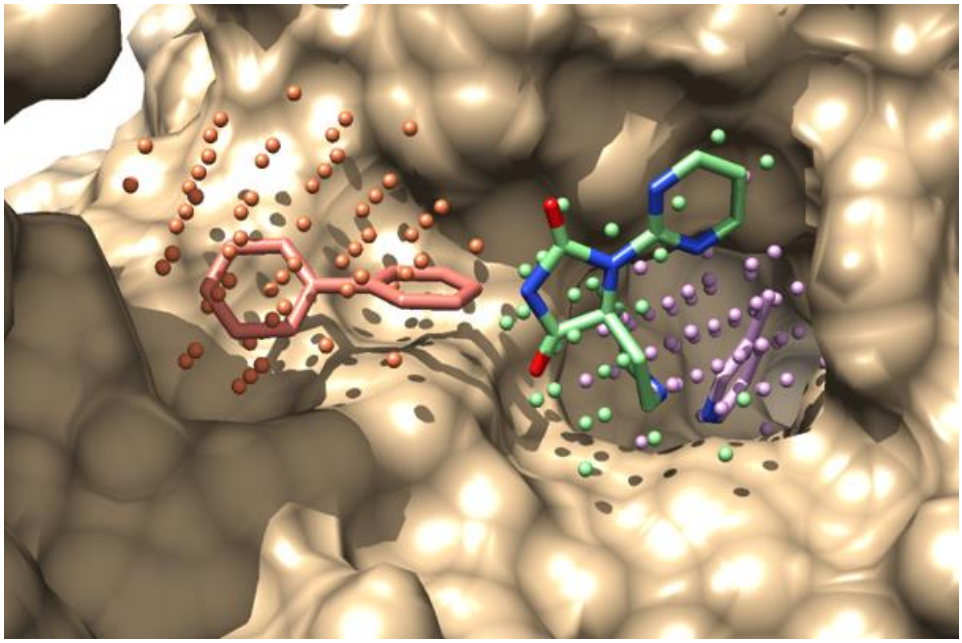
ProCare

& utils

A Point Cloud Registration Approach to Compare and Align Protein Cavities

Version 1.0.0



Content

1. Description and theory

ProCare[1] is a Python package for protein cavities comparison and alignment, based on a point cloud registration algorithm.[2–4] A protein cavity by IChem VolSite module is represented as an ensemble of points that imprint the cavity shape and pharmacophoric properties (**Figure 1**).[5]

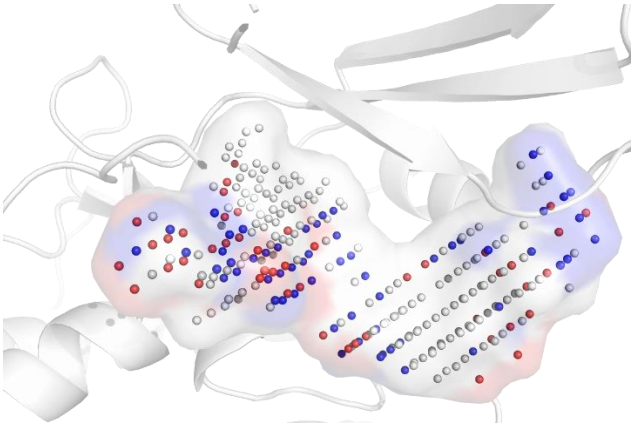


Figure 1. Example of CDK8 (PDB code: 5HBH) ATP cavity representation by IChem VolSite. Each point is associated to one of the eight possible pharmacophoric features according to the nearest protein atom. Blue: h-bond donor, positive ionizable; red: h-bond acceptor and donaceptor, negative ionizable; white: hydrophobic, aromatic, dummy. The transparent molecular surface of the cavity was illustrated par Pymol 2.1 (Schrödinger, New York, USA).

A comparison is carried in 4 steps (**Figure 2**):

1. Computation of the points descriptors. A point descriptor is encoding both shape and position of the eight pharmacologic features (h-bond donor, acceptor and donaceptor, positive ionizable, negative ionizable, hydrophobic, aromatic, dummy).

