# Chapter 6

# Contact Area-Based Structural Analysis of Proteins and Their Complexes Using CAD-Score

Kliment Olechnovič and Česlovas Venclovas

## Abstract

Quantifying discrepancies between computationally derived and native (reference) structure is an essential step in the development and comparison of protein modeling and protein-protein docking methods. Measuring conformational differences of proteins or protein complexes is also important in other areas of structural biology such as molecular dynamics and crystallography. There are multiple scores to do that. However, nearly all of them, whether superposition-based (e.g., RMSD) or superposition-free, use distances to measure similarity. CAD-score is conceptually different as it uses physical contacts represented as contact areas. Such representation makes it possible to quantify differences of both structures and surfaces (e.g., protein-protein interfaces and binding sites) using the same framework. A number of studies have found CAD-score to be among the most robust scores. The method is implemented both as a web server and as standalone software available at http://bioinformatics.lt/software/cad-score. Here, we describe how to use the standalone CAD-score software for comparison and analysis of protein structures, interfaces, and binding sites.

**Key words** Protein structure, Protein-protein interactions, Voronoi tessellation, Interatomic contacts, Contact area, Global similarity score, Local similarity score

## 1 Introduction

Comparison of different structures (conformations) for the same protein or protein complex is a common task in both computational and experimental structural biology. For example, measuring discrepancies between computational models and corresponding native (reference) structures is at the heart of development and comparison of protein structure prediction and/or refinement methods. Other common uses include comparison of experimental structures solved in different crystal forms, at different temperature or pH, with and without bound ligand, etc. Analysis of a molecular dynamics simulation also involves comparison of structures obtained along the simulation course.

Zoltán Gáspári (ed.), *Structural Bioinformatics: Methods and Protocols*, Methods in Molecular Biology, vol. 2112,
https://doi.org/10.1007/978-1-0716-0270-6_6, 

Over the years, multiple scores have been developed for performing such comparisons. Some of the scores, such as RMSD [1], GDT-TS [2, 3] or TM-score [4], are based on global structure superposition. Others, like Local Distance Difference Test (LDDT) [5], are superposition-free and focus on local deviation. Despite some differences, the majority of such methods use distances to derive a similarity score. Contact Area Difference (CAD) score [6] is conceptually different as it uses areas of physical contacts to quantify differences between the reference structure (target) and the one being evaluated (model). CAD-score is superposition-free measure and can be used for the evaluation of both local and global structural similarity. Moreover, since CAD-score is based on contact areas, it can be directly applied not only for structures but also for surfaces such as protein-protein interfaces or protein binding sites. A recent comprehensive analysis revealed a number of advantages of CAD-score over various other scores [7]. For example, CAD-score shows robust performance on structures displaying large local deviations and multidomain proteins with flexible linkers, the cases presenting a serious problem for superposition-based global scores. Another important advantage of CAD-score is that it strongly favors models with realistic stereo-chemical features, the property that might be particularly important for the analysis of homology modeling and refinement results.

## 2 CAD-Score Definition

### 2.1 Contacts

Contacts in CAD-score are derived from protein structure represented as a set of atomic balls, each ball having a van der Waals radius depending on the atom type. A ball can be assigned a region of space that contains all the points that are closer (or equally close) to that ball than to any other. Such a region is called a Voronoi cell, and the partitioning of space into Voronoi cells is called Voronoi tessellation [8]. Two adjacent Voronoi cells share a set of points that form a surface called a Voronoi face (Fig. 1a, b). A Voronoi face can be viewed as a geometric representation of a contact between two atoms; the area of the Voronoi face corresponds to the contact area. Voronoi cells of atomic balls are constrained inside the boundaries defined by the solvent-accessible surface as described in the recent paper [9].

The resulting constrained Voronoi faces can be combined into residue-residue contacts involving either all atoms (Fig. 1c) or only a subset, for example, side chain atoms (Fig. 1d). For practical purposes, three standard subsets of residue atoms are defined (A, "all atoms"; S, "side chain atoms"; and M, "main chain atoms") resulting in six nonredundant categories of residue-residue contacts (A-A, A-S, S-S, A-M, M-M, M-S). The most useful categories are those that include side chain-side chain interactions, namely, A-A, A-S, and S-S.

