Version: November 1, 2020

Instruction for using VESPER

VESPER is a computational tool using local vector-based algorithm that can accurately identify the global and local alignment of EM maps.

This package includes:
Program files:
VESPER (VESPER executable program compiled and tested on Linux Ubuntu 18.04)
cluster_score.py (calculate Z-score for each model in VESPER output)
transform_em_map.py (generate transformed EM map)
VESPER_code/ (folder for VESPER source codes)

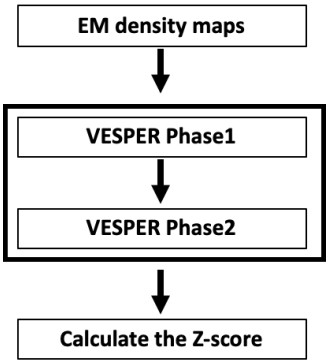
Input files:
example_data/emd_8724.map (EMD-8724, complete V-ATPase)
example_data/emd_8409.map (EMD-8409, V_o region of V-ATPase)

Output files:
example_data/8724_8409_s7a30.pdb (VESPER output)
example_data/8724_8409_s7a30.pdb.normzscore (Z-score for top 10 models)
example_data/target_transform_model_1.mrc, target_transform_model_2.mrc, …
(transformed EM map for top 10 models)

The source code of VESPER is provided at our GitHub (https://github.com/kiharalab/VESPER).
A webserver is also available at our lab website (http://kiharalab.org/em-surfer/vesper.php).

1 Protocol

The architecture of VESPER is summarized in the flowchart below.



Flowchart of VESPER

This document provides a detailed explanation of each step of the VESPER architecture and programs needed to run those steps. It concludes by giving a step-by-step application walk-through on a pair of experimental EM maps.

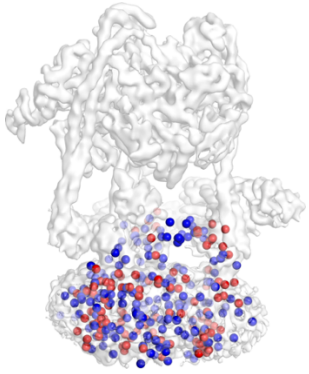
1.1 VESPER Phase1 - Conversion to unit vectors

To identify the optimal superimposition of two EM maps, each map is represented by a set of unit vectors computed with the mean shift algorithm. Each density point is represented as a unit vector that shows the direction of the movement toward a local representative point calculated by the mean shift algorithm. This vector representation provides information of underlying local molecular structures around each voxel.

Vector representation in VESPER

1.2 VESPER Phase 2 - Find the best superimposition using FFT

Given a pair of EM maps, the goal of VESPER is to find the pose transformation that maximizes the agreement of the local density landscape of the two maps. The best superimposition of two maps is searched using FFT. For each rotation of a map using an interval of 30° as default (users can change this interval), a translation scan is performed using FFTs to optimize the sum of dot products of matched vectors (the DOT score). Then, for each of the 10 top scoring models from the FFT search, VESPER performs a finer rotational angle search with a 5° interval around each axis (if the angle interval used for the initial coarse-grained search is larger than 5°). Finally, the top 10 (default) or a user-specified number of scoring superimpositions will be output. The figure below shows the top scoring model from VESPER.



Best superimposition of two maps

1.3 Calculation of Z-score

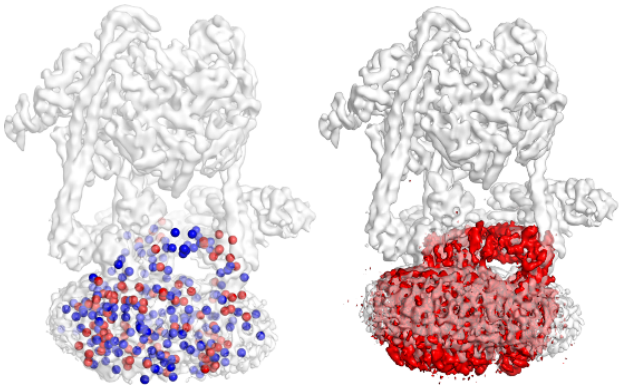
The procedures we use to calculate the Z-score is as follows. In VESPER search process, the target map is placed in a rotational pose with the angle interval (specified in Angle Spacing). For each rotational pose, the target map is translated by the translational interval (specified in Voxel Spacing) to optimize the summation of dot products of matched vectors (the DOT score). The largest DOT score among all the translations for a rotational pose is stored. We then perform single-linkage clustering of all the DOT scores with a cutoff of 20% of the difference between the maximum and minimum DOT scores of the entire distribution, and calculate the mean and standard_deviation of the largest cluster. The Z-score of the each of the top 10 models is defined as (DOT_Score - mean)/standard_deviation.

```
Normalized z-score for top 10 models:
#0      10.477225138782385
#1      10.052819302572953
#2      9.24642431705002
#3      9.24045013579707
#4      9.12777707736647
#5      8.912109134135044
#6      8.607067439359513
#7      8.594402175103264
#8      8.53609416607449
#9      8.474321131919003
```