2. Initial RANSAC search by sampling randomly a few points at a time in the mobile cavity to match the most similar points in the reference cavity. Point similarity is estimated by the Euclidian distance of their respective descriptors. Given that the most similar points are searched, a point is always associated to a presumed equivalent in the target cavity, regardless of the distance. Then, false positive matches will be filtered out by checking the sample topology (pairwise distances should match withing a certain distance tolerance). Then an initial alignment is estimated and scored by a property-agnostic scoring function (fitness = proportion of aligned points in source cavity). This search is repeated until convergence.

3. Refinement of the initial alignment by the iterative closest point algorithm. A point is associated to its nearest neighbor in the Euclidian space.

4. Proposed alignment is scored by different schemes and metrics, accounting for the number of associated source/target pairs of points of the same or compatible pharmacophoric feature.

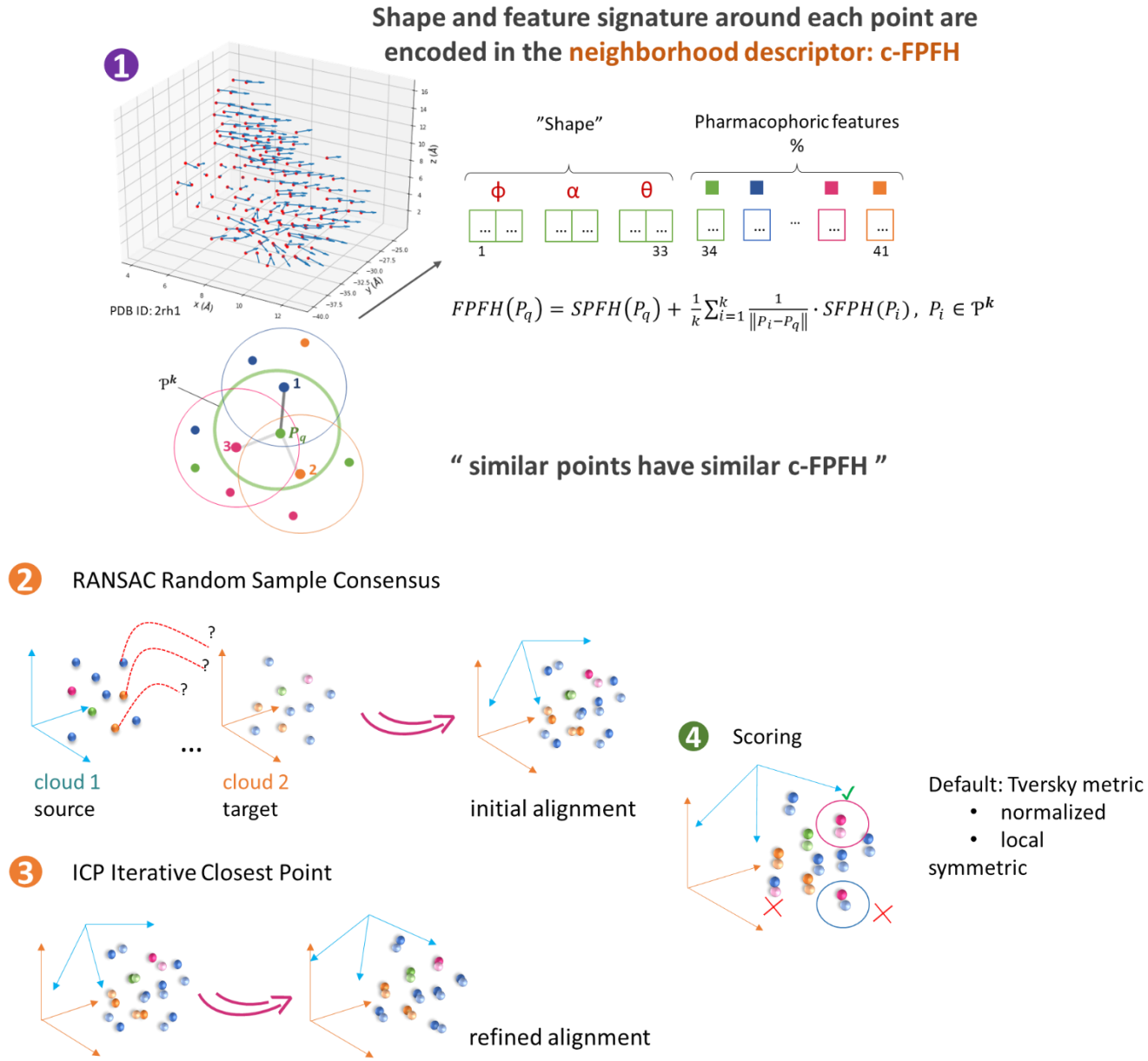


Figure 2. ProCare steps for comparing protein cavities.