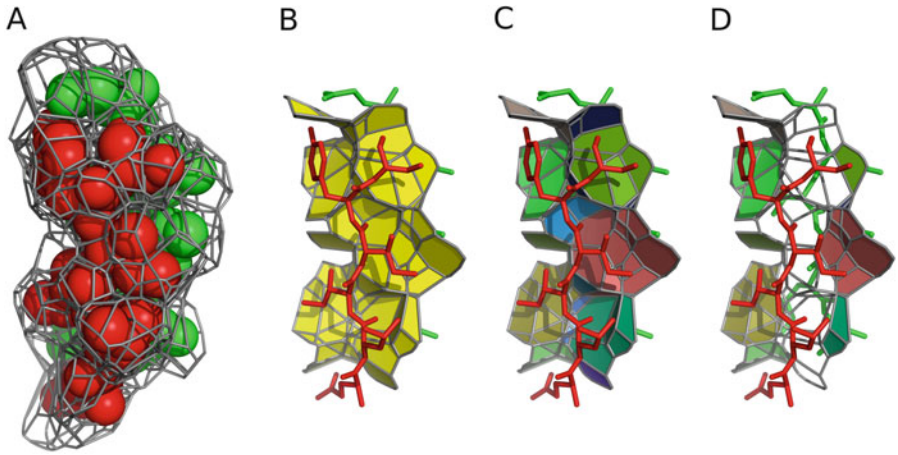


**Fig. 1** Example of interatomic and inter-residue contacts, with two groups of atoms distinguished by red and green coloring. (**a**) Voronoi cells of atomic balls. (**b**) Interatomic contacts between two groups of atoms. (**c**) Grouping of interatomic contacts into inter-residue contacts. (**d**) Contacts between residue side chains

### 2.2 Structure Scores

Given reference structure $T$ (target) and structure to be compared, $M$ (model), let $G$ denote the set of all the pairs of residues $(i,j)$ that have a nonzero contact area $T_{(i,j)}$ in the target structure. Then for every residue pair $(i,j) \in G$, the corresponding contact area $M_{(i,j)}$ in the model is calculated. $M_{(i,j)}$ is assigned zero if there is no contact between residues $i$ and $j$ in the model or if either residue ($i$ or $j$) is missing from the model. The CAD-score for the model structure is then defined as:

$$\text{CAD-score}(G) = 1 - \frac{\sum \min\left(|T_{(i,j)} - M_{(i,j)}|, T_{(i,j)}\right)}{\sum T_{(i,j)}} \qquad (1)$$

Values of Eq. 1 are always within the [0,1] range. If model and target structures are identical, CAD-score$(G) = 1$. At the other extreme, if not a single contact is reproduced with sufficient accuracy, CAD-score$(G) = 0$.

Scores for individual residues are calculated by applying Eq. 1 to a residue-specific subset of $G$. Thus, the score for residue $i$ equals CAD-score$(G_i)$, where $G_i$ is a set of all pairs $(i,j) \in G$. Per-residue scores can be smoothed along the sequence using a sliding window technique.

### 2.3 Scores for Interfaces

A straightforward way to compare the inter-chain interfaces of two protein complexes is to apply Eq. 1 to a set of inter-chain contacts. Let $I$ and $J$ denote the sets of interface residues of the first and the second subunits (chains), respectively, in the target protein complex. Then the set of target interface contacts $G_{I,J}^{\text{iface}}$ and the interface similarity score CAD-score$^{\text{iface}}$ are defined as:

$$G_{I,J}^{\text{iface}} = G \cap (I \times J) \qquad (2)$$

$$\text{CAD-score}^{\text{iface}}(I,J) = \text{CAD-score}(G^{\text{iface}}_{I,J}) \tag{3}$$

It is also possible to quantify how each interface residue is exposed to the other chain by summing the corresponding contact areas. For a specific residue $i \in I$, the exposure value in the target structure is $T_i = \sum_{(i,j)\in G^{\text{iface}}_{I,J}} T_{(i,j)}$. The set of $T_i$ values for all $i \in I$ describes the binding site of the first chain in the target structure. The corresponding binding site in the model structure is defined in the same way, but using the model interface contacts. Then the similarity score of the target and the model binding sites is computed:

$$\text{CAD-score}^{\text{site}}(I) = 1 - \frac{\sum \min\left(|T_i - M_i|, T_i\right)}{\sum T_i} \tag{4}$$

Values of Eq. 4 can range from 0 (completely different binding site) to 1 (binding site with the same exposure of residues, but not necessarily the exact same inter-chain contacts).

Less detailed and, therefore, less stringent similarity measures can be defined using total interface contact areas (Eq. 5) and total binding site areas (Eq. 6):

$$\text{CAD-score}^{\text{iface-area}}(I,J) = \min\left(1, \frac{\sum M_{(i,j)}}{\sum T_{(i,j)}}\right) \tag{5}$$

$$\text{CAD-score}^{\text{site-area}}(I) = \min\left(1, \frac{\sum M_i}{\sum T_i}\right) \tag{6}$$

These latter two similarity measures essentially look whether the interface (binding site) corresponds to the same surface patch without paying attention to the exact contribution by individual residues.

### 2.4 CAD-Score Web Server

The CAD-score web server is accessible without any restrictions at the following URL: http://bioinformatics.ibt.lt/cad-score. It provides a simple and intuitive graphical user interface for running the original ("classic") implementation of CAD-score [6]. The server outputs tables of scores and provides interactive plots for exploring local contact differences. The CAD-score web server has an online tutorial, and in addition there is a separate paper devoted entirely to the description of the server [10]. Therefore, the focus of this chapter is solely on the standalone CAD-score software, which offers maximal flexibility in structural analyses.