Example of Z-score result

1.4 Visualize the superimposition

In VESPER output, each vector is represented by two spheres: one for start position and one for end position. To show the goodness of fit between two maps, the vectors can be colored by the dot product values of matched vectors. The dot product of a pair of matched vectors ranges from -1 to 1 with 1 for the perfect match, 0 for two perpendicular vectors, and -1 for two vectors pointing the opposite directions. It also generates the EM map for each of top 10 models at transformed position.



Visualization of vectors (left) and transformed map (right)

2 Usage Guide

(1) Identify the best superimposition between two EM maps by VESPER. (Phase1 and Phase2)

```
Usage: VESPER -a [MAP1.mrc (large)] -b [MAP2.mrc (small)] [(option)]

---Options---
-t [float] : Threshold of density map1 def=0.000
-T [float] : Threshold of density map2 def=0.000
-g [float] : Bandwidth of the gaussian filter
             def=16.0, sigma = 0.5*[float]
-s [float] : Sampling grid space def=7.0
-A [float] : Sampling Angle interval def=30.0
-c [int  ] : Number of cores for threads def=2
-N [int  ] : Refine Top [int] models def=10
-S         : Show topN models in PDB format def=false
-V         : Vector Products Mode def=true
-L         : Overlap Mode def=false
-C         : Cross Correlation Coefficient Mode def=false
             Using normalized density value by Gaussian Filter
-P         : Pearson Correlation Coefficient Mode def=false
             Using normalized density value by Gaussian Filter and average density
-F         : Laplacian Filtering Mode def=false
-E         : Evaluation mode of the current position def=false
```

(2) Calculate the Z-score for each of top 10 models in VESPER output.

```
python cluster_score.py:
usage: cluster_score.py [-h] -i INPUT_FILE [-c CUTOFF] [-o OUT_NAME]

Calculate the normalized z-score for top 10 models from VESPER. Normalized
z-scores for top 10 models are written into the output file.

optional arguments:
  -h, --help            show this help message and exit
  -i INPUT_FILE, --input INPUT_FILE
                        Required. Name of input file.
  -c CUTOFF             Optional. Clustering cutoff ranging from 0 to 1.
                        Default = 0.2.
  -o OUT_NAME, --output OUT_NAME
                        Optional. Name of output file. If not specified, the
                        output file would be named as input filename followed
                        by .normzscore.
```

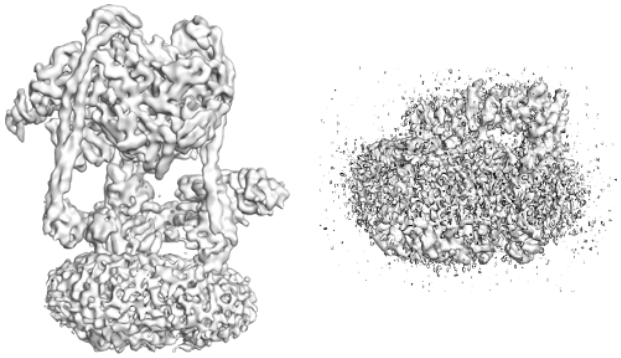
(3) Generate transformed EM map.

```
usage: transform_em_map.py [-h] -i1 REF_MAP -i2 TARGET_MAP -t VESPER_RESULT
                           [-odir OUT_DIR]

Transform the target map given rotation and translation information in the
VESPER output file.

optional arguments:
```

```
  -h, --help            show this help message and exit
  -i1 REF_MAP, --input1 REF_MAP
                        Required. Name of the reference map file.
  -i2 TARGET_MAP, --input2 TARGET_MAP
                        Required. Name of the target map file.
  -t VESPER_RESULT      Required. Name of the result file from VESPER.
  -odir OUT_DIR         Optional. Directory for the transformed target map
                        files. If not specified, the transformed target map
                        files would be written to the current director
```

3 Example

Here we take the matching between complete V-ATPase (EMD-8724) and V_o region (EMD-8409) as an example. The map files for these two structures are provided in example_data/ folder.