2. Install

The installation procedure is summarized in:
#easy install
https://github.com/kimeguida/ProCare/blob/master/install.sh

- Install conda if not already there

See: https://docs.conda.io/en/latest/miniconda.html

- Create a conda environment

```
$ conda env create -n procare -f procare_environment.yml
```

- Activate procare conda environment

```
$ conda activate procare
```

If successful, your prompt will turn into: `(procare)$`

- Download procare

```
$ git clone https://github.com/kimeguida/ProCare.git
$ cd ProCare/
```

- Install procare in the conda environment

```
(procare)$ pip install procare_python_package/
```

3. Help and issues

Signal issues at https://github.com/kimeguida/ProCare/issues

4. Usage

4.1. Compare cavities with procare_launcher.py

```
(procare)$ python procare_launcher.py -s 2rh1_cavity.mol2 -t
5d6l_cavity.mol2 --transform --ligandtransform 2rh1_ligand.mol2
```

Outputs:

- procare_scores.tsv (tab-separated): simplified scores output
- procare.tsv: complete output containing transformation matrices elements, pharmacophore contributions, scores
- using the `--transform` option will output rotated cavity mol2 (cfpfh_2rh1_cavity.mol2)
- using the `--ligandtransform` option with a ligand/protein/anything mol2 file as argument will output aligned ligand/protein/anything in the rotated cavity reference frame in mol2 format (cfpfh_2rh1_ligand.mol2). Can handle multiple files:

  ```
  […] --ligandtransform 2rh1_ligand.mol2 <anything>.mol2
  ```

Help and options:

```
(procare)$ python procare_launcher.py –help

(procare)$ python procare_launcher.py [options] <values>
```

Input options

```
-s, --source <cavity>          source cavity (mobile) mol2

-t, --target <cavity>          target cavity (stationary)
                               mol2
```

Alignment options

```
-rv, --ransacvalid        maximum RANSAC validation
      default : 500

-ri, --ransaciter         maximum RANSAC iteration
      default: 4000000

-rn, --ransacn            RANSAC clique size
      default: 4
```

```
-gt, --globaltranstype    transformation estimation method
                          for global registration
      default: TransformationEstimationPointToPoint
-gd, --globaldist         distance (Å) threshold to define
                          aligned points in global
                          registration
      default: 1.5

-cs, --checkersim         relative distance tolerance of
                          RANSAC clique: between 0 and 1
      default: 0.9

-it, --icptranstype       transformation estimation method
                          for ICP registration
      default: TransformationEstimationPointToPoint
-id', '--icpdist'         distance (Å) threshold to define
                          aligned points in ICP
                          registration
      default: 3

-ir, --icprmse            relative RMSE threshold for ICP
                          terminaison
      default: 10e-6

-if, --icpfitness         relative fitness score threshold
                          for ICP terminaison
      default: 10e-6

-ii, --icpiter            maximum ICP iteration
      default: 100


-nr, --normalrad          radius (Å) for local surface
                          normal estimation
      default: 3.1

-nm, --normalmaxn         maximum number of neighbors to
                          consider for local surface normal
                          estimation
      default: 471 (based on 4.5 Å radius)


-fr, --featurerad         radius (Å) for local surface
                          feature calculation
      default: 3.1
```

```
-fm, --featuremaxn        maximum number of neighbors to
                          consider for local surface
                          feature calculation
      default: 135 (based on 3 Å radius)
```

Output options

```
-o, --output              complete output containing
                          transformation matrices elements,
                          pharmacophore contributions,
                          scores
      default: procare.tsv

-so, --scoreoutput        simplifiled scores output
      default: procare_scores.tsv

-p, --paramid             user-defined ID for parameters
      default: default

-c, --classification      class for retrospective
                          screening: 0 or 1
      default: NAN

--transform               if set, output roto/translated
                          cavity. Does not require <value>


--ligandtransform <mol2>  apply transformation to other
                          object(s) mol2
```