### 2.5 Standalone CAD-Score Software

At present, there are two distinct software implementations of the CAD-score method. In the first, "classic" implementation (https://bitbucket.org/kliment/cadscore), contacts are constructed for every atom by subdividing the expanded atom sphere according to the Voronoi neighbors. In the more recent implementation, which is a part of a larger package called Voronota (https://bitbucket.org/kliment/voronota), contacts are derived directly

from the Voronoi faces, as described in Subheading 2.1. Although the actual values for contact areas and similarity scores in these two implementations differ, the two versions correlate very strongly.

The "classic" implementation has been tested in CASP [11] and CAMEO [12] projects. Recently, it has also been extensively compared to other reference-based similarity measures [7]. On the other hand, the new implementation uses more intuitive and symmetric definition of contacts, making it especially suitable for analysis and comparison of interfaces. The new implementation has been extensively tested and employed in the comparison and clustering of protein-protein interfaces [13]. The scores defined by Eqs. 4–6 are attainable only through the new implementation.

This chapter describes the use of new CAD-score implementation. However, the software can always be run in the "classic" mode, which is enabled by simply using the **`--old-regime`** flag in the command line.

## 3 CAD-Score Usage

### 3.1 Installation

The latest version of CAD-score is implemented as the **`voronota-cadscore`** script, which is a part of the Voronota package. The package can be downloaded from https://bitbucket.org/kliment/voronota/downloads and installed (or run without installing) on any modern Linux or macOS system (also, *see* **Notes 1** and **2**). Ubuntu 18.04 and newer Voronota can be downloaded and installed with a single command: **`sudo apt install voronota`**.

### 3.2 Global Scoring of 3D Structures

For a basic yet realistic example, let us use a dataset from the CASP12 experiment. CASP12 target and model structures are available correspondingly from "targets" and "predictions" folders at http://predictioncenter.org/download_area/CASP12/. Let us consider the heterodimeric target structure "T0921-T0922.pdb" and its models. For clarity, let us rename "T0921-T0922.pdb" to "target.pdb" and rename the model files "TS188_1" and "TS208_1" to "model1.pdb" and "model2.pdb," respectively. These target and model structures already have the same residue numbering and the same chain naming, key requirements for proper use of CAD-score (*see* **Notes 3** and **4** for more details on how the **`voronota-cadscore`** script reads and interprets input PDB files). Below is an example of the global CAD-score calculation for "model1.pdb":

```
voronota-cadscore -t "target.pdb" -m "model1.pdb"

target.pdb model1.pdb AA 212 0.358224 13334.2 8287.38
```

The same result can be presented as a table with a header that explains the values; output can be aligned by passing it to the standard **column** command (also, *see* **Note 5**):

```
voronota-cadscore -t "target.pdb" -m "model1.pdb" --output-header | column -t

target_file model_file query_code residues score    target_area model_area
target.pdb  model1.pdb AA         212      0.358224 13334.2     8287.38
```

"query_code" indicates the category of residue-residue contacts as described in Subheading 2.1. "residues" is the number of target residues that were included in the evaluation. "score" is the global CAD-score value calculated by Eq. 1. "target_area" and "model_area" are total sums of considered contact areas for the target and the model.

### 3.3 Using Query Codes

Different query codes can be requested using the **--contacts-query-by-code** option, and all possible query codes may be used at once with the **--use-all-query-codes** flag (also, *see* **Note 6**):

```
voronota-cadscore -t "target.pdb" -m "model1.pdb" --use-all-query-codes \
--output-header | column -t

target_file model_file query_code residues score    target _area model_area
target.pdb  model1.pdb AA         212      0.358224 13334.2      8287.38
target.pdb  model1.pdb AS         212      0.268364 9736.82      5192.76
target.pdb  model1.pdb SS         194      0.19971  4908.5       2055.24
target.pdb  model1.pdb AM         212      0.404073 8425.71      5708.45
target.pdb  model1.pdb MM         212      0.479336 3597.39      2736.15
target.pdb  model1.pdb MS         212      0.266683 4828.31      2707.13

voronota-cadscore -t "target.pdb" -m "model1.pdb" --contacts-query-by-code "SS" \
--output-header | column -t

target_file model_file query_code residues score   target_area model_area
target.pdb  model1.pdb SS         194      0.19971 4908.5      2055.24
```