Input maps of V-ATPase (left) and V_o region (right)

Find the best superimposition with VESPER

User can run the following command to find the top 10 scoring superimpositions between EMD-8724 and EMD-8409. Here the first map or reference map, EMD-8724, is fixed. The target map, EMD-8409, is superimposed onto the reference map.

```
./VESPER -a ./example_data/emd_8724.map -b ./example_data/emd_8409.map -t 0.04 -T
0.048 -s 7 -A 30 -c 5 -S > ./example_data/8724_8409_s7a30.pdb
```

Computational time: 6.2 seconds on Linux machine with Intel Xeon CPU E5-2680. By default, VESPER writes the vector information for each of top 10 models after local refinement into VESPER output. In a VESPER output, vector information for the first model starts with two lines like the ones shown below.

```
#0 R={0.0 0.0 5.0} MTX={0.996194700 -0.087155725 0.000000000 0.087155725 0.996194700
0.000000000 -0.000000000 0.000000000 1.000000000} T={37.335 24.247 93.200} sco=
144.563 zsco= 10.477246
Overlap= 0.0871 249/2859 CC= 0.206020 PCC= 0.068408 Score= 144.6
```

In the first line, 0 is the index of the first model. Here the model index starts from 0. MTX shows the rotation matrix of MAP2 relative to MAP1. T shows the translation vector of MAP2 relative to MAP1. sco shows the DOT score, which is the summation of dot products of matched vectors between two maps.

After these two lines, the output file shows the vector information for the first model. Each vector is represented by two atoms, one for start position (CA) and one for end position (CB). The number in the last column shows the fitness score, which is dot product between this vector and the matched vector in MAP1. Fitness score ranges from -1 to 1: 1 for a perfect match, 0 if two vectors are perpendicular or there is no matched vector, and -1 if two vectors are in the opposite direction. One example is shown below. In this case, there is no matched vector in MAP1 for this specific vector. Thus, fitness score in the last column is 0.00. Similar information is provided for other 9 models in VESPER output.

```
ATOM      1  CA  ALA     1     101.600 164.600 248.600  1.00  0.00
ATOM      2  CB  ALA     1     106.065 166.626 243.604  1.00  0.00
```

Calculation of Z-score

To calculate the Z-score for each of the top 10 scoring models in VESPER output, user can run the command below. Computational time: 1.5 seconds on Linux machine with Intel Xeon CPU E5-2680.

```
python cluster_score.py -i ./example_data/8724_8409_s7a30.pdb
```

```
Normalized z-score for top 10 models:
#0      10.477225138782385
#1      10.052819302572953
#2      9.24642431705002
#3      9.24045013579707
#4      9.12777707736647
#5      8.912109134135044
#6      8.607067439359513
#7      8.594402175103264
#8      8.53609416607449
#9      8.474321131919003
```

Z-score result

Visualize the superimposition in Pymol

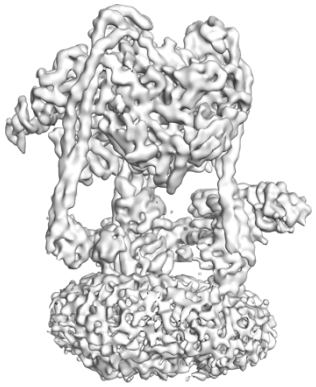
To transform the map for V_o region, user can run the command below. It generates the EM map for each of top 10 models at transformed position. Computational time: 35.6 seconds on Linux machine with Intel Xeon CPU E5-2680.

```
python transform_em_map.py -i1 ./example_data/emd_8724.map
-i2 ./example_data/emd_8409.map -t ./example_data/8724_8409_s7a30.pdb
-odir ./example_data/
```

1) emd_8724.map: This is the map file for the reference map. The reference map is fixed in VESPER search process. The target map is superimposed onto the reference map.