Utils

4.2. Inspect superposed points with: procare_aligned_points.py

Superposed points are source/target points of the same pharmacophoric feature and within 1.5 Å distance of each other.

```
(procare)$ python procare_aligned_points.py
-c1 cfpfh_2rh1_cavity.mol2 -c2 5d6l_cavity.mol2
-o1 aligned_2rh1_cavity.mol2 -o2 aligned_5d6l_cavity.mol2
```

Outputs:

- superposed points in cavities (aligned_2rh1_cavity.mol2 and aligned_5d6l_cavity.mol2)

Help and options:

```
(procare)$ python procare_aligned_points.py –help

(procare)$ python procare_aligned_points.py  [options] <values>
```

Input options

```
      -c1, --cav1 <cavity>            source cavity mol2

      -c2, --cav2 <cavity>            target cavity mol2
```

Output options

```
      -o1, --ocav1 <cavity>           output mol2 of aligned
                                      points in source

      -o2, --ocav2 <cavity>           output mol2 of aligned
                                      points in target
```

4.3. Reapply stored transformation matrices to objects with procare_apply_transformation.py

```
(procare)$ python utils/procare_apply_transformation.py -f
procare.tsv -a 2rh1_ligand.mol2 2rh1_cavity.mol2 -l 1
```

Outputs:

- rot_2rh1_ligand.mol2 and rot_2rh1_cavity.mol2 as rotated mol2 objects 2rh1_ligand.mol2 and 2rh1_cavity.mol2

Help and options:

```
(procare)$ python procare_apply_transformation.py –help

(procare)$ python procare_apply_transformation.py [options]
          <values>
```

Input options

```
      -f, --fileprocare          ProCare output file,
                                 e.g. procare.tsv

      -l, --line                 line index where results are
                                 stored in file, e.g. 2. File
                                 indexes start with 0 (header)

      -a, --appliedtomol2        mol2 file(s) to apply
                                 transformation
```

Output options

```
      --prefix                   prefix to roto/translated
                                 output(s)
      default: rot (rot_<name>)
```

4.4. Rescoring of previously superposed cavities using other scoring schemes with procare_rescoring.py

```
(procare)$ python utils/procare_rescoring.py
-s cfpfh_2rh1_cavity.mol2 -t 5d6l_cavity.mol2 -d 2
```

Outputs:

- procare_rescoring.tsv: score file

Help and options:

```
(procare)$ python procare_ rescoring.py –help

(procare)$ python procare_ rescoring.py [options] <values>
```

Input options

```
-s, --source <cavity>          source cavity (mobile) mol2

-t, --target <cavity>          target cavity (stationary)
                               mol2

-d, -- distance                distance (Å) threshold for
                               scoring
      default: 1.5
```

Output options

```
-o, -- ofile                   output rescoring file
      default: procare_rescoring.tsv
```

5. References

1. Eguida, M.; Rognan, D. A Computer Vision Approach to Align and Compare Protein Cavities: Application to Fragment-Based Drug Design. *J. Med. Chem.* **2020**, *63*, 7127–7142.
2. Rusu, R. B.; Cousins, S. 3D Is Here: Point Cloud Library (PCL). In *2011 IEEE International Conference on Robotics and Automation*; IEEE, 2011; Vol. I, pp 1–4.
3. Rusu, R. B.; Blodow, N.; Beetz, M. Fast Point Feature Histograms (FPFH) for 3D Registration. *2009 IEEE Int. Conf. Robot. Autom.* **2009**, 3212–3217.
4. Zhou, Q.-Y.; Park, J.; Koltun, V. Open3D: A Modern Library for 3D Data Processing. *arXiv:1801.09847* **2018**.
5. Desaphy, J.; Azdimousa, K.; Kellenberger, E.; Rognan, D. Comparison and Druggability Prediction of Protein–Ligand Binding Sites from Pharmacophore-Annotated Cavity Shapes. *J. Chem. Inf. Model.* **2012**, *52*, 2287–2299.

6. Case studies

1. Eguida, M.; Rognan, D. Unexpected Similarity between HIV-1 Reverse Transcriptase and Tumor Necrosis Factor Binding Sites Revealed by Computer Vision. *J. Cheminform.* **2021**, *13*, 1–13
2. Eguida, M.; Schmitt-Valencia, C.; Hibert, M.; Villa, P.; Rognan, D. Target-Focused Library Design by Pocket-Applied Computer Vision and Fragment Deep Generative Linking. *J. Med. Chem.* **2022**, *65*, 13771–13783.