### 3.4 Caching and Reusing Contacts

Calculated contacts may be cached in a specified directory to be reused when possible. Reading contacts from a cache directory is much faster than recomputing them. In the Bash script below, the contacts for "target.pdb" are calculated only once when scoring the first model, stored in the "tmp" directory, and reused when scoring the second model:

```
for model in "model1.pdb" "model2.pdb"
do
  voronota-cadscore --cache-dir "tmp" -t "target.pdb" -m "$model"
done

target.pdb model1.pdb AA 212 0.358224 13334.2 8287.38
target.pdb model2.pdb AA 212 0.448115 13334.2 9738.18
```

### 3.5 Tolerating Non-matching Sequences

In order to compare contacts, the CAD-score method assigns a unique identifier to every contact. A contact identifier is a pair of residue identifiers. By default, a residue identifier is comprised of the chain name, the residue sequence number, the insertion code (if present), and the residue name. If for some residue in the target structure there is no residue in the model with the exact same identifier, the CAD-score algorithm considers the residue to be completely absent from the model structure. However, this rule can be softened by using the **`--ignore-residue-names`** flag. It forces the software to ignore residue names when matching residue identifiers. For example, it allows comparison of wild-type structures with their mutants. Using this flag, in principle, structures with entirely different sequences can be compared (also, *see* **Note 7**). This possibility was not tested for global structure scoring, but it was shown to be very useful for comparison of inter-chain interfaces of homologous protein complexes [13]. For closely related protein complexes, the standard CAD-score definition may be used, but as relationships become more distant, similarities between protein-protein interfaces can be effectively assessed only using less stringent CAD-score variants defined by Eqs. 4–6.

### 3.6 Focused Scoring

The CAD-score software allows the user to specify which contacts to include in the evaluation. In other words, it is possible to restrict the *G* parameter for Eq. 1. This is done using the **`--contacts-query`** option as shown in examples below (also, *see* **Note 8**):

```
#assessing contacts between chains A and B
voronota-cadscore -t "target.pdb" -m "model1.pdb" --cache-dir "tmp" --output-header \
--contacts-query "--match-first c<A> --match-second c<B>"

target_file model_file query_code residues score    target_area model_area
target.pdb  model1.pdb AA         54       0.178782 898.033     356.19

#assessing contacts between two residue sets in chain B
voronota-cadscore -t "target.pdb" -m "model1.pdb" --cache-dir "tmp" --output-header \
--contacts-query "--match-first c<B>&r<39:51> --match-second c<B>&r<39:66,75:87>"

target_file model_file query_code residues score    target_area model_area
target.pdb  model1.pdb AA         29       0.390721 834.729     586.403
```

Let us dissect the second example. The argument to the **--contacts-query** option is a string describing two constraints. The first constraint, specified as **--match-first c<B>&r<39:51>**, means that one side of any included contacts must be a residue that is from chain B and has a sequence number in the 39–51 range. The second constraint, specified as **--match-second c<B>&r<39:66,75:87>**, means that the other side of any included contacts must be a residue that comes from chain B and has a sequence number in either 39–66 or 75–87 range. The second constraint can be rewritten using both "**&**" (logical and) and "**|**" (logical or) operators: **--match-second c<B>&r<39:66>|c<B>&r<75:87>**.

There are more possibilities for specifying contact queries. They can be explored using the graphical contact query generator `support/generate-arguments-for-query-contacts.html` that is included in the Voronota package.

### 3.7 Scoring of Interfaces and Binding Sites

When scoring inter-chain interfaces, the calculation of binding site similarity score, as defined by Eq. 4, can be enabled using the **--enable-site-based-scoring** flag. This adds additional values to the output as shown in the following example:

```
voronota-cadscore -t "target.pdb" -m "model1.pdb" --cache-dir "tmp" \
--contacts-query "--match-first c<A> --match-second c<B>" \
--enable-site-based-scoring --output-header

target_file model_file query_code residues score     target_area model_area
target.pdb  model1.pdb AA         54       0.178782  898.033     356.19

            site_residues site_score site_target_area site_model_area
            31            0.337558   898.033          675.073
```

In the above example, the binding site is defined by the interface residues of chain A because chain A was indicated with **`--match-first`**. Swapping "`c<A>`" and "`c<B>`" in the contact query forces the evaluation of a different binding site, the one in chain B:

```
voronota-cadscore -t "target.pdb" -m "model1.pdb" --cache-dir "tmp" \
--contacts-query "--match-first c<B> --match-second c<A>" \
--enable-site-based-scoring --output-header

target_file model_file query_code residues score    target_area model_area
target.pdb  model1.pdb AA         54       0.178782 898.033     356.19

     site_residues site_score site_target site_model_area
     23            0.557076   898.033     795.095
```

In both of the above examples, the "`score`" values (the values of Eq. 3) are the same, while the "`site_score`" values (the values of Eq. 4) are different. They may differ radically if, for example, in a model of a protein heterodimer only the binding site in chain A, but not the one in chain B, appears at the dimer interface.