To view this map file in Pymol, open Pymol first, then run the following commands in Pymol:

```
bg_color white
set normalize_ccp4_maps, 0
load emd_8724.map
isosurface 8724_isosurface, emd_8724, 0.04
```

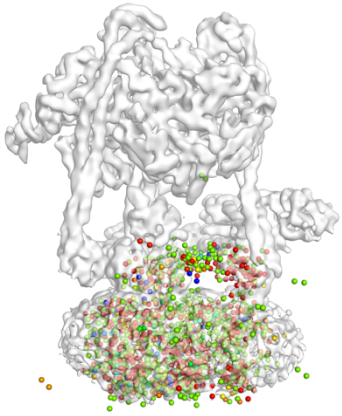


Load the map for V-ATPase

2) 8724_8409_s7a30.pdb: This PDB file shows the vector representation for the target map in each of the top 10 models identified by VESPER.

To view this file together with the reference map, firstly open Pymol and run the commands listed in 1), then run the commands below:

```
load 8724_8409_s7a30.pdb
set transparency, 0.4
hide cartoon, 8724_8409_s7a30
show spheres, 8724_8409_s7a30
spectrum b, rainbow, 8724_8409_s7a30
```

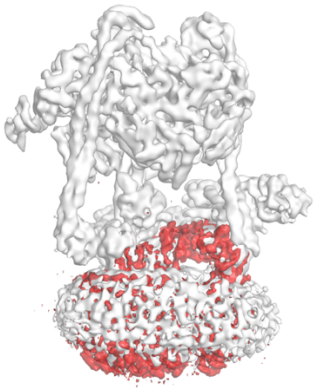


Color vectors

3) target_transform_model_*.mrc: each of those mrc files shows one of the top 10 models identified by VESPER for the target map. “target_transform_model_1.mrc” is the best model identified by VESPER, “target_transform_model_2.mrc” is the second best model, “target_transform_model_2.mrc” is the third best model, …

To view the best model together with the reference map and the vector pdb file, open Pymol and run the commands listed in 1) and 2), and then run the commands below:

```
load target_transform_model_1.mrc
isosurface model1_isosurface, target_transform_model_1, 0.048
color red, model1_isosurface
```



Load transformed map

VESPER alignment with other object functions

User can specify other object functions (Overlap, Cross correlation, Pearson correlation and Laplacian filter score) in VESPER alignment by the following options.

```
-L          : Overlap Mode def=false
-C          : Cross Correlation Coefficient Mode def=false
              Using normalized density value by Gaussian Filter
-P          : Pearson Correlation Coefficient Mode def=false
              Using normalized density value by Gaussian Filter and average density
-F          : Laplacian Filtering Mode def=false
```

The following command performs map alignments between EMD-8724 and EMD-8409 by the Pearson Correlation Coefficients.

```
./VESPER -a ./example_data/emd_8724.map -b ./example_data/emd_8409.map -t 0.04 -T
0.048 -s 7 -A 30 -c 5 -P > ./example_data/8724_8409_s7a30.pdb
```

Computing score for the aligned maps

User can compute the DOT score of the aligned maps with -E option. To use -E option, VESPER requires the same coordinate system and map size for map1 and map2. UCSF Chimera can generate the resampled map by “vop” command. For example, emd_8409.map is resampled based on emd_8724.map by the following command:

```
open ./example_data/emd_8724.map
open ./example_data/emd_8409.map
vop #1 resample onGrid #0
volume #2 save resampled_emd_8409.mrc
```

Then, VESPER can compute DOT score between emd_8724.map and resampled_emd_8409.mrc.

```
./VESPER -a ./example_data/emd_8724.map -b ./resampled_emd_8409.mrc -t 0.04 -T 0.048
-s 7 -A 30 -c 5 -E > score.txt
```

In the VESPER output, vector information is shown in one line.

```
#0 R={0.0 0.0 0.0} MTX={1.000000000 0.000000000 0.000000000 0.000000000 1.000000000 0.000000000 -
0.000000000 0.000000000 1.000000000} T={0.000 0.000 0.000} sco= 12.563 zsco= -4.432988
```

4 Contact Information

If you have any questions about your calculation result or VESPER in general, please contact:
Daisuke Kihara (dkihara@purdue.edu).

VESPER website: http://kiharalab.org/em-surfer/vesper.php
Kihara lab: http://kiharalab.org/