATH/top_all27_prot_lipid.rtf
pdbalias residue HIS HSE
pdbalias atom ILE CD1 CD
segment A {pdb ./3BO8_A.pdb}
coordpdb ./3BO8_A.pdb A
segment B {pdb ./3BO8_B.pdb}
coordpdb ./3BO8_B.pdb B
segment C {pdb ./3BO8_C.pdb}
coordpdb ./3BO8_C.pdb C
guesscoord
writepdb ./3BO8.cleaned.pdb
ENDMOL
```

Each chain has to be treated individually, here there are three (A,B,C).
Make file executable and … execute it.
Output will be in this case 3BO8.cleaned.pdb
This will contain hydrogen atoms!

4) Translate into internal format

```
epitope_threader.exe -pdb2mol  PARAMETER  TOPOLOGY IN-PDB  OUTFILE
```

PARAMETER: CHARMM parameters, most likely: par_all27_prot_lipid.prm
TOPOLOGY: CHARMM topology file, most likely: top_all27_prot_lipid.rtf
IN-PDB: output of step 3)

5) Fuse into centroids

```
epitope_threader.exe -ca_centroid   INPUT   OUTPUT
```

INPUT: output from step 4)
OUTPUT: file in internal format

6) Calculate binding energy matrix

```
epitope_threader.exe -score  INPUT  CHAIN  DEFAULT_AA  OUTPUT
```

INPUT: output of step 5)
CHAIN: should be 'A' because of step 2)
DEFAULT_AA: output of step 1)

7) Scan peptide sequences

```
epitope_threader.exe  -scan  INPUT-MATRIX  INPUT-EPITOPES  OUTPUT
```

INPUT-MATRIX: output of step 6)
INPUT-EPITOPES: file containing sequences to be scanned

Masterscript:

This script is designed for larger runs over multiple models and epitope sets.

USAGE:

```
PATH/run_epitope_threader.py  IN_MODELS  IN_EPITOPES EPITOPE_LENGTH
OUT_PATH  OUT_SCAN_SCORES
```

brief description of arguments:

- IN_MODELS: file containing list of models without .pdb ending, one per line, (PATH only if necessary) e.g.:

  ```
  PATH/1afo
  PATH/1zif
  ```

- IN_EPITOPES: file containing list of epitopes to be scanned, one epitope per line, file needs to start with number of epitopes contained, e.g.:

  ```
  3
  ABCDEFGH
  IJKLMNOP
  QRSTUVXY
  ```

- EPITOPE_LENGTH: number of amino acids of the peptide in models, has to match length of epitope sequences in list (8 in previous example)

- OUT_PATH: path for output files

- OUT_SCAN_SCORES: file for output of scores, the actual result of algorithm

Knowledge based potentials

The second topic that is covered by the epitope threader is to derive statistical or knowledge based potential.

One can derive three different types of potentials:

1. Backbone dihedrals
2. Amino acid pair potential
3. Amino acid solvation

The usage is rather simple:

```
epitope_threader.exe -kb:pot:dihedral     PDB-LIST
epitope_threader.exe -kb:pot:aapair       PDB-LIST
epitope_threader.exe -kb:pot:solvation    PDB-LIST
```

PDB-LIST: file containing all pdbs to used. The choice of pdbs will obviously impact quality and/or field of application and is therefore CRUCIAL.

Generally one tries to be complete but avoid redundancies in pdb data set as these introduce artificial bias.
See PISCES server at Dunbrack Lab. One can build lists defined by own criteria, or download pre-compiled lists of pdbs.

http://dunbrack.fccc.edu/PISCES.php

There are two main types of strategies

1. Use entire PDB and select by quality related criteria
2. Pass pre-filtered list of PDB

Latter opens the door to specialized potentials, eg.

- Membrane vs soluble proteins
- MHC-peptide complexes

However, one has make sure that the list has a statistically sufficient size, which limits the ability to derive specialized potentials. The required size depends crucially on the kind of potential. The amino acid pair potential has more degrees of freedom than the amino acid solvation. It is doubtable that one could derive a specialized potential based solely on MHC-peptide structures.

Trouble shooting

Message for ‘-sort’:
“WARNING: more than one chain has expected size, check output, last matching chain found will be considered epitope and assigned chain A!!!”

Explanation / Solution:
PDB structures of the MHC-peptide complex should contain 3 chains, PDBs of the MHC-peptide-TCR complex 5 chains. There is a chance that PDBs contains multiple copies of a complex (crystallographic data). Check with VMD, PYMOL or other molecular viewers. If they are highly similar (superimpose and check RMSD, should be small) simply remove unwanted chains. If they differ significantly split them and use individually.

Installation

On Linux/Mac:

```
‘mkdir  NAME’
‘mv  epitope_threaderVERSION.tgz NAME/.’
‘cd  NAME’
‘mkdir  exe’
```

Unpack tar ball:

```
‘tar  xvzf   epitope_threaderVERSION.tgz’
```

Then built by simply calling (using gcc):
‘make’

The executable program can be found in subdirectory ‘exe/’.

In subdirectory ‘toppar/’ there are the CHARMM topology and parameter files, needed for calculating the binding energy matrices.

The masterscript ‘run_epitope_threader.py’ can be found in main directory.
The only thing one needs to adjust is the path variable ‘exec_path’, if you do not call it from the main installation directory, which is likely.