The value of Eq. 5 is not present in the output of **voronota-cadscore**, but it can be easily calculated from the "`model_area`" and the "`target_area`" values: CAD-score$^{\text{iface-area}}$ = min(1, model_area/target_area). Similarly, the value of Eq. 6 can be calculated from the "`site_model_area`" and the "`site_target_area`" values: CAD-score$^{\text{site-area}}$ = min(1, site_model_area/site_target_area).

Instead of specifying the exact interacting regions of an interface, it is possible to ask for all the inter-chain interactions that can be found in the target structure. This is done using the **`--contacts-query-inter-chain`** flag. It also allows calculating the binding site similarity score, where the binding site is a union of all the found interface residues (in other words, the union of all the chain-specific binding sites):

```
voronota-cadscore -t "target.pdb" -m "model1.pdb" --cache-dir "tmp" \
--contacts-query-inter-chain --enable-site-based-scoring --output-header

target_file model_file query_code residues score    target_area model_area
target.pdb  model1.pdb AA         54       0.178782 898.033     356.19

   site_residues site_score site_target_area site_model_area
   54            0.447317   1796.07          1470.17
```

To attain comprehensive understanding of structural differences when assessing multimeric models, it is advisable to look at both structure and interface-related scores. For example, let us look at Table 1 with different CAD-score values for dimeric structures

**Table 1**
**Structure, interface, and binding site evaluation for two models of a dimeric structure using CAD-score**

| Score description | | Formula | model1.pdb | model2.pdb |
|---|---|---|---|---|
| Structure | Score | CAD-score($G$) | 0.358 | 0.448 |
| | Score for chain A | CAD-score($G_A$) | 0.376 | 0.424 |
| | Score for chain B | CAD-score($G_B$) | 0.375 | 0.593 |
| Inter-chain interface | Score | CAD-score$^{\text{iface}}$($A$,$B$) | 0.179 | 0.070 |
| | Area score | CAD-score$^{\text{iface-area}}$($A$,$B$) | 0.397 | 0.189 |
| Binding site | Score for chain A | CAD-score$^{\text{site}}$($A$) | 0.338 | 0.266 |
| | Area score for chain A | CAD-score$^{\text{site-area}}$($A$) | 0.751 | 0.750 |
| | Score for chain B | CAD-score$^{\text{site}}$($B$) | 0.557 | 0.528 |
| | Area score for chain B | CAD-score$^{\text{site-area}}$($B$) | 0.885 | 0.760 |

"model1.pdb" and "model2.pdb." The second model is better according to the global quality of both the overall structure and its individual chains. However, the inter-chain interface and binding sites are better predicted in the first model. Different levels of detail in the representation of binding sites help to further understand the differences. The binding site in chain A is more accurate in the first model according to detailed representation of contacts, but overall binding site areas are of approximate accuracy in both models. In contrast, the accuracy of binding site in chain B in both models is comparable according to the detailed representation, but the overall area of the binding site is better reproduced in the first model.

### 3.8 Evaluation of Homo-Oligomeric Models

Comparing homo-oligomeric structures often presents an additional challenge, because the correspondence of the chain names in the model to the chain names in the target may not be optimal. Different arrangements of model chain names may lead to different similarity scores, and the optimal arrangement is the one that results in the highest similarity score. The CAD-score software can rearrange model chain names for higher global scores, this feature is turned on with the **`--remap-chains`** flag, and the resulting rearrangement can be recorded using the **`--remap-chains-output`** option.

For example, let us consider the homotrimeric target structure "T0860o.pdb" and its model "T0860TS203_1o." Below are the inter-chain interface scoring results without and with rearranging the model chain names:

```
#without rearranging model chain names
voronota-cadscore -t "T0860o.pdb" -m "T0860TS203_1o" --cache-dir "tmp" \
--contacts-query-inter-chain --output-header | column -t

target_file model_file    query_code residues score     target_area model_area
T0860o.pdb  T0860TS203_1o AA         159      0.0336622 3201.16     147.817

#with rearranging model chain names
voronota-cadscore -t "T0860o.pdb" -m "T0860TS203_1o" --cache-dir "tmp" \
--remap-chains --remap-chains-output "remapping.txt" \
--contacts-query-inter-chain --output-header | column -t

target_file model_file    query_code residues score   target_area model_area
T0860o.pdb  T0860TS203_1o AA         159      0.40027 3201.16     2025.39

cat "remapping.txt"

A C
B B
C A
```

This example shows how the scores can go from poor to reasonable ones after simply rearranging the model chain names.

### 3.9 Residue-Level Local Scoring

Per-residue scoring can be performed at the same time as the global or the focused scoring. One way to output residue scores is by writing them in place of the B-factor values for the target and/or the model coordinates in the PDB format:

```
voronota-cadscore -t "target.pdb" -m "model1.pdb" --cache-dir "tmp" \
--output-residue-scores-pdb-t "model1_local_scores_on_target.pdb" \
--output-residue-scores-pdb-m "model1_local_scores_on_model.pdb"

target.pdb model1.pdb AA 212 0.358224 13334.2 8287.38
```

The above command writes score values only for the evaluated residues. The produced PDB files can be displayed and colored, for example, in PyMol [14]. Note that PyMol interprets missing B-factor values as zeros. A possible visualization of local scores for two models is shown in Fig. 2. Below is a PyMol script that displays a structure using a color gradient (red-white-blue colors for worst-medium-best scores):

```
load model1_local_scores_on_target.pdb
spectrum b, red_white_blue, all, 0, 1
```

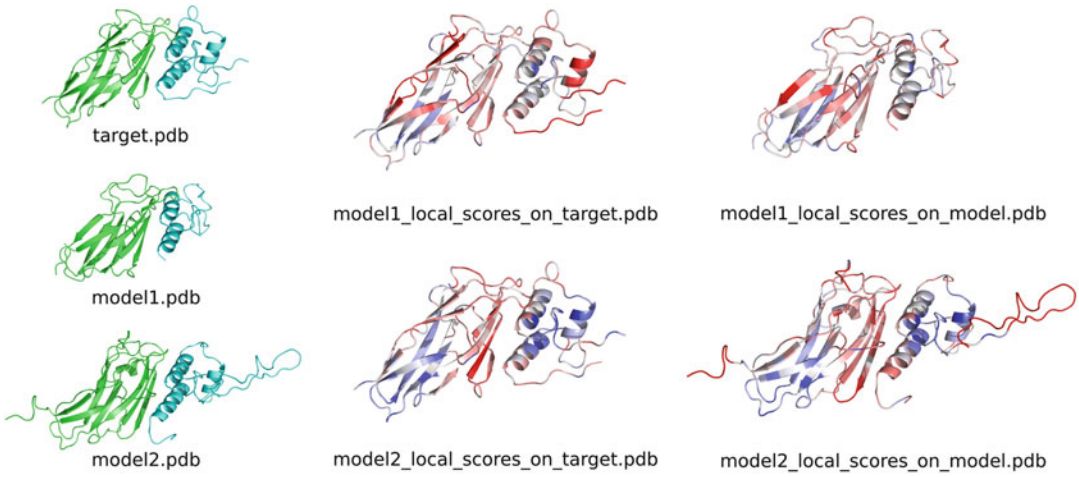


**Fig. 2** Example of structure coloring by local CAD-score values, done using PyMol. Blue-red coloring corresponds to high-low scores

When performing focused scoring (e.g., interface scoring), it may be helpful to write a default B-factor value (e.g., 99) in the output PDB files for residues that were not evaluated. One way to do this is to use the --**input-filter-query** option as shown below:

```
voronota-cadscore -t "target.pdb" -m "model1.pdb" --cache-dir "tmp" \
--input-filter-query "--set-adjuncts score=99" \
--contacts-query-inter-chain \
--output-residue-scores-pdb-t "model1_interface_local_scores_on_target.pdb" \
--output-residue-scores-pdb-m "model1_interface_local_scores_on_model.pdb"

target.pdb model1.pdb AA 212 0.358224 13334.2 8287.38
```

This sets the B-factor values of the not scored residues to 99. Then the coloring of the scored and not scored residues can be controlled separately; an example PyMol script is shown below:

```
load model1_interface_local_scores_on_target.pdb
select scored_residues, b<99
spectrum b, red_white_blue, scored_residues, 0, 1
color gray, (not scored_residues)
```

### 3.10 Detailed Analysis of Contacts

The **voronota-cadscore** script produces similarity scores, but does not output the raw contact data that the scores are derived from. The contacts for a single structure can be produced using the **voronota-contacts** script. Below is an example with a minimal set of options:

```
#calculate contacts
voronota-contacts -i "model.pdb" > contacts.txt

#print first five lines of the output
cat contacts.txt | head -5 | column -t

c<A>r<16>a<1>R<THR>A<N> c<A>r<138>a<914>R<ILE>A<CA>  0.963464 5.6312  . .
c<A>r<16>a<1>R<THR>A<N> c<A>r<138>a<917>R<ILE>A<CB>  3.00998  5.38076 . .
c<A>r<16>a<1>R<THR>A<N> c<A>r<138>a<920>R<ILE>A<CD1> 3.67065  4.46106 . .
c<A>r<16>a<1>R<THR>A<N> c<A>r<139>a<921>R<THR>A<N>   0.11352  5.17191 . .
c<A>r<16>a<1>R<THR>A<N> c<solvent>                   37.4663  5.9     . .
```

The first two columns of the output contain descriptors of the contacting atoms, the third column contains contact areas (in squared angstroms), and the fourth one contains distances between the centers of the atoms. The remaining two columns

contain additional contact-related tags (labels) and values. A single dot "." is printed when there are no tags or values to display.

For a more convenient field-based parsing (e.g., with the **awk** tool), the output can be further passed to the **voronota expand-descriptors** command that transforms each descriptor of an atom into a space-separated list of seven values (chain name, residue sequence number, insertion code, atom serial number, alternative location indicator, residue name, atom name); dots are printed in place of unavailable values:

```
voronota-contacts -i "model.pdb" \
| voronota expand-descriptors | head -5 | column -t

A 16 . 1 . THR N A       138 . 914 . ILE CA  0.963464 5.6312  . .
A 16 . 1 . THR N A       138 . 917 . ILE CB  3.00998  5.38076 . .
A 16 . 1 . THR N A       138 . 920 . ILE CD1 3.67065  4.46106 . .
A 16 . 1 . THR N A       139 . 921 . THR N   0.11352  5.17191 . .
A 16 . 1 . THR N solvent . . . . . .         37.4663  5.9     . .
```

Atom-level contact can be summarized as residue-level ones:

```
voronota-contacts -i "model.pdb" --contacts-query "--inter-residue" \
| head -5 | column -t

c<A>r<16>R<THR> c<A>r<17>R<ALA>  26.3112 1.33342 central.
c<A>r<16>R<THR> c<A>r<18>R<LYS>  1.39527 4.28654 . .
c<A>r<16>R<THR> c<A>r<138>R<ILE> 12.6313 4.05087 central.
c<A>r<16>R<THR> c<A>r<139>R<THR> 30.6752 3.25293 central.
c<A>r<16>R<THR> c<solvent>       156.25  5.69    . .
```

The possibilities for the **--contacts-query** option are the same as for the analogous option of the **voronota-cadscore** script (some examples are presented in Subheading 3.6). Possible querying parameters can be viewed by running the **voronota query-contacts --help** command. For example, using **--contacts-query "--inter-residue --no-same-chain --no-solvent"** limits its output to the contacts between residues of different chains without including contacts with the solvent.

The **voronota-contacts** command allows producing a script for drawing contacts in PyMol. In the example below, a script to display contacts between chains A and B in yellow color is generated:

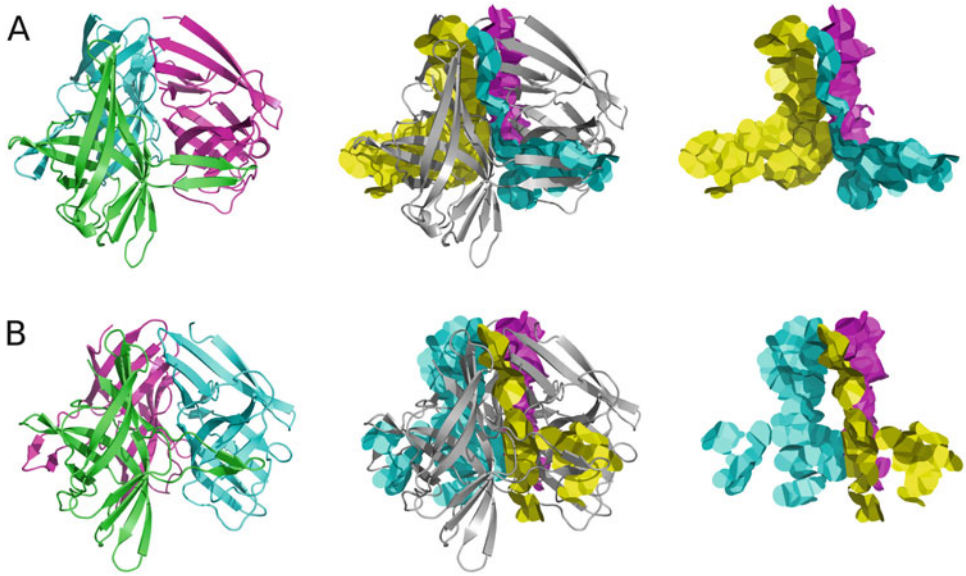


**Fig. 3** Inter-chain interface contacts drawn in PyMol for two homotrimeric structures: (**a**) CASP12 target structure "T0860.pdb"; (**b**) model structure "T0860TS203_1o.pdb"

```
#calculate contacts and generate a drawing script
voronota-contacts --cache-dir "tmp" -i "model.pdb" \
--contacts-query "--match-first c<A> --match-second c<B>" \
--output-drawing "draw_interface_AB.py" \
--drawing-parameters "--drawing-name interface_AB --default-color 0xFFFF00" \
> contacts.txt

#launch PyMol with the structure and the drawing
pymol model.pdb draw_interface_AB.py
```

The use of the **`--cache-dir`** option allows to generate several drawings for different queries without recomputing contacts every time. In order to load several drawings into PyMol, the names of the drawings should be distinct: providing the **`--drawing-name`** parameter is advised; otherwise, the name is set to "contacts." An example of multiple drawings in one scene is shown in Fig. 3, where interface contacts for different pairs of chains are displayed in distinct colors.

## 4 Notes

1. On macOS, it is advised to use Voronota 1.19 or a newer version, because earlier versions were not tested on macOS.
2. On Windows 10, the most convenient way to run CAD-score is through the Windows Subsystem for Linux.
3. The **`voronota-cadscore`** script reads both ATOM and HETATM records from PDB files. The script ignores TER records and determines chains just by the one-letter chain names from the ATOM or HETATM records.
4. If an input PDB file contains multiple MODEL blocks, then, by default, the **`voronota-cadscore`** script reads only the first MODEL block. This behavior can be changed with the **`--multiple-models`** option. It forces the script to treat input files as PDB biological assemblies (complexes assembled from the chains in every encountered MODEL block). This alters the internal representation of chain names: MODEL 1 chain names are left unchanged, and the names of the chains from the subsequent MODEL blocks are augmented with block numbers (e.g., chain "A" from MODEL 2 is renamed to "A2," chain "A" from MODEL 3 is renamed to "A3," and so on).
5. Command execution examples are presented as for the Bash shell that is the default shell for most Linux and macOS distributions.
6. Symbol "\" in a command example indicates that the command continues in the next line; "\" is not needed if a command is written in one line.
7. The **`voronota-cadscore`** script does not automatically align sequences or renumber residues in target and model structures. The correspondence between residues is determined simply based on their numbering and chain assignments in PDB files.
8. In most cases, it is necessary to enclose the argument to the **`--contacts-query`** option in quotes. Quotes are required for any argument that contains spaces or other special symbols (like "<," ">," "\&," and "|").

## Acknowledgment

This work was supported by the Research Council of Lithuania [S-MIP-17-60].

## References


1. Kabsch W (1976) A solution for the best rotation to relate two sets of vectors. Acta Crystallogr A 32(5):922–923. https://doi.org/10.1107/S0567739476001873
2. Zemla A, Venclovas Č, Moult J, Fidelis K (1999) Processing and analysis of CASP3 protein structure predictions. Proteins (Suppl 3):22–29
3. Zemla A, Venclovas Č, Moult J, Fidelis K (2001) Processing and evaluation of predictions in CASP4. Proteins (Suppl 5):13–21. https://doi.org/10.1002/prot.10052
4. Zhang Y, Skolnick J (2004) Scoring function for automated assessment of protein structure template quality. Proteins 57(4):702–710. https://doi.org/10.1002/prot.20264
5. Mariani V, Biasini M, Barbato A, Schwede T (2013) lDDT: a local superposition-free score for comparing protein structures and models using distance difference tests. Bioinformatics 29(21):2722–2728. https://doi.org/10.1093/bioinformatics/btt473
6. Olechnovič K, Kulberkytė E, Venclovas Č (2013) CAD-score: a new contact area difference-based function for evaluation of protein structural models. Proteins 81 (1):149–162. https://doi.org/10.1002/prot.24172
7. Olechnovič K, Monastyrskyy B, Kryshtafovych A, Venclovas Č (2018) Comparative analysis of methods for evaluation of protein models against native structures. Bioinformatics 35:937. https://doi.org/10.1093/bioinformatics/bty760
8. Olechnovič K, Venclovas Č (2014) Voronota: a fast and reliable tool for computing the vertices of the Voronoi diagram of atomic balls. J Comput Chem 35(8):672–681. https://doi.org/10.1002/jcc.23538
9. Olechnovič K, Venclovas Č (2017) VoroMQA: assessment of protein structure quality using interatomic contact areas. Proteins 85 (6):1131–1145. https://doi.org/10.1002/prot.25278
10. Olechnovič K, Venclovas Č (2014) The CAD-score web server: contact area-based comparison of structures and interfaces of proteins, nucleic acids and their complexes. Nucleic Acids Res 42(Web Server issue): W259–W263. https://doi.org/10.1093/nar/gku294
11. Kryshtafovych A, Monastyrskyy B, Fidelis K (2014) CASP prediction center infrastructure and evaluation measures in CASP10 and CASP ROLL. Proteins 82(Suppl 2):7–13. https://doi.org/10.1002/prot.24399
12. Haas J, Roth S, Arnold K, Kiefer F, Schmidt T, Bordoli L, Schwede T (2013) The Protein Model Portal—a comprehensive resource for protein structure and model information. Database (Oxford) 2013:bat031. https://doi.org/10.1093/database/bat031
13. Dapkūnas J, Timinskas A, Olechnovič K, Margelevičius M, Dičiūnas R, Venclovas Č (2017) The PPI3D web server for searching, analyzing and modeling protein-protein interactions in the context of 3D structures. Bioinformatics 33(6):935–937. https://doi.org/10.1093/bioinformatics/btw756
14. Schrödinger, LLC (2015) The PyMOL molecular graphics system, version 1.8